BOTSWANA INTERNATIONAL UNIVERSITY OF SCIENCE & TECHNOLOGY

Faculty of Pure and Applied Sciences

Department of Computing and Informatics

AN INTELLIGENT ENERGY EFFICIENCY FRAMEWORK IN CLOUD COMPUTING USING DEEP REINFORCEMENT LEARNING

by

NOMSA PUSO

Student ID number: 22100155 MSc (COMPUTER SCIENCE) (BIUST)

A Dissertation/Thesis Submitted to the Faculty of Pure and Applied Sciences in Partial Fulfilment of the Requirements for the Award of the Degree of Master of Science in Computer Science of BIUST

Supervisor(s): Dr Tshiamo Sigwele, Dr Dimane Mpoeleng

Department of Computing and Informatics Faculty of Pure and Applied Sciences, BIUST sigwelet@biust.ac.bw, mpoelengd@biust.ac.bw

January 2025

TABLE OF CONTENTS

ABSTRACT iv
DECLARATION AND COPYRIGHT v
CERTIFICATION vi
ANTIPLAGIARISM CHECK vii
ACKNOWLEDGEMENTS viii
DEDICATION ix
LIST OF FIGURES x
LIST OF TABLES xii
GLOSSARY OF TERMS xiii
CHAPTER 1: INTRODUCTION 15
1.1 CHAPTER INTRODUCTION 15
1.2 BACKGROUND 15
1.3 PROBLEM STATEMENT 16
1.3.1 General problem statement 16
1.3.2 Specific Problems Statement: Gaps from literature 16
1.4 OBJECTIVES 17
1.4.1 General Objective 17
1.4.2 Specific Objectives 17
1.5 RESEARCH QUESTIONS 18
1.6 CONTRIBUTION TO THE BODY OF KNOWLEDGE 19
1.6.1 Research Papers 19
1.6.2 Thesis Output Contribution 19
1.7 SIGNIFICANCE OF STUDY 19
1.8 RELEVANCE TO THE DISCIPLINE 20
1.9 EVIDENCE OF ORIGINALITY 21
1.10 ORGANIZATION OF THE THESIS 21
CHAPTER 2: LITERATURE REVIEW 23
2.1 CHAPTER INTRODUCTION 23
2.2 LITERATURE REVIEW RESEARCH METHOD 23
2.2.1 Non-Machine Learning Cloud Energy Optimization Approaches 24

2.2.2 Machine Learning Based Cloud Energy Optimization *29*
2.3 Literature Review Table 32
2.4 LITERATURE REVIEW ANALYSIS 35
2.4.1 The Most Adopted ML Models from Literature *35*
2.4.2 The Most PREVALENT Objectives from Current Approaches *36*
2.4.3 The Most Adopted ML Datasets from Literature *37*
2.4.4 The Most Adopted ML Model Evaluation Metrics from Literature. *37*
2.4.5 The Most Prevalent Limitations with the Current Approaches *39*
2.4.6 Most Adopted ML Tools from Literature *40*
2.4.7 Related Works Performance *41*
2.5 CHAPTER CONCLUSION 44
CHAPTER 3: PROPOSED FRAMEWORK 46
3.1 CHAPTER INTRODUCTION 46
3.2 RESEARCH JUSTIFICATIONS 46
3.3 THE PROPOSED FRAMEWORK ARCHITECTURE 47
3.4 PROPOSED FRAMEWORK MATHEMATICAL FORMULATIONS 48
3.4.1 User Jobs and their Characteristics *49*
3.4.2 Cloud Resources *50*
3.4.3 Job execution mechanism *50*
3.4.4 QoS Success Condition *51*
3.4.5 Energy Cost Formulation *51*
3.5 THE PROPOSED JOB SCHEDULER USING DRL 52
3.5.1 DRL Basics *52*
3.5.2 The IDBS Scheduler Architecture *54*
3.5.3 Decision Phase *54*
3.5.4 Action Space *55*
3.5.5 State Space *55*
3.5.6 Reward Function *55*
3.5.7 Experience Replay *56*
3.5.8 Target Network and Evaluation Network *57*
3.5.9 IDBS Scheduling Algorithm *57*
3.6 CHAPTER SUMMARY 61
CHAPTER 4: RESULTS AND DISCUSSIONS 62
4.1 CHAPTER INTRODUCTION 62
4.2 EXPERIMENTAL SETUP 62
4.3 EVALUATION METRICS 63

4.3.1 Success Rate (SR) *63*
4.3.2 Average Response Time *63*
4.3.3 Energy Cost *64*
4.3.4 QoS or Deadline Satisfaction Ratio (DSR) *64*
4.3.5 Resource Utilization *64*
4.3.6 Reward Metric *65*
4.3.7 Scalability *65*
4.4 MAPPING EVALUATION METRICS TO THE RESEARCH PROBLEM 66
4.4.1 Energy Consumption Problem *66*
4.4.2 Cloud Job Scheduling Problem: *66*
4.4.3 Model Optimization Problem *66*
4.5 EXPERIMENTAL PARAMETERS 67
4.5.1 Comparison Frameworks *68*
4.6 RESULTS 1: VARYING MEAN JOB ARRIVAL RATE 68
4.7 RESULTS 2: VARYING VM PROPORTION 73
4.8 RESULTS 3: VARYING JOB PROPORTION 77
4.9 CHAPTER SUMMARY 82
CHAPTER 5: CONCLUSION AND RECOMMENDATION 83
5.1 CHAPTER INTRODUCTION 83
5.2 RESULTS SUMMARY 83
5.3 OBJECTIVE ACHIEVEMENT 84
5.4 PROBLEMS ENCOUNTERED AND SOLUTIONS 87
5.5 FUTURE RESEARCH RECOMMENDATIONS 88
REFERENCES 89

ABSTRACT

The rapid expansion of cloud computing has resulted in a large increase of energy consumption, posing serious environmental and economic concerns. Data centers consume about 7% of worldwide electricity, a number that is anticipated to climb to 13% by 2030, contributing 2% of global emissions, comparable to the aviation industry. Current frameworks face challenges such as high energy consumption, poor job scheduling, non-adaptability to dynamic environments, and model optimization issues. These limitations demand the development of more efficient and adaptable solutions. This research proposes an intelligent energy efficiency framework for cloud computing based on Deep Reinforcement Learning (DRL). By dynamically adjusting to real- time workloads, this framework is designed for optimizing the energy consumption while maintaining the quality of service (QoS). This proposed DRL model formulates job scheduling through state and action spaces, incorporating real-time job characteristics such as arrival rates, types, and deadlines. The framework includes decision-making phases and reward functions to ensure effective and adaptable scheduling. The results show that the proposed DRL framework significantly outperformed related works demonstrating a 38% reduction in cloud energy consumption, a 98.4% jobs success rate, an improvement in QoS of 115%, a 66% reduction in response times, and a resource utilization of 90.7%. Additionally, the framework's scalability and adaptability to dynamic cloud conditions were validated through rigorous testing under various scenarios, confirming its robustness in handling fluctuating workloads and diverse job types. These outcomes address the identified problems by significantly enhancing energy efficiency, job scheduling effectiveness, and overall system performance.

Keywords – Deep Reinforcement Learning (DRL), Cloud Computing, Energy Efficiency, Job Scheduling, Quality of Service (QoS), Resource Utilization.

DECLARATION AND COPYRIGHT

BOTSWANA INTERNATIONAL UNIVERSITY OF SCIENCE & TECHNOLOGY

POSTGRADUATE SCHOOL

Declaration Regarding the Work and Copyright

DECLARATION REGARDING THE WORK AND COPYRIGHT

Candidate (please write in caps or type) _____ NOMSA PUSO_____

Student ID: _____**22100155** _____

Thesis Titled: **AN INTELLIGENT ENERGY EFFICIENCY FRAMEWORK IN CLOUD COMPUTING USING DEEP REINFORCEMENT LEARNING**

I, the **Candidate**, certify that the Thesis is all my own original work and that I have not obtained a degree in this University or elsewhere on the basis of any of this work.

(If the thesis is based on a group project, then the student must indicate the extent of her / his contribution, with reference to any other theses submitted or published by each collaborator in the project, and a declaration to this effect must be included in the thesis)

This dissertation/thesis is copyright material protected under the Berne Convention, the Copyright and Neighbouring Rights Act, Act. No. 8 of 2000 and other international and national enactments, in that behalf, on intellectual property. It must not be reproduced by any means, in full or in part, except for short extracts in fair dealing; for researcher private study, critical scholarly review or discourse with an acknowledgement, without the written permission of the office of the Postgraduate School, on behalf of both the author and the BIUST.

Signed: ---------- Date: ----------31-AUG-20234----------

Primary Supervisor (please write in caps or type) _____DR TSHIAMO SIGWELE_____

I, the Candidate's **Primary Supervisor**, hereby confirm that I have inspected the above titled thesis and, to the best of my knowledge, it is based on the original work of the candidate.

Signed: ---------- Date: ----------08-OCT-2024----------

CERTIFICATION

In order to fulfil the requirements for the Master of Science in Computer Science degree at the BIUST, the undersigned certifies that he or she has read the dissertation or thesis titled: *An Intelligent Energy Efficiency Framework In Cloud Computing Using Deep Reinforcement Learning*, and hereby recommends for acceptance by the Faculty of Science.


……..…..

……..DR TSHIAMO SIGWELE…...

Date: …. 25 January 2025 ……..

ANTIPLAGIARISM CHECK

Used Software: ***Turnitin feedback studio***
Result: ***Similarity Index Score of 10%***

BOTSWANA INTERNATIONAL UNIVERSITY OF SCIENCE & TECHNOLOGY

FORM PGS - 330

POSTGRADUATE SCHOOL

Antiplagiarism check

MEMORANDUM

Postgraduate candidate: **NOMSA PUSO** **Student ID: 22100155**

Degree level: PhD [ ], or Master [✓] *(tick the appropriate)*

Thesis Titled: **AN INTELLIGENT ENERGY EFFICIENCY FRAMEWORK IN CLOUD COMPUTING USING DEEP REINFORCEMENT LEARNING**

The Antiplagiarism Check had been completed and the results is detailed below.

Software and version used: **TURNITIN FEEDBACK STUDIO**

Outcome: **10% SIMILARITY INDEX SCORE**

Resolution: Comment from the Supervisor needed: YES [ ], or NOT [✓] *(tick the appropriate)*

Officer who completed the check (please write in caps or type) **KELEBONYE BAGAI**

Signed: Date: **08-OCTOBER-2024**

COMMENTS BY THE PRIMARY SUPERVISOR (*if requested*): **NONE**

Primary supervisor (please write in caps or type): **DR TSHIAMO SIGWELE**

Signed:---------- Date: **08-OCTOBER-2024**

ACKNOWLEDGEMENTS

Words cannot fully express my heartfelt gratefulness to the God almighty, who, through His grace, blessings, guidance and strength, made it possible for me to successfully complete this thesis.

Also, I extend my profound appreciation to my supervisors Dr Tshiamo Sigwele and Dr Dimane Mpoeleng for their unwavering guidance, invaluable insights and continuous encouragement throughout the entire research process. Moreover, I want to thank my research group, coworkers and friends for their assistance and motivation which made this journey a memorable and enriching experience.

Lastly, I would wish to express my deepest gratitude to my mother, my family whose love, support, understanding and encouragement served as a continual source of strength.

DEDICATION

I humbly dedicate this work to my mother, as well as my siblings; without your support this, would not have been possible. My sincere thanks especially to my mom for her support, love and inspiration to strive for excellence. I appreciate all that you have done for me towards the success of this work. I love you always.

LIST OF FIGURES

Figure 1. Energy-Efficient Approaches for Cloud Computing 24
Figure 2. DVFS Aware consolidation methods. 25
Figure 3. VM Placement methods. 26
Figure 4. VM Migration methods. 28
Figure 5. VM Selection methods. 29
Figure 6. ML types and techniques or algorithms. 30
Figure 7. Most adopted ML models from related works. 36
Figure 8. Most Adopted research objectives from related works. 36
Figure 9. Most adopted ML datasets from related works. 37
Figure 10. Most adopted ML metrics from literature. 38
Figure 11. Most prevalent limitations/Gaps in current related works. 39
Figure 12. Most adopted ML tools from literature. 41
Figure 13. Performance analysis of related works based on energy consumption metric. 43
Figure 14. Performance analysis of related works based on accuracy metric. 43
Figure 15. Performance analysis of related works based on latency metric. 43
Figure 16. Performance analysis of related works based on Service Level Violation. 44
Figure 17. Performance analysis of related works based on Resource Utilization 44
Figure 18. Proposed Scheduling Framework. 48
Figure 19. The proposed IDBS job scheduling model based on DRL. 54
Figure 20. RESULTS 1A: Varying the mean job arrival rate vs model average response time. 69
Figure 21. RESULTS 1B: Varying the mean job arrival rate vs model success rate. 70
Figure 22. RESULTS 1C: The impact of varying job arrival rates on VM resource utilization 71
Figure 23. RESULTS 1D: The impact of varying job arrival rates on energy cost. 72
Figure 24. RESULTS 1E: The impact of varying job arrival rates on QoS. 73
Figure 25. RESULTS 2A: The impact of varying VM type proportions on average response time... 74
Figure 26. RESULTS 2B: The impact of varying VM type proportions on model success rate. 74
Figure 27. RESULTS 2C: The impact of varying VM type proportions on resource utilization. 75

Figure 28. RESULTS 2D: The impact of varying VM type proportions on energy cost. *76*
Figure 29. RESULTS 2E: The impact of varying VM type proportions on QoS. *77*
Figure 30. RESULTS 3A: The impact of varying job proportions on average response time. *78*
Figure 31. RESULTS 3B: The impact of varying job proportions on model success rate *79*
Figure 33. RESULTS 3C: The impact of varying job proportions on resource utilization. *80*
Figure 33. RESULTS 3D: The impact of varying job proportions on energy cost *80*
Figure 34. RESULTS 3E: The impact of varying job proportions on QoS *81*

LIST OF TABLES

Table 1. Related Works on ML Energy Efficiency in Cloud Computing. *33*
Table 2. Notations for the proposed scheduling model. *49*

GLOSSARY OF TERMS

ACRONYM	DESCRIPTION
ABC	Artificial Bee Colony
ADVMC	Adaptive Virtual Machine Consolidation Mechanism
AI	Artificial Intelligence
ART	Average Response Time
BIO-VMP	Bio-Inspired Virtual Machine
CE	Carbon Emissions
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DNN	Deep Neural Network
DQN	Deep Q-Network
DRL	Deep Reinforcement Learning
DSR	Deadline Satisfaction Ratio
DVFS	Dynamic Voltage Frequency Scaling
EC	Energy Cost
EEHVMC	Energy Efficiency Heuristic Using Virtual Machine Consolidation
FCFS	First-Come First-Served
FP-NSO	Flower Pollination-Based Non-Dominated Sorting Optimization
FPO	Flower Pollination
GM	Global Manager
HML	Host Medium Loaded
HOL	Host Overloaded
HUL	Host Under-Loaded
I/O	Input And Output Devices
IC	Influence Coefficient
IDBS	**I**ntelligent **D**eep Reinforcement Learning **B**ased Job **S**chedular
IDE	Integrated Development Environment
IT	Information Technology

ACRONYM	DESCRIPTION
ICVMS	Influence Coefficient Based Virtual Machine Selection
LM	Local Manager
LSTM	Long Short-Term Memory
MDP	Markov Decision Process
ML	Machine Learning
MLP	Multilayer Perceptron
MOABC	Multi-Objective Artificial Bee Colony
MRCU	Maximum Ratio of CPU Utilization
NIST	National Institute of Standards And Technology
NSGA-II	Non-Dominated Sorting Technique-Based Genetic
PADRL	Prediction-Aware Deep Reinforcement Learning
PM	Physical Machine
Q-FUNCTION	Action-Value Function
QoS	Quality Of Service
RF	Random Forest
RILTA	Reinforcement Learning Based Task Assignment
RL	Reinforcement Learning
RR	Round Robin
SDN	Software-Defined Networking
SLA	Service-Level Agreement
SR	Success Rate
SVM	Support Vector Machine
TD	Temporal Difference
V-FUNCTION	State-Value Function
VCPU	Virtual Central Processing units
VM	Virtual Machine
VMP	Virtual Machine Placement
VMM	Virtual Machine Monitor
WSVM	Wavelet Support Vector Machine

CHAPTER 1: INTRODUCTION

1.1 CHAPTER INTRODUCTION

This chapter provides an overview of the entire research work. It presents the research background, problem statement, objectives, research questions, research contributions, significance of the study, relevance to the discipline, evidence of originality, and organization of the thesis.

1.2 BACKGROUND

Cloud computing is a fast-growing technology combining two significant trends: Information Technology (IT) efficiency and business agility [1][2]. According to the National Institute of Standards and Technology (NIST), Cloud computing is a unique model that allows for easy, on-demand network access to a shared pool of configurable computing resources that can be quickly provisioned and released with the least amount of management work and service provider interactions. These resources are servers, storage, applications, services and networks. The cloud provides unrestricted data storage as well as the capacity to hide a large volume of data from other users. Users can get on-demand access to the necessary files, documents, and applications. Instead of purchasing expensive infrastructure, users only pay for the services provided by the cloud suppliers. A data center's power consumption is divided into three categories: servers, cooling systems and data centre networks [1]. Servers consume 40% to 55% while cooling systems take 15% to 30% of the power [1]. The network consumes 10% to 25% of the power and it was also reported that cloud data centers alone consume approximately 7% of global electricity and are expected to rise to 13% by 2030 [3]. This sector is also responsible for an estimated 2% of global emissions, comparable to the aviation industry. It is anticipated that the energy consumption of data centers will exceed 140 billion kilowatt-hours annually [4]. The energy-saving scheduling of data centers is critical for cloud service providers and will also contribute to environmental sustainability. This research proposes an intelligent energy efficiency framework in cloud computing using Deep Reinforcement Learning (DRL) called IDBS (Intelligent Deep Reinforcement Learning Based Job Schedular) which is designed for optimizing the consumption of energy while maintaining the quality of service (QoS). The proposed DRL model will formulate job scheduling through state and action spaces, incorporating real-time job characteristics such as arrival rates, types, and deadlines. DRL has real world applications such as robotics, finance and healthcare that have become a natural choice to support energy efficiency in cloud computing.

1.3 PROBLEM STATEMENT

1.3.1 General problem statement

Studies as elaborated in the background section above have reported that cloud data centers alone consume approximately 7% of global electricity and are expected to rise to 13% by 2030. This sector is also responsible for an estimated 2% of global emissions, which is comparable to the aviation industry. The cloud data center components consume energy from servers, cooling systems, and data centre networks. The servers consume 40% to 55% of the power while cooling systems take 15% to 30% of the power and the network consumes 10% to 25% of the power. Experts predict that data centers will consume over 140 billion kilowatt-hours of energy annually. Energy-efficient data center scheduling is critical for cloud service providers as it will also contribute to environmental sustainability.

1.3.2 Specific Problems Statement: Gaps from literature

The specific problem statement deals with the gaps in the current related work and it's the main research problem that this research work addresses. Containers and Virtual Machines (VMs) are both resource virtualization technologies that make applications independent from IT infrastructure resources and the scope of the thesis is on VMs. In the Literature Review Chapter, the research reviewed and critically analyzed several frameworks on energy efficiency in cloud computing using machine learning, considering criteria such as research objectives, performance, evaluation metrics, and framework limitations. This research work will use these criteria to construct the research problem statement. Below are the gaps from literature that this research would address.

Energy Consumption Problem: The major research gap in literature is the high energy consumption of the current frameworks and 28% of the research objectives from the literature review work are on optimizing energy consumption. Of all the reviewed literature, the energy consumption evaluation metric is the most widely adopted, contributing 21%, showing that energy consumption is a major concern in the cloud.

Cloud Job Scheduling Problem: The other major research gap to address is the job scheduling or resource allocation problem, which collectively contributes to 17% of the related works research objectives and 10% of related works limitations. The current frameworks suffer poor job scheduling due to non-adaptability to dynamic and real-time cloud environments where jobs arrive rapidly in real-time. Almost all the reviewed scheduling or resource allocation frameworks concentrated on handling non-realistic and non-real-time transactional workloads where jobs arrive in batches in fixed patterns. They depend on datasets that are not real-time. The current

frameworks are non-scalable and non- adaptive to the dynamic nature of the cloud, which requires autoscaling to adjust resources in response to workload demands. The non-real-time decision-making and static cloud job scheduling nature of the current work make allocating user jobs to cloud resources a very costly task in terms of quality of service and energy consumption. In addition, current frameworks consider homogeneous workloads with only one job type and homogeneous cloud resources with one type of virtual machine (VM).

Model Optimization Problem: In addition, there is a limitation on model optimization with a 15% contribution from literature. Model optimization entails the minimization and maximization of critical metrics that are detrimental to the framework's optimal performance. According to the literature, there has been poor performance on major evaluation metrics in energy-efficient machine learning frameworks. These metrics have low model success rate or accuracy (28%), poor response time and quality of service (8%), poor resource utilization (4%), and low average commutative reward (4%). The percentages show how often these metrics are used in literature. Furthermore, there is a significant deficiency of single objective optimization that only focuses on maximizing just one metric of energy consumption, without considering other crucial metrics such as quality of service (QoS).

1.4 OBJECTIVES

1.4.1 General Objective

To develop Deep Reinforcement Learning (DRL) framework for optimizing energy consumption in cloud computing while maintaining the Quality of Service (QoS).

1.4.2 Specific Objectives

1. To determine the limitations/gaps, most adopted machine learning (ML) models, datasets, evaluation metrics, and current related works performance from the literature.

2. To develop a real-time job scheduling model based on DRL to minimize energy consumption in cloud computing while maintaining QoS.
3. To evaluate the proposed framework and comparing it with current works in terms of energy consumption, model success rate, response time, QoS and resource utilization.

1.5 RESEARCH QUESTIONS

Research Questions for Objective 1: *To determine the limitations/gaps, most adopted ML models, datasets, evaluation metrics, and current related works performance from the literature.*

1. What are the limitations of the ML related works in minimizing cloud computing energy consumption?
2. What are the most adopted ML models used in literature to minimize cloud energy consumption?
3. What are the most adopted ML datasets used in the related works to minimize cloud energy consumption?
4. What are the most adopted metrics in literature for evaluating ML frameworks cloud data center energy minimization?
5. How do related works perform based on the most adopted evaluation metrics?

Research Questions for Objective 2: *To develop a real-time DRL based job scheduling model to minimize energy consumption in cloud computing while maintaining QoS.*

1. How will the proposed cloud job scheduling model be mathematically formulated?
2. What and how will the real-time job/workload characteristics be generated?
3. How will the proposed scheduling model minimize cloud energy consumption?
4. How is the QoS measured, and what determines whether it is maintained?
5. How will the DRL phases map with the cloud job scheduling model in terms of the decision phase, state phase, action space, reward function, and training phase?
6. How will the proposed framework deal with the dynamic and unpredictable nature of the cloud?

Research Questions for Objective 3: *To evaluate the proposed framework by comparing with current works in terms of energy consumption, model success rate, response time, QoS and resource utilization.*

1. What parameters will be used and how will the experiment be performed for evaluation.
2. Compared to related frameworks, how does the proposed DRL framework perform in terms of various metrics like energy consumption, model success rate, response time, QoS and resource utilization.
3. How does the proposed model perform in terms of scalability and the changing cloud conditions?

1.6 CONTRIBUTION TO THE BODY OF KNOWLEDGE

1.6.1 Research Papers

1. Journal Article 1(Published):

   Nomsa Puso, Tshiamo Sigwele, and Oba Zubair Mustapha. "Machine Learning Centered Energy Optimization in Cloud Computing: A Review." ***Indonesian Journal of Electrical Engineering and Informatics (IJEEI),*** vol. 11, no. 3, 2023.
   DOI: 10.52549/ijeei.v11i3.5037

2. Journal Article 2 (Submitted):

   Nomsa Puso, Tshiamo Sigwele, Mpoeleng Dimane, Tshenyego Onkemetse. " A Smart DRL Based Job Scheduler for Green Cloud Computing." ***International Journal of Computers and Applications (IJCA),*** vol. 00, no. 0, 2024.
   PDF LINK: https://drive.google.com/file/d/1EFWPFdksKcliEPr7kSUUxm0s5QJ039EE/view?usp=sharing

3. Conference paper 3 (Published):

   Nomsa Puso, Tshiamo Sigwele, Dimane Mpoeleng, "Energy Efficiency for Cloud Data Centers Using Machine Learning in Botswana", ***Teaching Research & Innovation Symposium 2023 (TRDAIS)***, Palapye, Botswana, pp. 1400-1415, 2023,
   DOI: https://repository.biust.ac.bw/handle/123456789/593

1.6.2 Thesis Output Contribution

A robust real-time job scheduling framework based on DRL that contributes to the minimization of energy consumption in cloud computing without degrading QoS. The framework offers an additional blueprint for future researchers to expand and replicate the research (algorithms are directives in the thesis) to further save energy in cloud computing data centers.

1.7 SIGNIFICANCE OF STUDY

This research work is significant because high energy consumption has become a rising concern in the operation of complex cloud data centers, due to the ever-expanding size of cloud computing facilities and ever-increasing number of users. As the demand for cloud service grows, so does the data centers energy consumption. Cloud computing infrastructures often need a large amount of power that contributes to environmental concerns. Hence, concentrating on energy efficiency can help in lowering the carbon footprint associated with cloud computing. Since energy-efficient practices and resource optimization often go hand in hand, DRL can optimize and dynamically adjust resource allocation, resulting in more effective use of computational resources. Cloud computing is well known for its own scalability and energy efficiency which is critical for sustaining this scalability without increasing the environmental effect proportionally. This research provides insights into how to make cloud services more scalable while minimizing energy

consumption. Meanwhile, the proposed DRL framework will significantly minimize or reduce energy consumption while maintaining the quality of service. Further significances of study can be extracted from the entire thesis as the findings deepen the current understanding of energy efficiency in cloud computing using machine learning.

1.8 RELEVANCE TO THE DISCIPLINE

Energy efficiency in cloud computing using machine learning connects with several disciplines, including computer science, cloud computing, sustainable computing, artificial intelligence, energy, and environmental sciences, as well as business and economics. Collaboration and interdisciplinary methods can help to improve energy consumption and promote sustainable practices in cloud computing. This work finds relevance to the research disciplines as follows:

1. **Computer Science and Engineering**

Energy efficiency in cloud computing entails creating and implementing algorithms, models, and systems to maximize resource utilization and reduce energy usage. Computer scientists and engineers play an essential role in developing machine learning approaches, algorithms, and tools for improving energy efficiency in cloud environments.

2. **Cloud Computing and Data Centers**

Energy efficiency is a top priority for cloud service providers and data centre operators. The discipline of cloud computing is concerned with the design, management, and operation of cloud infrastructures, and energy efficiency is a critical component of assuring cost-effectiveness and sustainability. Machine learning techniques can help improve resource management, workload distribution and power optimisation in cloud systems.

3. **Sustainable Computing**

Sustainable computing seeks to minimize the environmental impact of computer systems and data centre. Energy efficiency in cloud computing is an important aspect of sustainable computing techniques. Machine learning can help achieve energy-efficient operations in cloud infrastructures and data centers, which aligns by the goals of sustainable computing.

4. **Artificial Intelligence and Machine Learning**

The use of machine learning methods and approaches to improve energy efficiency in cloud computing is consistent with the discipline of artificial intelligence. Researchers and practitioners in artificial intelligence and machine learning can investigate and develop models, algorithms, and tools to improve energy management and optimisation in cloud systems.

5. Energy and Environmental Sciences

Energy efficiency in cloud computing has an impact on both energy consumption and the environment. Researchers in energy and environmental sciences can use cloud infrastructure's energy profiles, resource utilization patterns, and efficiency measures to design energy-saving methods. Machine learning may be used to analyze and predict energy consumption, contributing to further research in this field.

1.9 EVIDENCE OF ORIGINALITY

The signed “Declaration and Copyright” section at the beginning of this thesis justifies that this work was done by Nomsa Puso. As a sign of originality, the accepted publications were peer reviewed and published in high quality journals by experts who also checked for plagiarism. Also, the adopted methods, ideas and tools were cited in this research work. The evidence of originality can also be deduced from this entire research thesis.

1.10 ORGANIZATION OF THE THESIS

The thesis is organised as follows:

Chapter 1: Introduction - This chapter introduces the research problem, outlining the significant increase in energy consumption due to the rapid growth of cloud computing. It highlights the importance of developing efficient and adaptable solutions to address the environmental and economic concerns associated with high energy usage. The chapter also presents the research objectives, research questions, and the overall structure of the thesis.

Chapter 2: Literature Review - This chapter provides a comprehensive review of the existing literature related to machine learning (ML) models, datasets, evaluation metrics, and current frameworks for minimizing energy consumption in cloud computing. It identifies the limitations and gaps in the existing approaches and discusses the most adopted ML models and evaluation metrics. This review sets the foundation for the development of the proposed DRL-based framework.

Chapter 3: Proposed Framework - This chapter details the methodology and implementation of the real-time Deep Reinforcement Learning (DRL)-based job scheduling model. It describes the mathematical formulation of the model, the generation of real-time job characteristics, and the mapping of DRL phases with the job scheduling model. The chapter also explains how the model minimizes energy consumption and maintains Quality of Service (QoS). Additionally, it discusses the implementation of the proposed DRL-based framework, covering the coding and simulation environment, the tools and libraries used, and the integration of various components. Challenges faced during implementation, such as debugging and dependency issues, are also discussed.

Chapter 4: Results and Discussion - This chapter presents the evaluation of the proposed framework by comparing it with related works frameworks. The frameworks compared include First-Come-First-Served (FCFS), Round Robin (RR), and JingDQN. It details the experimental setup, parameters used, and the metrics for performance evaluation. The results demonstrate the framework's superior performance in terms of energy consumption reduction, response times, resource utilization, and job completion success rates. Additionally, the chapter provides a detailed discussion of the results, linking them back to the research objectives and questions. It explains how the proposed framework addresses the identified problems, enhances energy efficiency, and improves job scheduling effectiveness. The scalability and adaptability of the framework to dynamic cloud conditions are also discussed.

Chapter 5: Conclusion and Recommendations - This chapter summarizes the key findings and contributions of this research and presents the recommendations for future research based in terms of the findings and limitations of this study. It highlights the achievements of the research objectives, addresses the research questions, and emphasizes the significance of the proposed DRL-based framework in optimizing energy consumption while maintaining QoS in cloud computing environments. This chapter outlines the main problems encountered during the research. It discusses issues related to acquiring paid research papers, code debugging, Python library dependencies, large-scale simulations, real-time data acquisition, and integrating various cloud simulation tools. The chapter suggests exploring the integration of renewable energy sources, advanced cooling techniques, Software-Defined Networking (SDN), edge computing, and digital twin technology to further enhance energy optimization in cloud computing. It also highlights the importance of expanding datasets to include diverse workloads and real-time data for enhancing the robustness and generalizability of the framework. The chapter concludes by reaffirming the framework's versatility, robustness, and potential for future advancements in cloud computing.

CHAPTER 2: LITERATURE REVIEW

2.1 CHAPTER INTRODUCTION

This chapter aims to answer the gap analysis research questions presented in Section 1.5 by conducting a comprehensive literature with a summary justification of the best options (methodologies, tools, languages, frameworks, evaluation metrics, etc.) to adopt for the development of the proposed deep reinforcement learning framework from the literature. The chapter will define energy efficient approaches for cloud computing in Section 2.2.1. This chapter will also present related works in Section 2.2.2 and give a literature review analysis for the most adopted models (60% adopted DRL model), objectives (the most dominant research objective is power or energy consumption reduction at 28%), datasets (40% being dominated by CloudSim [1]), machine learning model evaluation metrics (Energy consumption at 13% followed by Accuracy at 4%, Power consumption at 4% and many more), prevalent limitations (Poor performance or poor model accuracy or loss of accuracy at 15%, model optimization problems at 15% also), and tools (52% has adopted the TensorFlow ML tool) (Section 2.3) for the DRL framework development process. The chapter concludes by a summary from the literature review with justification.

2.2 LITERATURE REVIEW RESEARCH METHOD

The objective of this review section is to provide a comprehensive overview of state-of-the- art research and advancements in machine learning techniques for energy optimization in cloud computing environments. The review will concentrate on the recent research, which is published in peer-reviewed journals or publications, conference proceedings and other credible sources. For selecting the literature, the time frame will be from 2017 to 2023. The review will follow an organised systematic approach to literature review, including the following key steps: (1) Literature searching using online academic databases such as SpringerLink, ScienceDirect, IEEE Xplore, Google Schoolar, ACM Digital Library and Elsevier's Cloud Computing Journal for identifying the relevant articles using keywords like "machine learning," "energy optimization," "cloud computing," "power-efficient," "resource allocation". (2) Inclusion Criteria of the articles that specifically address machine learning techniques applied to energy optimization in cloud computing between 2017 and 2023. (3) Exclusion Criteria of the articles that do not focus on machine learning or energy optimization, or those that are not related to cloud computing. (4) Data Extraction of key information from the selected articles or journals with the title, authors and years, machine

learning models, research objectives, evaluation metrics, model limitations, datasets and tools used. (5) Synthesis: Organizing the acquired information into a summarized table then extracting trends and patterns of the most adopted models, objectives, metrics, tools, and datasets. For a detailed overview of cloud computing, the reader is directed to [5][6][7][8].

2.2.1 Non-Machine Learning Cloud Energy Optimization Approaches

Figure 1 shows several non-machine learning based energy efficiency techniques in cloud computing, categorized into Dynamic Voltage Frequency Scaling (DVFS), Aware consolidation, VM placement, VM migration, and VM selection. These techniques are described in the subsequent sections.

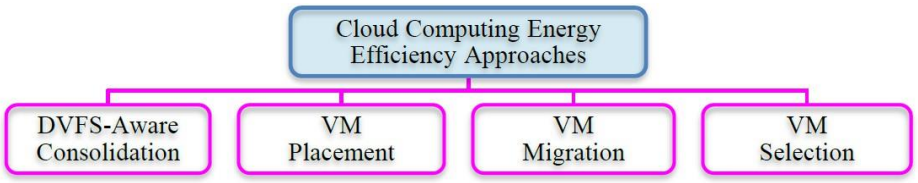


Figure 1. Energy-Efficient Approaches for Cloud Computing

(1) DVFS-Aware Consolidation Energy Efficiency Methods*:* Dynamic Voltage Frequency Scaling (DVFS) is a state-of-the-art energy-saving method for reducing the consumption of power in current computer systems. Through this method, CPUs run at different clock frequencies and various combinations of voltages based on requirements of the system performance at a given time [9]. Consolidation refers to the live migration of virtual machines from one host to another with minimum execution-related delays [10]. DVFS-Aware Consolidation converts the frequency and voltage automatically to cut down on CPU heat generation and power consumption. Furthermore, less heat generated enables the cooling system to be switched off while consuming less energy [9]. In order to minimize energy or power consumption, limit VM migrations and reduce service level agreement violations, the author in [9] proposed an energy efficiency heuristic employing VM consolidation (EEHVMC). The fundamental concept categorizes host computers according to their memory and CPU consumption. The host machines are divided into three primary groups based on two criteria concerning the memory and CPU usage: host overloaded (HOL) and host medium loaded (HML). Lastly, the host underloaded (HUL) machines. VMs were moved to the HML as HOL to reduce the power intake in data centers. The suggested method reassigned VMs to the HML from the HUL hosts and switched the inactive hosts into a power-saving mode [9]. For energy-efficient cloud data centers, the author [11] created an adaptive VM consolidation mechanism called ADVMC which is based on DRL technique. It implements both the VM placement and selection for reducing energy

consumption and eliminating the SLA violations of users as compared to many other VM consolidation strategies. Based on workload detection, there are three types of hosts in a cloud data center namely: overloaded-hosts, underloaded-hosts, and regular-hosts whose workload falls between that of overloaded-hosts and underloaded-hosts [11].

The author in [12] proposed an energy-efficient and quality of service aware VM consolidation strategy for improving resource utilization and saving energy on the cloud data centers. Dynamic VM consolidation is a highly effective strategy for increasing resource utilization and maintaining a regular operational state while upholding SLAs for all hosts. To efficiently reallocate VMs to hosts, this dynamic VM consolidation strategy is divided into several steps which are: identifying overloaded host, choosing the migrated VM from the overloaded host, identifying the underloaded host then placing the VM. Figure 2 shows a summary of the evaluation metrics, strengths and limitations of DVFS Aware consolidation method for cloud computing energy optimization.

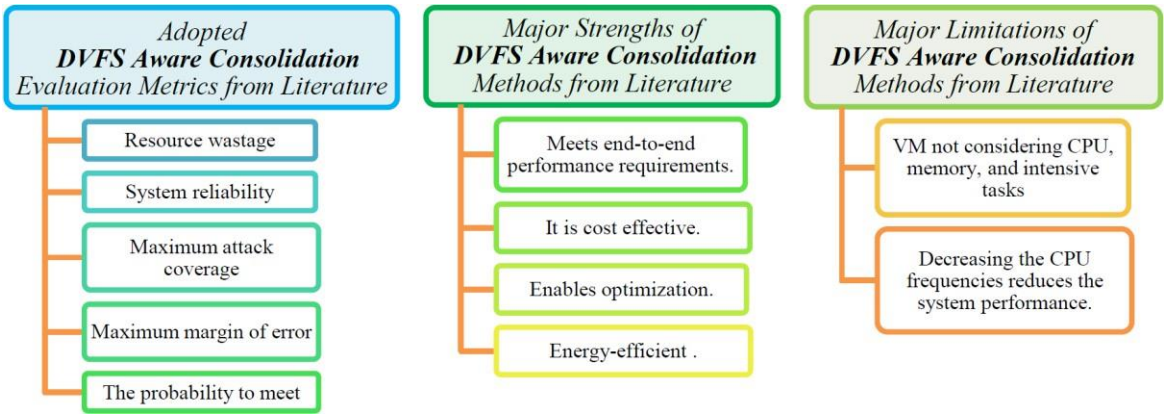


Figure 2. DVFS Aware consolidation methods.

(2) VM Placement Energy Efficiency Methods: VM placement methods are sometimes called VM Allocation or VM Scheduling. In large cloud data centers, VM placement (VMP) is the process of choosing the most suitable Physical Machine (PM) to host the user's requested virtual machines [13]. It is a technique that executes VMs to enhance power efficiency and maximize resource utilization. The author in [13] proposed a holistic resource management strategy named bio-inspired virtual machine (Bio-VMP) for cloud environment that is both sustainable and energy efficient. This technique used a novel flower pollination-based on non-dominated sorting optimization (FP-NSO) method which maximizes the resource utilization while minimizing carbon emissions (CE) and energy consumption of the cloud data center. It is founded on the principles of non-dominated sorting technique-based genetic method (NSGA-II) and flower pollination optimization (FPO). Additionally, it investigates the most possible and optimal virtual placement allocations subjected to energy consumption, resource utilization, and CE [13]. A prediction-aware deep reinforcement learning

(DRL) based VM placement technique (PADRL) was created by the author in [11], in order to identify acceptable hosts for virtual machines to be migrated in an effective manner. The two main primary components of the PADRL approach are deep Q-learning (DQN) based VMP and long short-term memory (LSTM) based state prediction. The virtual machines that required to be migrated were set up on suitable hosts, incorporating all the virtual machines from underloaded-hosts and a few from overloaded-hosts. This method is achieved in terms of energy-saving and reduction of service level agreement violations in data centers [11].

The author in [14] presented a multi-objective VMP for achieving the best VMs to PM mapping using the dominance-based multi-objective artificial bee colony (MOABC) technique. This technique balanced the overall consumption of energy, system reliability and resource waste to satisfy the requirements of QoS and SLA. The list of migrating virtual machines, destination physical machines, number of solutions as well as the maximum number of repetitions is the first input parameters used by the suggested algorithm [14]. The author in [12] proposed a virtual machine placement heuristics strategy called CUECC to choose a target host that offers the greatest reward by considering both real-time CPU usage and variations in energy consumption during the deployment of virtual machines. The global manager (GM) gathered the operational status of hosts within cloud data center, while local manager (LM) monitored the hosts status to check if they are underloaded or overloaded. Therefore, the least underloaded host migrated entirely all the VMs to other hosts then switched the host to an inactive state for saving energy. Figure 3 shows a summary of the evaluation metrics, strengths and limitations of VM Placement method for cloud computing energy optimization.

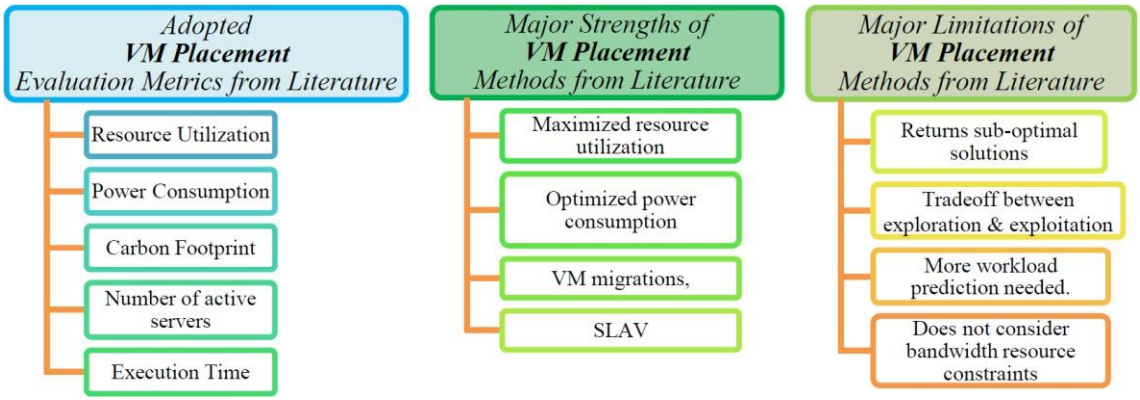


Figure 3. VM Placement methods.

(3) VM Migration Energy Efficiency Methods: Virtual Machine (VM) migration is an enchanting feature of virtualization technology that plays an essential role in the administration of cloud data centers, it shifts the operational VMs from a physical machine to one another [15]. Author in [15] proposed a plethora of VM migration strategies, aiming at serving the QoS-

driven user requirements and reviewing the most recent and cutting-edge load balancing, energy-aware, SLA-aware, and network-aware live VM methods based on artificial intelligence (AI). Several methods have been proposed to achieve effective VM migration and these techniques are divided into non-live and live VM migration. The live VM migration as the name implies keeps providing services to users while the virtual machine is being moved to ensure continuous connectivity and efficient resource utilization. Whereas the non-live VM migrations stop all the virtual machine services while it is in the process and resumes them when it is done. The VM migration controller manages all aspects of the VM migration process, whether it is non-live or live [15]. To address the issues faced due to the increment of cloud data centers, the author in [16] offered an energy-aware virtual machine allocation and migration strategy. This approach employs a machine learning-based artificial bee colony (ABC) algorithm which is used to rank the VMs according to their load while prioritizing energy efficiency as a vital metric. The most efficient virtual machines were selected based on the load and energy dynamics, allowing applications to be migrated between virtual machines. Additionally, active virtual machine servers helped prevent the resource underutilization by reducing the idle resource time [16].

The author in [17] proposed a VM migration method called V2PQL which is grounded in Q-learning technique and the Markov decision process. One of the main benefits of this virtual technology is that it freed VMs from the underlying hardware, allowing flexibly management of resources and maximizing the use of shared resources. By shifting VMs from overloaded PMs to light-loaded ones, a cloud system achieved load balancing. VMs that were operating on a light load PM were merged into another physical machine for saving energy by lowering the number of running PMs. The author in [18] proposed a cross-data center VM migration approach called EVMA which addresses energy consumption issues in a multi-data center environment. Initially, the destination data center for the VM migrations was selected first based on the bandwidth between data centers and this approach of choosing an overload host as well as VM was chosen based on the historical CPU load. Studies revealed that the algorithm performed well in lowering the data center's energy consumption while maintaining QoS [18]. Figure 4 shows a summary of the evaluation metrics, strengths and limitations of VM Migration method for cloud computing energy optimization.

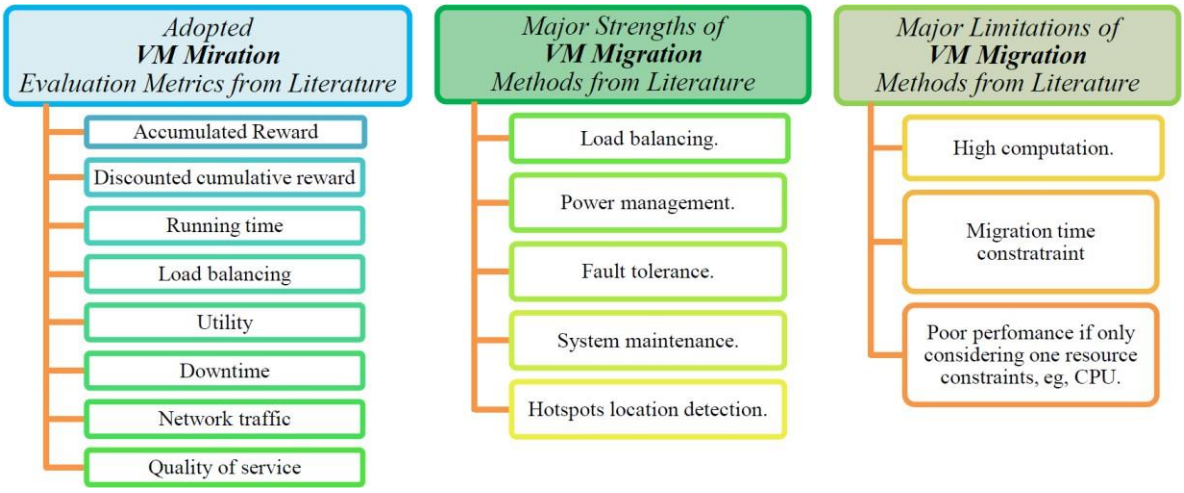


Figure 4. VM Migration methods.

(4)VM Selection Energy Efficiency Methods: Virtual Machine Selection is a crucial strategy for enhancing power proficiency and asset usage in cloud infrastructures [19]. It is concerned with assigning computational tasks to VMs. The varying utilization of central processing units (CPUs) in virtual machines is a significant factor contributing to the inconsistent CPU utilization of hosts. The virtual machine monitor (VMM) takes over when the host is identified as overloaded. The author in [12] used VM selection strategy called AUMT for minimizing energy consumption in overloaded data centers, reducing the number of migrations and enhancing the quality of service (QoS). This strategy is based on the average CPU utilization of virtual machines as well as migration times. Meanwhile, the author in [14] introduced an innovative heuristic virtual machine selection policy that considers migration times and the total number of migrations to mitigate the negative effects of migrations on the quality of service. A key aspect of this policy is examining task completion times when selecting the virtual machines for migration. To this end, a constraint was established for selecting only those virtual machines that could not finish their tasks before the mean first passage time on the current physical machine (PM), thereby avoiding unnecessary further virtual machine migrations that could improve the quality of service (QoS) [14].

The author in [11] explored a virtual machine selection algorithm based on a dynamic influence coefficient (ICVMS) that prioritizes VMs with the highest influence coefficient (IC) values, those that have the greatest impact on migration. This approach facilitates the swift and precise removal of excessive workloads from overloaded hosts. A selection of VMs was identified for migration to help return the overloaded host to a normal operating state. The author in [9] proposed a new virtual machine selection method called MRCU (maximum ratio of CPU utilization to memory utilization) to identify virtual machines for migration when a host is overloaded with CPU-intensive tasks. The MRCU approach considers both the central processing unit and memory components. Since a higher

CPU workload result in larger power consumption, this algorithm chooses a virtual machine with the largest CPU value for migration and its objective is to save energy [9]. Figure 5 shows a summary of the evaluation metrics, strengths, and limitations of VM Selection method for cloud computing energy optimization.

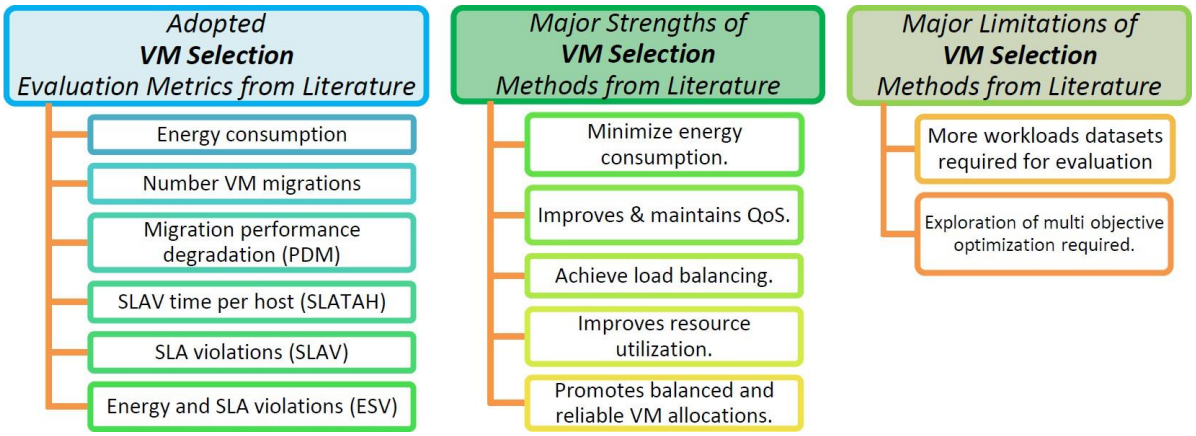


Figure 5. VM Selection methods.

2.2.2 Machine Learning Based Cloud Energy Optimization

Machine Learning (ML) is a kind of artificial intelligence that simulates human learning [20]. It enables computers to develop their predictive abilities up to the point where they can carry out tasks on their own without having to be programmed. ML-driven software applications may forecast future results based on historical training data [21]. It takes a lot of data, computing power, and Infrastructure to train an accurate ML model. ML models can be trained using a cloud ML platform, which offers computation, storage, and services needed. ML is made more affordable, flexible, and accessible due to cloud computing, which also enables faster ML algorithm development. The benefits of Machine Learning in the Cloud are as follows; According to authors in [22], many businesses have the resources to develop ML models internally using open- source frameworks like Scikit Learn, TensorFlow, or PyTorch. Even if internal teams are competent at creating algorithms, they frequently struggle to scale models for usage with real- world workloads and deploy them in production, which frequently calls for sizable computing clusters. ML workloads that come in bursts function well on the pay-per-use cloud [22]. Enterprises can quickly scale up as initiatives go into production and demand rises due to the cloud's ease of ML experimentation. The Limitations of Machine Learning in the Cloud are as follows; Although ML has transformed several industries, it frequently falls short of expectations. This can be attributed to several factors, including a lack of appropriate data, a lack of data availability, data bias, challenges with confidentiality, poorly planned goals and algorithms, improper tools and personnel, a shortage of resources, and evaluation

concerns [23]. It can be challenging to switch systems across clouds or services when running ML models on the cloud.

To accomplish this, the data must be moved in a way that preserves model performance. Little changes in the input data can often have a big impact on ML models [24]. The same security issues apply to cloud-based ML as they do to any cloud computing platform. Attackers may hack cloud-based ML systems, which are frequently exposed to public networks and susceptible to manipulation of ML results or infrastructure cost increases.

Nevertheless, there are several applications for ML. ML algorithms are essential in situations when deployment is required for development. One of the main factors influencing ML solutions' widespread acceptance is their dynamic nature. These algorithms can be used to replace some human tasks because of how flexible they are. The finest illustration of this is the fact that chatbots that use natural language processing are replacing customer service representatives. These chatbots operate by evaluating consumer inquiries and responding to them automatically [23]. ML focuses on the process through which computers learn how to carry out activities without having received specialized training [23]. Computers are used to analyze and identify trends from data. Figure 6 shows various ML types and techniques or algorithms. ML algorithms can be trained using various techniques, each of which has advantages and disadvantages of its own. The ML algorithms are generally categorized into three groups: supervised, unsupervised, and reinforcement learning.

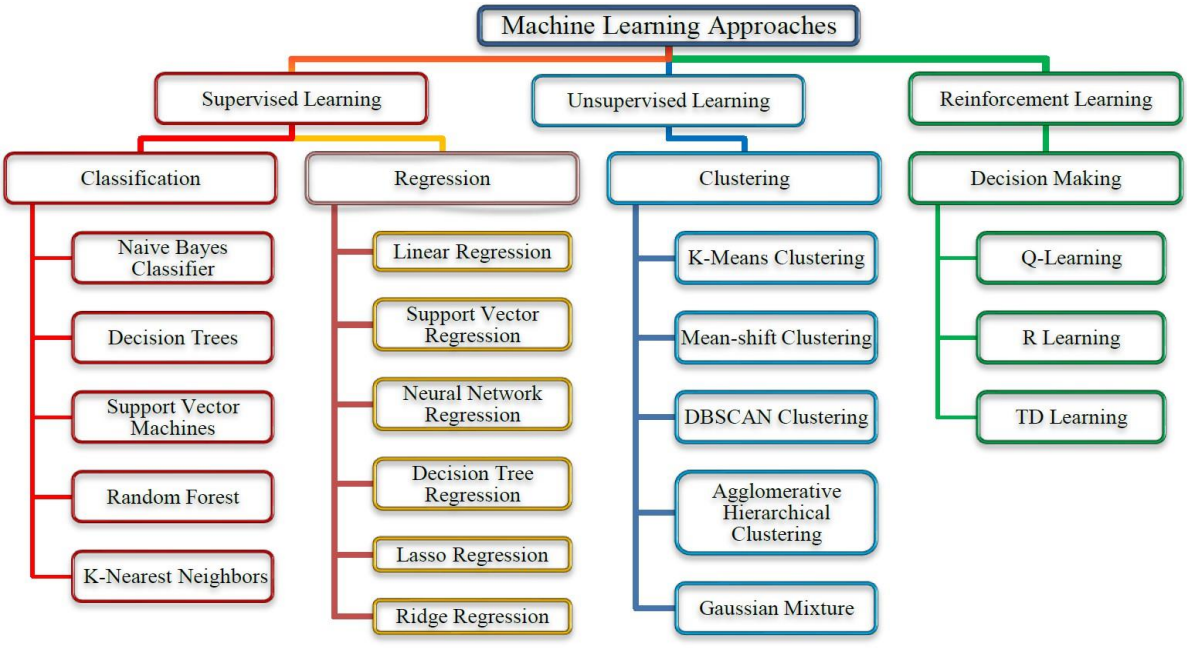


Figure 6. ML types and techniques or algorithms.

(1) Supervised Learning: The first category in the ML hierarchy is supervised learning. When given a dataset, which includes both the inputs and the matching labelled outputs, then it is modeled by supervised algorithms [23]. The training samples in the dataset have inputs that have been mapped to the matching outputs. Each training example in the mathematical model is described by an array called a feature vector, and the training data is represented by a matrix through an iterative process. A function that is created by supervised learning algorithms can be used to forecast the results of brand-new inputs. That task has been learned by an algorithm that gradually improves the precision of its predictions over time. Classification and regression are other divisions of supervised learning [23]. Supervised ML allows the collection or production of data output from previous experience and helps optimize performance criteria [25]. Also, it helps solve various types of real-world computation issues. A lot of computation time is required for supervised learning training. Unlike unsupervised learning, it cannot cluster or classify data by independently determining its features [25].

(2) Un-Supervised Learning: The second category in the ML hierarchy is unsupervised learning. Unsupervised learning techniques start with a data set that only contains the inputs and looks for patterns, such as grouping or clustering of data points [23]. In this type of learning, algorithms are trained on test data that have not been labelled. Unsupervised learning algorithms operate by finding commonalities rather than responding to feedback. It is used in the identification of anomalies, clustering, and determination of the probability density function. It deals with clustering. It is employed to reveal hidden patterns that are highly valuable to the industry and have various real-time applications, as it can detect insights beyond human capabilities [20]. In comparison to supervised learning tasks, it is less complex. However, since there are no labels or output metrics to validate its effectiveness, the value of the results is not always guaranteed. The output and classification of an unsupervised task cannot be clearly defined and are heavily reliant on the model and the machine used. Consequently, results are often less accurate [20].

Reinforcement Learning: The third category in the ML hierarchy is Reinforcement Learning (RL). According to the author in [26] an agent is trained in RL (a decision-making technique) to acquire a desired behavior within an interactive environment by responding to the events it experiences. In essence, the agent gets rewarded for every action taken in a specific state, and this reward represents the effectiveness of that action in that context. With enough practice, the agent learns which activities produce the highest cumulative reward over time. The environment in which the RL agent functions is frequently modeled using the Markov Decision Process (MDP). As a result, it is believed that state transitions and rewards are governed by the Markov property, which holds that

the future state and reward only depend on the present state and the action the agent takes in the current. Reinforcement learning can tackle extremely complex problems that traditional methods cannot solve [26]. This approach is particularly effective for achieving long-term outcomes, which are often difficult to attain. The model can correct errors made during the training process. However, in real physical systems, the curse of dimensionality significantly limits reinforcement learning, necessitating substantial amounts of data and computational resources [26]. In addition, an overload of states resulting from excessive reinforcement learning may adversely impact the results.

2.3 Literature Review Table

This section will present a review of related works which adopted or developed ML models to solve the energy efficiency optimization problem in cloud computing. This is a review of articles over the last five years. The extracted information will be presented in the form of a table, as shown in Table 1. Each column on the table will represent the features of the related work in question. The table rows represent several related works (the authors) and the years they were published. The table will present the reviewed article with its author and the years, the ML model that was proposed to solve the energy efficiency problem in cloud computing, the proposed objective, the evaluation metrics, e.g., accuracy, the limitation of the proposed model, and the datasets and tools adopted

Table 1 below shows the reviewed related works on ML models for energy efficiency in cloud computing. A detailed analysis of related works from Table 1 will be presented in the subsequent sections, showing graphically the most adopted ML models, the most adopted objectives, the most adopted ML evaluation metrics with ranges, e.g., ML model accuracy ranges, prevalent ML model limitations, and most adopted ML datasets and tools. This review will then specify and suggest the discovered trends and patterns of the reviewed features as a recommendation to current and future researchers.

Explanation on Literature Review Columns: The first column shows the name of the authors for the framework and the year the work was published. The machine learning models column shows different machine learning technologies and models that were used or adopted to develop the framework. The objective column shows the purpose of the work. The evaluation metrics and performance show the different types of metrics and their performance improvements in the form of percentages. The model limitations show the research gaps. The datasets and tools sections show different sources of data and the machine learning tools that have been adopted.

Table 1. Related Works on ML Energy Efficiency in Cloud Computing.

	Authors & Years	Machine Learning Models	ML Research Objectives	- ML Evaluation Metrics & Performance	ML Model Limitations	ML Tools
1	Liu et al., (2017) [27]	Deep Reinforcement Learning (DRL)	To adaptively allocate resources in cloud computing system.	- Power Consumption (16% reduction) - Latency - (10% reduction)	Partially solves the resource allocation problem	Dataset: Actual Google cluster-usage traces ML Tool: Keras
2	Cai et al., (2017) [20]	Page Rank and K-Means	To minimize the consumption of energy in data centres.	- - Energy consumption (25% reduction)	High Computation	Dataset: Used the customized dataset ML Tool: Hadoop
3	Zhong et al., (2017) [25]	Wavelet Support Vector Machine (WSVM)	Analyze the cycle and frequency of the input signal by substituting the kernel function of the support vector machine (SVM)	- CPU Usage (3%) - Mean absolute error (6.5% minimized) - Root mean square error (30% minimized)	Does not address workload prediction.	Dataset: Google cloud computing center dataset. ML Tool: TensorFlow
4	McGough et al., (2018) [28]	Multilayer Perceptron (MLP) and Random Forest (RF)	To minimize the energy consumption of data center.	- Accuracy (70% achieved) - Energy consumption (51.4% reduction) - Overhead (4.9% increase)	Utilizes real trace logs for complex. Scenarios that can arise within the presented platform.	Dataset: 2010 exemplar dataset. ML Tool: Scikit-Learn
5	Li et al., (2018) [29]	Deep Reinforcement Learning	Reduce power consumption by computation speed and hardware footprint optimization	- Power usage (54.1% reduced) - Average job latency (18.7% increase)	The action space in the DRL framework needs to be reduced.	Dataset: Google cluster workload traces ML Tool: TensorFlow
6.	Zhang et al., (2018) [30]	Linear and Logistic Regression	To model and analyze the multi- dimensional cloud resource allocation problem.	- Accuracy (98% achieved) - CPU utilization (100% achieved) - Storage utilization (100% achieved) - Memory utilization (100% achieved)	The resource allocation algorithm does not satisfy the strategy proof of the auction mechanism.	Dataset: Grid Workloads Archive. ML Tool: GNU Octave 4.2.1
7.	Jararweh et al., (2018) [31]	Logistic Regression Model	To predict physical machine overloading	- Energy consumption (67%), No. of VM migrations (91%), No. of host shutdowns (reduced by (86%), SLA violation (45%)	The proposed algorithm was not applied with varying workloads that represent different cloud customers.	Dataset: CloudSim 4.0 dataset. ML Tool: TensorFlow
8.	Moreno-Vozmediano et al., (2019) [32]	Support Vector Machine (SVM) Regression	To optimize the latency of the service and reduce VM over-provisioning.	- Mean Absolute Error (94% MAE reduction) - Root Mean Squared Error (11% RMSE reduction) - SLA Violation (8% reduction)	Only based on normalized polynomial kernels.	Dataset: Real web service logs from Complutense University of Madrid. ML Tool: Weka
9.	Rajalakshmi et al., (2019) [33]	Reinforcement Learning	To enhance the effectiveness of the VM consolidation algorithm in reducing energy consumption.	- SLA violation (8.5% reduction) - Energy consumption (32% reduction)	The number of hosts cannot be expanded.	Dataset: CloudSim PLANET LAB. ML Tool: TensorFlow
10.	Sui et al., (2019) [34]	K-Means Clustering Algorithm	To reduce cost and response times	- Average utilization, (100% Improvement) - VM migration number (94.5%) - Energy consumption (49.13%)	Only addressed the prediction of the distribution of resource demand.	Dataset: CloudSim 4.0 dataset. ML Tool: Eclipse 4.5.1
11.	Thein et al., (2020) [35]	Fuzzy Logic and Reinforcement Learning	To ensure the efficient management of physical resources.	- Power Usage Effectiveness (between 1.78 and 1.96) - Resource utilization (Above 50%) - SLA Violation (24% reduction)	For a very large number of infrastructure resources, the scheduling process may become slow.	Dataset: PlanetLab Virtualized Research datasets. ML Tool: MLBox
12	Tong et al., (2020) [36]	Deep Q Learning	To solve the problem of scheduling tasks in cloud computing environment.	- Makespan (5% Improvement) -Load Standard deviation (32% Improvement)	Does not cut costs when after reaching task deadlines	Dataset: CloudSim 4.0 dataset ML Tools: TensorFlow

13	Ding et al., (2020) [24]	Q Learning	Proposed a Q-Learning based task scheduling framework for energy-efficient cloud computing.	- Energy consumption (25% reduction) - Average response time (68.7% reduction) - CPU utilization rate (20% increase)	Did not consider scalability with the increase in the number of VMs.	Dataset: CloudSim 4.0 dataset. ML Tool: TensorFlow
14	Asghari et al., (2020) [37]	Cooperative Reinforcement Learning (RMFW Model)	To reduce user costs, consumption of energy and perform load balancing of resources.	- Scheduling time (61% reduction) - Makespan (20% reduction) - Cost (32% reduction) - Energy (50% reduction)	Loss of accuracy due to discretization of state space.	Dataset: CloudSim 4.0 dataset. Tool: TensorFlow
15.	Madhusudha et al., (2021) [38]	Random Forest (GA-RF Model)	To minimize power consumption while maintaining load balance yet maximizing resource utilization.	- Energy Consumption (39% reduction) - Execution Time (37% reduction)	The model was not tested with various ML and deep learning approaches	Dataset: Real workload traces from PlanetLab. ML Tool: TensorFlow
16	Yan et al., (2021) [39]	Deep Q-Learning (Dueling-DDQN)	To reduce power consumption, ensuring resource load balance,	- Average Reward (20% Improvement) - Power Consumption (30% reduction)	Do not have specific optimization goals.	Dataset: CloudSim 4.0 dataset. ML Tool: TensorFlow (TF)
17.	Wang et al., (2021) [4]	Deep Reinforcement Learning	Proposed a DRL model that utilizes QoS feature learning to enhance resource scheduling data centre.	- Energy consumption (22% improvement) - SLA rate (4.5%–26% improvement)	Affected by response time, reliability, and other parameters.	Dataset: 1998 World Cup website workload ML Tool: TensorFlow
18.	Caviglione et al., (2021) [40]	Deep Reinforcement Learning (DRL VMP)	To selects the most suitable PMs to deploy VMs requested by users	- QoE (50% improvement) - Consumed power (7% reduction) - Total reward (75% improvement)	Optimization problem (indirectly acted by means of heuristics).	Dataset: Traces taken from Netalia (www. Netalia.it). ML Tool: TensorFlow
19.	Shaw et al., (2022) [3]	Reinforcement Learning	To solve the VM consolidation problem for improved cloud resource management.	- Energy Consumption (47% reduction) - Energy efficiency (Improved by 25%) - Service violations (Reduced by 63%)	A limitation of the tradeoff between exploration and exploitation.	Dataset: Real workload data from PlanetLab. ML Tool: TensorFlow
20.	Jayanetti et al., (2022) [26]	Deep Reinforcement Learning	Energy-efficient resource scheduling in edge-cloud environment.	- Energy consumption (56% improvement) - Execution time (46% improvement)	Only operates in a centralized manner.	Dataset: CloudSim 4.0 dataset. ML Tool: Keras
21.	Chen et al., (2023) [41]	Deep Reinforcement Learning (DQTS)	To dynamically and collaboratively schedule high-dimensional cloud objectives.	- Makespan 2484 (16.6% improvement) - Fairness (Improve by 5.3%.)	Performance degradation in larger-scale workflow scheduling	Dataset: CyberShake, Epigenomics, ML Tools: PyTorch
22.	Cui et al., (2023) [42]	Deep Reinforcement Learning (DQN)	Optimizing computing, caching, and communication resources	- Network Delay (44% reduction) - Reward per episode (39% improvement)	A tradeoff between energy consumption and network not considered	Dataset: Not Specified ML Tools: Not Specified
23.	Sen et al, (2019) [43]	Reinforcement Learning based Task Assignment approach (RILTA)	Ensures the timeliness guaranteed execution of intelligent cognitive assistants' tasks with high energy efficiency.	- Guarantee ratio (22%) - Energy consumption -Task allocation time(25%) - Task running time(51%) - Percentage deadline (63%)	Cannot always find the optimal solution, has high variance error bars.	Dataset: Generated via uniform distribution from iFogSimTool: iFogSim
24.	Kumar et al., (2022) [44]	ML frameworks among them: Supervised learning, neural networks, support vector regression, etc.	A Review: Reviews the ML based cloud improvement through dynamic load allocation, task scheduling, energy optimization, live migration, etc.	The review table presented compares reviewed literature based on the year, authors, learning model, benefits, limitations, platforms adopted and the performance.	No dataset comparison. No analysis of performance,	Dataset: The review does not compare any datasets. ML Tool: several tools compared.

25.	Demirci, (2016) [45]	Presenting several ML schemes: Supervised, Unsupervised, Reinforcement and Hybrid	Review: Provides a comparative classification of ML models in the cloud for energy reduction.	The review table presented compares reviewed literature based on the year, authors, learning model and objectives.	Does not compare based on metrics, performance, tools limitation, datasets. Outdated articles up to 2016.	Dataset: No comparison. ML Tools: No comparison.
26.	Khan et al., (2022) [46]	Presents several ML centric resource management in cloud computing	A review: The review compares several ML approaches based on experiments configuration, datasets, performance, and limitations.	Energy Consumption (Best reviewed article has 38% energy reduction) Several other performance metrics of articles were presented; accuracy, latency, etc.	No analysis of most adopted datasets, tools, metrics, performance, etc. Outdated with articles up to 2021. Missing values on performance for some article	Dataset: Several datasets were compared with. ML tools: Several tools were compared with.
27.	Soni et al., (2022) [47]	Presents several ML schemes generally in cloud computing	A review: Compares machine learning techniques in emerging cloud computing integrated paradigms	Several performance metrics; power consumption, latency, delay, accuracy, cost, utilization, etc.	No comparison on; performance but only accuracy performance presented. No comparison based on ML tools.	Dataset: MNIST, LSTM, Cifar10, etc. ML tools: No comparison.
28.	Nomsa Puso **PROPOSED** (2025)	Proposed An Intelligent Energy Efficiency Framework in Cloud Computing Using Deep Reinforcement Learning	Maximize Job Success Rate, and VM Utilization Minimize Energy Cost, Job Response time.	Job Success Rate- 98.4%, VM Utilization – 90.7%, Energy Cost 38%, Response time, 66%.	More metrics can be added, Security not considered	Poison Process Generated, Various Machine Learning tools

2.4 LITERATURE REVIEW ANALYSIS

This section presents a detailed analysis of the literature review presented previously in Table 1. This section will provide a detailed analysis of the most adopted ML models, the prevalent objectives of current approaches, the most adopted evaluation metrics used in current approaches with ranges, e.g., the minimum and maximum accuracy that has ever been achieved in current works, the most prevalent limitations of current related works, the most adopted ML datasets and tools.

2.4.1 The Most Adopted ML Models from Literature

Figure 7 shows an analysis of the most adopted ML models from the literature for cloud computing energy optimization. The analysis shows that Deep Reinforcement Learning (DRL), Random Forest (RF), K-means, Support Vector Machine (SVM) Regression, and Linear and Logistic Regression were among the ML models adopted. Figure 7 shows that 60% of related works adopted the DRL model to solve energy efficiency problems on the cloud. The reason why DRL is the most adopted model is because, compared to other models, the DRL algorithm [43] – [45]:

- Can handle complex and high-dimensional state and action spaces.
- Learn complex policies that are difficult to specify manually by learning to maximize a reward signal over a long-time horizon.

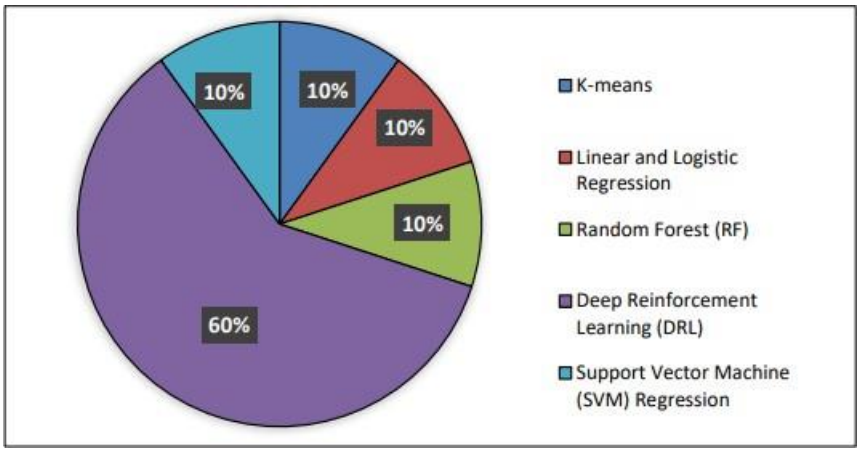


Figure 7. Most adopted ML models from related works.

2.4.2 The Most PREVALENT Objectives from Current Approaches

Figure 8 shows an analysis of research articles from literature with the most prevalent objectives of using ML for cloud energy minimization. It can be observed from Figure 8 that the most dominant research objective is power or energy consumption reduction at 28%. Other objectives in addition to energy reduction are load balancing at 14%, task/resource scheduling at 10%, VM consolidation at 10%, response time at 7%, resource allocation at 7%, reduce costs at 7%, SLA and resource utilization at 4%, low overhead at 4% and many others. The main reasons for energy consumption reduction as the dominant research objective in literature are because:

- By minimizing energy usage, data centers can reduce their operating expenses and pass on some of the savings to their customers.
- By reducing energy consumption, data centers can operate with fewer resources and plan for future expansion more efficiently.
- By reducing the load on servers and other hardware components, data centers can extend their lifespan and reduce the risk of hardware failures.

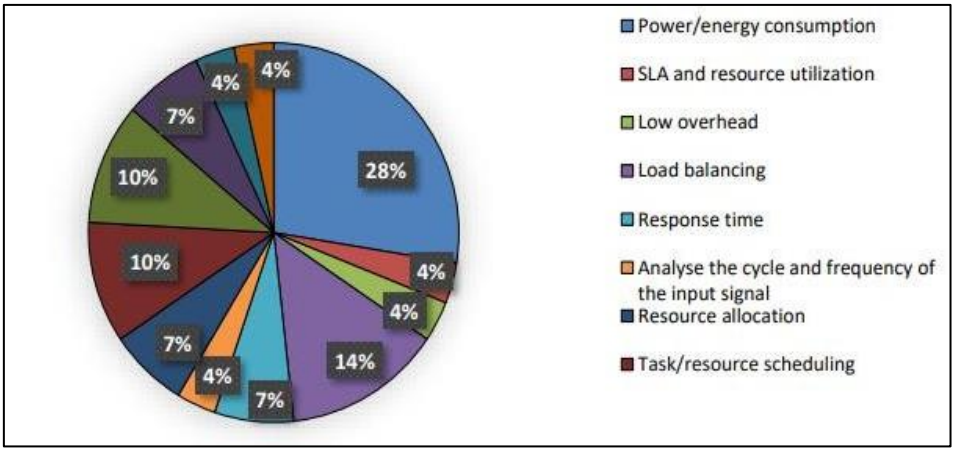


Figure 8. Most Adopted research objectives from related works.

2.4.3 *The Most Adopted ML Datasets from Literature*

Figure 9 shows a summary analysis of the most adopted ML datasets from the literature, with a larger percentage of 40% being dominated by the CloudSim datasets simulation toolkit, whilst other datasets were taken from different sources and databases. The main reasons for the CloudSim dataset being dominant in literature is because:

- CloudSim is a widely used simulation tool to model and simulate cloud computing environments and is used to simulate a variety of cloud computing scenarios. The dataset includes parameters such as the number of hosts, the number of virtual machines, the amount of memory and storage, and the network bandwidth.
- CloudSim dataset is based on real-world cloud computing environments, which makes the simulations more realistic and accurate.
- CloudSim dataset is adaptable and can be customized to meet specific research requirements. Researchers can modify the input parameters to simulate different cloud computing scenarios and evaluate the performance of different cloud computing architectures and algorithms.

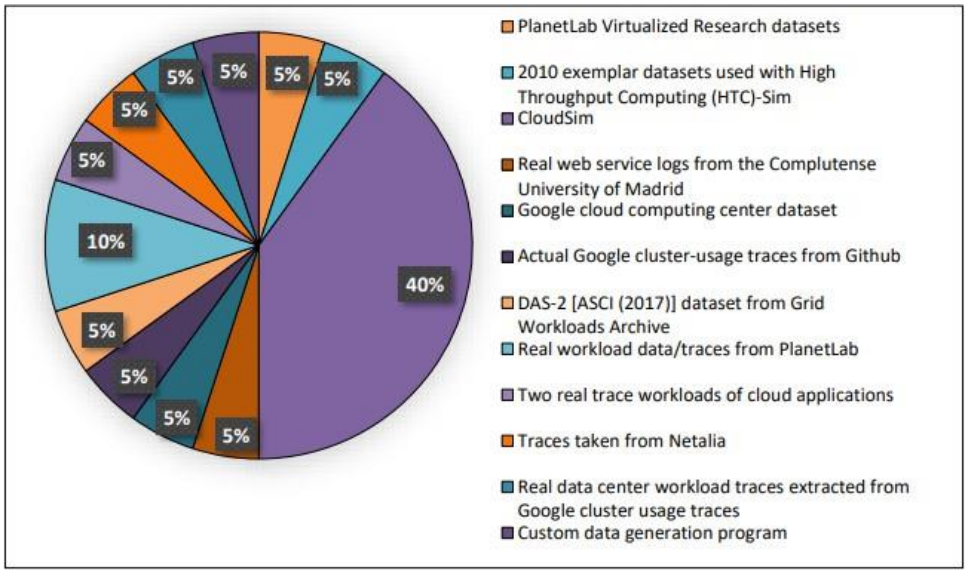


Figure 9. Most adopted ML datasets from related works.

2.4.4 *The Most Adopted ML Model Evaluation Metrics from Literature.*

Figure 10 shows a summary of the most adopted ML evaluation metrics for cloud energy minimization from literature, which is comprised of energy consumption at 13%, accuracy at 9%, power consumption at 4%, execution time at 4%, resource utilization at 4%, average reward at 4%, Makespan at 4%, power usage at 4%, average job latency at 4% and many others. The most adopted metric is energy consumption (28%) followed by accuracy (9%), and the explanation for this is that;

for energy consumption, the reasons are similar to why energy consumption is the dominant objective, and this was presented in the previous sections. The reason for accuracy being the next dominant metric is that:

- The accuracy of a model is a primary measure of its performance. High accuracy means that the model is making accurate predictions, while low accuracy indicates that the model is making errors.
- Accuracy is a convenient metric for comparing the performance of different models. It allows us to determine which model is better suited for a particular task based on their accuracy scores.
- Accuracy is also used in the training process for optimizing the model's parameters. By monitoring the accuracy of the model during training, we can adjust the parameters to enhance the model's performance.

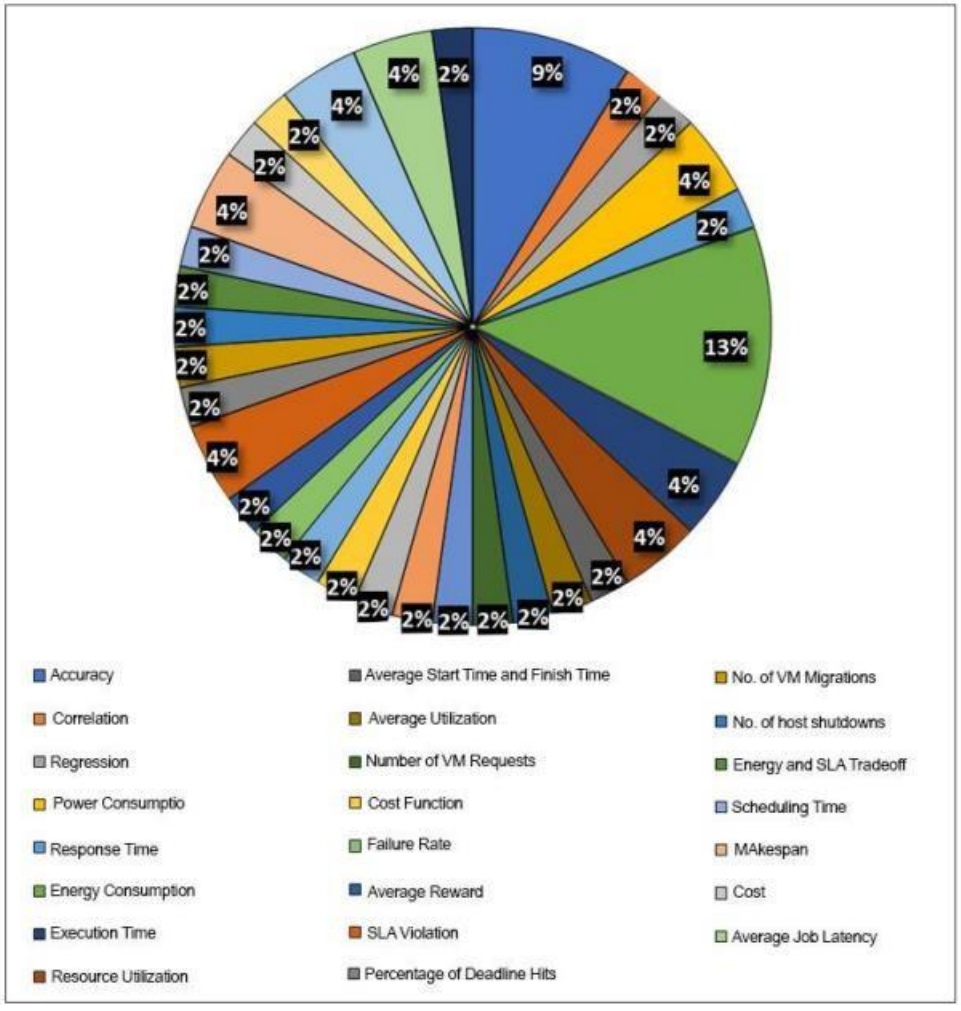


Figure 10. Most adopted ML metrics from literature.

2.4.5 The Most Prevalent Limitations with the Current Approaches

Figure 11 shows a percentage summary of the most prevalent limitations of current ML-based approaches for cloud computing energy reduction. A larger portion of current literature approaches are being dominated by poor performance or poor model accuracy or loss of accuracy (15%), model optimization problems (15%) followed by resource allocation problems at (10%) and other limitations like high computation (5%), centralized approaches (5%) which are a single point of failure, etc. The poor performance/ loss of accuracy is a major limitation because.

- When a model loses accuracy, it is less effective at making predictions. This can result in reduced performance and lower quality results.

- A model with reduced accuracy is more likely to misclassify data. In some applications, misclassification can have serious consequences.
- Loss of accuracy can also be a symptom of overfitting or underfitting. Overfitting happens when the model is overly complex and closely aligns with the training data, resulting to insignificant performance on new data. Underfitting happens when the model is overly simple and fails to identify the underlying patterns in the data, resulting in low accuracy. When a model loses accuracy, its predictions become less reliable. This can make it difficult to trust the model's output and can lead to poor decision- making.

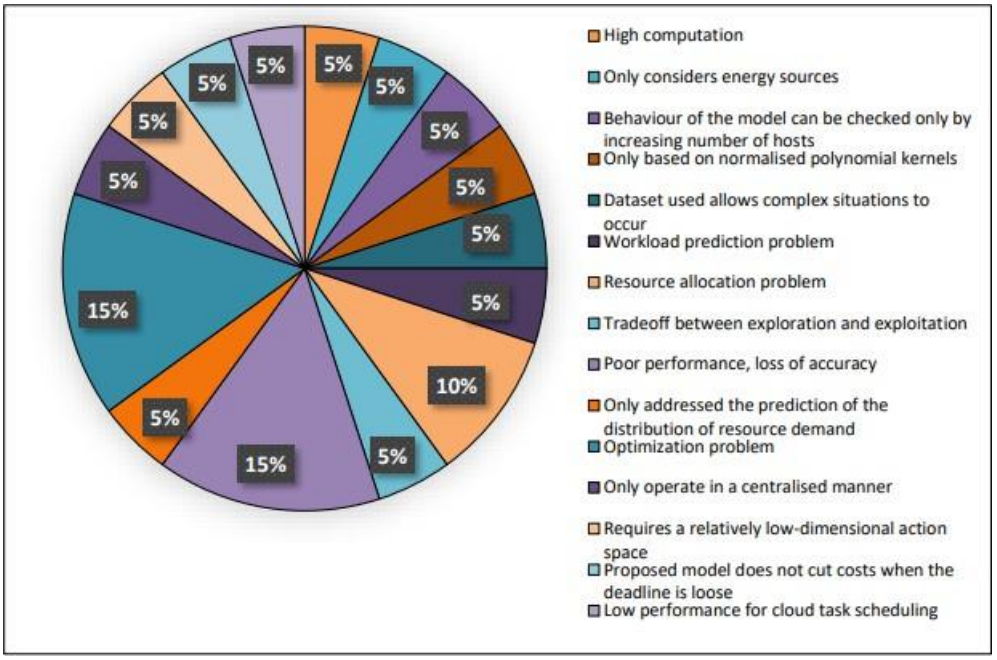


Figure 11. Most prevalent limitations/Gaps in current related works.

The prevalence of optimization problems at 15% of the literature as shown in Figure 11 are also major limitations in ML models for cloud computing energy minimization due to the following.

- Time Complexity: Optimization problems can be very time-consuming to solve, especially for

large datasets and complex models. This can make it difficult to train models in a reasonable amount of time.

- Resource Requirements: Optimization problems require significant computational resources, such as memory and processing power. This can limit the scalability of machine-learning algorithms and make them difficult to implement on low-power devices.
- Local Minima: Optimization problems can have multiple local minima, which can make it difficult to find the global minimum. This can result in suboptimal solutions that do not fully capture the underlying patterns in the data.
- Sensitivity to Initialization: Optimization problems can be sensitive to the initialization of the model's parameters. This can lead to different solutions for different initializations, which can make it difficult to compare the performance of different models.

In addition, the resource allocation problem, which is prevalent in 10% of current literature, is also a major limitation in ML models for cloud computing energy minimization because it can greatly affect the performance and costs of cloud services. Resource allocation involves the distribution of computing resources, such as processing power, memory, and storage, for various applications as well as users in a cloud environment. If resources are not allocated efficiently, it can result in wasted resources and increased costs for cloud providers and their customers. If resources are not allocated optimally, it can result in degraded performance, which can lead to user dissatisfaction and loss of business. Resource allocation is particularly challenging in cloud environments with dynamic workloads. In such environments, the demand for resources can vary significantly over time, making it difficult to allocate resources effectively.

2.4.6 Most Adopted ML Tools from Literature

Figure 12 shows a summary of the most adopted ML tools for cloud energy minimization from the literature. As shown in Figure 12, 52% of the current literature has adopted the TensorFlow ML tool. TensorFlow is among the most widely used open-source machine learning frameworks available today, offering several advantages over other reviewed ML tools hence its prevalent adoption in literature because it offers high flexibility, scalability, and seamless integration with other software libraries. It supports various tasks, is scalable, and integrates with popular programming languages. Its active developer community provides resources, and it supports production environments for easy deployment and scaling.

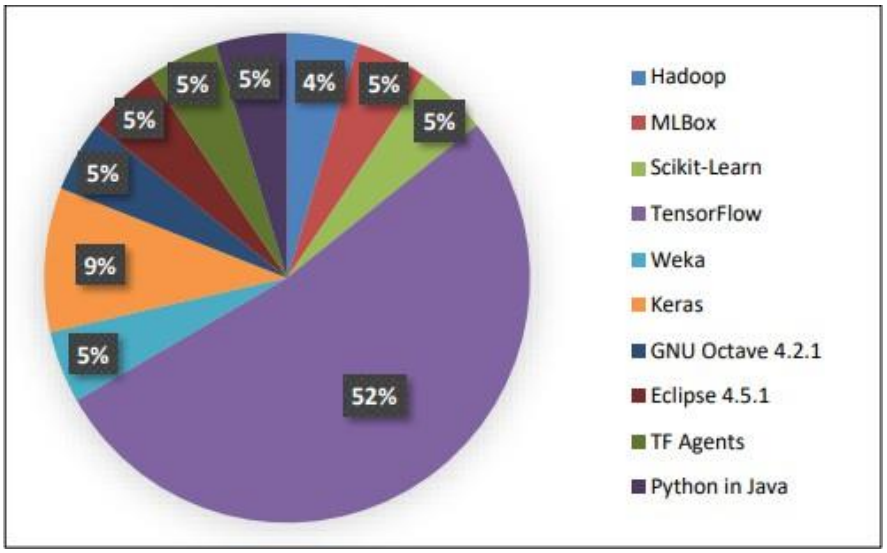


Figure 12. Most adopted ML tools from literature.

Even though DRL has been the most adopted in literature, from Table 1, Jararweh et al [31] achieved an energy reduction performance of 67% using Logistic Regression Model which is the best and maximum achieved from the reviewed literature showing that adoption and performance are different. The DRL algorithms balance exploration (trying out new tactics) and exploitation (leveraging current techniques), which is why many authors have adopted them. DRL can capture highly complex and dynamic cloud settings with a huge number of variables and interactions that logistic regression and other models may struggle with. DRL can adapt to the challenges of cloud environments, which include resource allocation, workload scheduling, and server provisioning. This is critical in the cloud setting, where the ideal trade-off between energy efficiency and performance must be found. DRL might assist with near-real-time decision-making. DRL can learn from historical data and previous experiences, which is useful for recognizing workload patterns and trends and making more informed decisions.

2.4.7 Related Works Performance

The performance metrics of related works and their values are clearly presented in Table 1 under the "ML Evaluation Metrics and Performance" column. This section will present the literature performance results and give a critical performance analysis to deduce any trends or patterns that emerge. The performance of related works was presented in the form of a percentage improvement for every metric in Table 1, which can either be a percentage increase (if the metric is maximized) or a percentage decrease (if a metric is minimized). The values of the performance percentages were calculated by critically examining the findings found in the article's results section on tables, graphs, figures, or clearly stated in the abstract, discussion, and conclusion sections. In any article

reviewed, the proposed framework was compared with the baseline framework within that article for any metric, and the percentage improvement was then calculated using the equation below.

$$Perfomance = \frac{ProposedFrameworkMetricM_{value} - BaselineFrameworkMetricM_{value}}{BaselineFrameworkMetricM_{value}}$$

Performance refers to measurable outcomes in the form of a percentage to show either improvement or degradation in performance. The M^{value} refers to the exact value of metric M from the results. If the *performance* value is a positive number and the objective is to maximize *M,* then there is good performance. If the *performance* value is a negative number and the objective is to maximize *M,* then there is poor performance. If the *performance* value is a negative number and the objective is to minimize *M*, then there is good performance. If the *performance*
value is a positive number, and the objective is to minimize *M*, then there is poor performance. If *performance* in *(1)* gives a value of zero, then both the baseline and the proposed frameworks have the same performance.

Energy Consumption: Figure 13 shows the performance analysis of reviewed frameworks that considered energy consumption as a metric, and the results show that 67% has been the maximum energy reduction performance in literature from [31], and the lowest performance was from [35] at 2%. The Figure shows that the average energy consumption performance is 38%.

Accuracy: Figure 14 shows the performance analysis of reviewed frameworks that considered accuracy as a metric, and the results show that 98% has been the maximum achieved accuracy performance in literature from [30], and the lowest performance was from [39] at 20% and the average accuracy achieved from related works is 55%.

Latency: Figure 15 shows the performance analysis of reviewed frameworks that considered latency/delay/response time as metrics, and the results show that 68.7% has been the maximum achieved performance by [24], and the lowest performance was from [27] at 10%. The average latency performance achieved form literature is 41%.

Service Level Agreement: Figure 16 shows the performance analysis of reviewed frameworks that considered SLA as a metric, and the results show that 63% has been the maximum achieved performance by [3], and the lowest performance is 8% by [32]. The average SLA achieved based is 45%.

Resource Utilization: In Figure 17, the maximum resource utilization was achieved by Sui [34] and Zhang [30] at 100% performance. Nevertheless, the lease performance from reviewed articles was attained by Zhong in [25] at 2%.

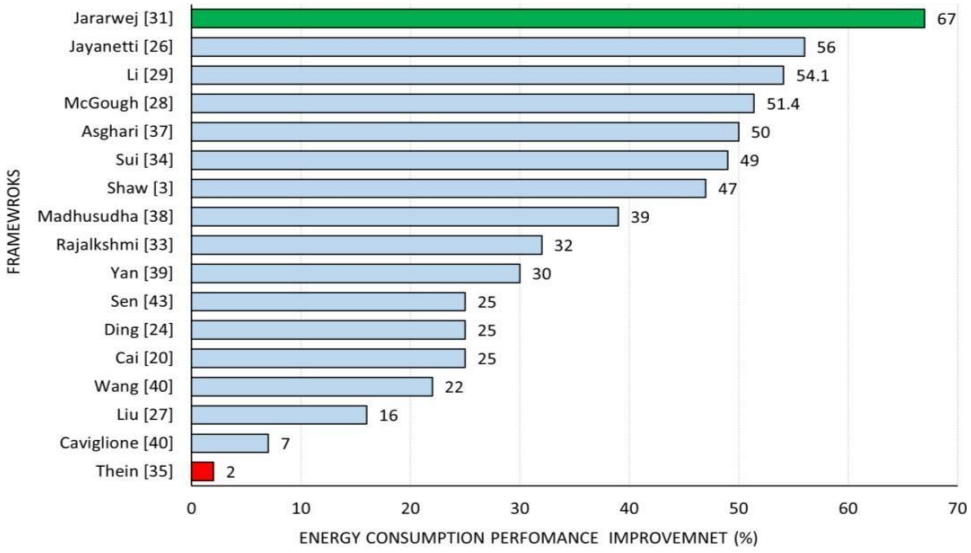


Figure 13. Performance analysis of related works based on energy consumption metric.

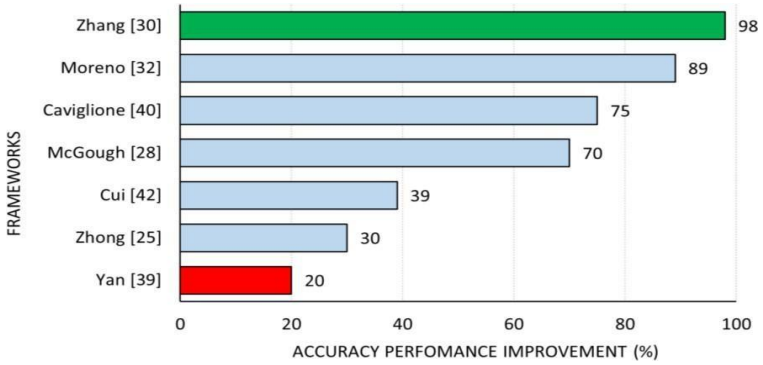


Figure 14. Performance analysis of related works based on accuracy metric.

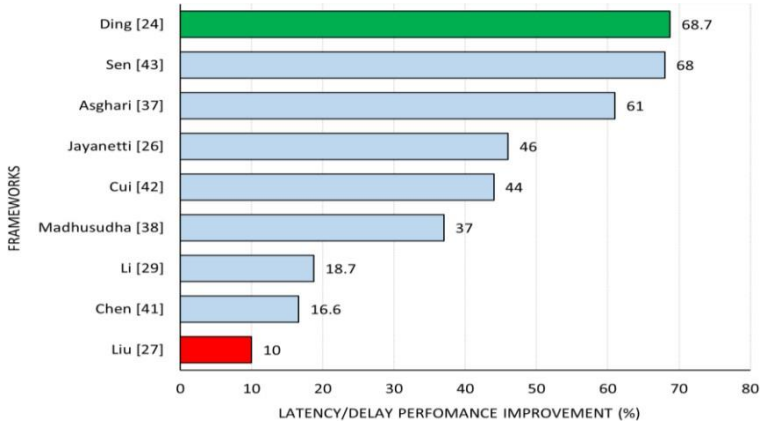


Figure 15. Performance analysis of related works based on latency metric.

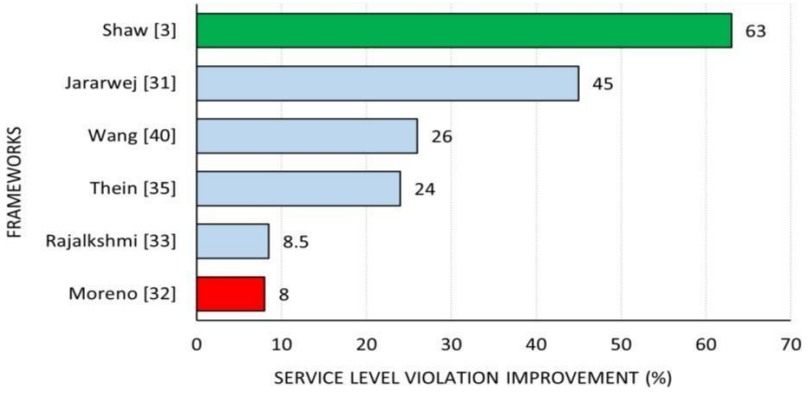


Figure 16. Performance analysis of related works based on Service Level Violation.

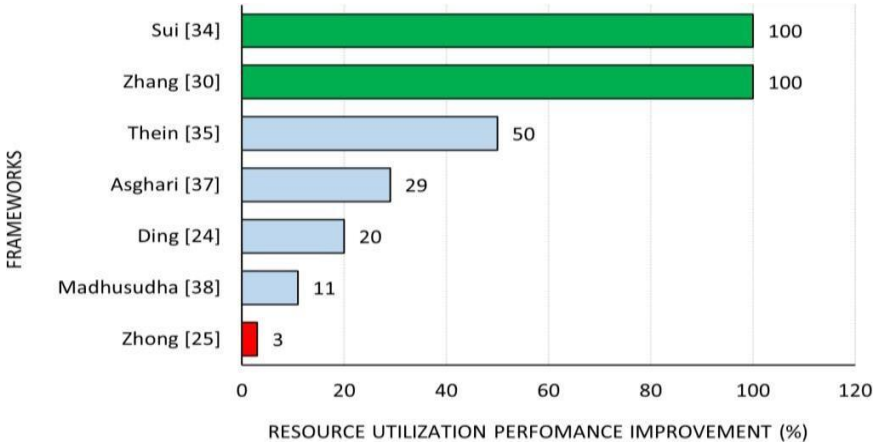


Figure 17. Performance analysis of related works based on Resource Utilization

2.5 CHAPTER CONCLUSION

The findings from the literature review in this chapter indicated that 67% has been the maximum energy reduction performance from literature with an average performance at 38%. These techniques leverage various algorithms, such as Deep Reinforcement Learning (DRL), Reinforcement Learning (RL), Random Forest (RF), and Support Vector Machines (SVM), to optimize resource utilization in cloud environments. The review analysis revealed that 60% of related works adopted the DRL model to solve energy efficiency problems on the cloud because DRL can adapt to changing environments and leverage large amounts of data to improve performance. The review analysis has also shown that the most dominant research objective is power or energy consumption optimization in cloud computing because high energy consumption leads to high costs and degraded performance. In addition, 40% of the literature has used the CloudSim datasets for their ML models because they are flexible, open source, and can be customized to suit specific research needs. The review also concluded that the most adopted or dominant evaluation metrics are energy consumption (28%), followed by accuracy (9%). A larger

portion of current literature approaches are dominated by the limitations of poor performance, poor model accuracy, or loss of accuracy (15%), model optimization problems (15%), resource allocation problems (10%), and other limitations like high computation (5%), centralized approaches (5%) that are a single point of failure. Furthermore, 52% of the current literature has adopted the TensorFlow ML tool for cloud computing ML- based optimization models due to its flexibility, scalability, large and active community support, etc. This review therefore concludes and recommends DRL as the best ML model to optimize energy in cloud computing, CloudSim as the best tool to generate the dataset, and TensorFlow as the best ML platform for building the DRL model. The future direction of energy optimization in cloud computing through machine learning algorithms is likely to concentrate on several key areas like exploring combinations of multiple optimization techniques to overcome the limitations of individual techniques, dynamically assigning resources in real time, and incorporating supplementary data sources such as weather data, occupancy patterns, and workload characteristics. Future research may aim on creating algorithms which optimize multiple objectives simultaneously, such as energy consumption, cost, and performance. Machine learning-based energy optimization may be integrated with other technologies, such as blockchain, edge computing, and IoT, to enhance the efficiency and scalability of cloud systems. As renewable energy sources become more prevalent, machine learning-based energy optimization algorithms may be developed to take advantage of these energy sources, further reducing the carbon footprint of cloud computing.

CHAPTER 3: PROPOSED FRAMEWORK

3.1 CHAPTER INTRODUCTION

This section presents the development of the proposed DRL based job scheduling framework. The section will address Research Objective 2: To develop a real-time job scheduling model based in DRL to minimize energy consumption in cloud computing while maintaining QoS. The Research Objective 2 will be archived by answering the following research question.

1. *How will the proposed cloud job scheduling model be mathematically formulated?*
2. *What and how will the real time job/workload characteristics be generated?*
3. *How will the proposed scheduling model minimize cloud energy consumption?*
4. *How is the QoS measured, and what determines whether it is maintained?*
5. *How will the DRL phases map with the cloud job scheduling model in terms of the decision phase, state phase, action space, reward function, and training phase?*
6. *How will the proposed framework deal with the dynamic and unpredictable nature of the cloud?*

3.2 RESEARCH JUSTIFICATIONS

This section will present the research justification for adopting the DRL model, the chosen evaluation metrics, the chosen real time dataset generation, the chosen ML tools used.

Justification for choosing the DRL model: From the literature, the DRL model has been the most adopted as it offers the potential for automated learning, decision-making, and adaptation in complex and dynamic environments. Its ability to handle high-dimensional data, learn from raw inputs, and optimize long-term cumulative rewards makes it a powerful approach for solving challenging problems in various domains.

Justification for choosing cloud energy consumption minimization objective: Having energy consumption reduction as the dominant research objective serves as a larger goal of achieving environmental sustainability, cost savings, resource conservation, scalability, and extended battery life. It aligns with regulatory requirements and reflects social responsibility, addressing both ecological and economic concerns in various fields.

Justification for the evaluation metrics: The proposed framework will be evaluated using the metrics such as energy consumption, success rate, response time, QoS, resource utilization and cumulative reward as they are mostly adopted from literature. Energy efficiency is crucial

in cloud computing and machine learning, prioritizing accuracy as the dominant metric ensures that machine learning models deliver reliable, unbiased, and high-quality predictions. By focusing on accuracy, researchers can address ethical considerations, comply with regulations, enhance user experience, and drive advancements in the field of machine learning.

Justification for choosing the real-time poison process generated dataset/workload: The dominance of CloudSim dataset from literature can be one option but this research will generate data using the poison process. Poisson process datasets offer dynamic, realistic, and adaptive scenarios, outperforming CloudSim's static, predefined simulations in complex cloud environments. A poison process can model random, unpredictable events that occur in cloud environments, such as spikes in user demand or sudden resource failures.

Justification for choosing TensorFlow: TensorFlow's broad community and ecosystem, scalability, flexibility, support for deep learning, model deployment capabilities, visualization tools, integration options, platform compatibility, and extensive documentation contribute to its popularity and set it apart from other reviewed machine learning tools.

Predicting workloads and then allocating appropriate resources to the accountable incoming jobs is another common strategy for lowering cloud computing energy usage.

3.3 THE PROPOSED FRAMEWORK ARCHITECTURE

Figure 18 shows the proposed job scheduling framework. As more apps are installed in the cloud, energy consumption also increases. The framework seeks to maximize energy efficiency while meeting job QoS requirements by consolidating workloads onto fewer VMs, allowing idle VMs to be turned off or put into low-power states. VM processing power is dynamically scaled based on workload, activating only the most energy-efficient servers and VMs for job requests, while less efficient ones remain offline. Effective load balancing spreads the computational load evenly across VMs, preventing overburdening and underutilization, and prioritizes the utilization of energy-efficient hardware and VMs for reducing the overall energy footprint. This research addresses real-time job scheduling within a dynamic public cloud environment. End users rent VMs from IaaS providers and submit job requests to the cloud broker. The scheduler assigns each job to a selected VM based on available resources, VM energy consumption, and user needs such as QoS and deadlines. The resource manager tracks resource usage to prevent over- or under-utilization. A job and VM monitor collects data on job execution and available computing resources. The scheduler uses this information to continuously schedule real-time

jobs efficiently. Table 2 shows the notation used for the mathematical formulation of the proposed DRL based job scheduling approach adopted from [48][49].

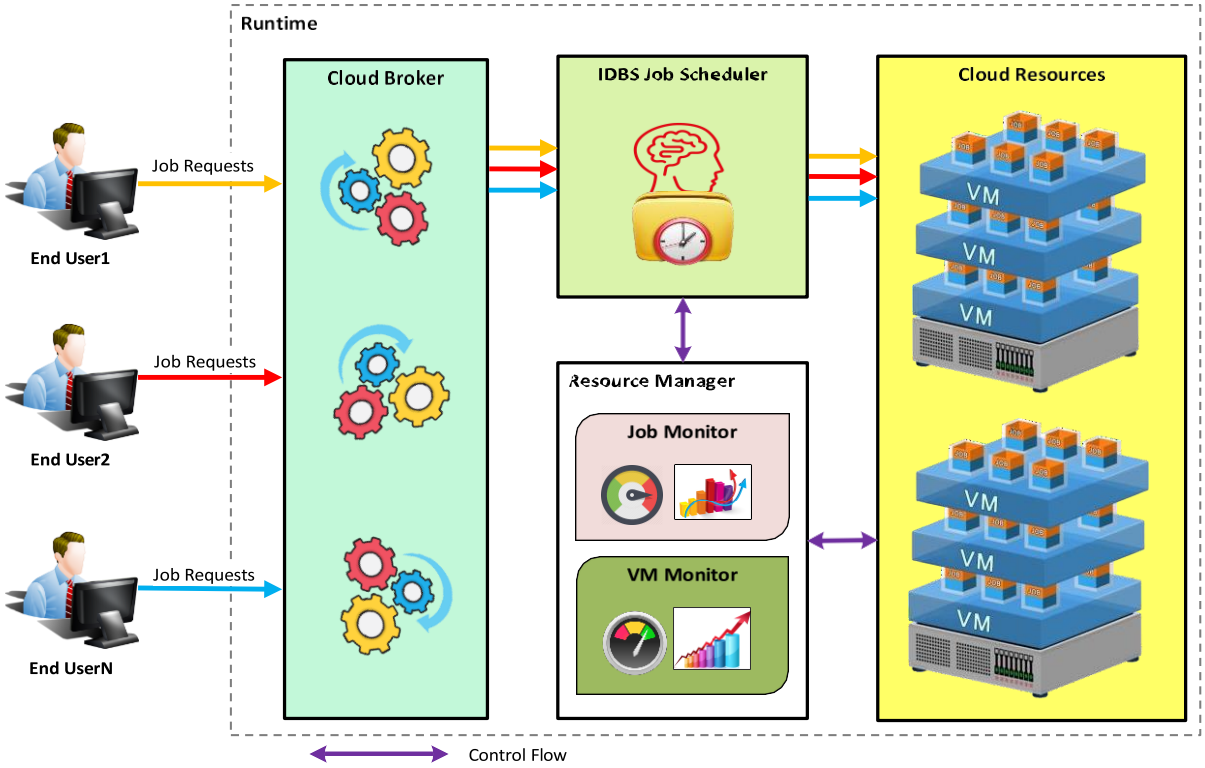


Figure 18. Proposed Scheduling Framework.

3.4 PROPOSED FRAMEWORK MATHEMATICAL FORMULATIONS

A mathematical model is a representation of a real-world system using mathematical concepts, equations, and structures to describe its behavior, relationships, and processes. It helps in predicting, analyzing, and understanding the system by simulating different conditions and outcomes. This section will describe the mathematical formulation in equation form for the non-DRL based blocks of the proposed framework. This includes how the user jobs are generated, mathematical formulation of the cloud resources describing what a VM really is in this research context, the mathematical equation formulation of the job scheduling mechanism of user jobs to virtual machines, the job execution and response time equations, QoS equations, and the energy consumption model/equations. Table 2 shows the notation used for the mathematical formulation of the proposed DRL based job scheduling approach. Mathematical models simplify analysis, optimize systems, and are universally applicable.

Table 2. Notations for the proposed scheduling model.

Notation	Meaning
Job_i	The *i*-th job
Job_i^{id}	The id of the Job_i
Job_i^{arrT}	The arrival time of Job_i
Job_i^{comp}	The required compute units for Job_i as number of instructions
Job_i^{qos}	The QoS requirement for Job_i in seconds, the job deadline.
Job_i^{type}	Job type of Job_i (I/O as 0 or compute intensive job as 1)
VM_j	The *j*-th instance or VM
VM_j^{id}	The id of the VM_j
VM_j^{comp}	Computational capability of VM_j measured in instructions per second
VM_j^{Type}	The type of VM_j, (I/O intensive as 0 or compute intensive as 1)
VM_j^{srtEng}	The fixed start-up energy cost of VM_j
VM_j^{exeEng}	The execution energy cost of VM_j per time unit
$VM_j^{EngCost}$	Energy cost of VM_j
T_i^{res}	The response time for Job_i
T_i^{exe}	The execution time for Job_i

3.4.1 User Jobs and their Characteristics

This section will describe all the components of a user Job. For clarity, workload or jobs are used interchangeably and are the tasks that users submit for processing in the cloud VMs [50]. The proposed cloud model comprises of multiple servers each with multiple VMs. Several jobs can be executed in one VM [48]. The jobs that users submit are independent of one another. It is considered that there is no prior knowledge of the incoming jobs, and a job arrives in real time, without a specific arrival pattern. For every job that arrives, the QoS time- requirement is defined to be met once the job is scheduled for execution. Each job Job_i from a user is formulated as follows:

$$Job_i = \left(Job_i^{id}, Job_i^{arrT}, Job_i^{comp}, Job_i^{QoS}, Job_i^{type}\right) \quad (1)$$

where the Job^{id} is the job id, Job^{arrT} is the job arrival time, Job^{comp} is the job computing requirements, Job^{QoS} is the job QoS requirements, and Job^{type} is the job type. There are two types of jobs: I/O-intensive job or computing intensive job.

3.4.2 Cloud Resources

The cloud resources in this research are the VMs or instances deployed in the cloud with certain characteristics. This section will describe all the components of a VM. It will be wiser to allocate a computing-intensive job to a virtual machine instance with a high-speed central processing unit and a minor job to one with a lower-end CPU [49]. Several public clouds, like Microsoft Azure [51] and Amazon EC2 [52], offer several substitute choices for virtual machines. For example, AWS EC2 can supply the instances with fast, medium, or slow CPUs, depending on the desired level of computational power [53]. This research will use two types of VMs available to users, those that are memory intensive and those that are input/output (I/O) intensive. The computation power of the instances is defined by the number of Virtual CPUs (VCPUs). Each virtual machine's attributes will be defined in the proposed model as follows:

$$VM_j = (VM^{id}, VM^{com}, VM^{type}, VM^{srtEng}, VM^{exeEng}) \quad (2)$$

where VM^{id} is the id of the VM_j to uniquely identify the VM, VM^{com} is the computational capability of the VM measured in instructions per second, VM^{type} is the type of VM which is either high I/O or high computing, VM^{srtEng} is VM 's fixed start-up energy and VM^{exeEng} is VM_j's executing energy associated with the running time.

3.4.3 Job execution mechanism

The role of the proposed scheduler is to assign the incoming real-time jobs to one of the available instances. In this scenario, we assume that as soon as a real-time job arrives, it is promptly directed by the scheduler to the appropriate instance. Job scheduling follows a first-come, first-served (FCFS) approach [54], meaning that priority setting for individual jobs is not considered [49]. Once assigned, the job is immediately processed by a VM. Contrary to the scheduler creating a queue to contain user-submitted jobs, in the proposed work, there is a pool of VMs, allowing for immediate and optimal assignment of a job to a virtual machine instance with available resources [48]. If all VMs are fully utilized, a new instance is instantiated, and the job is immediately assigned to that instance with no waiting time. Pre-emption of jobs have not been considered in this work. A VM can process several jobs depending on its capacity or type [48][49]. If all VMs are fully utilized, a new instance is created, and the job is assigned immediately, eliminating waiting time. Pre-emption is not considered. A VM can process multiple jobs based on its capacity. Jobs are not assigned to busy VMs to avoid delays; instead, they are assigned to new instances. The response time for job is defined as follows:

$$T_i^{res} = T_i^{exe} \tag{3}$$

Where is T_i^{res} is Job_i's response time and T_i^{exe} is Job_i's execution time defined as:

$$T_i^{exe} = \frac{Job_i^{comp}\left(Job_i^{type} \odot VM_j^{type} + 1\right)}{2VM_j^{comp}} \tag{2}$$

where Job^{comp} is the number of instructions, Job^{type} is the job type, VM^{type} is the type of a VM. In this proposed model, both a job and a VM can be classified as either I/O intensive or computing intensive [48][49]. There will be a penalty if a job is allocated to a virtual machine, but they are of different types. To illustrate the impact on the execution time when a job of one type is assigned to a VM of another type, then the execution time is defined using the exclusive NOR $\odot$ which indicates that if the types of the job and VM do not match, the processing speed will be decreased (the type of a job or VM is set to 0 or 1) and the computational capacity of the VM is V^{com}.

3.4.4 QoS Success Condition

It is essential to ensure that jobs are executed in a manner that meets user QoS requirements. In this research, the QoS requirements of a job is the job execution deadline expected by the user which is submitted with the job. If the job results are returned within the deadline, then the job execution will be successful, else the job execution failed. Therefore, successful job execution occurs when it is completed within the stipulated QoS requirement. Additionally, to enhance the overall QoS for job executions, our scheduling approach aims to maximize the success rate while also considering the financial costs associated with cloud instances.

3.4.5 Energy Cost Formulation

The total VM energy consumed, that is, the energy cost $VM_j^{EngCost}$ for each VM is formulated as:

$$VM_j^{EngCost} = VM_j^{srtEng} + \sum_{i=0}^{i=I}\left(VM_j^{exeEng} T_i^{\text{exe}}\right) \tag{3}$$

where VM_j^{srtEng} is the VM start-up energy cost, VM_j^{exeEng} is the VM executing energy cost, and T_i^{exe} is the executing time of the current job and I is the total number of jobs being currently processed in VM_j.

3.5 THE PROPOSED JOB SCHEDULER USING DRL

3.5.1 DRL Basics

This subsection will acquaint the ready with the basic DRL in job scheduling. In cloud computing a job scheduler optimizes the allocation and execution of computational tasks on VMs [55] for ensuring efficient resource utilization, load balancing, QoS performance optimization, cost efficiency, scalability, energy efficiency, reliability, and adherence to service level agreements (SLAs). Deep Reinforcement Learning (DRL) is crucial in cloud computing because it can dynamically adapt to the unpredictable nature of job submissions, with random arrival times and varying workloads [56][50]. DRL continuously learns and improves its scheduling policies in real time, making it capable of optimal resource allocation and performance optimization even as conditions change [57]. This adaptability ensures efficient utilization of resources, reduces operational costs, and enhances system scalability and performance, effectively managing the complex and fluctuating demands of a public cloud environment. The volume of jobs and the arrival time are unpredictable since users may submit any job to the public cloud at any moment [56]. Consequently, using a reinforcement learning (RL) approach to accommodate the dynamics of the job schedule is appropriate [58]. Deep reinforcement learning can train the agent to discover the optimal policy through interactions with the cloud environments without requiring any prior information [59][60]. Reinforcement learning in this work is utilized for making real time decisions to optimally schedule the jobs in real time into appropriate VMs while minimizing energy usage and concurrently improving QoS. There is no label on training data, in contrast to supervised learning. The training data must be investigated by the agent itself [49].

The agent uses deep neural networks, which are a core aspect of deep learning, for handling complex and high-dimensional state and action spaces, allowing it to approximate the best policies or value functions and solve challenging tasks that traditional RL methods struggle with [49]. The agent observes the state S_t and performs action A_t at each time step t. The agent receives a reward of R_t, and the state of environment is changed from S_t to S_{t+1} [64]. An agent acts in a way to maximize the reward, the agent then receives a reward, either positive or negative, depending on the action taken. Depending on the agent's activity, the reward can be good or bad, and the observation indicates the environment's next state [65]. To perform the task in the most efficient manner possible, the agent attempts to choose the optimum course of action based on the favorable reward points. Because of this, reinforcement learning is a useful strategy that always seeks to learn over time and maximize its rewards [58]. As a result, the reward function is essential and requires to be properly constructed. The DRL agent's objective is to

maximize its expected reward over time:

$$\mathbb{E}[\Sigma_0^{+\infty}\gamma r^t] \tag{4}$$

Where $\mathbb{E}$ denotes the future reward expectation, which is the average or expected value over all possible outcomes or trajectories that the agent might experience. The variable $\gamma \in (0,1]$ is the discount factor, a value between 0 and 1, which determines the importance of future rewards [66]. A lower gamma makes the agent prioritize immediate rewards, while a higher gamma makes it consider long-term rewards more. The variable r^t represents the reward received at time step t which is the feedback signal that the agent receives from the environment after taking an action. In reinforcement learning, an agent's decision is made based on the policy within the action space, given the state of environment, S, is characterized as the probability distribution $\pi(a|s)$. An approximator, or function with numerous adjustable parameters, is typically used when the probability distribution of a complex problem is too large to be stored. Deep neural networks are commonly utilized as approximators in many reinforcement learning problems [67]. In this study, Q-learning serves as the foundational implementation, being one of the most widely used reinforcement learning models that determines the appropriate action based on the current state. Q-learning is a well-known reinforcement learning method that requires no prior system information [49]. In this approach, the agent makes decision based on past experiences by storing each state-action combination in a Q-function, Q (s, a) [55]. The agent aims to achieve a long-term reward at state *s* through action *a*. For every action-state combination, the agent maintains a value function $Q(s, a)$ that indicates the reward for the action performed in the state, s. Additionally, the agent can estimate the q-values for each action with the aid of the value function [66]. An agent uses the q-values to choose the course of action that will ultimately result in the largest reward [55]. The Q-table is the table where the q-values are kept. DRL has become the highpoint of gaming accomplishments, including data center cooling and Atari games [68]. The value function $Q(s, a)$ is updated at each step and is defined as follows:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha\left(r_{t+1} + \gamma \max_{\hat{a}\in\mathcal{A}} Q_t(s_{t+1}, \hat{a})\right) - Q_t(s_t, a_t) \tag{5}$$

Where α is the learning rate and $\alpha \in (0,1]$ and r_{t+1} is the reward obtained by the a_{t+1} on s_{t+1} and γ is the discount factor, which has a value in the range of (0,1].

3.5.2 The IDBS Scheduler Architecture

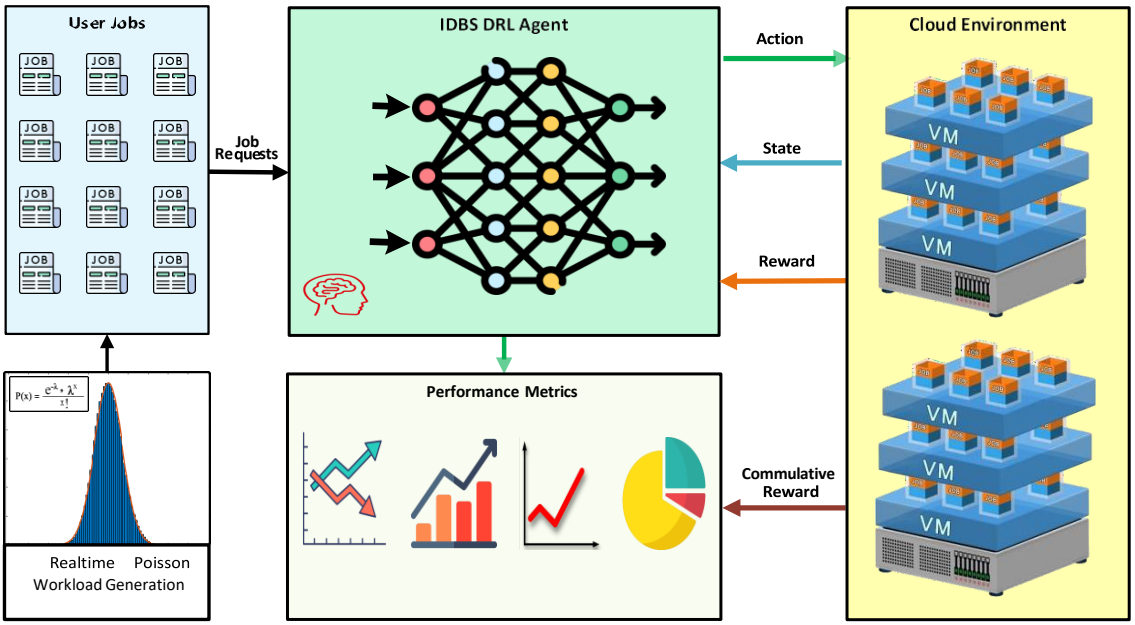


Figure 19. The proposed IDBS job scheduling model based on DRL.

Figure 19 shows the proposed cloud job scheduling model block diagram using DRL. The proposed job scheduler is called Intelligent DRL Based Job Schedular (IDBS). The DRL agent, trained with job data, makes scheduling decisions based on cloud conditions, optimizing energy, QoS, and resource utilization. The suggested framework employs Deep Reinforcement Learning (DRL) for optimizing job scheduling in cloud computing. It adapts to unpredictable job submissions with random arrival times and varying workloads, continuously improving scheduling policies in real time. This ensures efficient resource use, reduces costs, enhances scalability, and manages complex public cloud demands. By leveraging DRL, the framework makes real-time scheduling decisions to assign jobs to VMs, minimizing energy usage and improving Quality of Service (QoS). The basics or the operation of DRL can be understood further in [49][55] [57][59][60][61][62][63].

3.5.3 Decision Phase

The jobs in the queue are scheduled using the DRL mechanism [64]. The reward is computed using the reward function, and the scheduling is based on the q-values assessed by a deep neural network [49]. The virtual machine (VM) that best meets the work criteria and uses the least amount of energy will be assigned to a particular job [49]. In the proposed approach, several jobs are scheduled to run on various virtual machines within cloud resources. Every job operates independently and must be assigned to a virtual machine inside a server for execution [58].

3.5.4 Action Space

The action space consists of all the virtual machines instances available in the cloud [63]. Action space guides job-to-VM assignments and VM management [70], optimizing job scheduling in cloud computing while adhering to QoS. The action space is described as follows:

$$A = [VM_1, VM_2, \ldots, VM_N] \tag{8}$$

Where N represents the total number of the available virtual machines within the cloud. The action space includes every possible VM in the cloud that can be selected for scheduling the job.

3.5.5 State Space

The state space includes all possible configurations of job and VM statuses [49]. The scheduling model leverages the state space to allocate jobs to VMs efficiently, ensuring optimal resource usage and adherence to DRM policies. By analyzing various states representing jobs and VM resource statuses, the model makes informed decisions, maximizing performance while meeting DRM requirements. The proposed model's state-space is specified as follows:

$$S = [Job_1, Job_2, \ldots, Job_i, \ldots, Job_i, Job_I] \cup [VM_1, VM_2, \ldots, VM_j, \ldots, VM_N] \tag{9}$$

Where I is the total number of jobs and N is the total number of VMs. Since the state space encompasses all conceivable states, it typically has a large dimension.

3.5.6 Reward Function

The reward function is like a scorekeeper in a game, telling how good or bad each move is [71]. It helps an algorithm learn by giving positive scores for good actions and negative scores for bad ones. The reward function in job scheduling is like a judge that evaluates how well a scheduling decision works [24]. It gives points based on factors like energy cost of VMs, completing jobs quickly, using resources efficiently, and meeting deadlines. These points help the scheduling algorithm learn to make better decisions for assigning jobs to virtual machines [63]. The reward function is a prediction of the outcome an agent will receive when it takes certain actions in different situations. To enhance accuracy and ultimately job success rate, the agent generally selects states with higher rewards. Initially, it explores random states for learning, then prioritizes those with the greatest rewards. Our model minimizes factors like energy consumption and job response times to craft the optimal reward function. This work proposes the following novel reward function:

$$reward = \left(e^{-\lambda * EnergyCost_j}\right)\frac{Job_i^{qos}}{T_i^{res}} \quad (6)$$

T_i^{res} is the response time of the current job. Job_i^{qos} is the job QoS and encompasses job deadlines, $EnergyCost_j$ is the energy cost of processing this job in VM_j. $EnergyCost_j$ represents the energy consumed by VMs during the execution of jobs. It reflects the resource utilization and efficiency of the cloud environment. The variable $EnergyCost_j$ has also been denoted as $VM_j^{EngCost}$. The variable λ is a hyper-parameter used to balance the energy cost and QoS. A higher λ places more emphasis on minimizing energy consumption, while a lower λ prioritizes meeting QoS requirements. The term $e^{-\lambda * EnergyCost_j}$ represents the energy cost component of the reward. It essentially scales the reward based on the exponential decay of energy cost with respect to λ. Higher values of λ result in a faster decay of the reward with increasing energy cost, prioritizing energy-efficient job scheduling decisions. The term (Job_i^{qos}/ T_i^{res}) represents the QoS and response time component of the reward. It signifies the ratio of job QoS to response time. Higher values of this ratio indicate better QoS relative to response time, which contributes positively to the reward. The overall reward is computed by multiplying these two components ($e^{-\lambda * EnergyCost_j}$) and ($Job_i^{qos}/T_i^{res}$) together. Therefore, actions that result in lower energy consumption, higher job QoS, and faster response times will yield higher rewards. A high reward indicates that the job scheduling decision led to efficient resource utilization, meeting job QoS requirements, and achieving fast response times, while keeping energy consumption low. Conversely, a low reward value suggests that the job scheduling decision resulted in suboptimal resource allocation, possibly leading to higher energy consumption, poorer job QoS, and longer response times. By adjusting the parameter λ, one can control the balance between energy cost reduction and job QoS/response time optimization based on specific priorities and requirements of the cloud environment and its users.

3.5.7 Experience Replay

In the proposed job scheduling for VMs, Q-learning optimizes decisions based on job-to-VM assignments, aiming to minimize energy cost, response time, and maximize VM resource utilization and DRL success rate. Each decision's outcome is rewarded and stored in replay memory Δ which has the capacity $N\Delta$ for learning. Sequential replay enhances learning efficiency by revisiting past experiences. Randomized mini batches stabilize learning, reducing variability and promoting stable decision-making. This iterative process refines scheduling

strategies, balancing performance metrics for efficient job allocation in cloud environments [70].

3.5.8 Target Network and Evaluation Network

The proposed DRL model employs two neural networks: the target network and the evaluation network. This setup aims to stabilize decision-making, minimize parameter fluctuations, and reduce divergence in network learning. The networks have distinct parameters but share the same structure. The **target network** acts as a reference, remaining temporarily frozen to provide stability [49]. During each Q-learning step, the target network generates target Q- values. Compared to the evaluation network, the parameters of the target network are either kept fixed or updated at a slower rate. The target network helps stabilize the learning process by providing consistent target Q-values, reducing parameter oscillations, and aiding convergence [49]. Subsequently, the parameters of the evaluation network are synchronized with those of the target network. This process ensures consistency in decision-making, enhancing efficiency in minimizing energy consumption while maximizing quality of service and minimizing response time. The **evaluation network** is another neural network within the reinforcement learning framework that is actively trained during the learning process. It evaluates states and actions, generating Q-values that guide decision-making in the reinforcement learning algorithm [72]. The parameters of the evaluation network are updated iteratively through the learning process to improve decision-making and optimize performance. The evaluation network learns from experience, adjusting its parameters to improve its ability to generate accurate Q-values and make optimal decisions [49]. The Q-values produced by the evaluation network are utilized to select actions during the exploration-exploitation process, driving the learning and decision-making in the reinforcement learning algorithm [72].

3.5.9 IDBS Scheduling Algorithm

The proposed DRL-based job scheduling algorithm is shown in Algorithm 1. The algorithm assigns jobs to cloud instances, while storing the state and actions in the Q-function for better decision-making. The deep reinforcement algorithm using Q-learning was chosen for creating a smart job scheduler that uses real-time data without requiring prior system knowledge. It also tracks the incoming jobs, updating it as new jobs arrive or completed. A detailed mathematical model is provided later. The proposed DRL scheduling model uses Algorithm 1 to predict the optimal scenario while considering energy consumption. It does this by utilizing both offline decision- making methods, such as experience replay and target networks, along with deep Q learning techniques to improve efficiency and reduce oscillations and divergence of the deep neural network parameters during training [69].

Algorithm 1. The proposed DRL Job Scheduling Algorithm

Initialization Parameters: *Initialize Exploration rate ϵ, learning rate α, learning frequency f, start learning time τ, mini-batch size $S\Delta$, replay period η, action space A*

Initialize Replay Memory, Δ *with capacity* $N\Delta$

Initialize Q-function: *Initialize Q(s,a) with random parameters* θ

Initialize Target Q-function: *Q^(s, a) with random parameters* θ'

Main Loop: *For episode = 1, ..., E:*

- ***Reset cloud server environment to initial state***
- ***Initialize metrics for the episode***
- ***Action Selection,*** a_j*:*

$$a_j = \begin{cases} random \ \ action, & with\ probability\ \epsilon \\ otherwise\ arg\ max_a Q(s_j, a; \theta), & with\ probability\ 1-\epsilon \end{cases}$$

- ***Job Scheduling:***
- *- Schedule job* j *to a VM based on action* a_j
- *- Receive reward* r_j
- *- Observe new state* $s_j + 1$
- ***Calculate Optimization Metrics:*** *Energy cost, utilization, response time, success rate*
- ***Store transition*** $(s_j, a_j, r_j, s_j + 1)$ ***in***
- ***Learning Check:*** *If* $j \geq \tau$ *and* $j \equiv 0 \ mod\ f$:
  - ***Target Network Update Check***
  - *if* $j \equiv 0 \ mod\ \eta$ *then* $\theta' \leftarrow \theta$
  - ***Sample Mini batch***
  - *Randomly select mini batch samples* S_Δ *from replay memory* Δ
  - ***Transition:*** *For each transition*
    - ***Compute Q-network Target Value***
    - $target_k = r_k + \lambda \max_{a'} Q(s_{k+1}, a'; \theta')$
    - ***Perform Gradient Descent***
    - $\theta \leftarrow \theta - \alpha \nabla \theta Loss$, *where* $\nabla \theta Loss$ *is the gradient*
  - *End Transition Loop*
  - ***Decay Exploration Rate***
  - $\epsilon_{t+1} = \epsilon_t \times decay_factor$, *Balance exploration & exploitation*
- *End Learning check*

End Main Loop

Algorithm 1 is explained as follows and each step is separated by blank line to make each step clear its own paragraph.

The **Initialization Parameters** in step 1 involve setting up and initializing various parameters essential for job scheduling and VM management, such as the exploration rate (ϵ), learning rate (α), learning frequency (f), start learning time (τ), mini-batch size ($S\Delta$), replay period (η), and action space (A). These parameters play crucial roles in guiding job scheduling decisions and optimizing VM resource utilization [65].

In step 2 (**Initialize Replay Memory**), the replay memory (Δ) is initialized with a capacity ($N\Delta$). Replay memory stores past experiences related to job scheduling, such as state-action-reward transitions, which are later used for efficient training and decision-making.

In step 3 (**Initialize Q-function**), the Q-function Q (s, a) is initialized with a random parameters θ. This function estimates the expected reward for assigning jobs to VMs, guiding decision-making during job scheduling based on the current state and available actions.

In step 4 (**Initialize Target Q-function**), the target Q-function Q ˆ (s, a) is initialized with random parameters (θ'). This function serves as a target for training the Q-function, helping stabilize the learning process and improving the accuracy of job scheduling decisions.

In step 5 **(Main Loop),** the main scheduling process is initiated, iterating over episodes (or time steps) where each episode involves the action of scheduling jobs to VMs.

Step 6 **(Reset cloud server environment to initial state),** involves resetting the cloud server environment, including VMs and job queues, to an initial state before each episode. It ensures consistency and reproducibility in the job scheduling process.

In step 7 **(Initialize metrics for the episode)**, metrics such as VM Energy cost, utilization, response time, success rate, etc., are initialized to monitor and optimize the performance of the job scheduling algorithm throughout each episode.

In step 8 **(Action Selection**), actions are selected based on the current state, representing decisions regarding job assignments to VMs. With a certain probability ϵ, random actions are chosen for exploration, while otherwise, actions maximizing the Q-value for the current state are selected for exploitation.

Step 9 to 12 (**Job Scheduling**) involves making decisions on assigning jobs to VMs based on the selected actions and receiving some rewards based on the VM selected for scheduling. Jobs are scheduled to VMs to optimize resource utilization, minimize energy consumption, and reduce job response times and a new state is then observed for the next scheduling.

In step 13 (**Calculate Optimization Metrics**), optimization metrics such as energy cost, VM

utilization, and job response time are calculated in terms of the current state of VMs and the scheduled jobs. These metrics provide insights into the efficiency and effectiveness of the job scheduling process.

In step 14 (**Store transition in replay memory**), transitions representing state-action-reward-next state tuples are contained in the replay memory for experience replay. These transitions capture the interactions between job scheduling decisions and their outcomes, facilitating learning and decision-making improvements over time.

The **Learning Check** in step 15 checks whether it's time to update the Q-network parameters based on the learning frequency f. If so, learning updates are performed to improve the accuracy of the Q-function and enhance job scheduling decisions [69].

In step 16 (**Target Network Update Check**), the target Q-network is periodically updated to align with the current Q-network's parameters, stabilizing the learning process and reducing oscillations in job scheduling decisions.

In step 17 (**Sample Mini-batch**), mini-batch samples are randomly selected from the replay memory for training the Q-function. These samples represent past job scheduling experiences and are used to update the Q-function parameters efficiently.

In step 19, the Compute **Q-network Target Value** step stabilizes learning by providing consistent target values, preventing oscillations, and improving resource allocation decisions through more reliable policy updates in step 20 (**Perform Gradient Descent**), gradient descent is performed on the loss function to update the Q-function parameters. This optimization process adjusts the Q-function to better approximate the expected rewards of different job scheduling actions.

In step 22 (**Decay Exploration Rate**), the exploration rate ϵ is decayed over time to balance exploration and exploitation in the job scheduling process. This allows the algorithm to gradually shift from exploring new job scheduling strategies to exploiting learned knowledge for more efficient decisions.

Step 23 (**End Learning check**) marks the end of the learning check loop, completing the update process for the Q-function parameters based on past job scheduling experiences.

Step (**End Main Loop**), signifies the end of the main scheduling process loop, completing the job scheduling process for the specified number of episodes or time steps.

3.6 CHAPTER SUMMARY

This chapter presented the proposed Job Scheduling Framework using DRL called IDBS. The framework architecture proposed all the DRL phases including decision phase, action phase, state space, reward functions, experience replay, target network were presented. The scheduling algorithm was also presented in detail.

CHAPTER 4: RESULTS AND DISCUSSIONS

4.1 CHAPTER INTRODUCTION

In this section, the experimental setup will first be defined for the performance evaluation the proposed approach will be compared with related works. The experimental setup and evaluation metrics will be defined. Then the experimental parameters will be presented. The evaluation metrics will also be mapped to the research problem followed by the presentation of the results and their discussions.

4.2 EXPERIMENTAL SETUP

The experimental setup aims to thoroughly evaluate the proposed deep reinforcement learning (DRL) framework for job scheduling in cloud environments. The framework will be tested under three different scenarios: varying job arrival rates, varying VM proportions, and varying job proportions. Each scenario is designed to assess different aspects of the framework's performance and adaptability. **Varying Job Arrival Rates:** This experiment will vary the mean job arrival rate to test the framework's scalability and efficiency under different workloads. Understanding how the framework handles different arrival rates is crucial for ensuring it can maintain low response times and high success rates in real-world scenarios with fluctuating job loads. **Varying VM Proportions:** In this scenario, the proportions of high CPU and high I/O VMs will be varied to evaluate the framework's ability to optimize resource allocation and maintain performance. This helps in understanding the framework's adaptability to different cloud resource configurations, which is vital for maximizing resource utilization and minimizing energy costs. **Varying Job Proportions:** In this experiment, the ratio of I/O-intensive to compute-intensive jobs will be changed to examine the framework's effectiveness in handling diverse job types. The goal is to ensure the framework can manage varying job compositions efficiently, minimize energy consumption and maintain high Quality of Service (QoS).

Visual Studio Code was chosen as the simulation platform, owing to its robust Integrated Development Environment (IDE). Its versatility in language support and framework extensibility expedites code development, debugging, and deployment tasks. This choice facilitated the implementation and testing of Deep Reinforcement Learning (DRL)-based job scheduling algorithms within cloud computing contexts. Additionally, Matplotlib, a powerful Python plotting library, was adopted for data visualization purposes. This tool aids in crafting

performance graphs vital for evaluating the performance of proposed DRL-based job scheduling model. NumPy was the foundational library for numerical computing. Its efficient data structures and array manipulation capabilities streamline data processing tasks within DRL-based job scheduling models, fostering performance enhancements and scalability. Deep learning frameworks such as PyTorch and TensorFlow furnish high-level abstractions for neural network construction and training. Leveraging these frameworks, the proposed model was implemented and fine-tuned. This approach aims at optimizing energy consumption and guaranteeing QoS in cloud computing.

4.3 EVALUATION METRICS

The adopted evaluation metrics are Success Rate (SR) (%) Average Response Time (ART) (ms), and Energy Cost (EC) (Joules), Resource Utilization (%), Quality of Service (QoS), Scalability with justification from the literature Review Section.

4.3.1 Success Rate (SR)

SR measures the effectiveness of job scheduling in meeting Quality of Service (QoS) requirements by determining the proportion of successfully completed jobs within a given time frame. A high SR indicates that a large percentage of jobs are executed satisfactorily, meeting or exceeding the specified QoS constraints.

$$SR = \frac{SuccessfulJobs}{Total\ Jobs} \tag{7}$$

High SR (>0.8) shows efficient job scheduling, where most incoming jobs are successfully completed within the expected time frame, ensuring customer satisfaction and adherence to service level agreements (SLAs). Low SR (≤0.8) indicates inefficiencies or failures in job scheduling, with a significant portion of jobs experiencing delays, resource shortages, or outright failures. This can lead to customer dissatisfaction, SLA violations, and increased operational costs.

4.3.2 Average Response Time

Average Response Time (ART) represents the average duration taken to process and complete jobs from their submission to their fulfillment. It serves as a crucial indicator of system responsiveness and efficiency.

$$ART = (\frac{\sum_{i-1}^{n} ResponseTime_i}{n} \tag{8}$$

Low ART indicates efficient job processing and prompt job completion, meeting or surpassing QoS expectations. Users experience minimal waiting times, leading to enhanced satisfaction and productivity. High ART reflects delays or inefficiencies in job processing, where jobs take longer than expected to be completed. This can result in degraded user experience, missed deadlines, and reduced system performance.

4.3.3 Energy Cost

Energy Cost (EC) measures the VM energy consumption associated with executing scheduled jobs. It reflects the environmental impact and operational expenses incurred by the cloud infrastructure.

$$EC = \frac{Energy\ Consumed}{Total\ Jobs} \tag{9}$$

Low EC signifies efficient energy usage, where jobs are executed with minimal energy consumption. This contributes to cost savings and environmental sustainability while maintaining QoS standards. High EC indicates excessive energy consumption, potentially resulting from inefficient resource allocation or suboptimal job scheduling. Higher energy costs can lead to increased operational expenses and ecological footprint.

4.3.4 QoS or Deadline Satisfaction Ratio (DSR)

QoS represented by DSR compares a job's actual execution time to its deadline. It evaluates the system's ability to meet job deadlines and fulfill QoS requirements.

$$QoS = DSR = \frac{ET}{D} \tag{10}$$

A high DSR (>1) indicates timely job completion, where jobs are executed within or before their specified deadlines. This demonstrates efficient job scheduling and adherence to QoS standards. A low DSR (≤1) signifies deadline breaches, where jobs exceed their allotted execution time or miss their deadlines. This can lead to SLA violations, customer dissatisfaction, and penalties.

4.3.5 Resource Utilization

VM Resource Utilization measures the effectiveness of resource allocation in virtual machines. It assesses the degree to which available resources are utilized to support job execution.

$$Utilization = \frac{Used\ Resources}{Total\ Resources} \tag{11}$$

High utilization (>0.8) indicates efficient resource allocation, where a significant portion of available resources is utilized to support job execution. This optimizes resource usage and contributes to overall system performance. Low utilization (≤0.8) suggests underutilization or inefficient resource allocation, where available resources remain idle or underutilized. This can lead to wasted resources, decreased system performance, and increased operational costs.

4.3.6 Reward Metric

Reward Metric evaluates the performance of the scheduling algorithm by quantifying the cumulative rewards obtained during the training or execution process. Rewards are provided based on how well the agent (scheduler) performs actions that lead to desirable outcomes, such as meeting QoS requirements, optimizing resource usage, and minimizing job processing time. A high reward indicates effective decision-making by the DRL agent, where actions lead to positive outcomes such as successful job completions within deadlines, efficient resource utilization, and minimized response times. A high reward reflects optimal scheduling strategies that align with QoS objectives and operational efficiency. Low Reward suggests suboptimal decision-making or ineffective policies employed by the DRL agent, resulting in negative outcomes such as job failures, missed deadlines, inefficient resource allocation, and increased response times. Low rewards indicate the need for improvement in scheduling strategies to enhance system performance and meet QoS requirements effectively.

4.3.7 Scalability

Scalability encompasses all the other metrics above and will be evaluated in terms of VM variation, job type variation and job arrival rate variation. Scalability refers to the ability of a system or framework to efficiently adapt and handle increasing demands or changes in workload characteristics without compromising performance or incurring significant overhead. In the context of cloud computing, scalability is essential for dynamically adjusting resources to meet fluctuating workload requirements in real-time. Scalability in **VM variation** entails the framework's ability to seamlessly accommodate changes in virtual machine configurations, such as varying CPU, memory, or storage capacities. A scalable solution should efficiently allocate resources across different VM types to meet workload demands without sacrificing performance or incurring unnecessary resources overhead. Scalability in **job type variation** refers to the framework's capacity to effectively manage and process diverse types of jobs with varying resource requirements. This involves dynamically allocating resources based on the specific characteristics and demands of each job type to ensure optimal performance and resource utilization. Scalability in **job arrival rate variation** involves the framework's

capability to adapt to fluctuations in the rate at which jobs are submitted or arrive in the cloud. A scalable solution should be able to dynamically scale resources, either increasing or decreasing in response to fluctuations in workload intensity, thereby ensuring efficient job scheduling, execution, and real-time resource allocation.

4.4 MAPPING EVALUATION METRICS TO THE RESEARCH PROBLEM

It is necessary to associate every research problem to a corresponding evaluation metric to quantify how well the proposed framework compared to related works in solving the gap, to what degree of solving the gap in terms of the gaps performance metric value achieved.

4.4.1 Energy Consumption Problem

The proposed framework addresses the pressing concern of high energy consumption in cloud infrastructures by minimizing Energy Cost (EC) compared to related works. This metric quantifies the framework's success in curbing energy expenditure, crucial for sustainable and cost-effective cloud operations. By achieving lower EC values, the framework demonstrates enhanced energy efficiency, contributing to reduced operational costs and environmental impact.

4.4.2 Cloud Job Scheduling Problem:

Success Rate (SR), Average Response Time (ART), Energy Cost (EC), Resource Utilization, Quality of Service (QoS) and Scalability metrics collectively evaluate the performance of the proposed framework in tackling the challenges of cloud job scheduling. SR measures the proportion of successfully completed jobs, reflecting the framework's efficiency in meeting workload demands. Lower ART signifies faster job processing, enhancing user satisfaction and system responsiveness. Minimized EC indicates optimized resource allocation, leading to reduced operational expenses and energy consumption. Improved Resource Utilization ensures efficient use of cloud resources, maximizing performance and cost-effectiveness. QoS metrics assess the framework's ability to meet service level agreements, ensuring reliable and timely job execution.

4.4.3 Model Optimization Problem

Success Rate (SR), Average Response Time (ART), Energy Cost (EC), Resource Utilization, and Quality of Service (QoS) metrics provide comprehensive insights into the performance of the proposed framework for model optimization. SR evaluates the accuracy and success rate of model predictions, crucial for reliable decision-making and task performance. Lower ART

reflects faster model inference, enhancing real-time responsiveness and user experience. Reduced EC signifies energy-efficient model training and inference processes, essential for sustainable and cost-effective machine learning operations. Optimized Resource Utilization ensures effective allocation of computational resources, maximizing efficiency and scalability. QoS metrics assess the reliability and performance of machine learning models, ensuring high-quality predictions and user satisfaction. To solve the problem of scalability in the dynamic real cloud environment, the scalability metric in terms of VM variation, job type variation and job arrival variation align well in quantifying the model performance in dynamic cloud environments. In terms of VM variation, the scalability in dynamic cloud environments assesses the framework's ability to efficiently allocate resources across diverse VM types, ensuring optimal performance and workload management in response to varying VM configurations. The job type variation quantifies how well the scheduling framework adapts to diverse job types by dynamically allocating resources based on specific job characteristics, optimizing performance and resource utilization efficiency. The job arrival rate variation evaluates the framework's capability to dynamically adjust resource allocation in response to fluctuating job arrival rates, ensuring optimal performance, resource utilization efficiency, and responsiveness in dynamic cloud environments. Scalability in model optimization assesses how well the framework handles increasing complexity, workload size, and computational demands while maintaining performance.

4.5 EXPERIMENTAL PARAMETERS

In the research experimentation, the effectiveness of the proposed approach is evaluated across various workloads and virtual machine (VM) configurations based on [48] and [49] as reference justifications . The total number of jobs is set to 8,000. The duration of each job follows Normal Distribution, with mean and variance parameters of 300 and 20, respectively. Job arrival times are modeled using Poisson Distribution, with mean arrival rates chosen from a set of values ranging from 10 to 30. The proposed model has two job types: I/O-sensitive jobs and High-I/O jobs, each varying in proportion. The proposed implementation leverages a Deep Q-Network (DQN), realized through a two-layer Multi-Layer Perceptron (MLP) with 20 neurons per layer. Following standard configurations for training DRL models, the replay memory size is set to 20, and the mini-batch size is set to 16. Key parameters include a learning rate of 0.01 and a trade-off parameter λ set at 1.5. The greedy parameter is gradually reduced from 0.9 to 0.05, while maintaining a discount factor of 0.9. Learning commences after 800 iterations, with updates occurring every 200 iterations. The proposed DRL framework will be compared to the following

scheduling methods from [48]: Random scheduling, which allocates jobs randomly to available VMs, Round Robin algorithm, which distributes jobs evenly across VMs in a cyclic order; Earliest algorithm, which schedules jobs based on their arrival times; and JingDQN [49], a deep Q-network-based job scheduling approach known for its reinforcement learning capabilities. Random scheduling, Round Robin (RR), and Earliest algorithms serve as the baseline methods for this comparison, while JingDQN represents an advanced DRL learning- based method against which the proposed DRL framework's performance will be measured.

4.5.1 Comparison Frameworks

The proposed DRL framework will be compared with baseline frameworks such as Random scheduling, Round Robin, and Earliest algorithms. In addition, the proposed will also be compared to an advanced DRL based job schedular in [49] which will be denoted JingDQN as it is proposed by Jing et al. JingDQN is a deep Q-network-based job scheduling approach known for its reinforcement learning capabilities. Random scheduling allocates jobs randomly to available VMs; Round Robin algorithm, which distributes jobs evenly across VMs in a cyclic order; Earliest algorithm, which schedules jobs based on their arrival times.

4.6 RESULTS 1: VARYING MEAN JOB ARRIVAL RATE

The purpose of this first experimentation with varying job arrival rates is to evaluate the proposed DRL framework's performance, scalability, and adaptability in real-time job scheduling. By testing under different workload conditions, the experiment ensures to test that the framework can maintain low response times, high Quality of Service (QoS), high success rate, high utilization, and lower energy consumption demonstrating its effectiveness and robustness compared to traditional and other DRL-based scheduling methods. Specifically, the mean job arrival rate varies from 10 to 30 to simulate different levels of workload intensity, assessing how well the DRL scheduler handles varying job loads in a real-time cloud environment. This section details the effects of different job arrival rates on key performance metrics such as average response time, success rate, and energy cost, utilization, and QoS. The compute units required for each job follow a normal distribution with a mean value of 300 and a variance of 20, while job arrival times are modelled using a Poisson distribution. The cloud environment consists of an equal proportion of high central processing unit and high I/O instances, as well as an equal proportion of I/O intensive and computing intensive jobs, creating a realistic and challenging scheduling scenario.

RESULT 1A: Varying the mean job arrival rate vs model average response time - Figure 20 shows the effects of varying the mean job arrival rates on the model response times. The

results demonstrate that the proposed framework consistently outperforms all related frameworks across varying mean arrival rate`s due to its efficient real-time scheduling mechanism. At lower job arrival rates, the proposed model outperforms Random by 75%, Round Robin by 68%, Earliest by 68%, and JingDQN by 53%, resulting in an average outperformance of 66%. As job arrival rates increase, the proposed framework maintains a lower average response time. At higher job arrival rates, the proposed model outperforms Random by 50%, Round Robin by 65%, Earliest by 58%, and JingDQN by 37%, with an average outperformance of 52.5%. The drop in performance from 66% to 52.5% at higher arrival rates is expected due to the increased load on VMs, as higher job arrival rates typically lead to more complex resource allocation and scheduling tasks, but the proposed DRL-based model still performs significantly better than others.

Related frameworks such as Round Robin and Random scheduling do not account for fluctuating job arrival rates, often causing bottlenecks and job delays. In contrast, the DRL framework predicts job loads and allocates resources dynamically, ensuring that jobs are assigned to less congested resources. This reduces queueing delays and ensures faster job processing, even during peak workloads. These results align with the research objective of addressing gap slow response times in cloud environments.

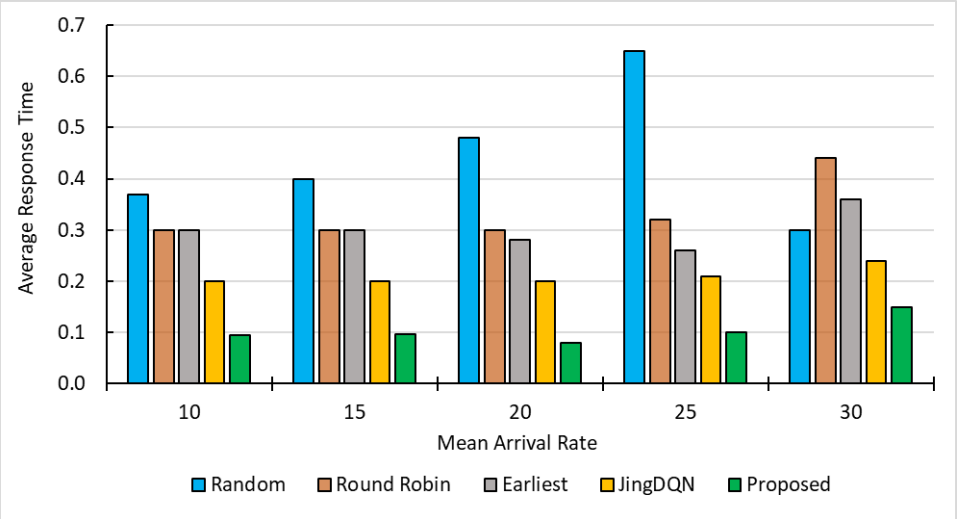


Figure 20. RESULTS 1A: Varying the mean job arrival rate vs model average response time.

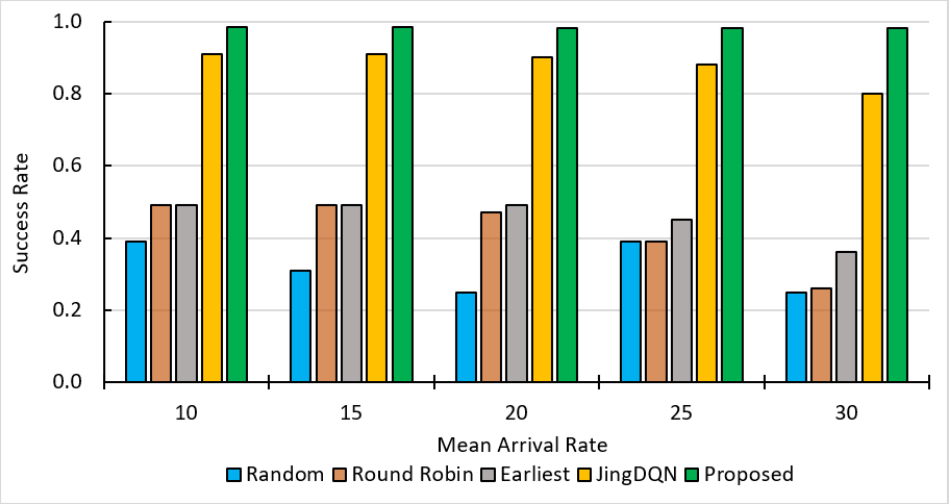


Figure 21. RESULTS 1B: Varying the mean job arrival rate vs model success rate.

RESULTS 1B: Varying the mean job arrival rate vs model success rate - Figure 21 illustrates the impact of varying the mean job arrival rate on the job success rate of the proposed framework. The results indicate that the proposed framework achieves the highest success rate across all frameworks for every mean arrival rate. At lower arrival rates, the proposed framework achieves a success rate of 98.4%, outperforming Random (39%), Round Robin and Earliest at 49%, and JingDQN (91%). As job arrival rates increase, the proposed framework maintains its superior performance with a success rate of 98%, still outperforming Random (25%), Round Robin (26%), Earliest (36%), and JingDQN (80%), highlighting its scalability and efficiency under heavy workloads. The DRL framework adjusts resource allocation and scheduling decisions based on job arrival patterns, ensuring a higher percentage of successfully completed jobs, even under heavy workloads.

The highest success rate achieved by the proposed model is 98.4%, showcasing its effectiveness in handling job scheduling and maintaining high completion rates under varying workloads. The proposed DRL-based model can dynamically adjust resource allocation, ensuring that jobs are efficiently distributed to the most suitable VMs, resulting in consistently high success rates across varying job arrival conditions.

RESULTS 1C: The impact of varying job arrival rates on VM utilization- Figure 22 shows the effect of varying job arrival rates on VM resource utilization. The results demonstrate that the proposed framework consistently achieves the highest utilization across all frameworks by dynamically adjusting resource allocation based on real-time dynamic cloud traffic demands.

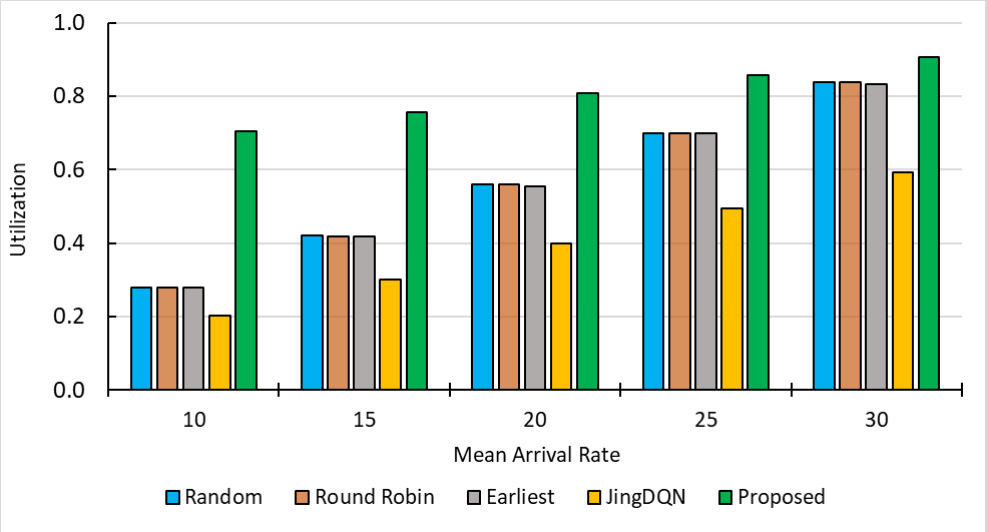


Figure 22. RESULTS 1C: The impact of varying job arrival rates on VM resource utilization

At a lower arrival rate, the proposed framework achieves a utilization rate of 70.4%, outperforming Random (28.0%), Round Robin (28%), Earliest (27.9%), and JingDQN (20.2%). As the job arrival rate increases, the proposed framework maintains its superior performance with a utilization rate of 90.7%, outperforming Random (83.9%), Round Robin (83.8%), Earliest (83.4%), and JingDQN (59.2%). The highest utilization achieved by the proposed model is 90.7% at higher arrival rates, showcasing its efficiency in maintaining high resource utilization even under heavy workloads.

Related frameworks often lead to underutilization or overloading of VMs due to static resource allocation policies. The DRL model continuously monitors VM usage and adjusts resource assignments, accordingly, leading to higher utilization rates and reducing idle time in the system. These results address the research gap of cloud optimization by optimizing cloud resource utilization, which is critical for improving energy efficiency and workload handling.

RESULTS 1D: The impact of varying job arrival rates on energy cost - Figure 23 shows the effect of varying the mean job arrival rates on energy cost. The results demonstrate that the proposed framework consistently outperforms all related frameworks across varying job arrival rates. At lower job arrival rates, the proposed model outperforms Random by 39%, Round Robin by 36%, Earliest by 36%, and JingDQN by 1%, resulting in an average outperformance of 28%. As the job arrival rates increase, the proposed model continues to maintain lower energy costs, outperforming Random by 40%, Round Robin by 41%, Earliest by 42%, and JingDQN by 19%, with an average outperformance of 35.5%. The slight drop in performance from 28% to 35.5% can be attributed to the increasing complexity of managing resources efficiently as the workload grows, which marginally impacts the energy cost savings.

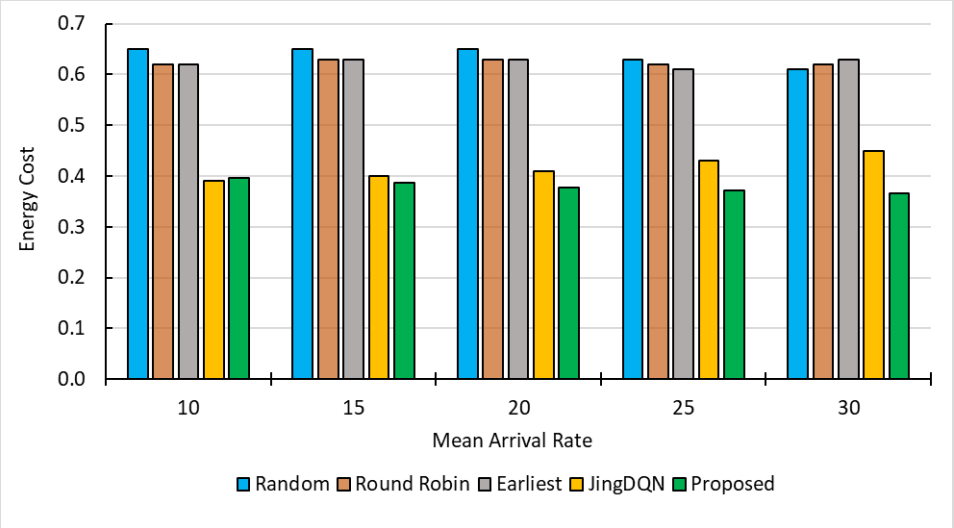


Figure 23. RESULTS 1D: The impact of varying job arrival rates on energy cost.

The related frameworks often waste energy by leaving VMs idle or underutilized. The DRL model ensures resources are used only when needed, dynamically scaling up or down based on job loads, thus minimizing energy waste. This results in a substantial decrease in overall energy consumption without sacrificing performance. The results address the critical gap in literature concerning high energy consumption in cloud computing. The proposed framework's superior performance is due to its DRL-based scheduling mechanism, which enables real-time decision-making and efficient resource allocation.

RESULTS 1E: The impact of varying job arrival rates on QoS - Figure 24 illustrates the effect of varying job arrival rates on QoS. The results demonstrate that the proposed DRL framework maintains higher QoS under varying job arrival rates by prioritizing job completion within deadlines. At a lower arrival rate, the proposed framework achieves a QoS score of 85.1%, outperforming Random (22%), Round Robin (27%), Earliest (27%), and JingDQN (40%). As the job arrival rate increases to 30, the proposed framework maintains its superior performance with a QoS score of 53.3%, still outperforming the other frameworks. The highest QoS score achieved by the proposed model is 100% at a job arrival rate of 20, demonstrating its ability to deliver high-quality service consistently under varying workloads. In addition, the model's efficiency in real-time job scheduling and resource allocation ensures that the QoS remains higher than the rest, despite the increased workload.

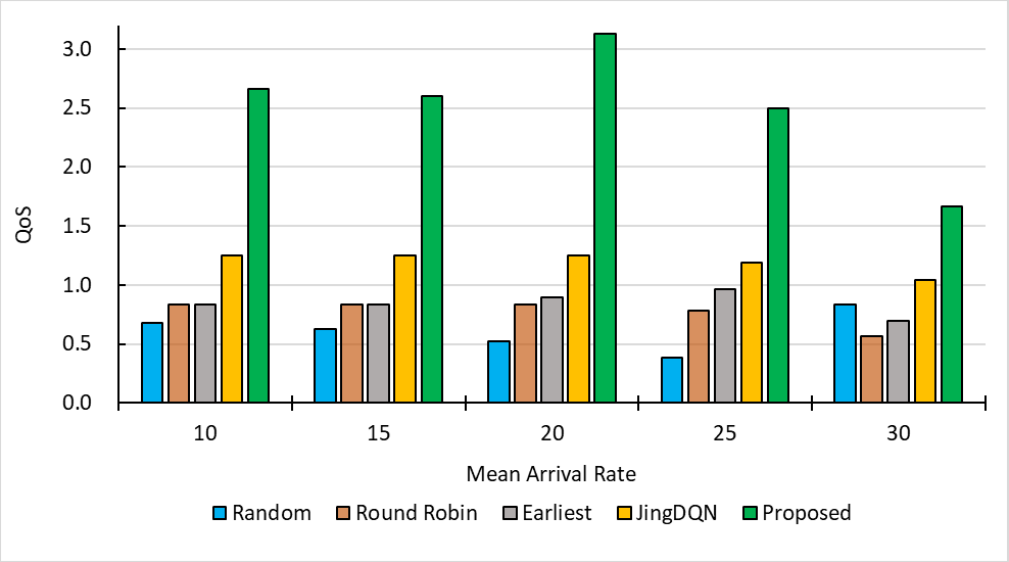


Figure 24. RESULTS 1E: The impact of varying job arrival rates on QoS.

Related scheduling methods fail to adapt to fluctuating job loads, leading to missed deadlines and degraded QoS. The DRL model, however, uses real-time decision-making to ensure that jobs are scheduled and completed within their deadlines, maintaining consistent QoS even as job arrival rates increase.

4.7 RESULTS 2: VARYING VM PROPORTION

To assess the performance of our approach under unbalanced virtual machine (VM) conditions, we adjust the proportion of High-I/O virtual machines from 10% to 90%, maintaining a constant average job arrival rate of 20, and equal proportions of I/O sensitive and compute sensitive jobs. Jobs were categorized into I/O-intensive and compute-intensive tasks and assigned accordingly.

RESULTS 2A: Varying VM type proportions on average response time - Figure 25 shows the effect of varying VM type proportions on the model response times. The results demonstrate that when VM proportions vary, the DRL framework demonstrates lower average response times by intelligently scheduling jobs to the most suitable VMs based on their capabilities. At a 10% VM type proportion, the proposed model outperforms Random by 71%, Round Robin by 55%, Earliest by 50%, and JingDQN by 48%, resulting in an average outperformance of 56%. As the VM type proportion increases to 90%, the proposed framework still maintains lower response times, outperforming Random by 77%, Round Robin by 64%, Earliest by 61%, and JingDQN by 59%, leading to an average outperformance of 65.25%.

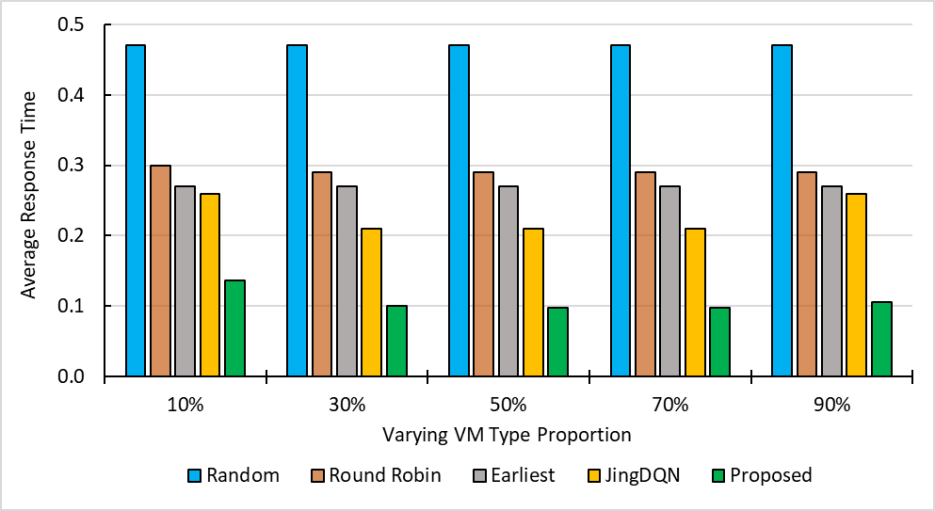


Figure 25. RESULTS 2A: The impact of varying VM type proportions on average response time.

The DRL model learns the strengths of high CPU and high I/O VMs, distributing jobs accordingly to reduce queuing times and improve overall response efficiency. The DRL-based model dynamically adjusts to varying VM types and proportions, ensuring that jobs are allocated to the most suitable and less-loaded VMs in real-time. This adaptability contributes to its superior performance even as VM proportions change.

RESULTS 2B: Varying VM type proportions on model success rate - Figure 26 shows the impact of varying VM type proportions on model success rate. The results demonstrate that the proposed DRL framework maintains a higher success rate under varying VM proportions by adapting to the unique characteristics of each VM type.

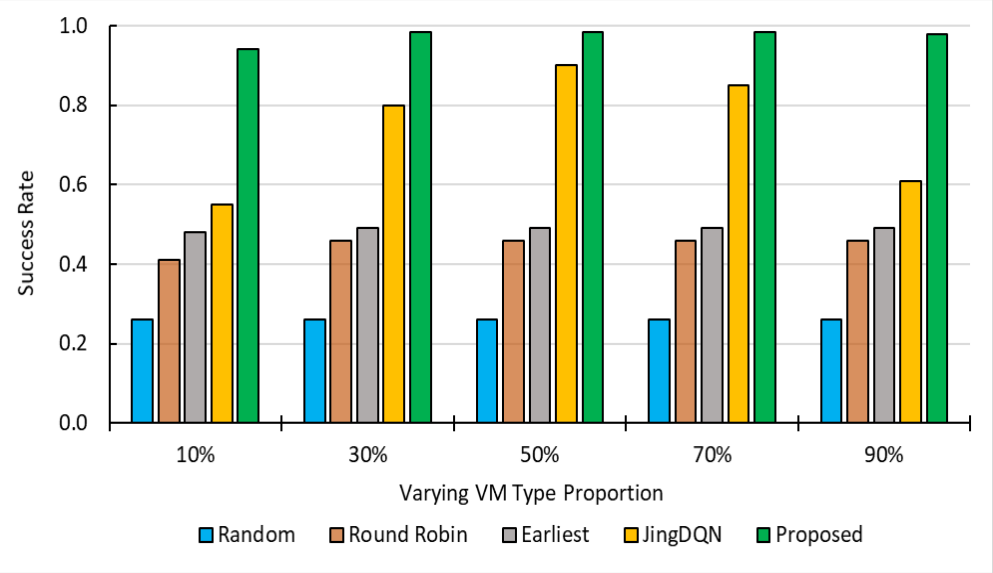


Figure 26. RESULTS 2B: The impact of varying VM type proportions on model success rate.

At a 10% VM type proportion, the proposed model achieves a success rate of 94.2%, outperforming Random (26%), Round Robin (041%), Earliest (48%), and JingDQN (55%). As the VM type proportion increases, the proposed model maintains its superior performance with a success rate of 97.7%. The highest success rate achieved by the proposed model is 98.4% at a VM type proportion of 50%. Related scheduling methods clearly struggle with heterogeneous VM environments, leading to poor job allocation and execution failures. The DRL model, however, effectively matches jobs with appropriate VMs, ensuring that jobs are successfully completed even when the proportion of VM types fluctuates.

RESULTS 2C: Varying VM type proportions on resource utilization - Figure 27 shows the impact of varying VM type proportions on VM resource utilization. The highest utilization rate achieved by the proposed model is 80.8% across VM type proportions indicating the framework's ability to efficiently allocate resources and maximize VM utilization under different configurations. At a 10% VM type proportion, the proposed model achieves a utilization rate of 80.8%, outperforming Random (55.7%), Round Robin (55.8%), Earliest (55.5%), and JingDQN (44.5%). As the VM type proportion increases to 90%, the proposed model maintains a utilization rate of 74.3%, still outperforming Random (55.7%), Round Robin (55.9%), Earliest (55.7%), and JingDQN (64.9%).

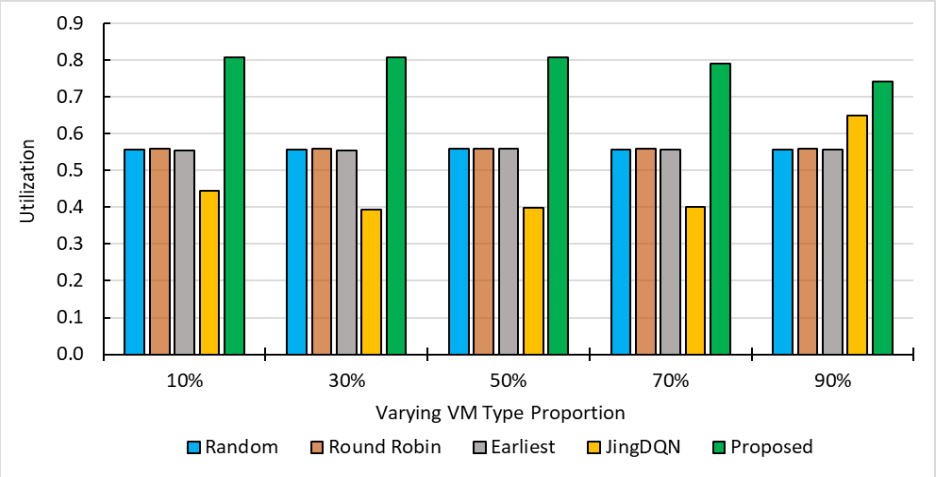


Figure 27. RESULTS 2C: The impact of varying VM type proportions on resource utilization.

Related frameworks fail to optimize VM usage in heterogeneous environments, often leading to overloading or underutilization of certain VMs. The DRL model continuously monitors VM usage and makes real-time adjustments, ensuring that resources are fully utilized across different VM configurations. The proposed model's success in maintaining high utilization rates is due to its ability to dynamically adapt to VM type variations, ensuring that resources are allocated optimally to handle different job types, leading to better overall system performance and reduced resource wastage.

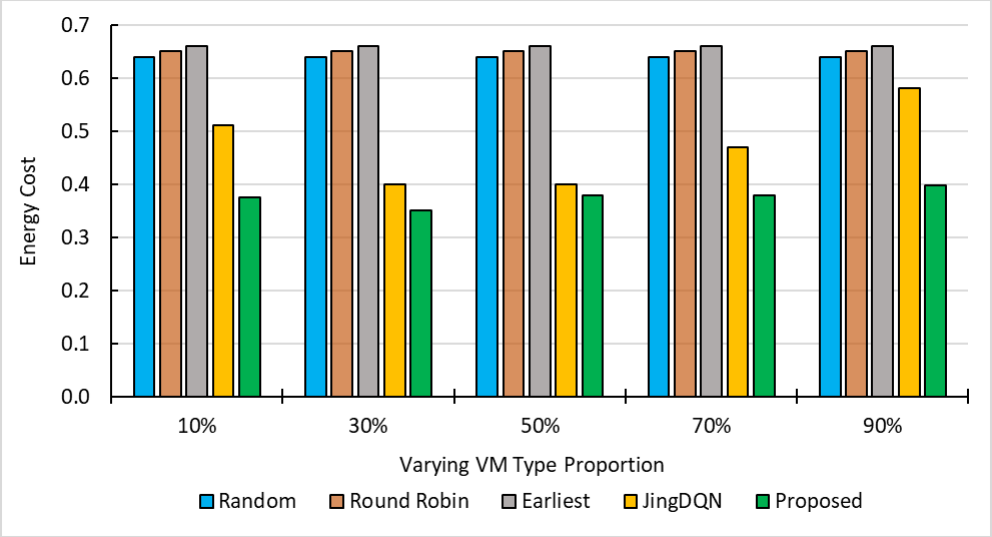


Figure 28. RESULTS 2D: The impact of varying VM type proportions on energy cost.

RESULTS 2D: Varying VM type proportions on energy cost - Figure 28 shows the effect of varying VM type proportions on energy cost. The results demonstrate that the proposed framework outperforms all related frameworks across different VM type proportions in terms of energy cost. At a 10% VM type proportion, the proposed model outperforms Random by 41%, Round Robin by 42%, Earliest by 43%, and JingDQN by 26%, resulting in an average outperformance of 38%. As the VM type proportion increases to 90%, the proposed framework continues to maintain lower energy costs, outperforming Random by 38%, Round Robin by 39%, Earliest by 40%, and JingDQN by 31%, with an average outperformance of 37%. The slight decrease in performance from 38% to 37% as VM type proportions increase as expected is due to the increased computational overhead in managing heterogeneous VM types.

Related frameworks often lead to inefficient resource utilization, with energy being wasted on underutilized or idle VMs. The proposed DRL-based model reduces energy consumption by efficiently allocating jobs to the most suitable VMs, ensuring that energy is not wasted on over- or underutilized resources. Even with varying VM type proportions, the proposed model's dynamic resource allocation contributes to its lower energy costs. These results address the gap in high energy consumption in cloud computing, one of the key objectives of this research.

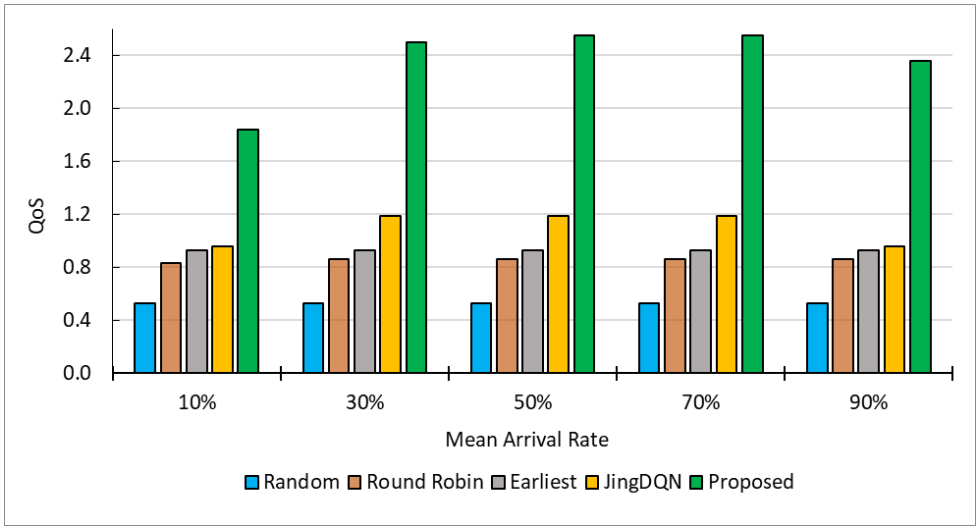


Figure 29. RESULTS 2E: The impact of varying VM type proportions on QoS.

RESULTS 2E: Varying VM type proportions on QoS - Figure 29 shows the impact of varying VM type proportions on Quality of Service (QoS). The results demonstrate that the proposed framework consistently achieves higher QoS compared to other frameworks across all VM type proportions. At a 10% VM type proportion, the proposed model achieves a QoS score of 58.8%, outperforming Random (17%), Round Robin (26.7%), Earliest (29.6%), and JingDQN (30.8%). As the VM type proportion increases to 90%, the proposed model maintains a QoS score of 75.4%, still outperforming Random (17.0%), Round Robin (27.6%), Earliest (29.6%), and JingDQN (30.8%). The highest QoS score achieved by the proposed model is 81.6% at both 50% and 70% VM type proportions, indicating the model's efficiency in ensuring high service quality under various VM configurations.

The proposed DRL framework maintains a higher QoS under varying VM proportions by intelligently managing job deadlines and allocating resources to meet QoS requirements. Related methods often miss deadlines or allocate jobs to inappropriate VMs, resulting in poor QoS. The DRL model dynamically adjusts scheduling decisions based on the current VM configuration, ensuring that jobs are completed within their deadlines and maintaining high QoS across different VM setups. These results align with the research objective of optimizing QoS, leading to better customer satisfaction and adherence to service level agreements SLAs.

4.8 RESULTS 3: VARYING JOB PROPORTION

This section investigates the impact of varying job proportions on the performance of the proposed deep reinforcement learning (DRL) framework for job scheduling. Like the above experiment, this study evaluates the framework's performance under varying job proportions. The proportion of I/O-sensitive jobs is varied from 10% to 90%, while the mean job arrival rate

is maintained at 20, and the proportions of high-I/O VMs and high-computing virtual machines are both held constant at 50%. This approach assesses how well the framework adapts to different workload compositions. Key performance metrics such as average response time, success rate, VM resource utilization, energy cost, and Quality QoS are evaluated under these varying job proportion scenarios.

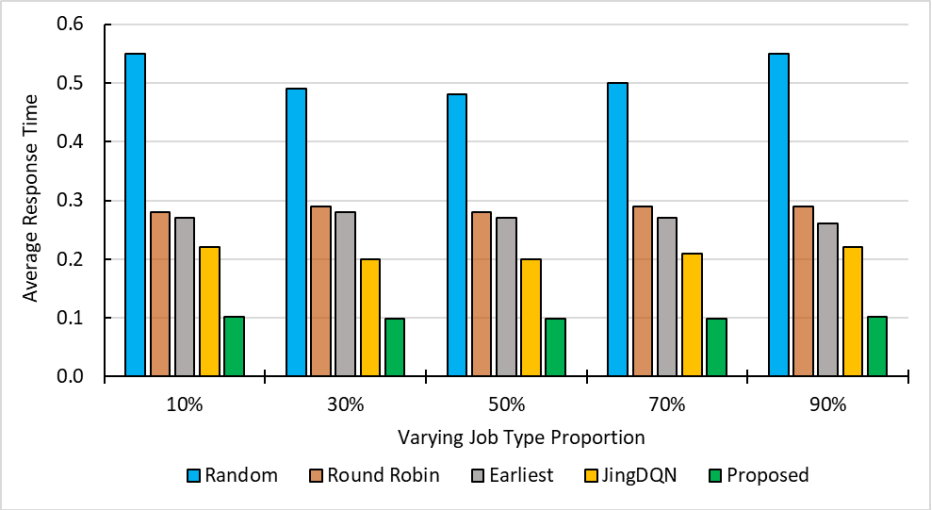


Figure 30. RESULTS 3A: The impact of varying job proportions on average response time.

RESULTS 3A: Varying job proportions on average response time - Figure 30 shows the effect of varying job type proportions on average response time. The results demonstrate that the proposed framework outperforms all related frameworks across varying job type proportions. At a 10% job type proportion, the proposed model outperforms Random by 81%, Round Robin by 64%, Earliest by 62%, and JingDQN by 54%, resulting in an average outperformance of 65.25%. As the job type proportion increases to 90%, the proposed framework still maintains lower response times, outperforming Random by 81%, Round Robin by 64%, Earliest by 61%, and JingDQN by 54%, maintaining an average outperformance of 65%. The DRL-based framework achieves lower average response times under varying job proportions by efficiently matching job types (I/O-intensive vs. compute-intensive) with the most suitable VMs. Related methods fail to account for the varying demands of different job types, leading to bottlenecks and delays. The DRL model prioritizes jobs based on their characteristics, ensuring that they are processed quickly and efficiently, reducing overall response times.

RESULTS 3B: Varying job proportions on model success rate - Figure 31 shows the impact of varying job proportions on model success rate. The highest success rate achieved by the proposed model is 98.5% and it remains highly stable across all job proportions, indicating its robustness in handling different job compositions.

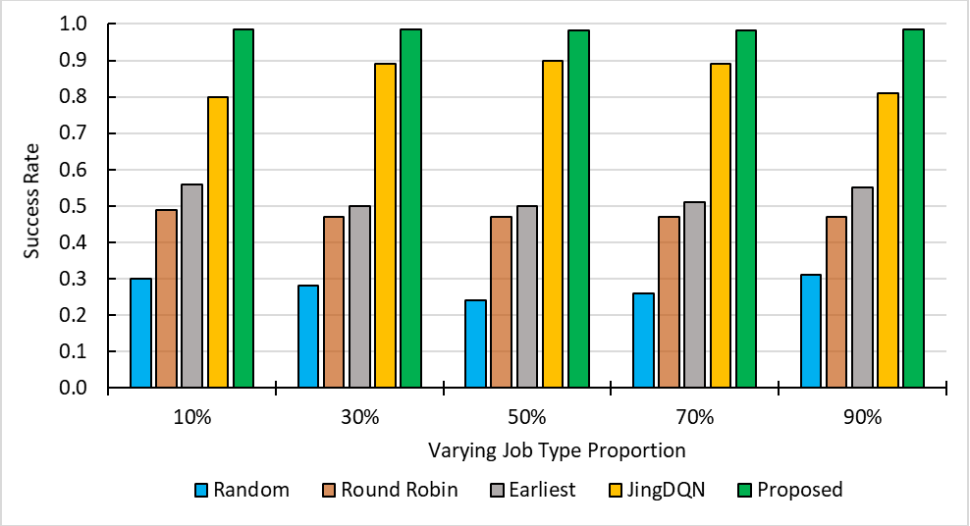


Figure 31. RESULTS 3B: The impact of varying job proportions on model success rate

At a 10% job type proportion, the proposed model achieves a success rate of 98.5%, outperforming Random (30%), Round Robin (49%), Earliest (56%), and JingDQN (80%). As the job type proportion increases to 90%, the proposed model maintains its superior performance with a success rate of 98.4%, still outperforming Random (31%), Round Robin (47%), Earliest (55%), and JingDQN (81%).

The DRL framework maintains a higher success rate when job proportions vary by intelligently scheduling jobs according to their specific requirements. Related frameworks often fail to differentiate between job types, leading to mismatches in resource allocation and job failures. The DRL model learns the differences between I/O-intensive and compute-intensive jobs, ensuring that each job type is allocated to the proper resources, resulting in higher success rate.

RESULTS 3C: Varying job proportions on Resource Utilization - Figure 32 illustrates the impact of varying job proportions on resource utilization. The outcomes indicate that the proposed framework consistently achieves the highest resource utilization across all job type proportions. At a 10% job proportion, the proposed model achieves a utilization rate of 80.2%, outperforming Random (55.9%), Round Robin (55.7%), Earliest (53.5%), and JingDQN (39.8%). As the job proportion increases to 90%, the proposed model still maintains a utilization rate of 80.2%, still outperforming others. The highest utilization rate achieved by the proposed model is 80.8% at a 50% job proportion. Under varying job proportions, the DRL framework achieves higher VM resource utilization by dynamically allocating resources based on the specific demands of each job type. Related methods often underutilize resources when job compositions vary, leading to inefficiencies. The DRL model adjusts resource allocations in real time, ensuring that VMs are fully utilized regardless of the job composition.

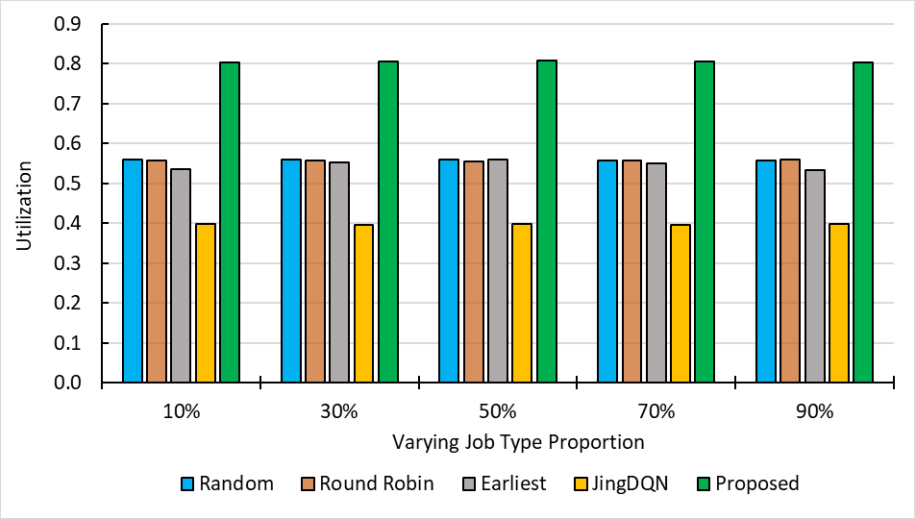


Figure 33. RESULTS 3C: The impact of varying job proportions on resource utilization.

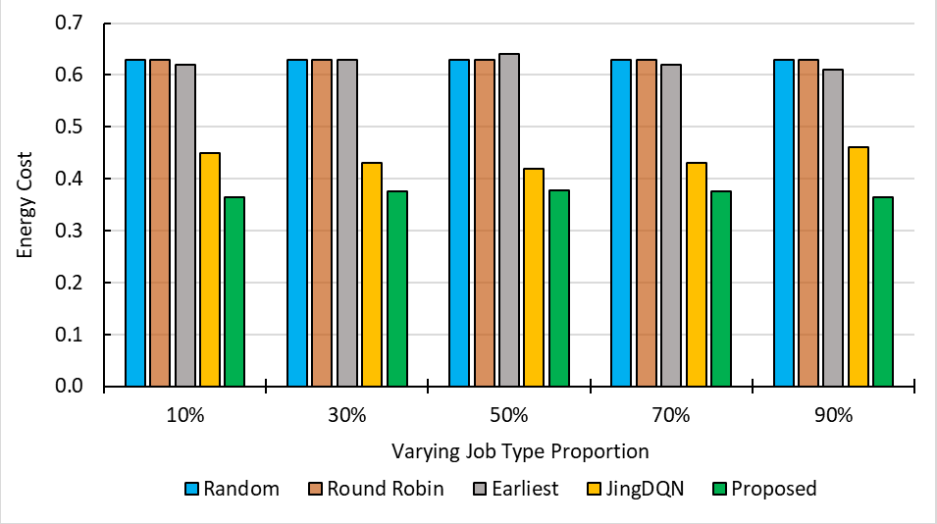


Figure 32. RESULTS 3D: The impact of varying job proportions on energy cost

RESULTS 3D: Varying job proportions on energy cost - Figure 33 shows the effect of varying job type proportions on energy cost. The outcomes indicate that the proposed framework consistently outperforms all other frameworks across different job type proportions in terms of energy cost. At a 10% job type proportion, the proposed model outperforms Random by 42%, Round Robin by 42%, Earliest by 41%, and JingDQN by 19%, resulting in an average outperformance of 36%. As the job type proportion increases to 90%, the proposed framework continues to maintain lower energy costs, outperforming Random by 42%, Round Robin by 42%, Earliest by 40%, and JingDQN by 21%, with an average outperformance of 36.25%.

The proposed DRL framework minimizes energy costs under varying job proportions by allocating resources efficiently based on the specific requirements of each job type. Related methods often over-provision resources for jobs, leading to energy waste. The DRL model

ensures that resources are distributed in a way that minimizes energy consumption, ensuring that jobs are processed efficiently while reducing energy costs.

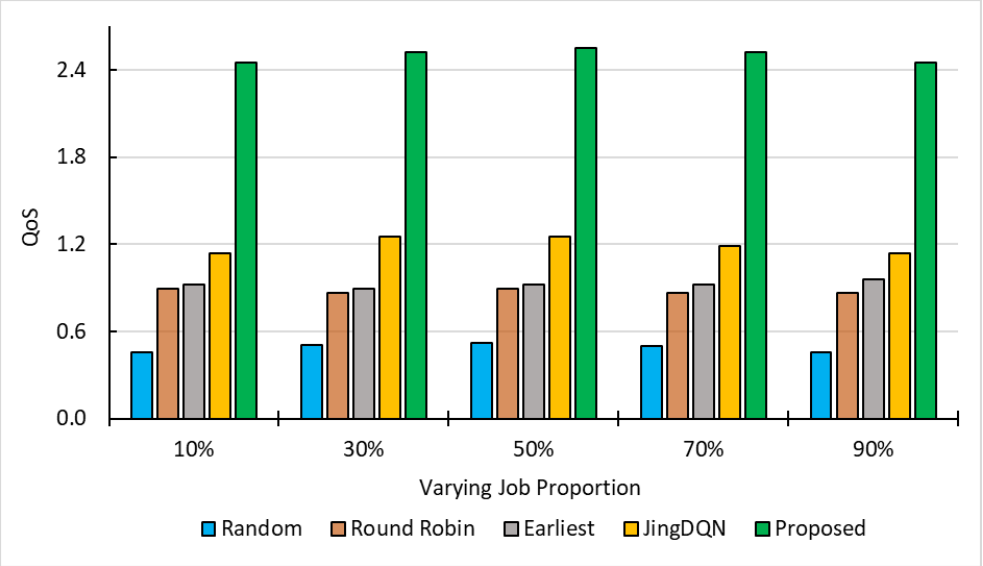


Figure 34. RESULTS 3E: The impact of varying job proportions on QoS

RESULTS 3E: The impact of varying job proportions on QoS - Figure 34 shows the impact of varying job proportions on QoS. The results demonstrate that the proposed framework consistently achieves the highest QoS across all job type proportions. At a 10% job proportion, the proposed model achieves a QoS score of 0.78.4%, outperforming Random (14.5%), Round Robin (28.6%), Earliest (29.6%), and JingDQN (36.4%). As the job proportion increases to 90%, the proposed model maintains a QoS score of 0.78.4%, still outperforming Random (14.5%), Round Robin (27.6%), Earliest (30.8%), and JingDQN (36.4%).

The highest QoS score achieved by the proposed model is 81.6% at a 50% job proportion, indicating its ability to maintain high service quality under different job compositions. The slight variation in QoS from 81.6% at 50% to 78.4% at 90% suggests that as the job proportions become more diverse, the complexity of maintaining perfect QoS increases. However, the proposed model's adaptability and real-time decision-making capabilities ensure that QoS remains high across all job compositions, outperforming the other frameworks consistently by effectively managing the different requirements of I/O-intensive and compute-intensive jobs. Related methods often fail to prioritize jobs appropriately, leading to missed deadlines and poor QoS. The DRL model learns to allocate resources dynamically based on job type, ensuring that jobs meet deadlines and maintain high QoS across varying job compositions. These results align with the research objective of optimizing QoS, ensuring that jobs are completed within their deadlines while meeting the expected service levels.

4.9 CHAPTER SUMMARY

When tested with different scenarios of varying job arrival rates, varying VM type proportions and varying job proportions, the DRL framework demonstrated significant performance in all the metrics significantly outperforming both JingDQN and baseline methods from related studies. The results show that the proposed DRL framework significantly outperformed related works demonstrating a 38% reduction in cloud energy consumption, a 98.4% jobs success rate, an improvement in QoS of 115%, a 66% reduction in response times, and a resource utilization of 90.7%. Additionally, the framework's scalability and adaptability to dynamic cloud conditions were validated through rigorous testing under various scenarios, confirming its robustness in handling fluctuating workloads and diverse job types.

CHAPTER 5: CONCLUSION AND RECOMMENDATION

5.1 CHAPTER INTRODUCTION

This chapter summarizes the key findings and contributions of the research on developing an intelligent energy efficiency framework in cloud computing using deep reinforcement learning (DRL). It highlights the achievement of research objectives, addresses the research questions, discusses encountered problems, and provides future recommendations.

5.2 RESULTS SUMMARY

The results show that the proposed DRL framework significantly outperformed related works demonstrating a 38% reduction in cloud energy consumption, a 98.4% jobs success rate, an improvement in QoS of 115%, a 66% reduction in response times, and a resource utilization of 90.7%.

When tested with varying job arrival rates, the proposed IDBS DRL framework achieved an average improvement of 67% across key performance metrics, significantly outperforming baseline methods in the related work. Specifically, it surpassed JingDQN, achieving a 98.4% success rate (8.1% higher) and reducing energy consumption by 13.5% at an arrival rate of 30. These insights demonstrate the proposed framework's superior capability in handling dynamic workloads while optimizing energy efficiency and maintaining high success rates.

In scenarios with varying VM type proportions, the DRL framework exhibited an average improvement of 75%, consistently outperforming other baselines from the related work. Compared to JingDQN, it reduced response times by 62%, improved success rates by 20%, and enhanced resource utilization by 88%, while cutting energy consumption by 22%. This highlights the framework’s adaptability and efficiency in diverse VM configurations, ensuring scalable and robust performance.

Under varying job proportions, the DRL framework demonstrated an average improvement of 70%, significantly outperforming both JingDQN and baseline methods from related studies. It reduced response times by 54% compared to JingDQN, boosted resource utilization by 103%, and lowered energy costs by 16%, with QoS scores 115% higher than JingDQN’s. These results show the framework’s effectiveness in optimizing resource management and energy usage,

making it highly suitable for dynamic and complex cloud environments.

5.3 OBJECTIVE ACHIEVEMENT

Objective 1: *To determine the limitations/gaps, most adopted ML models, datasets, evaluation metrics, and current related works performance from literature.* This objective aimed to address the critical issue of high energy consumption in cloud computing and the lack of comprehensive reviews on ML-based approaches. The research identified several limitations in current ML approaches, such as poor model optimization (15%), high computation costs (5%), and centralized approaches (5%). The most adopted ML models were Deep Reinforcement Learning (60%), Random Forest, and Support Vector Machines. The CloudSim dataset was the most frequently used (40%), and energy consumption was the most adopted evaluation metric (28%). These findings highlighted the need for a more adaptive and efficient framework, setting the stage for developing the proposed DRL-based model. The research questions for Objective 1 were:

Research Question 1 for Objective 1: *What are the limitations of the ML related works in minimizing cloud computing energy consumption?* Current ML-based frameworks suffer from poor adaptability to dynamic environments, high computational complexity, poor model optimization (15%), and limitations in handling real-time workloads, leading to inefficiencies in energy consumption reduction. These limitations contribute to suboptimal performance across many frameworks. (Extracted from: Literature Review - Section 2.2.3: The Most Prevalent Limitations with the Current Approaches)

Research Question 2 for Objective 1: *What are the most adopted ML models used in literature to minimize cloud energy consumption?* Deep Reinforcement Learning (DRL) is the most adopted model, making up 60% of related works due to its ability to handle high-dimensional state-action spaces. Other frequently used models include Random Forest (RF), K-Means, and Support Vector Machines (SVM). (Extracted from: Literature Review - Section 2.2.4: The Most Adopted ML Models from Literature)

Research Question 3 for Objective 1: *What are the most adopted ML datasets used in the related works to minimize cloud energy consumption?* The CloudSim simulation toolkit is the most adopted dataset (40%) because of its flexibility in simulating diverse cloud computing scenarios. It allows researchers to customize parameters like the number of hosts and virtual machines to evaluate energy efficiency in realistic cloud environments. (Extracted from: Literature Review - Section 2.2.5: The Most Adopted ML Datasets from Literature)

Research Question 4 for Objective 1: *What are the most adopted metrics in literature for evaluating ML frameworks cloud data center energy minimization?* Energy consumption (13%) is the most used evaluation metric, followed by accuracy (9%), power consumption (4%), execution time (4%), and resource utilization (4%). Energy consumption is a major concern as it directly impacts cloud service costs and sustainability. (Extracted from: Literature Review - Section 2.2.6: The Most Adopted ML Model Evaluation Metrics from Literature)

Research Question 5 for Objective 1: *How do related works perform based on the most adopted evaluation metrics?* Deep Reinforcement Learning (DRL) models show the best performance, achieving a maximum 67% reduction in energy consumption. Accuracy improvements reach 98%, with latency reduced by up to 68.7%, and resource utilization improved by 100% in some models.(Extracted from: Literature Review - Section 2.2.7: Related Works Performance)

Objective 2: *To develop a real-time DRL-based job scheduling model to minimize energy consumption in cloud computing while maintaining QoS.* This objective aimed to tackle the inefficiencies in resource allocation and the inability to handle dynamic and real-time cloud environments. The proposed model was mathematically formulated using state and action spaces, with job characteristics such as arrival rate, type, and deadlines. The DRL framework included phases such as decision-making, action space, and reward function to ensure adaptability. By dynamically allocating resources based on real-time workloads, the model achieved significant improvements outperforming related works as presented in the Results Section. The research questions for Objective 2 were:

Objective 2 Research Question 1: *How will the proposed cloud job scheduling model be mathematically formulated?* The proposed model represents job scheduling using state-action spaces, incorporating real-time job characteristics like arrival rates, types, and deadlines. The model dynamically adjusts resource allocation based on these inputs to minimize energy consumption. (Extracted from: Proposed Framework - Section 3.3: Cloud Scheduling Mathematical Model)

Objective 2 Research Question 2: *What and how will the real-time job/workload characteristics be generated?* Real-time job characteristics, such as arrival rates, types, and deadlines, are generated dynamically based on the specific workloads processed in the cloud. These characteristics are modeled within the state space of the scheduling framework. (Extracted from: Proposed Framework - Section 3.4: Application Workloads (Job Characteristics))

Objective 2 Research Question 3: *How will the proposed scheduling model minimize cloud energy consumption?* The model minimizes energy consumption by dynamically adjusting resource allocation to match real-time workloads. It uses reinforcement learning to schedule jobs efficiently based on job characteristics and system state, optimizing energy use while maintaining Quality of Service (QoS). (Extracted from: Proposed Framework - Section 3.5: The Proposed Intelligent DRL Based Job Scheduler)

Objective 2 Research Question 4: *How is QoS measured, and what determines whether it is maintained?* Quality of Service (QoS) is measured by metrics like the Deadline Satisfaction Ratio (DSR) and resource utilization. A job is considered successful if its deadline is met and resources are efficiently allocated, ensuring that QoS requirements are satisfied. (Extracted from: Proposed Framework - Section 3.6: QoS Success Condition)

Objective 2 Research Question 5: *How will the DRL phases map with the cloud job scheduling model in terms of the decision phase, state phase, action space, reward function, and training phase?* The DRL model includes decision-making, where jobs are assigned; state space, representing real-time workloads; action space, defining job scheduling; and a reward function that optimizes energy savings and QoS. Training is conducted through experience replay. (Extracted from: Proposed Framework - Section 3.7: Proposed DRL-based Scheduling Algorithm)

Objective 2 Research Question 6: *How will the proposed framework deal with the dynamic and unpredictable nature of the cloud?* The DRL-based framework dynamically adjusts job scheduling and resource allocation to respond to changes in workload and system state. This adaptability ensures efficient energy use and maintenance of QoS in unpredictable cloud environments. (Extracted from: Proposed Framework - Section 3.8: Dynamic and Unpredictable Cloud Adaptation)

Objective 3: *To evaluate the proposed framework by comparing it with current works in terms of energy consumption, model success rate, response time, QoS, and resource utilization.* This objective aimed to provide empirical evidence supporting the framework's effectiveness. The parameters used for evaluation included job arrival rates, types, and resource configurations. The results show that the proposed DRL framework significantly outperformed related works demonstrating a 38% reduction in cloud energy consumption, a 98.4% jobs success rate, an improvement in QoS of 115%, a 66% reduction in response times, and a resource utilization of 90.7%. The framework also showed excellent scalability and adaptability to changing cloud conditions, proving its robustness and efficiency in diverse scenarios. The research questions for Objective 3 were:

Objective 3 Research Question 1: *What parameters will be used and how will the experimentation be performed for evaluation?* Evaluation involves measuring energy consumption, success rate, response time, QoS (Deadline Satisfaction Ratio), and resource utilization under different scenarios, such as varying job arrival rates, virtual machine proportions, and job types. (Extracted from: Results and Discussion - Section 4.1: Experimental Setup and Evaluation Metrics)

Objective 3 Research Question 2: *Compared to related frameworks, how does the proposed DRL framework perform in terms of various metrics like energy consumption, model success rate, response time, QoS and resource utilization?* The results show that the proposed DRL framework significantly outperformed related works demonstrating a 38% reduction in cloud energy consumption, a 98.4% jobs success rate, an improvement in QoS of 115%, a 66% reduction in response times, and a resource utilization of 90.7%. Additionally, the framework's scalability and adaptability to dynamic cloud conditions were validated through rigorous testing under various scenarios, confirming its robustness in handling fluctuating workloads and diverse job types. These outcomes address the identified problems by significantly enhancing energy efficiency, job scheduling effectiveness, and overall system performance.

Objective 3 Research Question 3: *How does the proposed model perform in terms of scalability and the changing cloud conditions?* The proposed DRL-based model demonstrates strong scalability and adaptability to changing cloud conditions by dynamically adjusting resource allocation based on real-time workload changes. It maintains performance under varying job arrival rates, virtual machine proportions, and job types, ensuring efficient energy consumption and high QoS. *(Extracted from: Results and Discussion - Section 4.4: Scalability and Dynamic Cloud Conditions)*

5.4 PROBLEMS ENCOUNTERED AND SOLUTIONS

Several problems were encountered during the research. One major issue was the difficulty in acquiring paid research papers, which limited access to comprehensive literature and valuable insights from recent studies. To address this, alternative sources and open-access journals were utilized, although this sometimes limited the depth of the review. Another significant problem was code debugging and dependency issues with Python libraries. The complexity of DRL algorithms often led to challenging bugs that required extensive debugging efforts. Additionally, ensuring compatibility between various Python libraries and managing dependencies caused delays. These issues were mitigated through iterative testing and the use of virtual environments to manage dependencies. Handling large-scale simulations was another challenge. Simulating

real-time cloud environments requires substantial computational resources and careful attention to detail. This was addressed by optimizing the simulation code and using high-performance computing resources where available.

5.5 FUTURE RESEARCH RECOMMENDATIONS

Exploring the integration of renewable energy sources within the cloud infrastructure will help optimize energy consumption and promote sustainability. Further sections of the cloud can be optimized for energy improvement, such as implementing advanced cooling techniques like liquid cooling and AI-driven cooling management to significantly reduce power consumption. Optimizing networking through Software-Defined Networking (SDN) can enhance efficiency and reduce energy usage, improving data flow and reducing latency in job scheduling. Additionally, adopting edge computing and serverless architectures can enhance the efficiency of job scheduling by reducing latency and offloading processing from centralized data centers to edge locations, leading to faster job execution. Utilizing digital twin technology to create virtual models of data centers can provide real-time monitoring and optimization of job scheduling processes, allowing for more precise control over resource allocation and energy usage.

REFERENCES

[1] N. K.-C. C.-A. O. & Evolution and undefined 2018, "Cloud Computing-An Overview & Evolution," academia.edu, vol. 1, no. 3, pp. 2456–3307, 2018, [Online]. Available: https://www.academia.edu/download/57126486/CSEIT183131.pdf

[2] J. Surbiryala and C. Rong, "Cloud computing: History and overview," in Proceedings - 2019 3rd IEEE International Conference on Cloud and Fog Computing Technologies and Applications, Cloud Summit 2019, 2019. doi: 10.1109/CloudSummit47114.2019.00007.

[3] R. Shaw, E. Howley, and E. Barrett, "Applying Reinforcement Learning towards automating energy efficient virtual machine consolidation in cloud data centers," Inf Syst, vol. 107, p. 101722, Jul. 2022, doi: 10.1016/j.is.2021.101722.

[4] B. Wang, F. Liu, and W. Lin, "Energy-efficient VM scheduling based on deep reinforcement learning," Future Generation Computer Systems, vol. 125, pp. 616–628, Dec. 2021, doi: 10.1016/j.future.2021.07.023.

[5] N. Al Mudawi, N. Beloff, and M. White, "Issues and challenges: Cloud computing e-government in developing countries," International Journal of Advanced Computer Science and Applications, vol. 11, no. 4, 2020, doi: 10.14569/IJACSA.2020.0110402.

[6] I. Odun-Ayo, O. Ajayi, and C. Okereke, "Virtualization in cloud computing: Developments and trends," in Proceedings - 2017 International Conference on Next Generation Computing and Information Systems, ICNGCIS 2017, 2018. doi: 10.1109/ICNGCIS.2017.10.

[7] N. Jain and S. Choudhary, "Overview of virtualization in cloud computing," in 2016 Symposium on Colossal Data Analysis and Networking, CDAN 2016, 2016. doi: 10.1109/CDAN.2016.7570950.

[8] M. Abu-Alhaija, N. M. Turab, and A. R. Hamza, "Extensive study of cloud computing technologies, threats and solutions prospective," Computer Systems Science and Engineering, vol. 41, no. 1, pp. 225–240, 2022, doi: 10.32604/csse.2022.019547.

[9] U. 'Arshad, M. 'Aleem, G. 'Srivastava, and J. C.-W. 'Lin, "Utilizing power consumption and SLA violations using dynamic VM consolidation in cloud data centers," Elsevier, pp. 1–14, Jul. 2022.

[10] S.' 'Supreeth and K. 'Patil, "VM Scheduling for Efficient Dynamically Migrated Virtual Machines (VMS-EDMVM) in Cloud Computing Environment," KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS, vol. 16, no. 6, pp. 1892–1912, Jun. 2022.

[11] J. Zeng, D. Ding, K. Kang, H. M. Xie, and Q. Yin, "Adaptive DRL-Based Virtual Machine Consolidation in Energy-Efficient Cloud Data Center," IEEE Transactions on Parallel and Distributed Systems, vol. 33, no. 11, 2022, doi: 10.1109/TPDS.2022.3147851.

[12] J. 'Wang, H. 'Gu, J. 'Yu, Y. 'Song, X. 'He, and Y. 'Song, "Research on virtual machine consolidation strategy based on combined prediction and energy-aware in cloud computing platform," Journal of Cloud Computing: Advances, Systems and Applications, pp. 1–18, 2022.

[13] A. K. 'Singh, S. R. 'Swain, D. 'Saxena, and C.-N. 'Lee, "A Bio-Inspired Virtual Machine Placement Toward Sustainable Cloud Resource Management," IEEE Systems Journal (Early Access), pp. 1–12, Mar. 2023.

[14] M. H. Sayadnavard, A. Toroghi Haghighat, and A. M. Rahmani, "A multi-objective approach for energy-efficient and reliable dynamic VM consolidation in cloud data centers," Engineering Science and Technology, an International Journal, vol. 26, 2022, doi: 10.1016/j.jestch.2021.04.014.

[15] M. 'Imran, M. 'Ibrahim, M. S. U. 'Din, M. A. U. 'Rehman, and B. S. 'Kim, "Live virtual machine migration: A survey, research challenges, and future directions," Elsevier, pp. 1–18, Aug. 2022.

[16] S. Talwani, K. Alhazmi, J. Singla, H. J. Alyamani, and A. K. Bashir, "Allocation and migration of virtual

machines using machine learning," Computers, Materials and Continua, vol. 70, no. 2, 2022, doi: 10.32604/cmc.2022.020473.

[17] C. H. Tran, T. K. Bui, and T. V. Pham, "Virtual machine migration policy for multi-tier application in cloud computing based on Q-learning algorithm," Computing, vol. 104, no. 6, 2022, doi: 10.1007/s00607-021-01047-0.

[18] H. Li, J. Liu, and Q. Zhou, "Research on energy-saving virtual machine migration algorithm for green data center," IET Control Theory and Applications, 2022, doi: 10.1049/cth2.12401.

[19] B. B. Naik, D. Singh, and A. B. Samaddar, "Multi-objective Virtual Machine Selection in Cloud Data Centers Using Optimized Scheduling," Wirel Pers Commun, vol. 116, no. 3, 2021, doi: 10.1007/s11277-020-07807-z.

[20] X.-B. Cai, Y.-X. Ji, and K. Han, "Energy Efficiency Optimizing Based on Characteristics of Machine Learning in Cloud Computing," ITM Web of Conferences, vol. 12, p. 03047, Sep. 2017, doi: 10.1051/itmconf/20171203047.

[21] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," Electronic Markets, vol. 31, no. 3, 2021, doi: 10.1007/s12525-021-00475-2.

[22] J. Schmitt, J. Bönig, T. Borggräfe, G. Beitinger, and J. Deuse, "Predictive model-based quality inspection using Machine Learning and Edge Cloud Computing," Advanced Engineering Informatics, vol. 45, 2020, doi: 10.1016/j.aei.2020.101101.

[23] A. Trisal and D. Mandloi, "MACHINE LEARNING: AN OVERVIEW," International Journal of Research -GRANTHAALAYAH, vol. 9, no. 7, 2021, doi: 10.29121/granthaalayah. v 9.i7.2021.4120.

[24] D. Ding, X. Fan, Y. Zhao, K. Kang, Q. Yin, and J. Zeng, "Q-learning based dynamic task scheduling for energy-efficient cloud computing," Future Generation Computer Systems, vol. 108, pp. 361–371, Jul. 2020, doi: 10.1016/j.future.2020.02.018.

[25] W. Zhong, Y. Zhuang, J. Sun, and J. Gu, "The cloud computing load forecasting algorithm based on wavelet support vector machine," in Proceedings of the Australasian Computer Science Week Multiconference, New York, NY, USA: ACM, Jan. 2017, pp. 1–5. doi: 10.1145/3014812.3014852.

[26] A. Jayanetti, S. Halgamuge, and R. Buyya, "Deep reinforcement learning for energy and time optimized scheduling of precedence-constrained tasks in edge–cloud computing environments," Future Generation Computer Systems, vol. 137, pp. 14–30, Dec. 2022, doi: 10.1016/j.future.2022.06.012.

[27] N. Liu et al., "A Hierarchical Framework of Cloud Resource Allocation and Power Management Using Deep Reinforcement Learning," in 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), IEEE, Jun. 2017, pp. 372–382. doi: 10.1109/ICDCS.2017.123.

[28] A. S. McGough, M. Forshaw, J. Brennan, N. Al Moubayed, and S. Bonner, "Using Machine Learning to reduce the energy wasted in Volunteer Computing Environments," Oct. 2018, [Online]. Available: http://arxiv.org/abs/1810.08675

[29] H. Li, R. Cai, N. Liu, X. Lin, and Y. Wang, "Deep reinforcement learning: Algorithm, applications, and ultra-low-power implementation," Nano Commun Netw, vol. 16, pp. 81–90, Jun. 2018, doi: 10.1016/j.nancom.2018.02.003.

[30] J. Zhang, N. Xie, X. Zhang, K. Yue, W. Li, and D. Kumar, "Machine learning based resource allocation of cloud computing in auction," Computers, Materials and Continua, vol. 56, no. 1, 2018, doi: 10.3970/cmc.2018.03728.

[31] Y. Jararweh, M. B. Issa, M. Daraghmeh, M. Al-Ayyoub, and M. A. Alsmirat, "Energy efficient dynamic resource management in cloud computing based on logistic regression model and median absolute deviation," Sustainable Computing: Informatics and Systems, vol. 19, pp. 262–274, Sep. 2018, doi: 10.1016/j.suscom.2018.07.005.

[32] R. Moreno-Vozmediano, R. S. Montero, E. Huedo, and I. M. Llorente, "Efficient resource provisioning for elastic Cloud services based on machine learning techniques," Journal of Cloud Computing, vol. 8, no. 1, 2019, doi: 10.1186/s13677-019-0128-9.

[33] N. R. Rajalakshmi, G. Arulkumaran, and J. Santhosh, "Virtual machine consolidation for performance and energy efficient cloud data center using reinforcement learning," Int J Eng Adv Technol, vol. 8, no.

3 Special Issue, 2019.

[34] X. Sui, D. Liu, L. Li, H. Wang, and H. Yang, “Virtual machine scheduling strategy based on machine learning algorithms for load balancing,” EURASIP J Wirel Commun Netw, vol. 2019, no. 1, p. 160, Dec. 2019, doi: 10.1186/s13638-019-1454-9.

[35] T. Thein, M. M. Myo, S. Parvin, and A. Gawanmeh, “Reinforcement learning based methodology for energy-efficient resource allocation in cloud data centers,” Journal of King Saud University - Computer and Information Sciences, vol. 32, no. 10, 2020, doi: 10.1016/j.jksuci.2018.11.005.

[36] Z. Tong, H. Chen, X. Deng, K. Li, and K. Li, “A scheduling scheme in the cloud computing environment using deep Q-learning,” Inf Sci (N Y), vol. 512, pp. 1170–1191, Feb. 2020, doi: 10.1016/j.ins.2019.10.035.

[37] A. Asghari, M. K. Sohrabi, and F. Yaghmaee, “A cloud resource management framework for multiple online scientific workflows using cooperative reinforcement learning agents,” Computer Networks, vol. 179, p. 107340, Oct. 2020, doi: 10.1016/j.comnet.2020.107340.

[38] M. H. S, S. Kumar T, S. M. F. D. S. Mustapha, P. Gupta, and R. P. Tripathi, “Hybrid Approach for Resource Allocation in Cloud Infrastructure Using Random Forest and Genetic Algorithm,” Sci Program, vol. 2021, pp. 1–10, Oct. 2021, doi: 10.1155/2021/4924708.

[39] J. Yan, J. Xiao, and X. Hong, “Dueling-DDQN Based Virtual Machine Placement Algorithm for Cloud Computing Systems,” in 2021 IEEE/CIC International Conference on Communications in China (ICCC), IEEE, Jul. 2021, pp. 294–299. doi: 10.1109/ICCC52777.2021.9580393.

[40] L. Caviglione, M. Gaggero, M. Paolucci, and R. Ronco, “Deep reinforcement learning for multi-objective placement of virtual machines in cloud datacenters,” Soft comput, vol. 25, no. 19, pp. 12569–12588, Oct. 2021, doi: 10.1007/s00500-020-05462-x.

[41] G. Chen, J. Qi, Y. Sun, X. Hu, Z. Dong, and Y. Sun, “A collaborative scheduling method for cloud computing heterogeneous workflows based on deep reinforcement learning,” Future Generation Computer Systems, vol. 141, pp. 284–297, Apr. 2023, doi: 10.1016/J.FUTURE.2022.11.032.

[42] T. Cui, R. Yang, C. Fang, and S. Yu, “Deep Reinforcement Learning-Based Resource Allocation for Content Distribution in IoT-Edge-Cloud Computing Environments,” Symmetry (Basel), vol. 15, no. 1, 2023, doi: 10.3390/sym15010217.

[43] T. Sen and H. Shen, “Machine learning based timeliness-guaranteed and energy-efficient task assignment in edge computing systems,” in 2019 IEEE 3rd International Conference on Fog and Edge Computing, ICFEC 2019 - Proceedings, 2019. doi: 10.1109/CFEC.2019.8733153.

[44] Y. Kumar, S. Kaul, and Y. C. Hu, “Machine learning for energy-resource allocation, workflow scheduling and live migration in cloud computing: State-of-the-art survey,” Sustainable Computing: Informatics and Systems, vol. 36, 2022, doi: 10.1016/j.suscom.2022.100780.

[45] M. Demirci, “A survey of machine learning applications for energy-efficient resource management in cloud computing environments,” in Proceedings - 2015 IEEE 14th International Conference on Machine Learning and Applications, ICMLA 2015, 2016. doi: 10.1109/ICMLA.2015.205.

[46] T. Khan, W. Tian, G. Zhou, S. Ilager, M. Gong, and R. Buyya, “Machine learning (ML)-centric resource management in cloud computing: A review and future directions,” Journal of Network and Computer Applications, vol. 204, p. 103405, Aug. 2022, doi: 10.1016/J.JNCA.2022.103405.

[47] D. Soni and N. Kumar, “Machine learning techniques in emerging cloud computing integrated paradigms: A survey and taxonomy,” Journal of Network and Computer Applications, vol. 205, p. 103419, Sep. 2022, doi: 10.1016/J.JNCA.2022.103419.

[48] F. Cheng, Y. Huang, B. Tanpure, P. Sawalani, L. Cheng, and C. Liu, “Cost-aware job scheduling for cloud instances using deep reinforcement learning,” Cluster Comput, vol. 25, no. 1, 2022, doi: 10.1007/s10586-021-03436-8.

[49] J. Yan et al., “Energy-aware systems for real-time job scheduling in cloud data centers: A deep reinforcement learning approach,” Computers and Electrical Engineering, vol. 99, 2022, doi: 10.1016/j.compeleceng.2022.107688.

[50] M. T. Islam, S. Karunasekera, and R. Buyya, “Performance and Cost-Efficient Spark Job Scheduling Based on Deep Reinforcement Learning in Cloud Computing Environments,” IEEE Transactions on

Parallel and Distributed Systems, vol. 33, no. 7, pp. 1695–1710, Jul. 2022, doi: 10.1109/TPDS.2021.3124670.

[51] A. Kharitonov, A. Nahhas, H. Müller, and K. Turowski, "Towards hybrid-cloud infrastructure composition for SAP systems landscapes in smart manufacturing," in Procedia Computer Science, Elsevier B.V., 2024, pp. 2376–2385. doi: 10.1016/j.procs.2024.02.056.

[52] T. Dancheva, U. Alonso, and M. Barton, "Cloud benchmarking and performance analysis of an HPC application in Amazon EC2," Cluster Comput, vol. 27, no. 2, 2024, doi: 10.1007/s10586-023-04060-4.

[53] R. Saini and R. Behl, "An Introduction to AWS—EC2 (Elastic Compute Cloud)," in Proceedings of the International Conference on Research in Management & Technovation 2020, 2020. doi: 10.15439/2020km4.

[54] A. V. Karthick, E. Ramaraj, and R. G. Subramanian, "An efficient multi queue job scheduling for cloud computing," in Proceedings - 2014 World Congress on Computing and Communication Technologies, WCCCT 2014, 2014. doi: 10.1109/WCCCT.2014.8.

[55] Y. Gu, F. Cheng, L. Yang, J. Xu, X. Chen, and L. Cheng, "Cost-aware cloud workflow scheduling using DRL and simulated annealing," Digital Communications and Networks, 2024, doi: 10.1016/j.dcan.2023.12.009.

[56] Q. Li and X. Peng, "Job Scheduling and Simulation in Cloud Based on Deep Reinforcement Learning," Xitong Fangzhen Xuebao / Journal of System Simulation, vol. 34, no. 2, pp. 258–268, Feb. 2022, doi: 10.16182/j.issn1004731x.joss.21-0337.

[57] G. Zhou, W. Tian, R. Buyya, R. Xue, and L. Song, "Deep reinforcement learning-based methods for resource scheduling in cloud computing: a review and future directions," Artif Intell Rev, vol. 57, no. 5, May 2024, doi: 10.1007/s10462-024-10756-9.

[58] G. Rjoub, J. Bentahar, O. Abdel Wahab, and A. Saleh Bataineh, "Deep and reinforcement learning for automated task scheduling in large-scale cloud computing systems," in Concurrency and Computation: Practice and Experience, 2021. doi: 10.1002/cpe.5919.

[59] Y. Huang et al., "Deep Adversarial Imitation Reinforcement Learning for QoS-Aware Cloud Job Scheduling," IEEE Syst J, vol. 16, no. 3, 2022, doi: 10.1109/JSYST.2021.3122126.

[60] L. Cheng, Y. Wang, F. Cheng, C. Liu, Z. Zhao, and Y. Wang, "A Deep Reinforcement Learning-Based Preemptive Approach for Cost-Aware Cloud Job Scheduling," IEEE Transactions on Sustainable Computing, 2023, doi: 10.1109/TSUSC.2023.3303898.

[61] H. Hou, S. N. Agos Jawaddi, and A. Ismail, "Energy efficient task scheduling based on deep reinforcement learning in cloud environment: A specialized review," 2024. doi: 10.1016/j.future.2023.10.002.

[62] G. Zhou, R. Wen, W. Tian, and R. Buyya, "Deep reinforcement learning-based algorithms selectors for the resource scheduling in hierarchical Cloud computing," Journal of Network and Computer Applications, vol. 208, p. 103520, Dec. 2022, doi: 10.1016/J.JNCA.2022.103520.

[63] S. Mangalampalli, G. R. Karri, M. Kumar, O. I. Khalaf, C. A. T. Romero, and G. M. A. Sahib, "DRLBTSA: Deep reinforcement learning based task-scheduling algorithm in cloud computing," Multimed Tools Appl, vol. 83, no. 3, 2024, doi: 10.1007/s11042-023-16008-2.

[64] F. Li and B. Hu, "DeepJS: Job scheduling based on deep reinforcement learning in cloud data center," in ACM International Conference Proceeding Series, 2019. doi: 10.1145/3335484.3335513.

[65] L. Cheng et al., "Cost-aware real-time job scheduling for hybrid cloud using deep reinforcement learning," Neural Comput Appl, vol. 34, no. 21, pp. 18579–18593, Nov. 2022, doi: 10.1007/s00521-022-07477-x.

[66] Y. Cheng, Z. Cao, X. Zhang, Q. Cao, and D. Zhang, "Multi objective dynamic task scheduling optimization algorithm based on deep reinforcement learning," Journal of Supercomputing, vol. 80, no. 5, 2024, doi: 10.1007/s11227-023-05714-1.

[67] L. Buşoniu, T. de Bruin, D. Tolić, J. Kober, and I. Palunko, "Reinforcement learning for control: Performance, stability, and deep approximators," 2018. doi: 10.1016/j.arcontrol.2018.09.005.

[68] Y. Ran, H. Hu, X. Zhou, and Y. Wen, “DeepEE: Joint optimization of job scheduling and cooling control for data center energy efficiency using deep reinforcement learning,” in Proceedings - International Conference on Distributed Computing Systems, 2019. doi: 10.1109/ICDCS.2019.00070.

[69] K. Kang, D. Ding, H. Xie, Q. Yin, and J. Zeng, “Adaptive DRL-Based Task Scheduling for Energy-Efficient Cloud Computing,” IEEE Transactions on Network and Service Management, vol. 19, no. 4, 2022, doi: 10.1109/TNSM.2021.3137926.

[70] A. Sheibanirad and M. Ashtiani, “A self-adaptive approach to job scheduling in cloud computing environments,” Scientia Iranica, vol. 31, no. 5, pp. 373–387, Mar. 2024, doi: 10.24200/sci.2023.59168.6090.

[71] Y. Wang, S. Dong, and W. Fan, “Task Scheduling Mechanism Based on Reinforcement Learning in Cloud Computing,” Mathematics, vol. 11, no. 15, 2023, doi: 10.3390/math11153364.

[72] Q. Gao and A. M. Schweidtmann, “Deep reinforcement learning for process design: Review and perspective,” 2024. doi: 10.1016/j.coche.2024.101012.