

https://dspace.org/

Cell-Free Massive MIMO NOMA Network with Machine Learning aided User Association and Power Allocation

Zungunde, James Itai

2025-10
University of the Witwatersrand, Johannesburg
https://hdl.handle.net/10539/50001
Zungunde, James Itai. (2025). Cell-Free Massive MIMO NOMA Network with Machine Learning aided User Association and Power Allocation. [Master's dissertation, University of the Witwatersrand, Johannesburg]. WIReDSpace. https://hdl.handle.net/10539/50001

Downloaded from a DSpace repository. https://dspace.org/

University of the Witwatersrand

Masters Dissertation

Cell-Free Massive MIMO NOMA Network with Machine Learning aided User Association and Power Allocation

Author:
James Itai Zungunde (1712958)
Supervised by:
Professor Fambirai Takawira and Doctor Chabalala Chabalala

A Dissertation submitted in fulfilment of the requirements for the degree of Master of Science in Engineering

in the

School of Electrical and Information Engineering

October 2025



Declaration

I, **James Itai Zungunde**, declare that this Dissertation titled, " **Cell-Free Massive MIMO NOMA Network with Machine Learning aided User Association and Power Allocation**" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this dissertation is entirely my own work.
- I have acknowledged all main sources of help.

Signed: 

Date: 01/10/2025

Abstract

This dissertation explores the applications of Machine Learning (ML) in Cell-Free Massive Multiple-Inputs-Multiple-Output (CF-mMIMO) Non-Orthogonal-Multiple-Access (NOMA) Network that incorporates Unmanned Ariel Vehicles (UAVs) as mobile Access Points (APs) and Simultaneous Wireless Information and Power Transfer (SWIPT) for energy harvesting User Equipment (UE). Firstly, we propose an objective function to maximise Downlink (DL) Sum Spectral Efficient (SSE) whilst considering UE Spectral Efficiency (SE) requirements for both downlink and uplink transmission. To maximise the objective function, we propose using a Deep Reinforcement Learning (DRL) algorithm, namely, a Deep Deterministic Policy Gradient (DDPG), to do joint power allocation and user association by using a hybrid action space. We find the optimal time allocation for energy harvesting for the frame structure through a heuristic approach. We show the DDPG model performs better than random power allocation and user association schemes. Secondly, we improve the performance of our DDPG model by using various bootstrapping methods described in literature. We also introduce a novel bootstrapped DDPG, we call a Multi-Head-Elastic-Step (MH-ES) DDPG. We compare the various bootstrapped DDPG models to the base DDPG and show that bootstrapping can improve the training performance and inference performance of the base DDPG. Lastly, we conclude our work by exploring future work to extend on our research.

I dedicate this to Coletta Valeria Kanyekanye

This dissertation is dedicated to my beloved late aunt, Coletta Valeria Kanyekanye. Thank you for believing in me and motivating me all those years. Thank you for your unconditional love that continues to inspire me to this day, I will always carry you in my heart.

Acknowledgements

I would like to extend my sincere gratitude to the following:

- My supervisor - Professor Takawira, it is with immense gratitude that I sincerely thank you for your excellent guidance. Your breadth of knowledge never ceases to amaze me and thank you for imparting some of that knowledge to me and for your priceless dedication to your students, you were there to guide me even beyond the scope of duty.
- My supervisor - Dr Chabalala, it gives me great pleasure in acknowledging the support and help you gave me throughout my studies. I can never thank you enough for your attention to detail which was instrumental to the quality of this dissertation, your expert advice and encouragement will be cherished always.
- Sentech - I would like to extend my deepest appreciation to the Sentech for sponsoring my studies.
- Wits School of Electrical and Information Engineering (EIE) - I am also grateful to the entire team of the school of EIE, thank you for providing a conducive environment and support throughout my studies.
- Wits University - Special thanks goes to Wits University for awarding me Post Graduate Academic merit award.
- To my fellow student and friend - Francis Phiri, thank you for experiencing this academic journey with me. Thank you for the many hours we spent discussing and debating our ideas, you were a source of motivation and support.
- To my mother, Dr Matildah Zungunde and my brother, Sean Zungunde, I thank you for the continued motivation and support throughout my Master's journey.

Contents

Declaration i

Abstract ii

Dedication iii

Acknowledgements iv

Contents v

List of Figures viii

List of Tables x

Abbreviations xi

Publications xv

1 Introduction 1
- 1.1 Background . . . 1
- 1.2 Problem Statement . . . 2
- 1.3 Research Questions . . . 5
- 1.4 Research Methodology . . . 7
- 1.5 Research Contributions . . . 7
- 1.6 Dissertation Organisation . . . 9

2 Background and Literature Review 10
- 2.1 Introduction . . . 10
- 2.2 Cell Free Massive MIMO . . . 10
- 2.3 Work related to Cell Massive MIMO . . . 13
- 2.4 Non-Orthogonal-Multiple-Access . . . 14
  - 2.4.1 Downlink PD-NOMA . . . 15
  - 2.4.2 Uplink PD-NOMA . . . 17
  - 2.4.3 Cell Free Massive MIMO with NOMA . . . 18
- 2.5 Simultaneous Wireless Information and Power Transfer . . . 19
  - 2.5.1 Time Splitting SWIPT . . . 20
  - 2.5.2 Power Splitting SWIPT . . . 21
  - 2.5.3 Cell Free Massive MIMO with SWIPT . . . 22
- 2.6 Unmanned Ariel Vehicles . . . 22

2.6.1 Cell Massive MIMO with UAVs . . . 23
2.7 Machine Learning . . . 24
2.7.1 Supervised Learning . . . 25
2.7.2 Unsupervised Learning . . . 26
2.7.3 Semi-supervised Learning . . . 27
2.7.4 Deep Reinforcement Learning . . . 27
2.7.5 Different Deep Reinforcement Learning Models . . . 29
2.7.5.1 Model Based and Model-Free . . . 30
2.7.5.2 On-Policy and off-Policy . . . 31
2.7.5.3 Value Function Based Methods . . . 31
2.7.5.4 Policy Function Based Methods . . . 32
2.7.5.5 Actor-Critic . . . 33
2.7.6 Cell Free Massive MIMO with Machine Learning . . . 34
2.8 Chapter Summary . . . 37

3 Deep Reinforcement Learning aided User Association and Power Allocation 40
3.1 Introduction . . . 40
3.2 System Model . . . 41
3.2.1 Channel Model . . . 46
3.2.1.1 Channel Estimation . . . 47
3.2.1.2 Uplink Training Phase . . . 48
3.2.2 Downlink Transmission . . . 49
3.2.3 Energy Harvesting and Energy Consumption Model . . . 52
3.2.3.1 Energy Consumption during Pilot Training . . . 52
3.2.3.2 Energy Consumption during Downlink Transmission 53
3.2.3.3 Energy Consumption during Energy Harvesting . . 53
3.2.3.4 Energy Consumption during Uplink Transmission . 54
3.2.4 Uplink Transmission . . . 55
3.2.5 Downlink Transmission Performance . . . 56
3.2.6 Uplink Transmission Performance . . . 57
3.2.7 NOMA Clustering . . . 58
3.3 Problem Formulation . . . 61
3.4 Deep Deterministic Policy Gradient . . . 62
3.4.1 Q-Learning . . . 64
3.4.2 Target Network . . . 65
3.4.2.1 State . . . 66
3.4.2.2 Action . . . 66
3.4.2.3 Actor and Critic Network Architectures . . . 68
3.4.2.4 Reward . . . 69
3.5 Complexity Analysis . . . 70
3.6 Simulation and Training Parameters . . . 71
3.7 Results and Analysis . . . 73
3.7.1 Hyper-parameter Tuning on Training Performance . . . 74

3.7.2 Impact of Energy Harvesting Time Allocation 81
3.7.3 DDPG Inference Performance 92
3.8 Conclusion 94

4 A Comparative Analysis of Bootstrapped DDPGs 96
4.1 Introduction 96
4.2 Background and Literature Review 97
4.2.1 Bootstrapping and Thompson Sampling 97
4.2.2 N-Step DDPG 100
4.2.3 Elastic Step DDPG 103
4.2.4 Multi-Head DDPG 106
4.2.5 Multi-Head Elastic-Step DDPG 107
4.3 Complexity Analysis 108
4.4 Results and Analysis 110
4.5 Conclusion 119

5 Conclusion and Future Work 121
5.1 Conclusion 121
5.2 Future Work 123

List of Figures

2.1 A basic CF-mMIMO architecture. 11
2.2 A cluster centric CF-mMIMO system. 12
2.3 A user centric CF-mMIMO system. 13
2.4 Downlink NOMA transmission. 17
2.5 An uplink NOMA transmission. 18
2.6 Two architectures for SWIPT. 20
2.7 Frame structure for the TS-SWIPT protocol. 21
2.8 Frame structure for the PS-SWIPT protocol. 21
2.9 Supervised learning architecture. 26
2.10 Unsupervised learning architecture. 27
2.11 Semi-supervised learning architecture. 27
2.12 DNN architecture. 28
2.13 DRL architecture. 29

3.1 System model of CF-mMIMO with NOMA clusters. 42
3.2 Time coherence interval frame structure. 43
3.3 Post processing actions from the actor network. 67
3.4 Actor and critic network structure. 68
3.5 The effect of learning rate on DDPG training performance. 74
3.6 The effect of soft update size on training performance. 75
3.7 The effect of discount value on training performance 77
3.8 DDPG training performance after hyper-parameter tuning. 78
3.9 DDPG training performance after hyper-parameter tuning. 80
3.10 Impact for energy harvesting time on policy performance. 81
3.11 Policy performance for the number of UE that meet minimum required DL SE. 83
3.12 Policy performance for the number of UE that meet minimum required UL SE. 84
3.13 DL SSE performance for different policies. 85
3.14 CDF and PDF of DL SSE. 86
3.15 CDF and PDF UL SSE. 87
3.16 PMF of the number UE that meet minimum required DL SE. . . . 88
3.17 PMF of the number UE that meet minimum required UL SE. . . . 89
3.18 The impact of UAV operating height on trained DDPG model performance. 90
3.19 The impact of UAV operating velocity on trained DDPG model performance. 91
3.20 Achievable DL SSE performance of DDPG model. 92

3.21 Achievable number of UE that meet the minimum required DL SE. 93
3.22 Achievable number of UE that meet the minimum required UL SE. 93

4.1 DDPG variants training performance. 113
4.2 Comparison of DDPG variants training times. 114
4.3 Comparison of the achievable DL SSE performance of DDPG variants. 115
4.4 Comparison of the achievable number of UE that meet the minimum required DL SE. 116
4.5 Comparison of the achievable number of UE that meet the minimum required UL SE. 116
4.6 Comparison of the DDPG variants' reward during inferencing. . . . 117
4.7 Comparison of DDPG variants' inferencing speeds. 118

List of Tables

1.1 Literature review comparison with proposed research. 4

3.1 Table of notation. 43
3.2 System model and simulation parameters 72
3.3 Optimal energy harvesting time for different AP transmit powers . 83

4.1 HDBSCAN Hyper-parameters 104
4.2 The number of total FLOPS required during inferencing 110
4.3 System model and simulation parameters 111

Abbreviations

1G	First Generation
2G	Second Generation
3D	Three Dimensional
3G	Third Generation
4G	Fourth Generation
5G	Fifth Generation
6G	Sixth Generation
A3C	Asynchronous Advantage Actor-Critic
AC	Actor-Critic
AI	Artificial Intelligence
ANN	Artificial Neural Network
AP	Access Point
BM	Bisection method
BS	Base Station
CAPS	Cluster Based AP Selection
CD-NOMA	Code Domain NOMA
CDF	Cumulative Density Function
CDMA	Code Division Multiple access
CF-mMIMO	Cell-Free Massive Multiple-Input-Multiple-Output
CPU	Central Processing Unit
CRA	Channel Resource Allocation
CSI	Channel State Information
CSUN	Cognitive Satellite UAV Network
D4PG	Distributed Distributional Deterministic Policy Gradients
DCCN	Double Cascade Correlation Network

DDPG	Deep Deterministic Policy Gradient
DDQN	Deep Double Q Network
DNN	Deep Neural Network
DL	Deep Learning
DP	Dynamic Programming
DRL	Deep Reinforcement Learning
ES	Elastic-Step
EE	Energy Efficiency
EH	Energy Harvesting
FDMA	Frequency Division Multiple Access
fpZF	full pilot Zero Forcing
GNN	Graph Neural Network
GPU	Graphics Processing Unit
HDBSCAN	Hierarchical Density Based Spatial Clustering of Applications with Noise
HAPS	High-Altitude Platform Station
IoT	Internet of Things
KTH	Kungliga Tekniska Hogsklan Royal Institute of technology
LDM	Layered-Division Multiplexing
LoS	Line of Sight
LR	Learning Rate
SARL	Single Agent Reinforcement Learning
SE	Spectral Efficiency
SOUP	Self-Adaptive Double Bootstrapping
SSE	Sum Spectral Efficient
SWIFT	Simultaneous Wireless Information and Power Transfer
MARL	Multi Agent Reinforcement Learning
MBRL	Model Based Reinforcement Learning
MC	Monte Carlo
MDP	Markov Decision Process
MFRL	Model-Free Reinforcement Learning
MH	Muti-Head

ML	Machine Learning
MMSE	Minimum Mean Square Estimation
mmWave	Millimetre Wave
NAFD	Network-Assisted Full Duplex
NLoS	Non-Line-of-Sight
NN	Neural Networks
NOMA	Non-Orthogonal-Multiple-Access
NS	N-Step (NS)
OFDMA	Orthogonal Frequency Division Multiple Access
OMA	Orthogonal Multiple Access
PA	Power Allocation
PCA	Principal Component Analysis
PDF	Probability Density Function
PD-NOMA	Power Domain NOMA
PMF	Probability Mass Function
PPP	Poisson Point Process
PS	Power Splitting
PZF	Partial Zero-Forcing
QNN	Quantum Neural Networks
QoS	Quality of Service
RF	Radio Frequency
RF-EH	Radio Frequency Energy Harvesting
RIS	Reflective Intelligence Surfaces
Relu	Rectified Linear Unit
RL	Reinforcement Learning
RV	Random Variable
SIC	Successive Interference Cancellation
TD	Temporal Difference
TD3	Twin-delayed Deep Deterministic Policy Gradient
TDD	Time Division Duplex
TDMA	Time Division Multiple Access

TS	Time Splitting
UA	User Association
UAV	Unmanned Ariel Vehicles
UC	User centric
UDN	Ultra-Dense Networks
UE	User Equipment
UL	Uplink
URLLC	Ultra Reliable Low Latency Communications
WIT	Wireless Information Transfer
WPT	Wireless Power Transfer

Publications

J. Zungunde, F. Takawira and C. S. Chabalala, "Cell-Free Massive MIMO NOMA Network with Deep Reinforcement Learning Aided User Association and Power Allocation," 2025 IEEE 3rd Wireless Africa Conference (WAC), Pretoria, South Africa, 2025, pp. 1-6

CHAPTER 1

Introduction

1.1 Background

Modern telecommunications systems play an important role in society. There is a growing need for faster, more reliable and efficient telecommunications systems to support the ever-growing needs of the businesses and consumers in different countries around the world. Although Fifth Generation (5G) mobile network has not been fully adopted in every country, it is rapidly being deployed across the globe with many first world countries having already adapted 5G. A shift in focus is now towards the development and standardisation of the Sixth Generation (6G) mobile network, with 2030 being scheduled by the International Telecommunications Union Radiocommunication (ITU-R) as the deadline for specifications of standards for 6G networks [1]. One of the promising concepts currently being investigated is Cell-Free Massive Multiple-Input-Multiple-Output (CF-mMIMO), that promises to eliminate network issues that affect edge users while increasing Spectral Efficiency (SE). Another technique that promises to increase SE is Non-Orthogonal-Multiple-Access (NOMA). However, maximising SE is not the only goal for next generation mobile networks, with the telecommunications industry being a large energy consumer, there is an effort to increase Energy Efficiency (EE) in telecommunication systems. According to joint research conducted by Ericsson, and Kungliga Tekniska Hogsklan (KTH) Royal Institute of technology, the telecommunications industry is responsible for approximately one percent (242TWh) of the world's total energy consumption [2]. That roughly translates to 169 metric tons of carbon dioxide (CO_2), which is roughly equivalent to the CO_2

emissions of 29 million cars [3]. With that in mind, there has been an effort in researching Radio Frequency Energy Harvesting (RF-EH) for telecommunications systems with the aim of increasing EE. One concept currently being investigated is Simultaneous Wireless Information and Power Transfer (SWIPT) that promises to increase EE. There has also been increased interest in leveraging Machine Learning (ML) to enhance telecommunications, in particular ML has been used to solve mathematically complex optimization problems like power allocation and user association. Unmanned Ariel Vehicles (UAVs) have also gained significant interest in telecommunications systems due to their relatively low cost and increased mobility. This study focuses on combining the aforementioned concepts with the aim of maximising the Sum Spectral Efficiency (SSE).

1.2 Problem Statement

In the literature, we find that many papers have focussed on some of the aspects we incorporate in a CF-mMIMO network system model such as NOMA, UAVs as Access Points (APs), ML optimization and SWIPT. The integration of the aforementioned concepts is driven by their ability to naturally complement each other's strengths and mitigate weaknesses. For instance, NOMA relies heavily on precise power allocation to outperform OMA; without proper optimization, its performance can degrade. Machine learning offers an efficient alternative to traditional optimization methods, which are often too slow and computationally intensive for real-time applications. Similarly, CF-mMIMO and SWIPT work well together. SWIPT enables energy harvesting from APs, but UE at the network edge may struggle to harvest sufficient power when APs are sparse. While increasing transmit power is inefficient, deploying more APs as in CF-mMIMO addresses this issue effectively by eliminating edge users and enhancing spectral efficiency. However, we also notice that none of the papers have incorporated all the aspects in their system model. This is particularly evident when we compare our work to studies in the literature reviewed that are closely aligned to our research objectives. In [4] the usage of Graph Neural Network (GNN) based AP selection in CF-mMIMO

is investigated. In [5], the authors use a Deep Neural Network (DNN) and train it in an unsupervised manner to learn the optimum user power allocation with objective of maximising the minimum user rate. In [6], the authors propose using a Deep Reinforcement Learning (DRL) model with hybrid action space for joint AP selection and power control with aim of maximising EE under SSE requirements. In [7], two efficient unsupervised machine learning solutions for user clustering algorithms, namely K-means++ and improved K-means++, are proposed for effectively clustering users into disjoint clusters in CF-mMIMO with NOMA, with the objective of maximising Downlink (DL) SSE are investigated. In [8], the authors investigate CF-mMIMO with UAVs acting as APs where UAV altitude is correlated to Line of Sight (LoS) and Non-Line-of-Sight (NLoS) propagation model, an optimal height and horizontal coordinates that maximise the DL sum rate are obtained through an alternating optimization method. In [9], the authors investigate UAVs with Wireless Power Transfer (WPT) in a CF-mMIMO system, where energy harvested from the DL WPT is used to support both Uplink (UL) data and pilot transmission while taking hardware impairments of the UAV into account is investigated. In [10], an optimal user pairing strategy to group users in CF-mMIMO with NOMA while considering the impact of both individual and linear-combination channel estimations is investigated.

As shown in Table 1.1, there has been a lot of research done related to CF-mMIMO with SWIPT, UAVs, NOMA, or ML individually. However, to the best of the author's knowledge there has not been a single paper that combines the afore-mentioned concepts in CF-mMIMO network with the objective of maximising DL SSE by doing joint power allocation and user association optimization using DRL, whilst considering imperfect Successive Interference Cancellation (SIC) and minimum UE SE requirements for both UL and DL data transmission. In addition to power allocation and user association, SWIPT frame optimization will also be done. However, the question of why we are optimizing user association and power allocation remains unanswered. The reason for optimizing user association and power allocation CF-mMIMO with NOMA is to maximise the downlink sum spectral efficiency. Random user association and power allocation results in

TABLE 1.1: Literature review comparison with proposed research.

Paper	Gateway	Access Technique	PA	UA	Optimization Solution	SWIPT	SWIPT Frame Optimization	Imperfect SIC
[22]	BS	TDMA	No	No	ML	Yes	No	No
[33]	BS	NOMA	Yes	Yes	ML	No	No	No
[35]	BS	NOMA	Yes	Yes	None	No	No	No
[36]	BS	NOMA	No	No	None	No	No	Yes
[40]	BS	NOMA	Yes	Yes	SCA	No	No	Yes
[7]	BS	NOMA	Yes	Yes	SCP	No	No	No
[45]	BS	NOMA	No	Yes	None	Yes	No	Yes
[46]	BS	NOMA	Yes	No	SCA	Yes	No	No
[48]	BS	NOMA	Yes	No	None	Yes	No	No
[59]	BS	NOMA	No	No	SCA	No	No	No
[73]	UAV	NOMA	No	No	TSA	No	No	No
[4]	BS	TDMA	No	Yes	ML	No	No	No
[81]	UAV	TDMA	No	No	ML	No	No	No
[82]	BS	TDMA	No	Yes	ML	No	No	No
[55]	UAV	TDMA	No	No	ML	No	No	No
Our Work	UAV	NOMA	Yes	Yes	ML	Yes	Yes	Yes

sub-optimal results. Another reason for optimizing user association is for scalability reasons. The basic idea of CF-mMIMO is to have many distributed APs in an area that jointly serve all UE within the area. This becomes inefficient and impractical, especially when the number of UE far exceeds the number of APs within area. Therefore, instead of every AP serving every user, each AP services a subset of users. However, randomly grouping users to APs leads to sub-optimal results [7]. This research, therefore seeks to combine CF-MIMO with NOMA, SWIPT for energy harvesting, UAVs as mobile APs and ML aided UA and PA whilst considering minimum Quality of Service (QoS) requirements for UL and DL transmission. It is important to highlight why we consider both the DL and UL SE requirements. DL SE requirements have traditionally been prioritized, as most users tend to download significantly more data than they upload. However, with the continued rise of IoT devices that frequently upload data to the cloud, the importance of UL transmission and consequently UL SE requirements is steadily increasing. Therefore, it is essential to consider both DL and UL SE requirements. A prime example of this is computational offloading in URLLC-enabled UE, such as autonomous vehicles, where both DL and UL SE, along with low latency, are

critical. To address this, our system model incorporates both DL and UL SE constraints, using a frame duration of 1 ms, in alignment with the latency requirements specified for URLLC in 5G. In the system model, we also adopt imperfect SIC to more realistically represent the implementation of NOMA, as perfect SIC is practically impossible to achieve in real-world systems. Numerous studies have highlighted the impact of imperfect SIC in NOMA, and this has been further explored in the context of Cell-Free Massive MIMO, as discussed in [36].

1.3 Research Questions

Based on the literature reviewed, the following questions need to be addressed in UAV aided CF-mMIMO with NOMA and SWIPT:

'How can ML be used for power allocation and user association in a UAV aided CF-mMIMO network with NOMA and SWIPT while considering the effects of imperfect SIC and minimum UE SE requirements for both uplink and downlink data transmission?'

The question is further expanded below:

1. How can a CF-mMIMO system be modelled to incorporate NOMA, UAVs and SWIPT for energy harvesting?
2. How can an optimization objective be formulated for power allocation and user association with the goal of maximising downlink SSE and SWIPT frame optimization whilst considering UE SE requirements for both downlink and uplink data transmission?
3. How can a DRL model be used to jointly optimize power allocation and user association with probabilistic channel gain between mobile UE and APs?
4. How can we find the optimal energy harvesting time allocation for the SWIPT frame?

5. How can we improve the training and inference performance of the chosen DRL model?

1.4 Research Methodology

The following presents the research methodology applied in conducting the research:

1. Extensive literature review is conducted with a focus around CF-mMIMO with NOMA, SWIFT, Energy Harvesting (EH) and DRL. Resource allocation problems related to CF-mMIMO are also reviewed in literature.

2. We examine gaps in the literature and formulate a research question based on the gaps we found in literature. We also formulate research objectives and a research plan in the form of a detailed research proposal.

3. We develop a CF-mMIMO system model that incorporates NOMA and SWIPT with mobile APs in form of UAVs.

4. We develop an objective function with the goal of maximising DL SSE whilst considering UE SE requirements for both DL and UL data transmission.

5. We use a DRL model to solve the optimization objective function by doing joint user association and power allocation. We first formulate a reward function for the DRL model, that encapsulated the objective function.

6. We evaluate the performance of DRL solution and compare it to random power allocation and user association schemes.

7. We improve the training and inference performance of DRL model by using bootstrapping techniques like using multiple actor heads.

1.5 Research Contributions

The main contributions of the research can be summarised as follows:

1. We develop a CF-mMIMO system that incorporates NOMA with imperfect SIC and SWIPT. UAVs are also incorporated in the system model as mobile

APs in addition to mobile UE. We also find an optimal time for energy harvesting in the system model in addition to power allocation and user association optimization. To the best of the author's knowledge, there has not been a single paper that combines the afore-mentioned concepts.

2. We formulate objective function with the aim to maximise downlink SSE by doing joint power allocation and user association optimization using DRL, whilst considering imperfect SIC and minimum UE SE requirements for both uplink and downlink data transmission. To the best of the author's knowledge, no paper has yet addressed the combination of the concepts in a CF-mMIMO network. Specifically, there has been no research focused on maximising DL SSE by jointly optimizing power allocation and user association through DRL in a CF-mMIMO network. Additionally, none of the existing studies account for the impact of imperfect SIC. Furthermore, the minimum UE SE requirements for both UL and DL transmissions have not been considered in conjunction with these factors.

3. We also conduct comparative analysis of various techniques used to improve the performance of DRL models in literature for the proposed system model. Various bootstrapping techniques have been used for improving DRL model performance. However, these bootstrapping techniques have only been tested on a variety of game like environments, like Cheetah, Walker, Pong, Hooper and Atari Games. These game environments often have a relatively very small action space when compared to resource allocation problems in wireless communications systems, particularly in our case, where we have a hybrid action space for joint power allocation and user association using a single DRL model with one agent. In other words, our DRL environment is more complex; thus, we have more complex action and state spaces. Therefore, there is no guarantee that the improvements presented in literature, will lead to notable improvements for more complex environments. Thus, we compare the various bootstrapping methods and show that the training and inferencing performance of DDPG can be improved.

1.6 Dissertation Organisation

The rest of the dissertation is organised as follows:

- Chapter 2: In this chapter background and literature review of key concepts vital to the proposed system model will be explored. The study gap based on the literature reviewed, will also be articulated.
- Chapter 3: In this chapter a CF-mMIMO system with NOMA, SWIPT and DRL aided user association and power allocation is modelled. An objective function for the optimization problem is defined. Results are analysed and discussed to evaluate the performance of the proposed DRL solution.
- Chapter 4: This chapter builds upon the work presented in Chapter 3 by exploring various DRL training techniques aimed at improving model performance. The focus is on enhancing the effectiveness of the DRL model introduced in the previous chapter.
- Chapter 5: In this chapter an overview of the work done is presented and concluding remarks on the work done. Future work and possible directions to extend the work done are also discussed in the chapter.

CHAPTER 2

Background and Literature Review

2.1 Introduction

This chapter explores key concepts that form the basis of the CF-mMIMO NOMA network system model presented in Chapter 3. Background on CF-mMIMO architectures are explored. In addition to background on CF-mMIMO, a background of NOMA, SWIPT, UAVs and DRL are explored including an extensive literature review exploring how each of the aforementioned concepts have been used and integrated with CF-mMIMO.

2.2 Cell Free Massive MIMO

CF-mMIMO has been identified as one the most promising next-generation network architectures that will enable 6G, with the emergence of smart cars, smart environments, e-health, AR/VR and future applications like holographic communication. Future networks will require higher throughput, greater spectral efficiency, reliability and most importantly, greater capacity to cater for the ever-expanding list of smart devices [11]. To meet the increased demand requirements of 6G with current network architectures, cell splitting or ultra dense deployment will be required. However, this causes inter-cell interference, edge user performance degradation and handover problems. CF-mMIMO promises to solve these problems by taking advantage of enhanced micro-diversity, favourable propagation and providing uniform service to all users in the network.

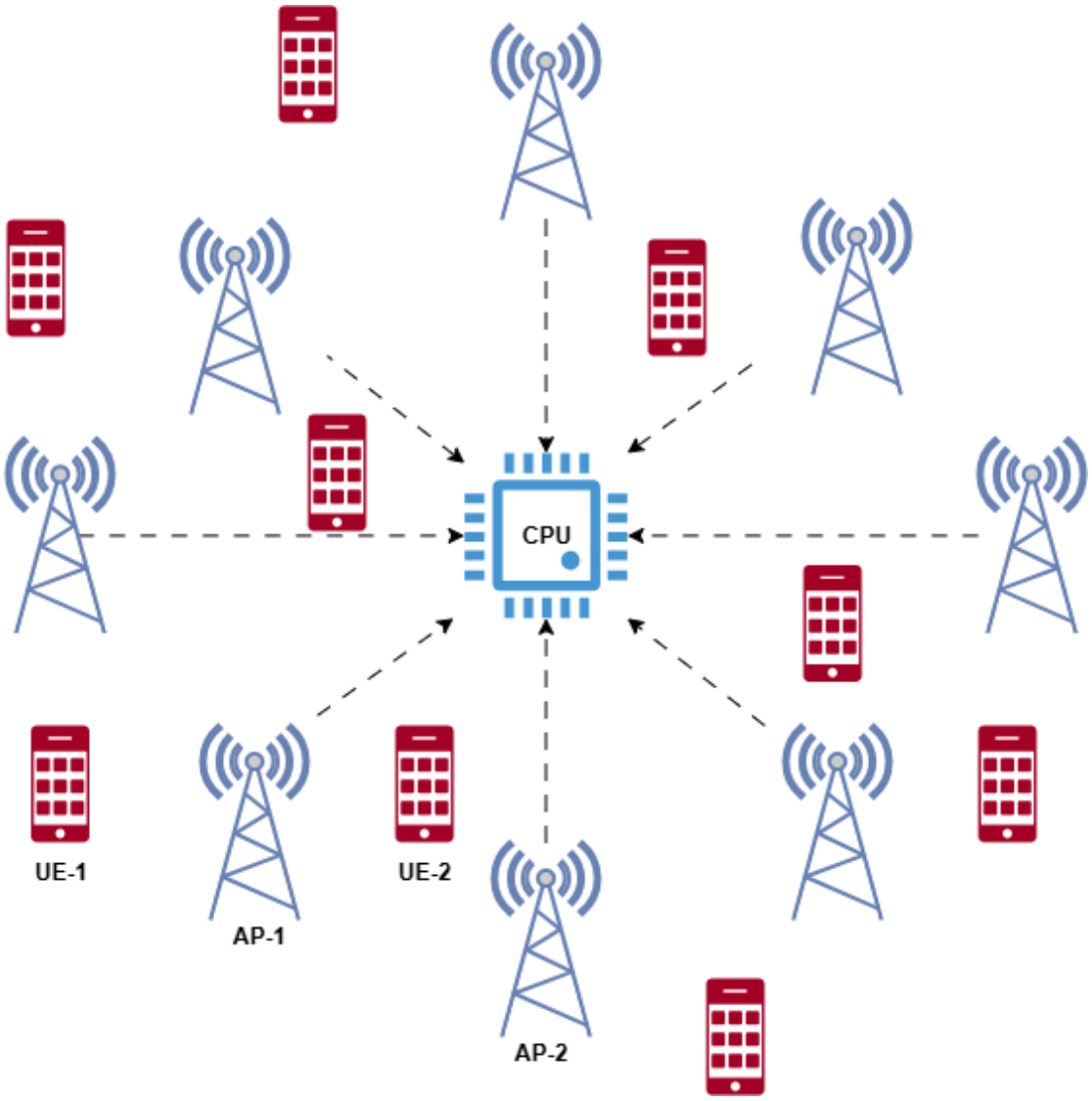


FIGURE 2.1: A basic CF-mMIMO architecture.

CF-mMIMO systems, typically consist of multiple APs at the same time-frequency resource block, that are connected to a Central Processing Unit (CPU), where the APs jointly serve each UE in a given area as shown in Figure 2.1 [11]. However, there are four different types of implementations of CF-mMIMO systems. The first level implementation is a fully distributed network where each AP is allocated and serves only one UE. In this configuration, the AP is responsible for detection using only local channel estimation [11]. In the second and third level implementations, the APs are responsible for performing channel estimations locally and applying arbitrary receive combiner to obtain estimates of the UE data, which is then collected in the CPU for joint detection [11]. The only difference between the

second and third levels is that, in the second level, the process is simplified: the CPU performs local detection by merely averaging the local estimates [11]. The fourth level implementation is the opposite of first level implementation, where instead of the network being fully distributed, the network is fully centralised. The pilot and data signals are received by all the APs and are transmitted to the CPU for channel estimation and signal detection [11].

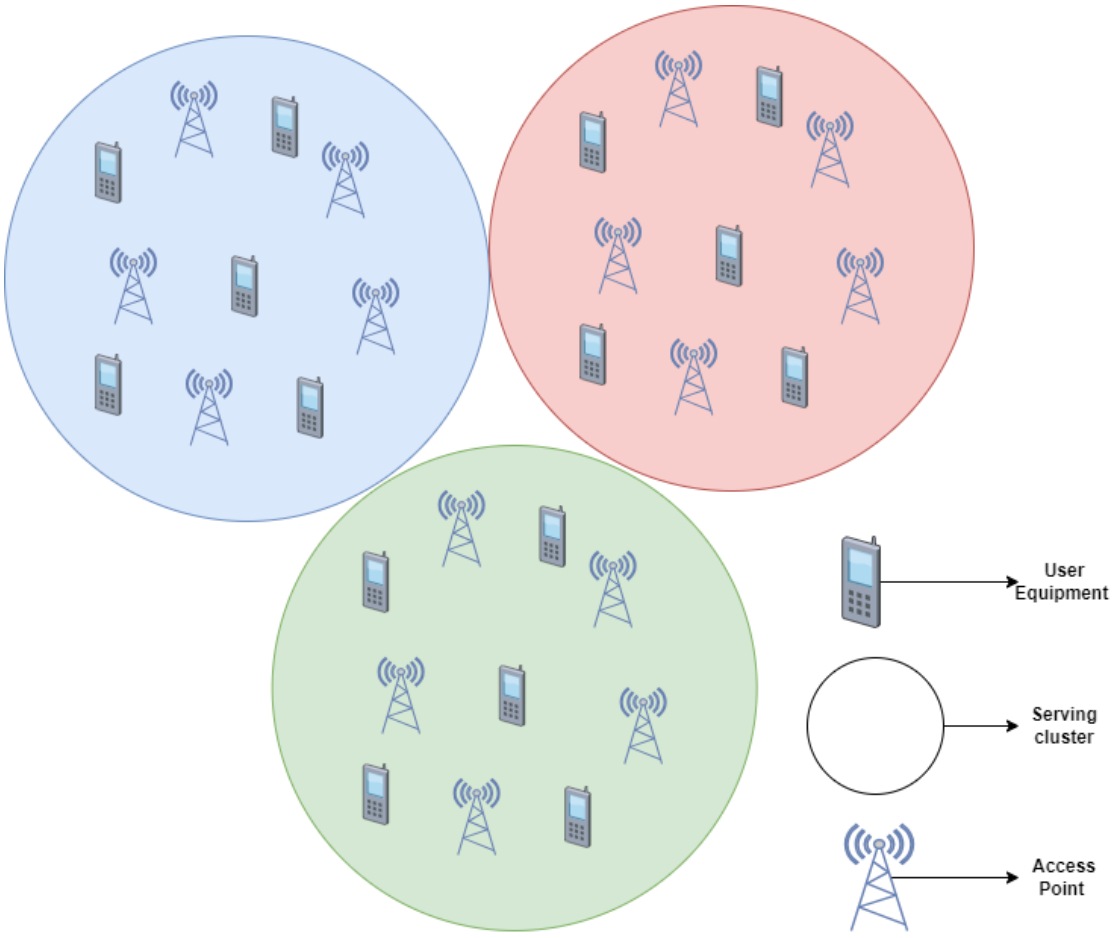


FIGURE 2.2: A cluster centric CF-mMIMO system.

The performance of CF-mMIMO has been shown to scale up as the number of APs increases relative to the number of UE [12]. However, this brings up the question about scalability. In [12], scalable CF-mMIMO that exploits the dynamic cooperation clustering in MIMO Network literature is explored. A User-Centric (UC) CF-mMIMO approach has been investigated in the literature, where the core idea is that, instead of all APs jointly serving all UE, each UE is served by only a selected subset of APs. A UC-CF-mMIMO is particularly important when

considering a finite power budget and when the number of UE is the same or greater than the number of APs [13]. There are two ways of implementing UC-CF-mMIMO, one approach is to clearly define AP clusters as shown in Figure 2.2, where UE can only be served by one AP cluster [13]. The second approach allows multiple APs to serve UE without clearly defined AP clusters as shown in Figure 2.3 [13].

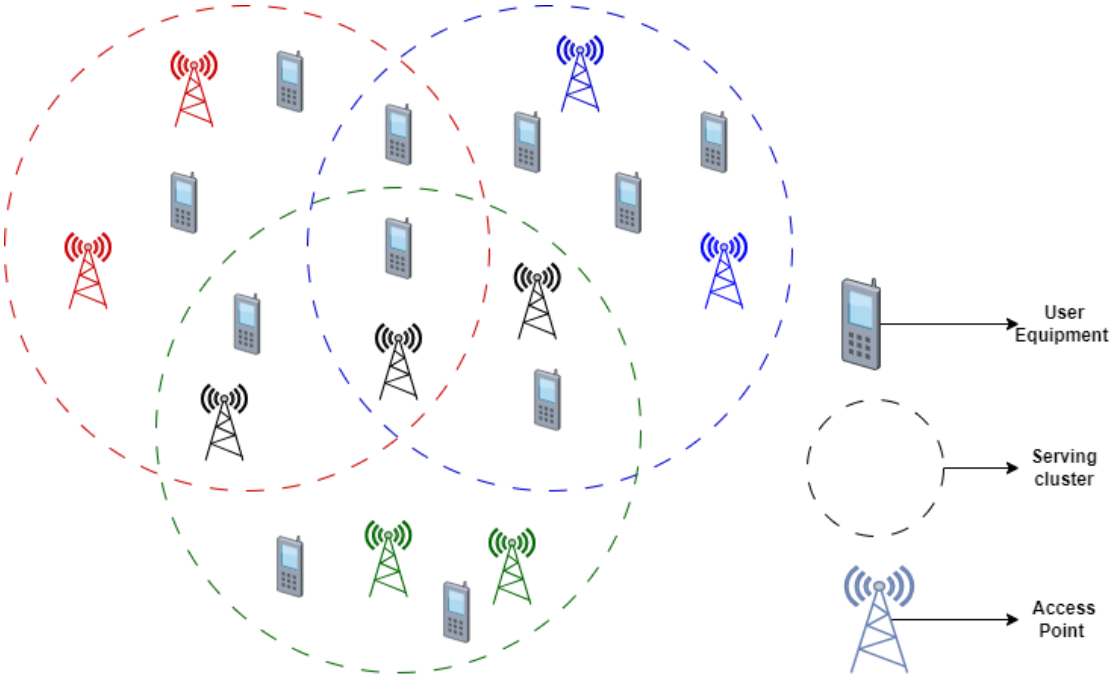


FIGURE 2.3: A user centric CF-mMIMO system.

2.3 Work related to Cell Massive MIMO

A lot of research has been done on CF-mMIMO focused on different aspects of CF-mMIMO. Traffic management in CF-mMIMO based 6G networks has been studied in [14], where a simulation that jointly considers the utilization of the communication resources at the radio edge and at front-haul is considered. A lot of research relating to conjugate beam-forming CF-mMIMO has also been conducted. In [15], a weighted Minimum Mean Square Estimation (MMSE) optimization of conjugate beam-forming that considers connectivity levels constraints to accommodate the finite rate front-haul links is studied. In [16], a joint user association and hybrid beam-forming design for CF-mMIMO with Millimetre Wave (mmWave) systems

without the full Channel State Information (CSI) is studied. In [17], a delete-based beam selection algorithm that considers the channel measurements available in cell free networks and the sum of Reference Signal Received Power (RSRP) at UE is utilized as the metric for beam selection is studied. In [18] and [19], DRL is used for distributed beam-forming techniques and self-organizing mmWave networks with hybrid beam-forming for CF-mMIMO respectively. In [20], joint access point beam-forming and a scheme for switching AP on/off to reduce network power consumption in CF-mMIMO mmWave networks is studied. In [21], the impacts of beam-forming schemes when combined with zero forcing baseband precoding on CF-mMIMO mmWave Ultra-Dense Networks (UDN) systems is investigated. Studies on the use of Reflective Intelligence Surfaces (RIS) in CF-mMIMO have also been explored in the literature. In [22], a decentralised cooperative beam-forming design for RIS in CF-mMIMO is investigated. In [23], beam-forming Optimization for RIS-aided SWIPT in CF-mMIMO is investigated. In [24], the performance analysis and system design of RIS assisted CF-mMIMO with two-timescale CSI to mitigate overhead due to instantaneous CSI with extremely high dimensions of estimated channels is investigated.

2.4 Non-Orthogonal-Multiple-Access

NOMA is a technique that promises to improve capacity, latency, connectivity, and SE when compared to Frequency Division Multiple Access (FDMA), Time Division Multiple Access (TDMA), Code Division Multiple Access (CDMA) and Orthogonal Frequency Division Multiple Access (OFDMA). CDMA, FDMA, TDMA and OFDMA all fall under Orthogonal Multiple Access (OMA), defined by their characteristic to allocate resources orthogonally [25]-[32]. OMA requires less complex receivers when compared to NOMA and has been used for First Generation (1G), Second Generation (2G), Third Generation (3G), Fourth Generation (4G) and 5G mobile networks [27]. However, OMA has certain limitations that make it less suitable for 6G networks. In particular, the number of users that can be supported simultaneously is restricted by the limited availability of orthogonal resources.

In addition to the capacity issues, orthogonality is affected by channel induced impairments. Received signals are usually non-orthogonal due to time-frequency expansion, thus in OMA complex measurements such as multi-user equalizers are required to restore orthogonality [27]. There have already been issues with OMA when trying to meet SE and capacity requirements of 5G. Hence, the need for NOMA, which can achieve SE and multi-user capacity requirements of 5G and beyond, with the aid of time sharing or rate splitting [29].

There are mainly two NOMA implementations namely, Power Domain NOMA (PD-NOMA) and Code Domain NOMA (CD-NOMA) [30]. Irrespective of which NOMA implementation is used, the underlying principle of allowing multiple users to be superimposed on the same time-frequency resource block remains the same [31]. In CD-NOMA, usually a Sparse Code Multiple Access (SCMA), works by allowing finite number of users to interfere with each other [31]. Each user is assigned a unique codebook, which is complex valued, multidimensional and sparse in nature. The codebooks are used to spread the modulated symbols from the users over the allocated resources and decoding in simultaneous detection of multiple users [31]. In PD-NOMA users are assigned different power levels according to channel quality and SIC is used to detect their signals. The optimal order to detect users' signals is from the strongest to the weakest user [32].

2.4.1 Downlink PD-NOMA

In DL PD-NOMA , a Base Station (BS) transmits superimposed users signals with different powers given by

$$x = \sum_{i=1}^{K} \sqrt{p_i} x_i, \tag{2.1}$$

where x_i is transmitted signal from the BS to the i-th user and p_i is the power scaling coefficient corresponding to the i-th user. The received signal at the i-th user is given by

$$y_i = h_i x + v_i, \tag{2.2}$$

where h_i is the channel gain, v_i is the Additive Gaussian Noise Power (AWGN) [32]. SIC is carried out at the receiver, with the optimal detection order proceeding from the strongest to the weakest user. This approach helps minimize interference contamination from users with lower normalized channel gains [33]. Without loss of generality, it can be assumed that the users are arranged according to the descending order $|h_1|^2/N_1 \geq |h_2|^2/N_2 \geq \cdots \geq |h_k|^2/N_k$ and the power allocated in the ascending order $p_1 \leq p_2 \leq \cdots \leq p_k$. The data rate of the i-th user is given by

$$R_i = W \log_2 \left(1 + \frac{p_i |h_i|^2}{N_i W + \left(\sum_{j=1}^{i-1} p_j \right) |h_i|^2} \right), \tag{2.3}$$

where W is the system bandwidth and N_i is the Gaussian noise plus inter-cell interference at the user [33]. In a two-user scenario, User-1 with best channel gain is assigned less power while User-2 with weaker channel gain is assigned more power. User-1 performs SIC of User-2 signal before decoding its own signal. User-2 only decodes its own signal and treats the signal for User 1 as noise as shown in Figure 2.4. The data rates for the users are given by

$$R_1 = \log_2 \left(1 + \frac{p_1 |h_1|^2}{N_1} \right), \tag{2.4}$$

$$R_2 = \log_2 \left(1 + \frac{p_2 |h_2|^2}{N_2 + p_1 |h_2|^2} \right). \tag{2.5}$$

where the bandwidth is normalised to 1Hz [33].

It is important to note that, greater channels gains do not necessarily mean greater signal strength as users with the best channel gains are assigned less power and users with the worst channel gains are assigned the most power in PD-NOMA. Users with the weakest channel gains are the least affected by intra-cell interference

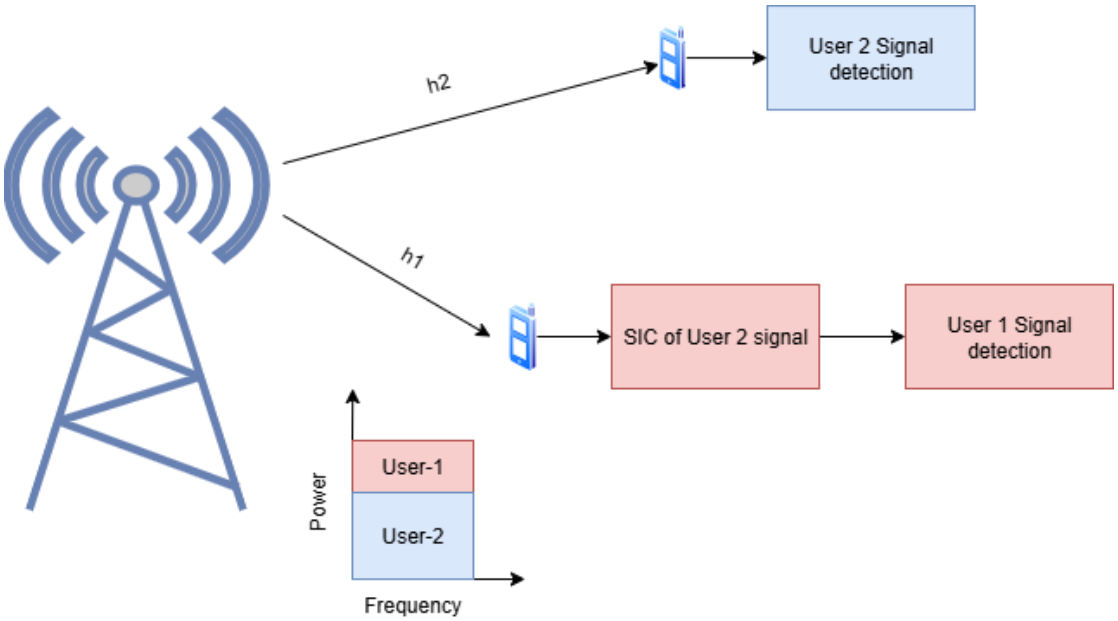


FIGURE 2.4: Downlink NOMA transmission.

and the users with best channel gains are affected the most by intra-cell interference due to the power differences.

2.4.2 Uplink PD-NOMA

In UL PD-NOMA, the signal received by the BS is given by

$$y = \sum_{i=1}^{K} h_i \sqrt{p_i}\, x_i + v, \tag{2.6}$$

where x_i and p_i are the signal and transmit power from the i-th user to the BS, respectively [33]. The power density v is associated with N_o, the Gaussian noise plus the inter-cell interference at the BS [32]. Without loss of generality, we can assume that during uplink, the users are arranged as follows: $p_1|h_1|^2 \geq p_2|h_2|^2 \geq \cdots \geq p_k|h_k|^2$ and the optimal decoding order for SIC is as follows: $x_1, x_2, \ldots, x_k$. In a two-user scenario, BS receives strong and weaker powers signals from User-1 and User-2, respectively. The BS starts by detecting User-1 signal and then subtracts it from the signal from the received signal to detect the weaker User-2 signal as shown in Figure 2.5. A power imbalance between the users is required

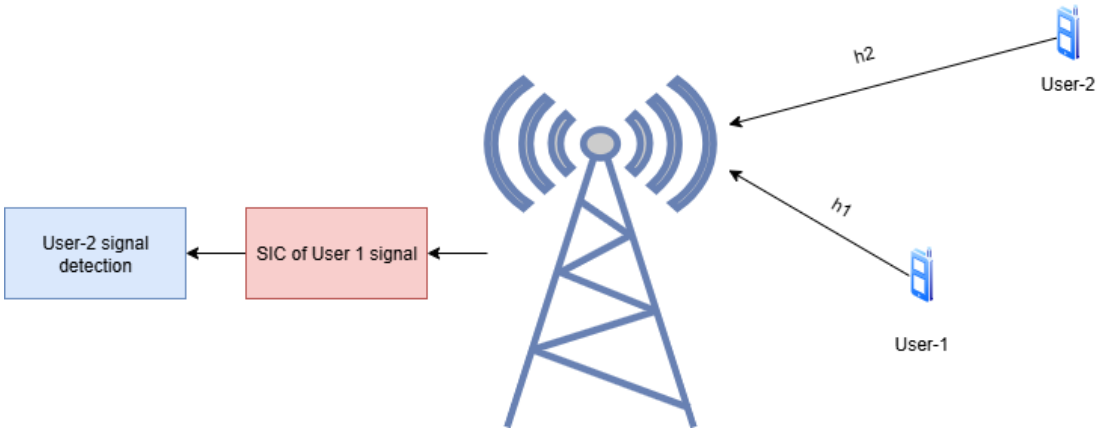


FIGURE 2.5: An uplink NOMA transmission.

to maximise Bit Error Rate (BER) performance. The achievable data rate of the i-th user is given by

$$R_i = W \log_2 \left(1 + \frac{p_i |h_i|^2}{N_o W + \left(\sum_{j=i+1}^{K} p_j \right) |h_j|^2} \right). \tag{2.7}$$

2.4.3 Cell Free Massive MIMO with NOMA

Although research has been done on CD-NOMA, most research has been centred around PD-NOMA particularly on research focussed on combining CF-mMIMO with NOMA. There have been a number of papers that have investigated the use of NOMA in CF-mMIMO. In [34], a user-centric approach where APs only serve a selected subset of UE rather than all of them in CF-mMIMO system with NOMA is investigated. In [35], the achievable rate of hybrid CF-mMIMO-NOMA underlaid a primary massive MIMO system is investigated. In [36], the authors investigate AP tiered for user grouping and power allocation in CF-mMIMO with NOMA system. In [37], the impact of non-reciprocal channels in CF-mMIMO with NOMA while considering pilot contamination and imperfect SIC is investigated. In [38], the achievable sum rate in CF-mMIMO with NOMA is derived while considering the effects of intra-cluster pilot contamination, intra-cluster interference

and imperfect SIC and compared to OMA. In [39], the UL transmission of a CF-mMIMO with NOMA in which multiple APs each equipped with multiple-antenna array and connected to CPU over a back-haul networks with optimal back-haul combining is investigated. In [40], conjugate beam-forming for active and passive beam-forming at the APs and IRS, respectively is investigated. In [41], the performance of CF-mMIMO with NOMA for Internet of Things (IoT) in spatially correlated Rician fading channels while considering the impact of imperfect SIC and pilot contamination is investigated. In [10], an optimal user pairing strategy to group users in CF-mMIMO with NOMA while considering the impact of both individual and linear-combination channel estimations is investigated. In [7], two efficient unsupervised machine learning based user clustering algorithms, namely K-means++ and improved K-means++, proposed to effectively cluster users into disjoint clusters in CF-mMIMO with NOMA is investigated.

2.5 Simultaneous Wireless Information and Power Transfer

SWIPT aims to combine concepts of WPT and Wireless Information Transfer (WIT) with the aim of optimizing synchronous transmission of information and energy [42]. SWIPT can be broken into two components, namely, Information Decoding (ID) and EH done at the receiver. The energy harvested is in the form of RF (Radio Frequency) waves that have been converted into electrical energy. Conventional transmitters and receivers cannot support SWIPT. There are two approaches for receiver design for SWIPT, one approach has separated information decoder and energy harvester units [43]. Another approach involves using a unified receiver circuit capable of performing both EH and ID simultaneously, as proposed in [43]. There are two basic operation modes of SWIPT namely, Power Splitting (PS) and Time Splitting (TS) as shown in Figure 2.6.

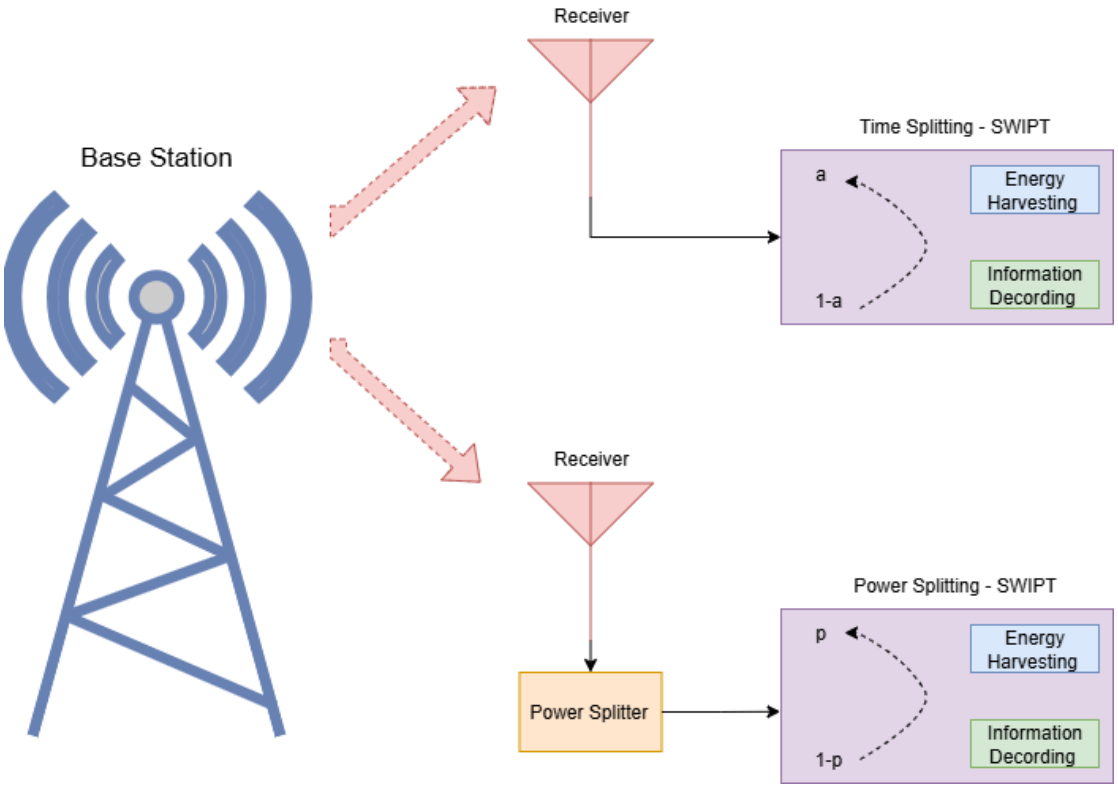


FIGURE 2.6: Two architectures for SWIPT.

2.5.1 Time Splitting SWIPT

The basic concept behind TS-SWIPT is that within a time coherence block, time slots are allocated for EH and ID respectively [43],[44]. For example, consider a scenario where the UE is harvesting energy from the received signal transmitted by the BS. Let T_d represent the time allocated for downlink transmission within a coherence block of duration T_c. The time allocated for EH, T_{EH}, is given by $T_{EH} = (1 - \alpha)T_d$ where α is time splitting coefficient and the time allocated for ID is given by $T_{ID} = T_d - T_{EH}$. During the EH phase no ID occurs and similarly during the ID phase no energy EH occurs. Typically, the EH occurs first followed by ID. The frame structure for TS-SWIPT is shown in Figure 2.7.

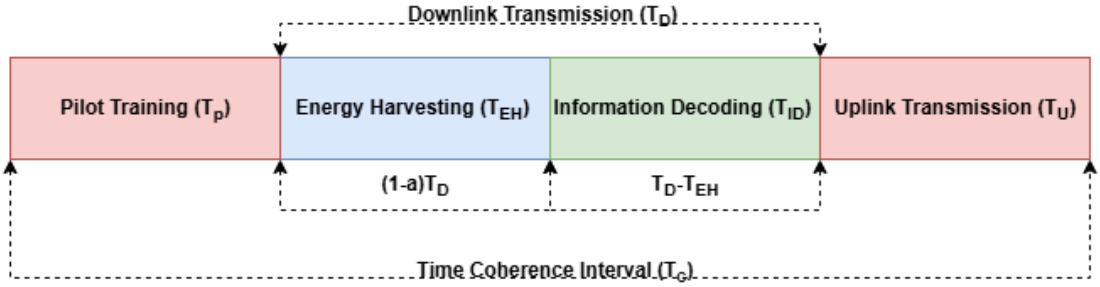


FIGURE 2.7: Frame structure for the TS-SWIPT protocol.

2.5.2 Power Splitting SWIPT

In PS-SWIPT the received power at the harvesting UE is split into two by a splitter at the receiver according to a pre-determined power splitting ratio [43],[44]. One part is used for EH and the other is used for ID. For example, consider a scenario where the UE is harvesting energy from the received signal transmitted by the BS. Let T_d represent the time allocated for downlink transmission within a coherence block of duration T_c. The time allocated for EH, T_{EH} , and time allocated for ID, T_{ID} , is identical and equal to $T_d = T_{EH} = T_{ID}$ as shown the frame structure shown in Figure 2.8. However, power splitting coefficient, ρ, is required to determine the amount of power allocated for EH and ID. For example, if the transmitted power from the BS is P, then the power allocated for EH, $P_{EH} = (1-\rho)P$, and the power allocated for ID is $P_{ID} = P - P_{EH}$.

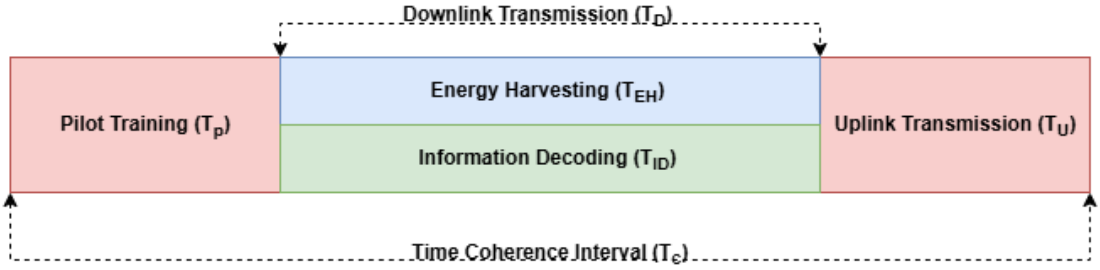


FIGURE 2.8: Frame structure for the PS-SWIPT protocol.

2.5.3 Cell Free Massive MIMO with SWIPT

Due to the EE of SWIPT and the promised increased QoS in CF-mMIMO, the concept of SWIPT coupled with CF-mMIMO has recently gained research attention. In [45], a CF-mMIMO with NOMA combined with SWIPT where each cluster head serves as a relay in order to decode the received signal and forward it to the second user in each cluster by using harvested energy is investigated. In [46], the authors investigate the energy-efficient resource allocation for Layered-Division Multiplexing (LDM) based non-orthogonal multicast and unicast transmission in CF-mMIMO, where UE perform SWIPT. In [47], CF-mMIMO with TS-SWIPT spatially random network, where the APs are randomly located and modelled using a Poisson Point Process (PPP) is investigated. In [48], the authors develop a statistical delay/error-rate QoS provision schemes over SWIPT-enabled CF-mMIMO 6G IoT networks under an infinite block length regime.

2.6 Unmanned Ariel Vehicles

UAVs commonly known as drones, have recently been considered for a variety of tasks in the literature due to their flexibility, rapid deployment capabilities and relatively low cost [49]. Due to their flexibility, UAVs have been considered for a range of applications such as monitoring and localization tasks in domains like agriculture, traffic and public safety [50]. UAVs can be equipped with a CPU and Graphics Processing Unit (GPU); this has made them an ideal candidate for providing off-loading services to latency and energy-sensitive applications in the emerging application of mobile edge computing technology [51],[52]. UAVs have also been considered for data collection in time sensitive applications [53]. UAVs have also been considered for wireless communication systems and have been shown to increase throughput, coverage and spectral efficiency [51],[52]. The basic idea of using UAV in communications systems is that they act as mobile BS or relays that can rapidly adapt to the changing environment [50]-[56]. UAVs in wireless communications system can be used in an isolated manner, or they can

be integrated with other wireless communication systems with a goal of increasing system performance [51],[52]. Whether UAVs are used in an isolated or integrated manner depends on the applications and task ahead. Isolated UAVs, for example, are well-suited for scenarios such as natural disasters or search and rescue missions in remote areas, where existing communication infrastructure is unavailable or has been destroyed. UAVs with communication equipment can provide temporary communication infrastructure allowing for communication between victims and rescuers [51]. An example where an integrated approach is beneficial is during periods of network congestion, such as peak hours. In such cases, UAVs can be deployed to enhance network capacity and extend coverage [51],[52]. However, the performance of UAVs are limited by how efficient and optimized their 3D placement, trajectory and collision avoidance systems are. The sustainability of UAVs is also a limiting factor as most UAVs have flight time of between 20-30 minutes. Another important issue when considering the use of UAVs in wireless communications systems is, effective integration with other communication systems and effective handover strategies between different network segments [51],[52]. Perhaps the biggest issue with using UAVs in wireless communication systems is their vulnerability to security threats such as eavesdropping and jamming attacks [51],[52],[56].

2.6.1 Cell Massive MIMO with UAVs

There has been a recent effort to investigate the usage of UAVs in CF-mMIMO, in [56], resource allocation and UAV placement based on particle swarm optimization is investigated under QoS and coverage constraints. In [57], power allocation and scheduling strategies for a CF-mMIMO where the channel between mobile station, either UAVs or ground users and the AP points follow a Rician distribution is investigated. In [8], the authors investigate CF-mMIMO with UAVs acting as APs where UAV altitude is correlated to LoS and NLoS propagation model, an optimal height and horizontal coordinates that maximise the downlink sum rate are obtained through an alternating optimization method. In [9], the authors

investigate UAVs with WPT in a CF-mMIMO system, where energy harvested from the downlink WPT is used to support both uplink data and pilot transmission while taking hardware impairments of the UAVs into account. In [58], power control in CF-mMIMO networks for UAV Ultra Reliable Low Latency Communications (URLLC) under the finite block length regime, where both the downlink and uplink use Partial Zero-Forcing (PZF) transmit/receive beam-forming and maximum ratio transmission/combining are utilized is investigated. In [59], the authors investigate a CF-mMIMO scheme UAV BS to manage the service inter-cell interference between aerial and terrestrial nodes, where the use of High-Altitude Platform Station (HAPS) as a CPU is proposed to provide a reliable LoS link. In [60], the authors investigate a multi-domain resource allocation for Cognitive Satellite UAV Network (CSUN) consisting of a satellite and a swarm of UAVs with the objective of improving the efficiency of massive access in wide areas.

2.7 Machine Learning

The use of ML in CF-mMIMO has slowly been gaining traction in recent years, with the number of journals and conference papers increasing year on year since 2019 [62]. ML in CF-mMIMO system has been used to tackle a variety of problems, in particular to solve mathematically complex optimization problems like user association, power and resource allocation [62]. Often, power allocation and user association optimization problems are non-linear and non-convex in nature that require an iterative approach to be solved optimally, if possible. However, this is computationally expensive and due to the iterative approach, not practical for low latency applications. ML promises to achieve similar results to iterative approach faster and at a fraction of the computational resources required [62].

Due to the advances that have been done around DRL, DRL has gained prominence in telecommunications research over recent years. DRL has proven to be highly effective for solving optimization problems, particularly in telecommunications research where it is applied to resource allocation and routing challenges [63],[64]. Notable examples of use cases for DRL in telecommunications research

include Power Allocation (PA) optimization, Network Slicing (NS), User Association (UA), network routing and Channel Resource Allocation (CRA) [65],[66]. The appeal of DRL lies in its ability to deliver near-optimal solutions much faster than traditional mathematical optimization techniques, which are often computationally expensive and too slow for real-time applications. Once trained, a DRL model can provide efficient results in a fraction of the time, making it well-suited for real-time scenarios.

ML is a subset of Artificial Intelligence (AI) that uses statistical models or algorithms to recognise patterns and generalise to unseen data [66]. When given a new input, the ML model should be able to perform a task without explicit instructions. In other words, the ML model learns what to do rather than being told what to do, when given an input. ML can be broken down into four basic paradigms [66]. These include:

- Supervised Learning
- Unsupervised Learning
- Semi-supervised Learning
- Reinforcement Learning

In what follows, we discuss the fundamentals of supervised learning, unsupervised learning, semi-supervised learning and DRL, with a particular emphasis on DRL. We conclude the section by reviewing the applications of ML in Cell-Free mMIMO systems.

2.7.1 Supervised Learning

In supervised learning, the ML algorithm is trained on a labelled dataset. During training the ML algorithm learns to identify pattens embedded between input and labelled output [67],[68]. Once the ML algorithm has been trained, it can use seen

or unseen data as input and produce an output in the form of a prediction or classification as shown in Figure 2.9. Some examples of supervised learning algorithms are decision trees, random forest, naive byes and support vector machine.

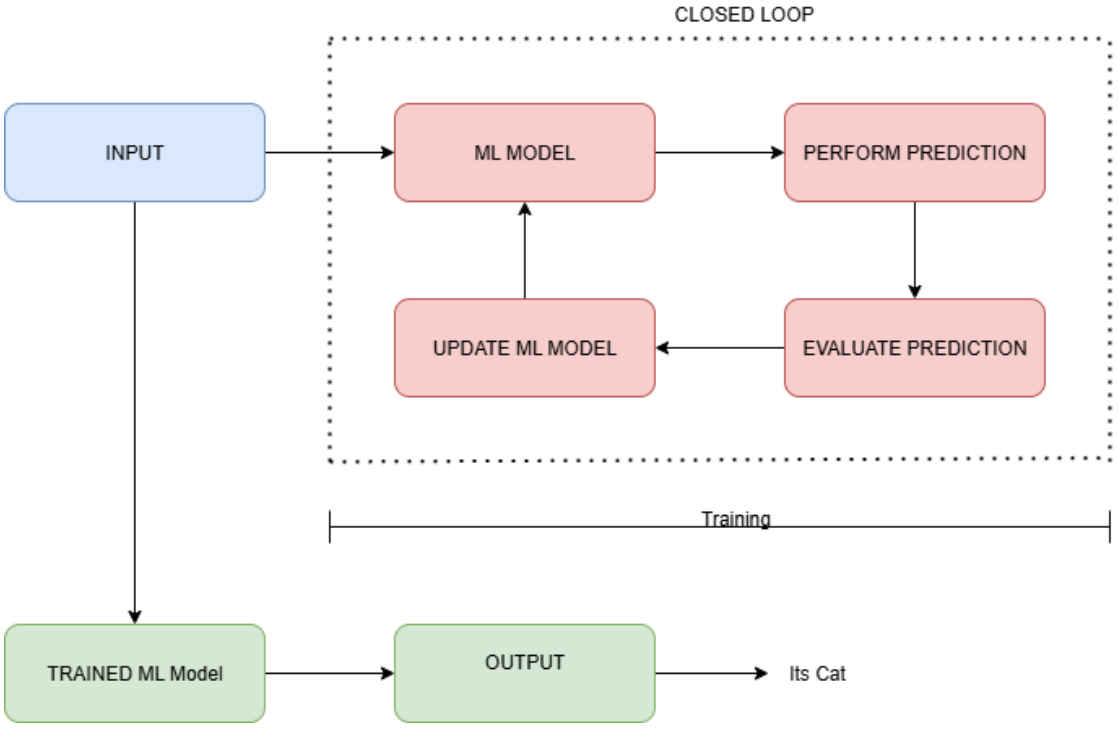


FIGURE 2.9: Supervised learning architecture.

2.7.2 Unsupervised Learning

Contrary to supervised learning, unsupervised learning does not require labelled dataset for training. Instead, unsupervised learning algorithms learn independently without any guidance such as labelled data as shown in Figure 2.10. Unsupervised learning ML algorithms learn the data's inherent structure; this makes them ideal for data exploration especially in unstructured datasets [67],[68]. Some examples of unsupervised learning algorithms are K-means clustering, Density-Based Spatial Clustering Applications with Noise (DBSCAN) and Principal Component Analysis (PCA).

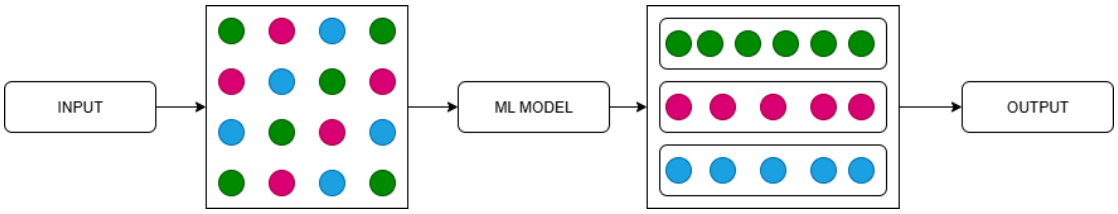


FIGURE 2.10: Unsupervised learning architecture.

2.7.3 Semi-supervised Learning

Semi-supervised learning lies somewhere between supervised and unsupervised learning. As such, semi-supervised learning can be thought of as a hybrid between supervised and unsupervised learning. In semi-supervised learning, a small amount of labelled data is used at the beginning to initiate the ML algorithm [67],[68]. Thereafter, the model predicts labels for a larger amount of unlabelled data as shown in Figure 2.11. Some examples of semi-supervised algorithms include graph based methods, generative models and Variational Auto-Encoders (VAEs).

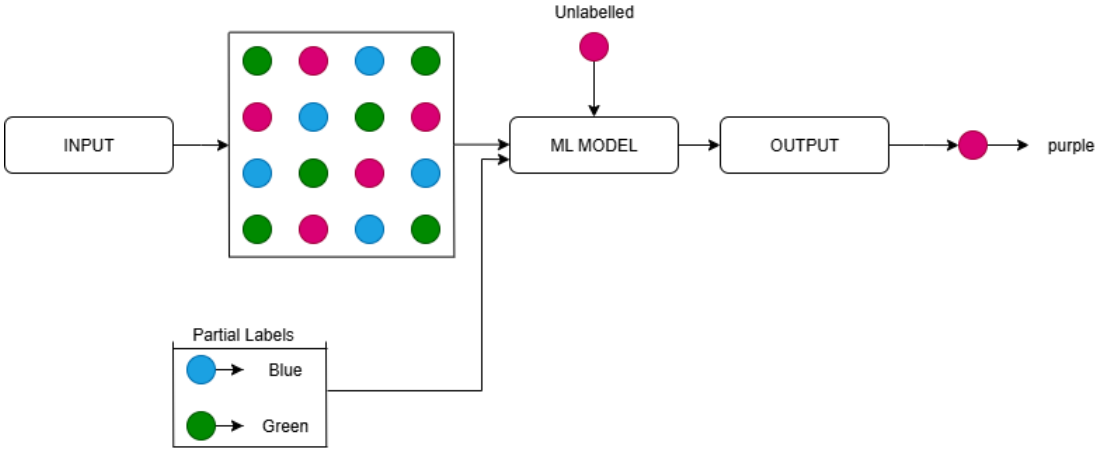


FIGURE 2.11: Semi-supervised learning architecture.

2.7.4 Deep Reinforcement Learning

As the name implies, DRL is a combination of Reinforcement Learning (RL) and Deep Learning (DL) which are subsets of ML. Deep Learning is a subset of ML, that uses multi-layered Neural Networks (NN) connected using synapses. A NN

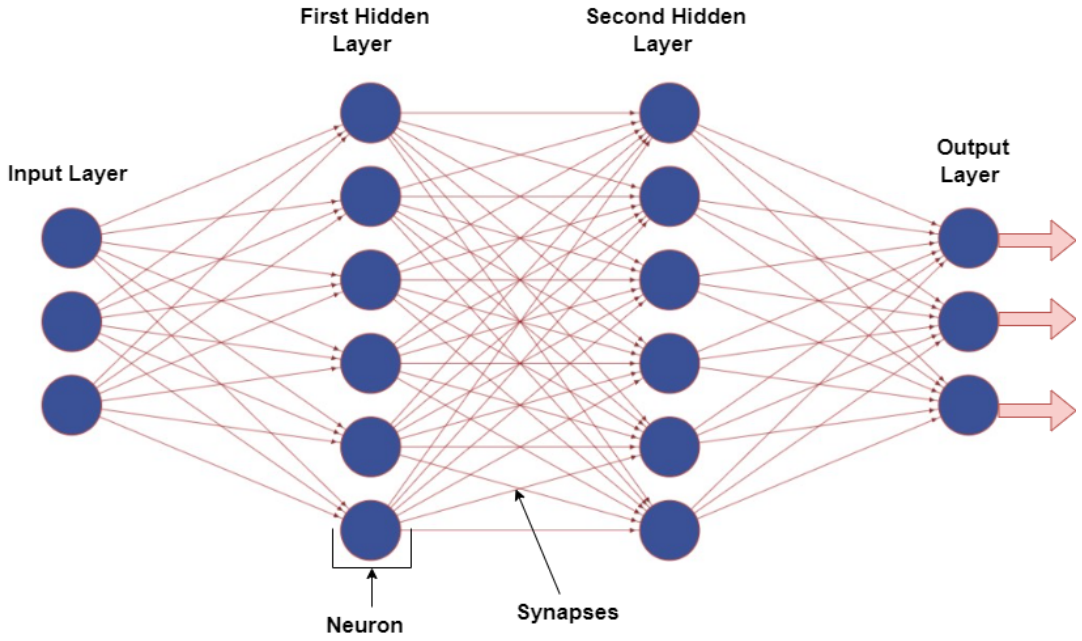


FIGURE 2.12: DNN architecture.

also known as Artificial Neural Network (ANN) is an attempt to mimic the human brain's ability to learn using connected nodes called neurons, hence the name [65],[66]. The Deep in DL comes from the DNN, typically a DNN has three or more neural layers as shown in Figure 2.12. To avoid confusion, the reason why DL is not included in as part of the four ML paradigms, is that DL can be combined with the all the basic ML paradigms, namely, supervised learning, unsupervised learning, semi-supervised learning and reinforcement learning. As such, DL should be considered as a natural extension of the basic ML paradigms.

RL is a ML algorithm that uses intelligent agents that learn through trial and error, to take appropriate actions in an interactive environment with the objective of maximising an accumulative reward, thus RL is goal oriented [69]. The learning process of RL can be mathematically modelled as a Markov Decision Process (MDP), where a MDP is simply a discrete time stochastic control process. Typically the MDP can be modelled as a tuple $\{S, A, P, R, \gamma\}$ where S is a finite set of States, A is finite set of actions, P is the transition probability from state s to state s' after action a is executed, R is the immediate reward and $\gamma \in [0, 1]$ is the discount factor of the reward. Given the MDP, at a given time-step, the RL model receives $S_t \in S$ from the environment and takes $A_t \in A$ as shown in Figure 2.13.

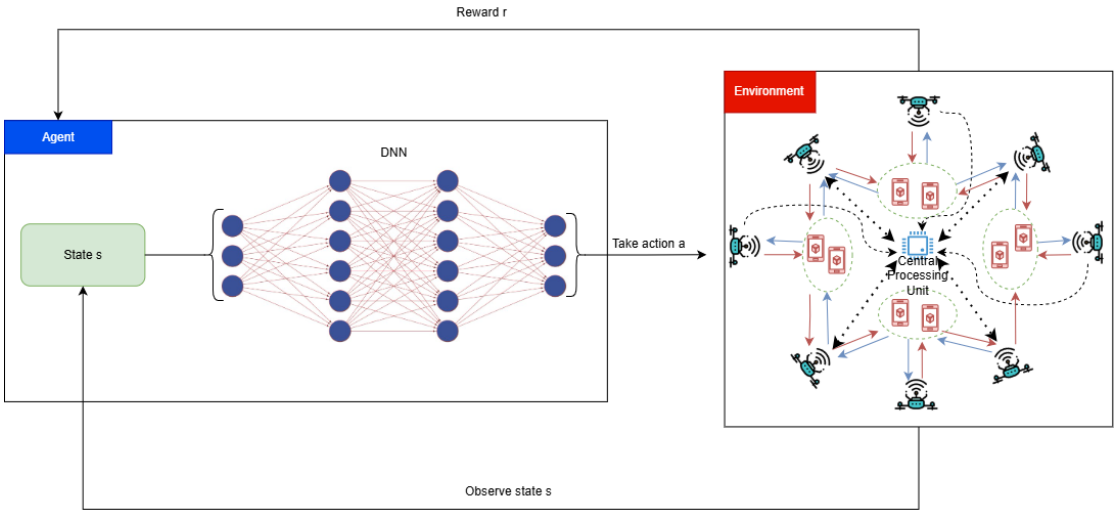


FIGURE 2.13: DRL architecture.

In turn the RL model receives a numerical reward $R_{(t+1)} \in R$ and receives the next state s'. The discount factor is used to control the impact of future rewards on the present, a larger discount value close to 1, encourages the agent to consider the long-term consequences of its actions, promoting strategies that may yield higher rewards in the future whereas a discount value close to 0 encourages the agent to focus on immediate actions. The expected discounted return G_t after time step t, is defined as

$$G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} = \sum_{k=0}^{T} \gamma^k r_{t+k+1}, \tag{2.8}$$

where T is a finite value or infinite value for finite and infinite horizon problems, respectively [70].

2.7.5 Different Deep Reinforcement Learning Models

There are many different types of DRL algorithms that have been used for a variety of applications. However, the DRL algorithms can be broadly classified based on their underlining structure, whether they are model-based or model-free, on-policy or off-policy and whether they use a value function-based method or policy function-based method or hybrid in the case of Actor-Critic (AC) models.

When choosing a DRL algorithm, one must consider the application and the action space, whether the action space is continuous or discrete. Each DRL algorithm has a set of distinct advantages and disadvantages.

2.7.5.1 Model Based and Model-Free

In Model-Free Reinforcement Learning (MFRL), the agent learns by directly interacting with the environment to learn optimal policies or value functions based on state transition and rewards [69],[71],[73]. MFRL can be broken into two categories, namely, value function based methods and policy function based methods. Model Based Reinforcement Learning (MBRL) contrary to MFRL, learns a transition model that allows for simulation of the environment without directly interacting with the environment [69],[71],[72]. The MBRL model then uses the learned environment model to simulate scenarios and actions. In general, MFRL is often used due to its simpler implementation when compared to MBRL, since it does not require a learning model. In addition to being simpler to implement, MFRL is less computationally demanding when compared to MBRL [71]. However, this is at the cost of adapting slower to changes in the environment due to relying on accumulated experiences. MFRL is also better when dealing with highly complex and unpredictable environments. However, MBRL has key advantages when compared to MFRL particularly for applications involving robotics. Since it might not be possible to have millions of direct interactions with the environment when training, for example, training a robotic arm, due to natural wear and tear from moving mechanical parts and the time required, a MFRL approach might not be the best option. Since MBRL learns the model of the environment, the learned models can be used to guide exploration purely based on simulation of the environment [69]. Thus, reducing the number of required interactions with the (real) environment. It must be mentioned that MBRL is incompatible with highly complex environments since it impossible to learn a model with high-dimensional state spaces [69],[71],[72].

2.7.5.2 On-Policy and off-Policy

On-Policy and off-policy refer to the training methods used by MFRL models. For on-policy methods, the policy π is used for both evaluation and exploration purposes, the behaviour policy and target policy are the same, where data is generated for the next iteration of policy optimization and subsequently discarded after policy optimization is completed [73],[74]. In other words, there is no buffer to store the previous actions, states and rewards. On-policy methods are more stable when working with continuous state space actions coupled with a function approximator such as neural networks and directly optimize the policy [73],[74]. However, on-policy methods require the policy to be stochastic or soft, and they tend to be more complex than off-policy methods, which makes them more challenging to implement. Off-policy methods contrary to on-policy methods, store samples generated by the behaviour policies into a buffer when interacting with the environment. The samples are then used during training to update the target policy. Another key difference between off-policy and on-policy methods to note, is since the training data may be from old behavioural polices, the behaviour policy is not the same as the target policy. In other words, off-policy methods use a different policy $\pi^{'} \neq \pi$ to generate the episodes and π is deterministic [73],[74]. Off-policy methods when compared to on-policy methods have a higher data efficiency.

2.7.5.3 Value Function Based Methods

The main idea behind value function-based methods, is that a value function is used to obtain global optimal action-value function. In other words, the value function finds the maximal value in terms of the best action. The maximal value is then used to learn the optimal policy indirectly [73],[74]. The optimal policy π^*, is simply the expected return for the best action a*, given a particular state. To estimate the optimal value function, Dynamic Programming (DP), Monte Carlo (MC) and Temporal Difference (TD) methods can be used. Value function-based methods are known for being good at dealing with applications with discrete action spaces with low dimensionality. However, value function methods suffer from

several pitfalls. For example, when dealing with continuous actions, the action space must be discretised. The problem with discretised actions spaces, is they suffer from a high dimensionality, this may lead to a lack of smoothness in control outcomes [75]. The action-value function $q_\pi^*(s,a)$ can be expressed as a Bellman equation, also known as the Bellman optimality equation. The action value function is given by

$$q_\pi^*(s,a) = \mathbb{E}\left[R_{t+1} + \gamma \max_{a'} q_\pi^*(S_{t+1}, a') \,\middle|\, S_t = s, A_t = a\right] \tag{2.9}$$

$$= \sum_{s',r} p(s', r \mid s, a)\left[r + \gamma \max_{a'} q_\pi^*(s', a')\right], \tag{2.10}$$

where $\mathbb{E}[\cdot]$ denotes the expectation over a random variable under the optimal policy π^* [73]. Similarly, the Bellman optimality for the state-value function, $\upsilon_\pi^*(s)$, can be expressed as

$$v_\pi^*(s) = \max_a \mathbb{E}\left[R_{t+1} + \gamma \upsilon_\pi^*(S_{t+1}) \,|\, S_t = s, A_t = a\right] \tag{2.11}$$

$$= \max_a \sum_{s',r} p(s', r \mid s, a)\left[r + \gamma \upsilon_\pi^*(s')\right]. \tag{2.12}$$

2.7.5.4 Policy Function Based Methods

Policy based methods, also known as policy search methods, contrary to the value function-based methods, directly optimise the parametrised policy function θ_t. The policy is optimized using a gradient free or gradient based optimization method also known as policy gradient theorem [73],[74]. The policy parameters are constantly updated to maximise the expected return. Since the optimal policy can be obtained using gradient based methods which are probability distributions of probability density on the action, which can be discrete or continuous [76].

This makes policy function-based methods ideal for both discrete and continuous action spaces. Policy networks, specifically deep policy networks are capable of learning complex behaviours. However, poor gradient updates may result in unwanted polices that may lead to drastic performance degradation. Using gradient based optimization, by learning the policy parameters with some measure $J(\theta)$, the optimal policy can be found. The approximate gradient accent is updated using

$$\theta_{t+1} = \theta_t + \alpha \nabla J(\theta_t), \tag{2.13}$$

where $\nabla J(\theta_t)$ is a stochastic estimation that approximates the gradient of $J(\theta_t)$ in terms of θ_t and α is the step size that influences the learning rate [73].

2.7.5.5 Actor-Critic

Actor-Critic (AC) methods are a hybrid between value-based and policy-based methods, that combine the advantages of both methods. As implied by the name, AC methods have an actor which follows a policy-based method and critic which follows a value-based approach to learn the policy and value function together [73],[74]. The actor is responsible for selecting the appropriate actions for a given state, while the critic evaluates the actions taken by the actor and provides feedback to the actor. The actor uses the value function of the critic's feedback to train the policy [73],[74]. The critic trains the value function and uses the TD method to update it in one step. The full step return is replaced by the one-step return using the one-step actor-critic algorithm given by

$$\theta_{t+1} = \theta_t + \alpha \left(R_{t+1} + \gamma \hat{v}(S_{t+1}, w) - \hat{v}(S_t, w) \right) \frac{\nabla_\theta \pi(A_t \mid S_t, \theta_t)}{\pi(A_t \mid S_t, \theta_t)}, \tag{2.14}$$

where w is the state-value weight vector learning by the actor-critic algorithm and (S_t, w) is the learned state value function that is used as the baseline [73]. Although, AC combines the advantages of both value-based and policy-based methods and goes a long way in alleviating the sampling efficiency problem by introducing a buffer for experience reply. AC models are still susceptible to instability during training due the coupling of the value evaluation and the policy updates.

2.7.6 Cell Free Massive MIMO with Machine Learning

In [76], the authors investigate Quantum Neural Networks (QNN) for optimal resource allocation in CF-mMIMO systems with the objective to maximise the minimum achieved sum rate. In [77], a hybrid DRL model, specifically a DDPG-Deep Double Q Network (DDQN) is used to solve dynamic AP clustering optimization problem in CF-mMIMO network. In [78], authors investigate the use of DRL to solve a joint communication and computing resource allocation problem in a CF-mMIMO system with the aim to minimize total energy consumption of the users, while meeting the respective user-specific deadlines. In [4] the usage of GNN based AP selection in CF-mMIMO is investigated. In [79], DRL for energy efficient beam-forming design in CF-mMIMO Networks is investigated. In [80], the authors investigate the usage of a DL to sum-rate-max and min-max power allocation in the uplink of CF-mMIMO network. In [81], a Cluster based AP Selection (CAPS) algorithm is used in CF-mMIMO network, with the aim of reducing the computations workload and pilot contamination by using K-mean++ clustering algorithm. In [82], the authors investigate DRL trajectory optimization for deployed UAVs in cell-free vehicular networks, with the aim of efficiently maximising vehicular coverage.

In [83], the authors propose using a Twin-delayed Deep Deterministic Policy gradient (TD3) based optimization scheme to reduce energy consumption by jointly optimizing offloading decision and the allocation of both the communication resources and computing resources. In [84], the authors propose a novel RL-based

scheme that benefits from the openness of O-RAN to provide two levels of centralised multi-agent decision making and decentralised single agent execution for proper service types by following the objective for maximising the system capacity while guaranteeing the defined QoS. In [85], the authors propose using a computationally efficient algorithm namely an ANN using a masked multi-head attention network module to solve a min-max downlink power control optimization problem subject to power constraints in the presence of pilot contamination. In [86], the authors use a DDN and train it in an unsupervised manner to learn the optimum user power allocation with objective of maximising the minimum user rate. In [87], the authors propose using a Double Cascade Correlation Network (DCCN) to optimize the RIS coefficients and based on the results from DDCN, an Inverse-variance DRL (IV-DRL) to address the UAV trajectory optimization problem in a UAV-RIS-aided CF-mMIMO hybrid NOMA/OMA network. In [88], the authors propose DRL approach Single Agent Reinforcement Learning (SARL) and Multi Agent Reinforcement Learning (MARL) to maximise EE by optimizing user association and content caching. In [89], the authors propose using TD3 for uplink power allocation with a goal of optimizing two conflicting metrics, namely, sum rate and fairness. In [90], the authors propose using a DDPG for uplink power control with the aim of maximising global sum rate under individual user QoS constraints. In [91], the authors propose using a Distributed Distributional Deterministic Policy Gradients (D4PG) to optimize beam-forming for uplink transmission for a cell-free network. In [92], the authors use DNN to perform power allocation optimization, where they use a heuristic power allocation based on large scale fading parameters as the input to the DNN. In [93], the authors proposing using a DDPG feed-forward NN to solve downlink max-min power control and max-product power control problems. In [94], the authors propose using DQN and DDPG to maximise SSE in the microwave domain. In [95], authors propose using model free DRL algorithms namely DDPG and SAC for optimal AP selection with the goal of maximising EE subject to minimum rate constraints for all devices in the system. In [96], the authors propose using a DRL model with hybrid action space for joint AP selection and power control with the aim of maximising

EE under SSE requirements. In [97], the authors propose RSMA DRL to optimize power allocation to common and private streams with objective of maximising EE whilst considering transceiver impairment which cause distortion in channel estimation and downlink transmission using RSMA. In [98], the authors propose a Prioritized Experience Reply (PER) with DDPG for joint power control and phase adjustments with objective of optimizing network EE whilst dealing with pilot attacks in RIS aided CF-mMIMO. In [99], the authors use a DQN to maximise continuous sum fair spectral efficiency by adjusting user duplex mode in Network-Assisted Full Duplex (NAFD) CF-mMIMO network. In [100], the authors propose a DDPG scheme joint on/off mode selection of APs and offloading ratio of users with the objective of minimizing total system delay under AP power limit constraint. In [101], the authors propose using DDPG for cooperation clustering and downlink power control with the aim of maximising the long-term global average proportional fairness. In [102], the authors propose using a centralised trained and decentralised MADDPG for jointly optimizing user power control and IRS's beam-forming. In [103], the authors propose using MARL for joint communication and computing resource allocation in CF-mMIMO enabled MEC network with a goal of minimizing energy consumption of the user while meeting ultra-low delay constraints.

In [104], the authors investigate a distributed design integrating user-centric association with local sub-connected hybrid precoding in mmWave CF-mMIMO systems, aiming to maximise the long-term global EE under quality-of-service and power budget constraints. The joint optimization problem is formulated as a Markov Game problem, and a novel Weighted Critic Update Multi-Agent Twin-delayed Deep Deterministic Policy Gradient (WCU-MATD3) algorithm is proposed to solve it. In [105], the authors explore the downlink power allocation challenge for CF-mMIMO systems. Focusing on EE optimization, using a distributed power allocation framework driven by DNN. In [106], the authors study a distributed downlink power control problem in user-centric cell-free systems, where each UE is served by a subset of APs to reduce front-haul requirements and computational complexity at the AP side. They propose a distributed downlink

power control method based on a DNN to enhance the long-term downlink data rate across the entire network. In [107], the authors propose a deep learning-based channel estimation approach that makes use of a deep neural network to provide a scalable and efficient channel estimation scheme. In [108], the authors propose using a deployment technique using SAC, a deep reinforcement learning method, to autonomously determine the optimal placement of the UAV within a given pre-defined coverage area, thereby maximising channel capacity and throughput. In [109], the authors propose a synergistic DRL cell-free framework, which utilizes the Advantage Actor-Critic (A2C) to jointly and dynamically adjust the idle/active states of APs and allocate the downlink transmit power. In [110], the authors introduce a reinforcement learning-based approach for optimal parent selection in Destination Oriented Directed Acyclic Graph (DODAG) based routing within cell-free massive MIMO networks. In [111], the authors propose Spatio-Temporal Graph Neural Networks (STGNN)-based CF-mMIMO networks with a maximal benefit-cost ratio for cost-effective development. In [112], the authors propose a GNN designed to rapidly maximise EE in CFmMIMO networks. In [113], the authors propose a Federated K-means (FedEst) framework for channel estimation that takes into account factors such as traffic load, user mobility, computational power, and network topology. In [114], the authors develop a novel approach using an unsupervised Heterogeneous Graph Neural Network (HetGNN), which effectively captures the complex relationships between AP and UE nodes, that directly optimizes AP selection through its loss function. In [115], the authors propose a novel deep learning-based Position-domain Channel extrapolation framework (named PCEnet) for CF-mMIMO systems. The user's position, which contains significant channel characteristic information, can greatly enhance the efficiency of channel acquisition.

2.8 Chapter Summary

In this chapter, the background of key concepts critical to the work proposed in this dissertation is provided. The background of CF-mMIMO and its different

architectures are explored. The differences between UC-CF-mMIMO and CF-mMIMO are explained. The differences between CD-NOMA and PD-NOMA are explained as well as the basics of how PD-NOMA works. SWIPT and its two architectures are explained in detail. We provide the detailed background on the key differences, namely, the frame structure differences between TS-SWIPT and PS-SWIPT. The use cases of UAVs in literature are discussed. A detailed background on DRL is provided. In addition to the provided background, an extensive literature review of concepts namely, NOMA, UAVs, SWIPT and DRL with CF-mMIMO was conducted.

Key findings in the literature review are as follows:

- CF-mMIMO promises to eliminate network issues that affect edge users whilst increasing SE. The main idea behind CF-mMIMO is that, all APs simultaneously serve all the UE in a given area. The performance of CF-mMIMO has been shown to scale up as the number of APs increases relative to the number of UE [7]. However, this brings up the question about scalability in scalable CF-mMIMO [11].

- A UC approach has been explored in literature with the main idea being, instead all APs jointly serving all UE, each UE is served by subset of APs. A UC-CF-mMIMO is particularly important when considering a finite power budget and when the number of UE is the same or greater than the number of APs [34],[36],[38].

- Another technique that promises to increase SE is NOMA. CF-mMIMO with NOMA has been shown to have performance gains when compared to CF-mMIMO using OMA [38]. However, the performance gains from combining CF-mMIMO with NOMA require precise PA as random power allocation may lead to suboptimal results. Similarly, random UA in UC-CF-mMIMO can lead to suboptimal performance.

- DRL has been used for resource allocation problems such power allocation and user association in CF-mMIMO particularly for UC-CF-mMIMO and

CF-mMIMO with NOMA [82],[93],[96]-[98],[103]. The idea of incorporating energy harvesting in CF-mMIMO has also been explored, particularly, SWIPT has been explored with aim of improving energy efficiency of CF-mMIMO networks [46],[49].

CHAPTER 3

Deep Reinforcement Learning aided User Association and Power Allocation

3.1 Introduction

In this chapter, we explore a UC-CF-mMIMO network with NOMA where we use UAVs as mobile APs, we incorporate SWIPT and use a DDPG for joint user association and power allocation. We investigate the impact of hyper-parameter tuning on model performance and compare the performance of the model to random power allocation in CF-mMIMO with and without random user association.

The contributions can be summarized as follows:

- We formulate an objective function to maximise DL SSE whilst considering minimum required DL and UL SE for all UE, as well as EH time allocation. To the best of the author's knowledge, no prior work in the existing literature has considered both the DL and UL SE constraints simultaneously, whilst also considering energy harvesting time allocation in CF-mMIMO.
- We use DDPG for joint power allocation and user association to maximise DL SSE and find an optimal time allocation, using a heuristic approach, for EH for the frame structure presented in the proposed system model. We also show the impact of hyper-parameter tuning on DDPG model's training performance.

In Section 3.2, we present the system model. In Section 3.3 we formulate the objective function, followed by the DRL solution to the objective function in Section 3.4. In Section 3.5, we analyse the inference computational complexity of the DDPG model. Section 3.6 outlines the simulation parameters, and Section 3.7 presents the results. Finally, we conclude in Section 3.8.

3.2 System Model

Consider CF-mMIMO system with NOMA as shown in Figure 3.1, where there is a set $\mathcal{M} \triangleq \{1, 2, 3 \dots M\}$ of UAVs connected to a CPU through perfect backhaul links that serve a set of $\mathcal{K} \triangleq \{1, 2, 3 \dots K\}$ of UE where $K = L \times N$. The UAVs are equipped with Z antennas while the UE are equipped with a single antenna. The UAVs act as mobile APs where the UAVs serve UE in the same time-frequency resource block through Time Division Duplex (TDD). Each time coherence interval, denoted by τ_c, consists of four phases as illustrated in Figure 3.2: UL training τ_p, DL energy harvesting τ_e, DL data transmission τ_d, and UL data transmission τ_u. In the system model, we adopt a TS-SWIPT architecture instead of a PS-SWIPT architecture. The primary reason is that TS-SWIPT is simpler and more cost-effective to implement from a hardware standpoint. In contrast, PS-SWIPT requires a power splitter at the receiver, which increases the complexity of the device circuitry and raises hardware costs. The UE are grouped into L clusters, with each cluster comprising N users. One UE can belong to only one cluster where the set of L clusters is denoted by $\mathcal{L} \triangleq \{1, 2, 3 \dots L\}$. We assume the UE and UAVs have the ability to move. The horizontal coordinates of the n-th UE in the l-th NOMA cluster are denoted by (i_{nl}, j_{nl}) and the horizontal coordinates of the UAVs are denoted by (i_m, j_m). It is assumed that all UAVs will operate at fixed altitude denoted by H, the vertical distance between the ground and UAVs. While some studies in the literature allow UAV operating heights to vary based on system demands, such research typically focuses on UAV trajectory optimization, which is beyond the scope of this work. Nonetheless, the impact of different fixed operating heights on energy harvesting will be evaluated later in

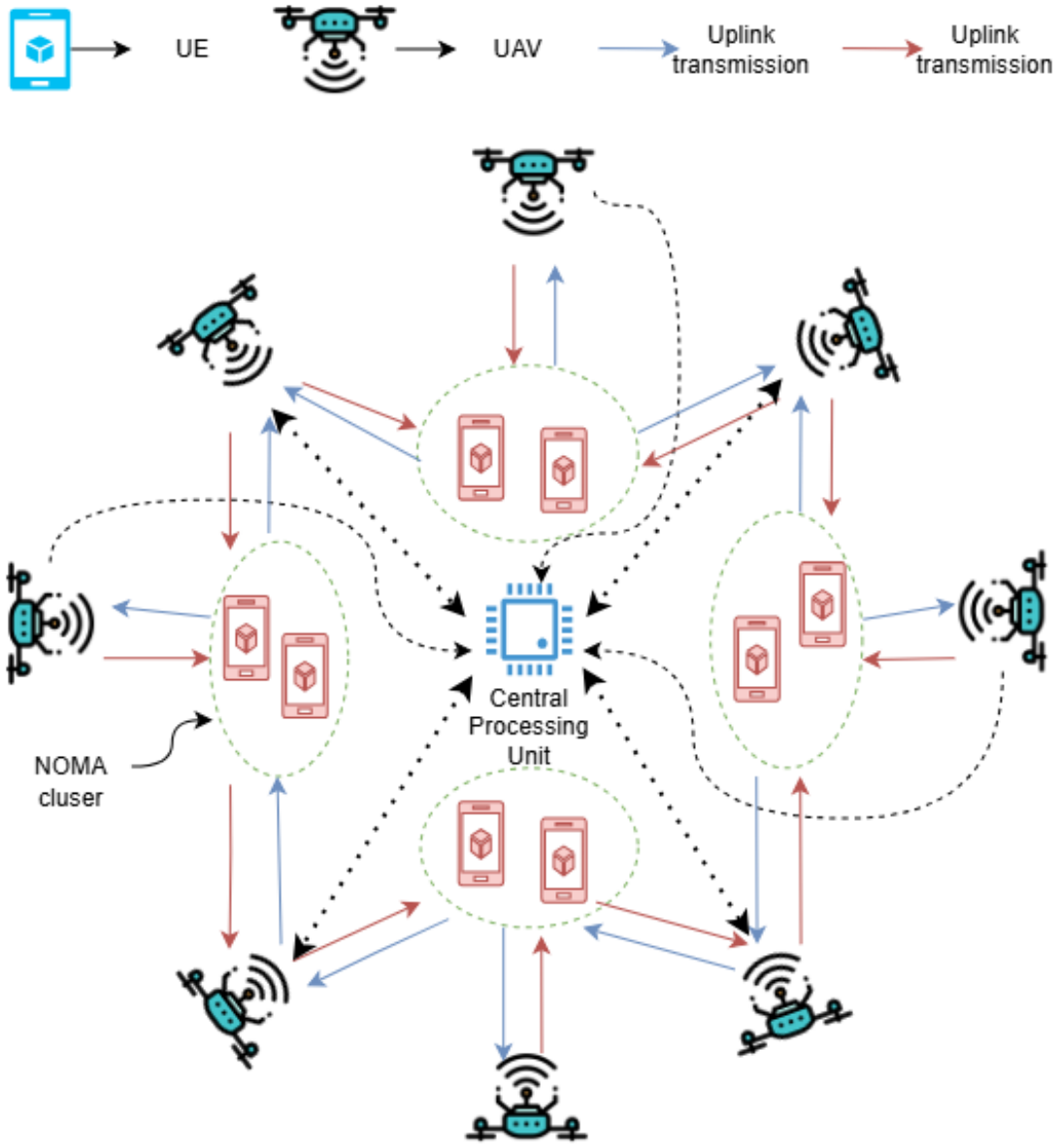


FIGURE 3.1: System model of CF-mMIMO with NOMA clusters.

the chapter. It is assumed that during the UL training phase, pairwise orthogonal pilot sequences are sent to the UAVs and are used for channel estimation. It is then assumed by leveraging channel reciprocity that the DL channels can be obtained.

Notations: Uppercase letters denote matrices $|\cdot|$,$(\cdot)^H$,$(\cdot)^T$,$(\cdot)^*$ and $\|\cdot\|_2$ correspond to the cardinality, the Hermitian transpose, the transpose, the conjugate and the L2-norm operators, respectively. $E\{\cdot\}$ represents the expectation operation. $\mathcal{CN}(\mu, \vartheta^2)$ stands for the for circularly symmetric complex Gaussian Random Variable (RV) with mean μ and variance ϑ^2.

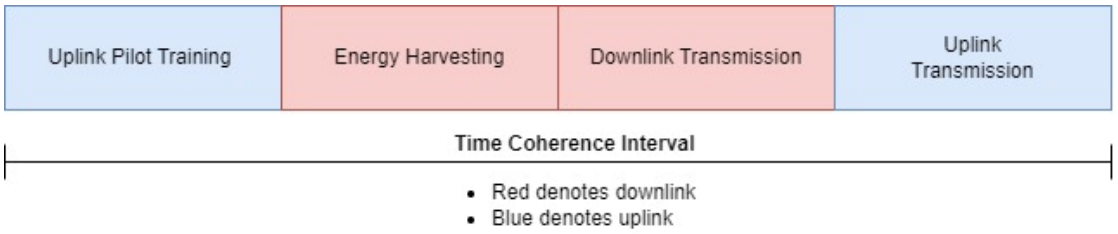


FIGURE 3.2: Time coherence interval frame structure.

TABLE 3.1: Table of notation.

Notation	**Description**	**Notation**	**Description**
M	Number of UAVs	γ_{mnl}	Channel estimates at the m UAV from the n-th UE in the l-th cluster
K	Number of UE	$\tilde{G}_m$	Full-rank matrix of full pilot zero-forcing precoder at the m-th UAV
L	Number of clusters	ϕ_l	l-th column of the identity matrix I_{τ_p}
τ_c	Time coherence interval	w_{ml}	Beam-forming vector at l-th cluster at m-th UAV
τ_p	Time allocated for UL training within τ_c	x_m	Transmitted signal from the m-th UAV
τ_e	Time allocated for energy harvesting within τ_c	A^D_{mnl}	User association allocation vector
τ_d	Time for DL data transmission within τ_c	ρ^{DL}_{mnl}	Normalised transmit power from the m-th UAV to the n-th UE in the l-th cluster
τ_u	Time for UL data transmission within τ_c	x^D_{mnl}	Data symbol intended for n-th UE in the l-th cluster

$[i_{nl}, j_{nl}]$	UE horizontal coordinates of the n-th UE in the l-th cluster	y_{nl}	Received signal at the n-th UE in the l-th cluster
$[i_m, j_m]$	UAV horizontal coordinates the m-th UAV	z_{nl}	Additive white Gaussian noise at the n-th UE in the l-th cluster
H	Vertical distance between ground and UAV	$\bar{y}_{nl}$	Received signal at n-th UE in l-th cluster after SIC
g_{mnl}	Channel gain between m-th UAV and n-th UE in l-th cluster	ζ_{nl}	SIC performance coefficient at n-th UE in l-th cluster
B_{mnl}	Large-scale fading coefficient between m-th UAV and the n-th UE in the l-th cluster	P_{mnl}^{EH}	Harvested power by n-th UE in l-th cluster from the m-th UAV
h_{mnl}	Small-scale Rayleigh fading coefficient between m-th UAV and the n-th UE in the l-th cluster	σ	Energy conversion efficiency coefficient
P_{mnl}^{LoS}	Probability of line-of-sight between the m-th UAV and UE	x_{nl}	Transmitted signal from n-th UE in l-th cluster
P_{mnl}^{NLoS}	Probability of non-line-of-sight between the m-th UAV and UE	ρ_{mnl}^{UL}	UL transmit power from n-th UE in l-th cluster to the m-th UAV
b_1, b_2	Environment constants	$\bar{y}_m$	Received signal at m-th UAV after SIC

d_{mnl}	Angle between the m-th UAV and the n-th UE in the l-th cluster	v_{nl}	Combining vector of the n-th UE in the l-th cluster
α_1	Line-of-sight path loss exponent	x^U_{mnl}	Data symbol intended for the m-th UAV from the n-th UE in the l-th cluster
α_2	Non-line-of-sight path loss exponent	z_m	Additive white Gaussian noise at m-th UAV
C_{mnl}	Horizontal distance between m-th UAV and n-th UE in the l-th cluster	SINR^D_{nl}	DL SINR of $0\text{n}-thUEinthe\text{l}-thcluster$
D_{mnl}	3D distance between m-th UAV and n-th UE in l-th cluster	R^{DL}_{nl}	DL SE of n-th UE in l-th cluster
$\varnothing_l$	Pilot sequence sent by UE in l-th cluster	$R^{\text{DL}}_{\text{sum}}$	SSE of all UE during DL transmission
Y^p_m	Training signal received at m-th UAV	SINR^U_{nl}	UL SINR of the n-th UE in the l-th cluster
ρ_{mnl}	Normalized UL training power of the n-th UE in the l-th cluster to the m-th UAV	R^{UL}_{nl}	UL SE of n-th UE in l-th cluster
W^p_m	Additive noise matrix at m-th UAV	$R^{\text{UL}}_{\text{sum}}$	SSE of all UE during UL transmission
A^p_{mnl}	Allocation vector to assign the m-th UAV to the n-th UE in the l-th cluster	Z	The number of antenna equipped at each AP

$\hat{g}_{mnl}$	Estimated channel gain between the m-th UAV and n-th UE in the l-th cluster	$P^m_{\max}$	Maximum transmit power at m-th UAV
e_{mnl}	Channel estimation error between the m-th UAV and n-th UE in the l-th cluster	$\bar{R}_{nl}$	Minimum required SE at n-th UE in l-th cluster
EH_{mnl}	Energy harvested by n-th UE in l-th cluster	ρ^t_{mnl}	Transmit power from m-th UAV to n-th the UE in the l-th cluster during EH

3.2.1 Channel Model

The channel gain, g_{mnl} between the n-th UE in the l-th cluster and m-th UAV is given by

$$g_{mnl} = \sqrt{B_{mnl}} h_{mnl}, \tag{3.1}$$

where B_{mnl} and h_{mnl} represent the effects of large-scale fading and small-scale Rayleigh fading, respectively [7]. We assume that h_{mnl} is Independent and Identically Distributed (i.i.d) $\mathcal{CN}(0,1)$ among different channels. The large-scale fading is governed by the probability of LoS transmission between the UAV and UE. The probability of LoS is given by

$$P^{LoS}_{mnl} = \frac{1}{1 + b_1 \times exp(-b_2(180 \times \frac{d_{mnl}}{\pi} - b_2))}, \tag{3.2}$$

where b_1 and b_2 are constants related to the environment, $d_{mnl} = arctan(\frac{H}{C_{mnl}})$ is the angle between the m-th UAV and the n-th UE in the l-th cluster, the

probability of NLoS transmission is denoted by $P_{mnl}^{NLoS} = 1 - P_{mnl}^{LoS}$ [8]. The large-scale fading channel gain is given by

$$\beta_{mnl} = P_{mnl}^{LoS} D_{mnl}^{\alpha_1} + P_{mnl}^{NLoS} D_{mnl}^{\alpha_2}, \tag{3.3}$$

where α_1 and α_2 are path loss exponents for the LoS and NLoS, respectively [8]. The term D_{mnl} denotes the three-dimensional distance between the m-th UAV and the n-th UE in the l-th cluster. It is calculated using the horizontal distance C_{mnl} between them and the fixed operating altitude of the UAV are given by

$$C_{mnl} = \sqrt{(i_m - i_{nl})^2 + (j_m - j_{nl})^2}, \tag{3.4}$$

$$D_{mnl} = \sqrt{{C_{mnl}}^2 + H^2}. \tag{3.5}$$

3.2.1.1 Channel Estimation

There are two phases relating to the channel, namely the UL phase and DL phase. The present research focuses on maximising the SE for DL and UL channels. However, the UL phase is also used for training, where all the UE send their training pilots to the UAVs for channel estimation. By allowing the UE in the same cluster to use the same pilot sequence, and ensuring that pilot sequences among different clusters are pairwise orthogonal (requiring $\tau_p \geq L$), the channel estimation overhead can be minimized. However, this introduces inter-cluster contamination, where UAVs estimating channels of UE belonging to the same cluster experience pilot contamination. The pilot sequence sent from the UE in the l-th cluster is denoted by $\boldsymbol{\phi}_l \in \mathbb{C}^{\tau_p \times 1}$ with $l \in \{1, 2, \ldots, \tau_p\}$, satisfying the orthogonality condition: $\|\boldsymbol{\phi}_l\|_2^2 = \tau_p, \quad \text{and} \quad \boldsymbol{\phi}_l^H \boldsymbol{\phi}_{l'} = 0 \quad \text{if } l \neq l'$.

3.2.1.2 Uplink Training Phase

The training signal received at m-th UAV is given by

$$Y_m^p = \sum_{l=1}^{L} \sum_{n=1}^{N} A_{mnl}^p \sqrt{\rho_{mnl}}\, g_{mnl} \boldsymbol{\phi}_l^H + W_m^p, \tag{3.6}$$

where, ρ_{mnl} and W_m^p represent the normalized transmit power of the n-th UE in the l-th cluster, and the additive noise matrix at the m-th UAV, respectively [7]. The elements of W_m^p follow a complex normal distribution, i.e., $\mathcal{CN}(0,1)$. A_{mnl}^p is the pre-determined allocation vector used to assign UAVs to UE. Given Y_m^p, the m-th UAV estimates g_{mnl}, the channel gain between the UAV and the UE. The received signal $\hat{\mathbf{y}}_m^p$ at the m-th UAV is given by

$$\hat{y}_m^p = Y_m^p \phi_l = \tau_p \sum_{n=1}^{N} A_{mnl}^p \sqrt{\rho_{mnl}}\, g_{mnl} + W_m^p \phi_l. \tag{3.7}$$

The MMSE estimate of the g_{mnl} is given by

$$\hat{g}_{mnl} = \mathbb{E}\left\{ g_{mnl} \left(\hat{y}_m^p\right)^H \right\} \left(\mathbb{E}\left\{ \hat{y}_m^p \left(\hat{y}_m^p\right)^H \right\} \right)^{-1} \hat{y}_m^p = \mathbf{u}_{mnl} \hat{y}_m^p, \tag{3.8}$$

where $u_{mnl} = \frac{A_{mnl}^p \sqrt{\rho_{mnl}}\, B_{mnl}}{\tau_p \sum_{n'=1}^{N} \rho_{mn'l} B_{mn'l} + 1}$. The estimation error vector of the channel gain g_{mnl} is given by

$$e_{mnl} = g_{mnl} - \hat{g}_{mnl}, \tag{3.9}$$

where the elements of e_{mnl} and $\hat{g}_{mnl}$ are i.i.d RVs distributed as $\mathcal{CN}\left(0, (B_{mnl} - \gamma_{mnl}) I_K\right)$ and $\mathcal{CN}\left(0, \gamma_{mnl} I_K\right)$, respectively[7]. There is no cooperation between UAVs for channel estimates; each UAV estimates the channel with each user independently

$$\gamma_{mnl} = \frac{A_{mnl}^p\, \tau_p\, \rho_{mnl}\, \beta_{mnl}^2}{\tau_p \sum_{n'=1}^{N} \rho_{mn'l} B_{mn'l} + 1}. \tag{3.10}$$

Due to pilot contamination arising from UAVs estimating channels for UE within the same cluster, the relationship between the channel estimates of UE n and UE n' (with $n \neq n'$) in the l-th cluster, at the m-th UAV, can be expressed as

$$\hat{g}_{mnl} = \frac{A^p_{mnl}\sqrt{\rho_{mnl}}\, B_{mnl}}{\sqrt{\rho_{mn'l}}\, B_{mn'l}} \hat{g}_{mn'l}. \tag{3.11}$$

3.2.2 Downlink Transmission

As mentioned, CSI to pre-code the transmit signals in the DL is acquired by leveraging channel reciprocity. A full pilot Zero Forcing (fpZF) precoder is adapted to cancel inter-cluster interference while considering intra-cluster interference. Each UAV computes the fpZF precoding using its local CSI only, leading to a distributed implementable algorithm. The full-rank matrix $\mathrm{G} \in \mathbb{C}^{K\times\tau_p}$ of fpZF precoder at the m-th UAV is given by

$$\widetilde{G}_m = Y^p_m \emptyset, \tag{3.12}$$

where $\phi = [\phi_1, \phi_2, \ldots..\phi_{\tau p}] \in \mathbb{C}^{K\times\tau_p}$ denotes the collection of τ_p orthogonal pilot sequences [7]. The channel estimate g_{mnl} can be written as

$$\hat{g}_{mnl} = u_{mnl}\widetilde{G}_m\varphi_l, \tag{3.13}$$

where φ_I is the l-th column of the identity matrix I_{τ_p}. From (3.12) and (3.13), the beam-forming vector w_{ml} at the l-th cluster at m-th UAV is given by

$$w_{ml} = \frac{\widetilde{G}_m(\widetilde{G}^H_m\widetilde{G}_m)^{-1}\varphi_l}{\sqrt{\mathbb{E}\|(\widetilde{G}^H_m\widetilde{G}_m)^{-1}\varphi_l\|^2_2}}, \tag{3.14}$$

where each AP has l precoding vectors, one per cluster (pilot). fpZF preorder has been shown to suppress the inter-cluster interference by only utilizing local

CSI rather than shared CSI amongst the APs, as well as increasing signal power when the number of antennas on each AP increases, due to the array gain [7]. The transmitted signal $x_m \in \mathbb{C}^{K\times 1}$ from the m-th UAV is expressed as

$$x_m = \sum_{l=1}^{L}\sum_{n=1}^{N} A_{mnl}^{D}\sqrt{\rho_{mnl}^{downlink}}\, w_{ml} x_{mnl}^{D}, \tag{3.15}$$

where $\rho_{mnl}^{downlink}$ is the normalised transmit power (normalised by the noise power at m-th UAV), x_{mnl}^{D} is the symbol intended for n-th UE in the l-th cluster from the m-th AP and A_{mnl}^{D} is the user association allocation vector for DL data transmission.

It is assumed that the UE is capable of simultaneously decoding information and harvesting RF energy during the DL phase. The signal received at the n-th UE in the l-th cluster is given by

$$\begin{aligned} y_{nl} &= \sum_{m=1}^{M} g_{mnl} x_m + z_{nl} \\ &= \underbrace{\sum_{m=1}^{M} A_{mnl}^{D}\sqrt{\rho_{mnl}^{downlink}}\, g_{mnl}^{H} w_{ml} x_{mnl}^{D}}_{Desired\ Signal}, \\ &+ \underbrace{\sum_{m=1}^{M}\sum_{n\prime\neq n}^{N} A_{mnl}^{D}\sqrt{\rho_{mnl}^{downlink}} g_{mn\prime l}^{H} w_{ml} x_{mn\prime l}^{D}}_{Intra-cluster\ interference\ before\ SIC} + \\ &\underbrace{\sum_{m=1}^{M}\sum_{l\prime\neq l}^{L}\sum_{n'=1}^{N} A_{mn\prime l\prime}^{D}\sqrt{\rho_{mn\prime l\prime}^{downlink}} g_{mn\prime l\prime}^{H} w_{ml} x_{mn\prime l\prime}^{D}}_{Inter-cluster\ interference} + z_{nl}, \end{aligned} \tag{3.16}$$

where z_{nl} $\mathcal{CN}(0,1)$ is the AWGN at n-th UE in the l-th cluster [7]. By adhering to the NOMA principles, the UE in l-th clusters are arranged in the descending order of channel gain $\|g_{m1l}\|_2 \geq \|g_{m2l}\|_2 \ldots \geq \|g_{mnl}\|_2$, where UE 1 and N are users with the strongest and weakest channel gains in l-th cluster. The n-th UE in the l-th cluster with the strongest channel gain first decodes the signals of UE with weaker gains, $n'l \geq nl$, then its own signal is successively decoded after removing the interference from the UE with weaker channel gains. Denoted by $SINR_{nl}^{n'l}$ and $SINR_{n'l}^{n'l}$ the signal-to-interference-plus-noise ratios (SINRs) in decoding the signal of UE $n'l$ by UE nl and itself, respectively.

Since perfect SIC is practically impossible to achieve due the to the effect of intra cluster-pilot contamination and channel estimation errors, the received signal at the n-th UE in the l-th cluster after SIC is given by

$$\bar{y}_{nl} = \underbrace{\sum_{m=1}^{M} A_{mnl}^{D} \sqrt{\rho_{mnl}^{downlink}}\, g_{mnl}^{H} w_{ml} x_{mnl}^{D}}_{Desired\ signal}$$

$$+ \underbrace{\sum_{m=1}^{M} \sum_{n'=1}^{n-1} A_{mn'l}^{D} \sqrt{\rho_{mn'l}^{downlink}} g_{mn'l}^{H} w_{ml} x_{mn'l}^{D}}_{Intra-cluster\ interference\ after\ SIC}$$

$$+ \underbrace{\sum_{m=1}^{M} \sum_{n'=n+1}^{N} \sqrt{\zeta_{n'l}} A_{mn'l}^{D} \sqrt{\rho_{mn'l}^{downlink}} g_{mn'l}^{H} w_{ml} x_{mn'l}^{D}}_{Intracluster\ interference\ due\ to\ imperfect\ SIC} \tag{3.17}$$

$$+ \underbrace{\sum_{m=1}^{M} \sum_{l' \neq l}^{L} \sum_{n'=1}^{N} A_{mn'l'}^{D} \sqrt{\rho_{mn'l'}^{downlink}} g_{mn'l'}^{H} w_{ml'} x_{mn'l'}^{D}}_{inter-cluster\ interference} + z_{nl},$$

where ζ_{nl} is the general SIC performance coefficient at n-th UE in the l-th cluster [7]. When $\zeta_{nl} = 0$, perfect SIC occurs and when the $\zeta_{nl} = 1$, worst possible SIC occurs (imperfect SIC) while $0 < \zeta_{nl} \leq 1$ indicates imperfect SIC.

3.2.3 Energy Harvesting and Energy Consumption Model

During each time coherence interval, it is assumed that each UE requires a minimum amount of energy to operate its circuitry. The total amount of energy required by UE to operate its circuitry during τ_c is given by

$$E^C_{total} = \tau_c \left|\sqrt{p_c}\right|^2 = \tau_p \left|\sqrt{p_c}\right|^2 + \tau_d \left|\sqrt{p_c}\right|^2 + \tau_e \left|\sqrt{p_c}\right|^2 + \tau_u \left|\sqrt{p_c}\right|^2, \tag{3.18}$$

where p_c is the constant power required to operate the UE circuitry, τ_p is the time allocated for pilot training, τ_d is the time allocated for DL transmission, τ_e is the time allocated for energy harvesting, τ_u is the time allocated for UL transmission and τ_c is the time coherence interval ($\tau_c \geq \tau_p + \tau_d + \tau_e + \tau_u$).

3.2.3.1 Energy Consumption during Pilot Training

During pilot training, each UE transmits training pilots to the APs for channel estimation. Therefore, in addition to the energy used by UE to operate its circuitry, energy is also required to transmit the training pilots. The amount of energy used for pilot training transmission by each UE is given by

$$E^P = \tau_p \left|\sqrt{\rho_{mnl}}\right|^2, \tag{3.19}$$

where ρ_{mnl} is the constant power allocated for pilot training and τ_p is the time allocated for pilot training. The total energy consumed during the pilot training process is simply the energy required by the UE to operate its circuitry and the energy required to for pilot transmission is given by

$$E_{total}^{P} = E^{P} + \tau_p \, |\sqrt{p_c}|^2. \tag{3.20}$$

3.2.3.2 Energy Consumption during Downlink Transmission

During DL transmission the UE receive transmitted signals from the APs. Therefore, the only energy consumed by the UE during DL transmission is the energy required to operate its circuitry is given by

$$E_{total}^{D} = \tau_d \, |\sqrt{p_c}|^2. \tag{3.21}$$

3.2.3.3 Energy Consumption during Energy Harvesting

The harvested energy is used for the UL training and UL data transmission. The energy harvested by the n-th UE in l-th cluster is given by

$$E_{nl}^{H} = \sigma \ \tau_e \left| \sum_{m=1}^{M} \sqrt{\rho_{mnl}^{t}} g_{mnl} x_m \right|^2, \tag{3.22}$$

where σ is the energy conversion efficiency coefficient, τ_e is the time allocated for energy harvesting and ρ_{mnl}^{t} the transmit power from the m-th UAV to n-th UE in l-th cluster during energy harvesting [46]. However, during energy harvesting, the UE still requires energy to operate its circuitry. Therefore, the net energy harvested is simply the energy harvested minus the energy used by UE during energy harvesting phase. The net harvested energy is given by

$$E_{total_{nl}}^{H} = E_{nl}^{H} - \tau_e \, |\sqrt{p_c}|^2. \tag{3.23}$$

3.2.3.4 Energy Consumption during Uplink Transmission

During UL transmission, UE transmit signals to APs. Therefore, the energy required by each UE during UL transmission is the energy to operate its circuitry and the energy required to transmit. The total energy consumption of each UE is given by

$$E^{U}_{total} = \tau_u \left| \sqrt{\rho^{uplink}_{mnl}} \right|^2 + \tau_u \left| \sqrt{p_c} \right|^2, \tag{3.24}$$

where ρ^{uplink}_{mnl} is the harvested power used by UE for UL data transmission and τ_u is the time allocated for UL transmission. The harvested UL transmission power is given by

$$E^{H}_{total_{nl}} = E^{U}_{total} + E^{P}_{total} + \ E^{D}_{total}$$

$$E^{H}_{total_{nl}} = \ \tau_u \left| \sqrt{\rho^{uplink}_{mnl}} \right|^2 + \tau_u \left| \sqrt{p_c} \right|^2 + E^{P}_{total} + \ E^{D}_{total}$$

$$\tau_u \left| \sqrt{\rho^{uplink}_{mnl}} \right|^2 = E^{H}_{total_{nl}} - (\tau_u \left| \sqrt{p_c} \right|^2 + E^{P}_{total} + \ E^{D}_{total}), \tag{3.25}$$

$$\bar{E}^{H}_{nl} = E^{H}_{total_{nl}} - (\tau_u \left| \sqrt{p_c} \right|^2 + E^{P}_{total} + \ E^{D}_{total})$$

$$\rho^{uplink}_{mnl} = \frac{\bar{E}^{H}_{nl}}{\tau_u}$$

.

3.2.4 Uplink Transmission

During the UL data transmission, energy harvested during energy harvesting phase is used to transmit the data from the n-th user in the l-th cluster to m-th UAV. Similar to the transmitted signal from (3.15), the transmitted signal $x_{mnl} \in \mathbb{C}^{K\times 1}$ from the n-th UE in l-th cluster is given by

$$x_{nl} = \sum_{m=1}^{M} A_{mnl}^{D} \sqrt{\rho_{mnl}^{uplink}} v_{nl} x_{mnl}^{U}, \tag{3.26}$$

where ρ_{mnl}^{uplink} is the harvested power, x_{mnl}^{U} data symbol intended for the m-th UAV from n-th UE in the l-th cluster. However, unlike (3.15) during the UL transmission, a combining vector denoted by v_{nl} is used instead of beam-forming vector denoted by w_{ml}. Similar to (3.17), the received signal at the m-th UAV is given by

$$\begin{aligned}
\bar{y}_m = & \underbrace{\sum_{l=1}^{L}\sum_{n=1}^{N} A_{mnl}^{D} \sqrt{\rho_{mnl}^{uplink}}\, g_{mnl}^{H} v_{nl} x_{mnl}^{U}}_{Desired\ signal} \\
& + \underbrace{\sum_{l=1}^{L}\sum_{n'=1}^{n-1} A_{mn'l}^{D} \sqrt{\rho_{mn'l}^{uplink}} g_{mnl}^{H} v_{nl} x_{mn'l}^{U}}_{Intra-cluster\ interference\ after\ SIC} \\
& + \underbrace{\sum_{l=1}^{L}\sum_{n'=n+1}^{N} A_{mn'l}^{D} \sqrt{\zeta_{n'l}} \sqrt{\rho_{mn'l}^{uplink}} g_{mnl}^{H} v_{n'l} x_{mn'l}^{U}}_{Intracluster\ interference\ due\ to\ imperfect\ SIC}
\end{aligned} \tag{3.27}$$

$$\underbrace{\sum_{m=1}^{M}\sum_{l\prime\neq l}^{L}\sum_{n'=1}^{N} A^{D}_{mn\prime l\prime}\sqrt{\rho^{uplink}_{mn\prime l\prime}}\, g^{H}_{mnl} v_{n\prime l\prime} x^{U}_{mnl}}_{inter-cluster\ interference} + z_m.$$

where $z_m \;\; \mathcal{CN}(0,1)$ is the AWGN at m-th AP and A^{D}_{mnl} is the user association allocation vector for DL data transmission. We assume the same user association allocation vector used for DL transmission will be used for UL transmission.

3.2.5 Downlink Transmission Performance

From (3.17), the DL SINR of the n-th UE in the l-th cluster is given by

$$SINR^{D}_{nl} = \frac{|DS|^2}{\mathbb{E}\{|BU|^2\}+\sum_{n\prime=1}^{n-1}\mathbb{E}\{|ICI|^2\}+\sum_{n'=n+1}^{N}\{|RICI|^2\}+\sum_{l\prime\neq l}^{L}\sum_{n'=1}^{N}\mathbb{E}\{|UI|^2\}+1}, \quad (3.28)$$

where

- DS = $\mathbb{E}\{\sum_{m=1}^{M} A^{D}_{mnl}\ \sqrt{\rho^{downlink}_{mnl}} g^{H}_{mnl} w_{ml}\}$, is the coherent beam-forming gain (desired signal) [7].
- BU = $(\sum_{m=1}^{M} A^{D}_{mnl}\ \sqrt{\rho^{downlink}_{mnl}}\ g^{H}_{mnl} w_{ml}$ - $\mathbb{E}\{\sum_{m=1}^{M} A^{D}_{mnl}\ \sqrt{\rho^{downlink}_{mnl}} g^{H}_{mnl} w_{ml}\})$, is the beam forming gain uncertainty [7].
- ICI = $\mathbb{E}\{\sum_{m=1}^{M} A^{D}_{mn'l}\sqrt{\rho^{downlink}_{mn'l}} g^{H}_{mn'l} w_{ml}\}$, is the intra-cluster interference after SIC [7].
- RICI = $\mathbb{E}\sum_{m=1}^{M}\sqrt{\zeta_{n'l}}\ A^{D}_{mn'l}\sqrt{\rho^{downlink}_{mn'l}} g^{H}_{mn'l} w_{ml}$, is the residual interference due to imperfect SIC [7].
- UI = $\sum_{m=1}^{M} A^{D}_{mn'l'}\sqrt{\rho^{downlink}_{mn'l'}} g^{H}_{mn'l'} w_{ml'}$, is the inter-cluster interference [7].

The closed-form expression for the DL SE of n-th UE in the l-th cluster is given by

$$R_{nl}^{Downlink} = \left(1 - \frac{(\tau_p + \tau_e + \tau_u)}{\tau_c}\right) \log_2\left(1 + SINR_{nl}^{D}\right). \tag{3.29}$$

From (3.29), the DL SSE of CF-mMIMO-NOMA is given by

$$R_{sum}^{Downlink} = \sum_{l=1}^{L} \sum_{n=1}^{N} R_{nl}^{Downlink}$$

$$= \sum_{l=1}^{L} \sum_{n=1}^{N} \left(1 - \frac{(\tau_p + \tau_e + \tau_u)}{\tau_c}\right) \log_2\left(1 + SINR_{nl}^{D}\right). \tag{3.30}$$

3.2.6 Uplink Transmission Performance

The UL transmission performance can be derived in a similar manner to the DL transmission performance, the UL SINR of the n-th UE in the l-th cluster is given by

$$SINR_{nl}^{U} = \frac{|DS|^2}{\mathbb{E}\left\{|BU|^2\right\} + \sum_{n'=1}^{n-1} \mathbb{E}\left\{|ICI|^2\right\} + \sum_{n'=n+1}^{N} \left\{|RICI|^2\right\} + \sum_{l' \neq l}^{L} \sum_{n'=1}^{N} \mathbb{E}\left\{|UI|^2\right\} + 1}, \tag{3.31}$$

where

- DS = $\mathbb{E}\{\sum_{m=1}^{M} A_{mnl}^{D} \sqrt{\rho_{mnl}^{uplink}} g_{mnl}^{H} w_{ml}\}$, is the coherent beam-forming gain (desired signal).
- BU = ($\sum_{m=1}^{M} A_{mnl}^{D} \sqrt{\rho_{mnl}^{uplink}} g_{mnl}^{H} w_{ml}$ - $\mathbb{E}\{\sum_{m=1}^{M} A_{mnl}^{D} \sqrt{\rho_{mnl}^{uplink}} g_{mnl}^{H} w_{ml}\}$), is the beam forming gain uncertainty.

- ICI = $\mathbb{E}\{\sum_{m=1}^{M} A^{D}_{mn'l}\sqrt{\rho^{uplink}_{mn'l}}g^{H}_{mn'l}w_{ml}\}$, is the intra-cluster interference after SIC.
- RICI=$\mathbb{E}\sum_{m=1}^{M}\sqrt{\zeta_{n'l}}\ A^{D}_{mn'l}\sqrt{\rho^{uplink}_{mn'l}}g^{H}_{mn'l}w_{ml}$, is the residual interference due to imperfect SIC.
- UI = $\sum_{m=1}^{M} A^{D}_{mn'l'}\sqrt{\rho^{uplink}_{mn'l'}}g^{H}_{mn'l'}w_{ml'}$, is the inter-cluster interference.

The closed-form expression for the uplink SE of n-th UE in the l-th cluster is given by

$$R^{uplink}_{nl} = \left(1 - \frac{(\tau_p + \tau_e + \tau_d)}{\tau_c}\right)\log_2\left(1 + SINR^{U}_{nl}\right). \tag{3.32}$$

From (3.32), the uplink SSE of CF-mMIMO-NOMA is given by

$$R^{uplink}_{sum} = \sum_{l=1}^{L}\sum_{n=1}^{N} R^{uplink}_{nl}$$

$$= \sum_{l=1}^{L}\sum_{n=1}^{N}\left(1 - \frac{(\tau_p + \tau_e + \tau_d)}{\tau_c}\right)\log_2\left(1 + SINR^{U}_{nl}\right). \tag{3.33}$$

3.2.7 NOMA Clustering

In NOMA, user clustering plays an important role, randomly pairing users into NOMA groups leads to suboptimal performance and may also lead to worse performance when compared to OMA [24]. In non-CF-mMIMO architectures, the users are usually paired according to their relative channel quality determined by their channel gains, where a channel quality imbalance in between the users is encouraged. SIC is performed at the receiver with the optimal detection order progressing from the strongest to the weakest user. This approach effectively reduces interference contamination from users whose normalized channel gains are

lower than those of the stronger users. However, traditional clustering algorithms in literature are not effective in a CF-mMIMO architectures, since the users are served by multiple APs simultaneously, clustering becomes difficult due to the fact that a strong user at one AP might be a weak user at another AP.

To address this issue papers in literature have come up with different algorithms specifically for NOMA clustering in a CF-mMIMO. In [9], the author uses an iterative optimization algorithm based on Successive Convex Programming (SCP) to find optimal NOMA pairs and as well as power allocation to maximise DL sum SSE. In [7], two effective unsupervised machine learning algorithms K-means++ and an enhanced version of K-means++ are proposed for user clustering in CF-mMIMO systems employing NOMA. The study evaluates their performance in forming disjoint user clusters with the goal of maximising DL SSE. In [116], two clustering algorithms namely, UE similarity combination algorithm and mean shift similarity combination algorithm, are proposed as effective clustering algorithms.

In the proposed system model, we adopt and slightly modify the UE similarity combination algorithm presented in [116], as shown in Algorithm-1. The main idea behind UE similarity combination algorithm is to pair UE according to their large-scale fading coefficients across all APs. Consider $\beta_{\mathbf{n}} \triangleq [\beta_{m1}, \beta_{m2}, \cdots \beta_{mn}]^T \in \mathbb{R}^{M\times 1}$ which represents the large-scale fading coefficients of all APs associated with UE$_n$.We assume thatβ_n is collected before performing user clustering and there can only be two UE in a cluster. We assume that $\beta_{\mathbf{n}}$ is from $UE_n, \forall_n \in N$ is distributed in an M-dimensional space and the m-th dimension is used to measure the β_{mn} from $UE_n, \forall_n \in N$ to AP_m. The M-dimensional space coordinate (β_{m1},β_{m2},...,β_{mn}) is used to represent the $UE_n, \forall_n \in N$ and the closer the UE are in the M-dimensional space, the more similar the UE are. As mentioned earlier, NOMA relies on channel quality imbalances between users. Therefore, it is necessary for UEs within the same cluster to have significant differences in the M-dimensional space. In [116] the UE with the most similar large scale channel gains denoted by β are put into one group and the other UE are divided into another group, where each group contains half the of UE. A UE is randomly selected and paired with one from the second group with the most dissimilar β,

this process is repeated until all UE are paired. However, as highlighted in [116], this approach has notable drawbacks. When there is a large disparity in β among UE in the two groups, the resulting clustering can become suboptimal or even highly ineffective. This issue is particularly pronounced in scenarios with a small number of UE in the system.

Algorithm 1 NOMA Clustering

Input: $\beta_n \ \forall_n \in N$, K is the number of UE in the system, $Bvec$ is the storage vector for β_n,
for $i \leftarrow K$ **do**
  Compute vector Euclidean norm $\|\beta_n\|$, and store value in $Bvec[i]$
end for
Sort UE according to $\|\beta_n\|$, in descending order using bubble sort:
for $j \leftarrow 1$ to $K-1$ **do**
  for $v \leftarrow 1$ to $K-1-j$ **do**
    if $Bvec[v] < Bvec[v+1]$ **then**
      $Bvec$, $Bvec[v+1] = Bvec[v+1], Bvec$
    end if
  end for
end for
Split $Bvec$ the user into two groups and pair users:
for $u \leftarrow 1$ to $K/2$ **do**
  Pair (UE_{u1}, UE_{u2})
end for

In our implementation, we arrange the users in descending order according to the Euclidean norm of the vector, $\|\beta_{\mathbf{n}}\|$, with large scale fading coefficient across all APs. We then divide the UE into two equal groups, one group with the strong users and the other group of weak users, where the first UE in group 1 (group with the strong users) has the largest Euclidean norm similarly for the first UE in the group 2 (group with the weak users). Instead of pairing UE with the largest differences between the two groups as done in [116], we pair the first user in the group 1 denoted by, UE_{11} with the first user in the group, denoted by, UE_{12}. This may seem counter-intuitive, but if users are paired from two groups based on the largest differences in β_n, the final cluster pair will end up having a relatively small difference in $\beta_{\mathbf{n}}$. Additionally, randomly selecting UE from the first group can result in suboptimal or even poor clustering performance, especially when the number of UE in the system is small, as noted in [116]. However, the proposed

clustering approach almost guarantees there is a sufficiently large difference in β_n across a cluster pairs. This particularly important when considering the objective function shown in (OP1) where there is an emphasis on UE meeting a minimum SE.

3.3 Problem Formulation

The objective is to maximise the DL SSE by doing power allocation, user association and SWIPT frame optimization. An optimization problem is formulated to maximise the DL SSE while on a constrained UAV transmit power budget, imperfect SIC and minimum requirements for UL and DL SE. $P \triangleq \{\rho_{mnl}^{downlink}\}$ is the UAV transmit power allocation matrix and $U \triangleq \{A_{ml}^{D}\}$ is the user association allocation matrix. In the system model we assume perfect CSI is available, thus the beam-former in (3.14) is constructed using perfect CSI.

$$\max_{\mathbf{P},\mathbf{U},\tau_e}, \quad R_{sum}^{downlink} = \sum_{l=1}^{L}\sum_{n=1}^{N} R_{nl}^{downlink} \tag{OP1}$$

$$s.t \sum_{l}^{L}\sum_{n}^{N} A_{mnl}^{D}\ \rho_{mnl}^{downlink} \leq P_{max}^{m}, \forall m \in \mathcal{M} \tag{C1}$$

$$R_{nl}^{downlink} \geq \bar{R}_{nl}^{D}, \forall n, \ , \forall l \tag{C2}$$

$$R_{nl}^{uplink} \geq \bar{R}_{nl}^{U}, \forall n, \ , \forall l \tag{C3}$$

$$A_{ml}^{D} \in [0, 1] \tag{C4}$$

$$\tau_u + \tau_e + \tau_d \leq \tau_c - \tau_p \tag{C5}$$

where P_{max}^m is the maximum total transmit power at the m-th UAV. For simplicity, we assume that the total maximum total transmit power at every UAV is the same. $\overline{R}_{nl}^D$ and $\overline{R}_{nl}^U$ are the minimum required SE at each UE for DL and UL transmission, respectively. It is assumed the channel gain will not change within one time coherence block. Therefore, the same optimized DL transmission user association matrix can be used for UL transmission. Since UL transmission uses all the harvested energy by the UE, there is no need for UL transmission power allocation. In addition to power allocation and user association optimization, the ratio between time allocated for DL data transmission, UL data transmission and energy harvesting will be optimized.

This is different from the work done in literature, where in [101], their objective function is to maximise the achievable DL sum rate by jointly designing a cooperation clustering scheme and power allocation subject to transmit power constraints. In [78], the objective function is to maximise sum rate subject to conditions required for successful SIC operations, by performing joint beam-forming and AP clustering. In [96], the objective function is to maximise EE subject to the SSE requirements, by doing joint AP selection and power allocation. In [95], the objective function is to maximise EE subject to the minimum required UE SE, by doing user association. In [94], the objective function is to maximise DL SSE subject transmit power budget, by doing power allocation. In [91], the objective function is to maximise the UL sum rate subject to minimum required UL SE, by doing power allocation. We note that none of the papers consider both UL and DL UE SE requirements and SWIPT frame optimization.

3.4 Deep Deterministic Policy Gradient

A DRL model was specifically chosen as it combines DL and RL. RL is a machine learning paradigm in which intelligent agents learn through trial and error

to take appropriate actions in an interactive environment, with the objective of maximising cumulative reward. RL is neither supervised nor unsupervised learning; rather, it is its own paradigm that lies somewhere in between the two. DL algorithms are based on ANNs, attempting to mimic the human brain's ability to learn by employing neural networks with three or more layers.

The DDPG algorithm is a model-free and off-policy DRL method that comprises four components: a Q-network, a deterministic policy function (actor), a target Q-network, and a target policy network (target actor). DDPG agents follow an actor-critic architecture. The critic estimates the value function, while the actor updates the policy in the direction suggested by the critic. Therefore, the actor corresponds to the policy network, and the critic corresponds to the Q-network. DDPG also employs a replay buffer to sample past experiences for updating neural network parameters. It uses the Bellman equation to learn the Q-function, and the learned Q-function is used to improve the policy [117]. In other words, DDPG concurrently learns both a Q-function and a policy. The underlying principle of DDPG is that, given an optimal action-value function $Q^*(s, a)$ in any given state s, the optimal action $a^*(s)$ can be found using:

$$a^*(s) = \arg\max_a Q^*(s, a). \tag{3.34}$$

If there are a finite number of discrete actions, the Q-values for each action can be computed, and the optimal action that maximises the Q-values can be calculated using (3.34). However, if the action space is continuous calculating $\max_a Q^*(s, a)$ becomes extremely difficult. In DDPG, the continuous action space is handled by differentiating with respect to the action argument. This allows for the setup of an efficient gradient-based learning rule for a policy $\mu(s)$. Therefore, to compute $\max_a Q^*(s, a)$, it can be approximated as:

$$\max_a Q^*(s, a) \approx Q^*(s, \mu(s)). \tag{3.35}$$

Algorithm 2 DDPG For Joint Power Allocation and User Association

Initialise policy parameters θ, Q-function parameters, empty replay buffer D, M is the max number of episodes, T the is max number of time-steps, γ is the discount factor
Initialise environment
for *episode* $\leftarrow 1$ to M **do**
    Observe states from environment S $= [g_{mnl_t}, A_{ml_{t-1}}]$
    Select appropriate action A $= [P, U]$
    Store (s, a, r, s') in reply buffer D;
    for *step* $\leftarrow 1$ to T **do**
        if s' *is terminal* **then**
            Reset environment states
        else
            for $K \leftarrow 1$ to *updates* **do**
                Sample mini-batch B from reply buffer D
                Compute targets;
                $y(r', s') = r + \gamma Q_{\emptyset targ}(s', \mu_{\emptyset targ}(s))$
                Update Q-function by one step of gradient decent;
                $\nabla_{\emptyset} \frac{1}{|B|} \sum_{(s,a,r,s\prime) \in B} \left(Q_{\emptyset}\left(s, a\right) - y(r', s')^2\right)$
                Update policy by one step of gradient accent;
                $\nabla_{\theta} \frac{1}{|B|} \sum_{(s,a,r,s\prime) \in B} \left(Q_{\emptyset}\left(s, \mu_{\theta}(s)\right)\right)$
                Update target networks;
                $\emptyset_{targ} \leftarrow \rho \emptyset_{targ} + (1 - \rho)\emptyset$
                $\theta_{targ} \leftarrow \rho \theta_{targ} + (1 - \rho)\theta$
            end for
        end if
    end for
end for

3.4.1 Q-Learning

DDPG uses the Bellman equation, which describes the optimal action-value function. The optimal action-value function $Q^*(s, a)$ given by

$$Q^*(s, a) = \mathbb{E}_{s' \sim P}\Big[r(s, a) + \gamma \max_{a'} Q^*(s', a')\Big], \tag{3.36}$$

where $s' \sim P$ is the next state, and s' is sampled by the environment from a distribution $P(\cdot \mid s, a)$. The Bellman equation is used to approximate $Q^*(s, a)$.

If the approximator is a neural network denoted by $Q_\phi(s, a)$, with parameters ϕ, and a dataset $\mathcal{D}$ of transitions (s, a, r, s', d) is given where d is an indicator variable

used to determine whether the next state s' is terminal, then we can use the Mean Squared Bellman Error (MSBE) to evaluate how closely Q_ϕ satisfies the Bellman equation, is given by

$$\mathcal{L}(\phi, \mathcal{D}) = \mathbb{E}_{(s,a,r,s',d)\sim\mathcal{D}}\left[\left(Q_\phi(s,a) - \left(r + \gamma(1-d)\max_{a'} Q_\phi(s',a')\right)\right)^2\right]. \quad (3.37)$$

The term $(1-d)$ is used to identify when s' is a terminal state, meaning a state after which the agent receives no further rewards where $(1-d)$ is used to determine whether s' is a terminal state, with a terminal state defined as one where the agent receives no additional rewards after the current state. Specifically, $d = 1$ indicates that s' is a terminal state [117].

3.4.2 Target Network

Q-learning algorithms use a target network to minimize the MSBE loss, with the objective of making the Q-function approximate the target network ϕ_{targ} more closely given by,

$$r + \gamma(1-d)\max_{a'} Q_\phi(s',a'). \quad (3.38)$$

In DDPG, the target network is updated once per main network update by Polyak averaging given by

$$\phi_{\text{targ}} \leftarrow \rho\,\phi_{\text{targ}} + (1-\rho)\,\phi, \quad (3.39)$$

where ρ is a hyper-parameter between 0 and 1.

To calculate the max over actions in the Target, the DDPG uses a target policy network to compute an action which approximately maximises $Q_{\phi_{\text{targ}}}$. The target

network policy network can be found the same way as the target Q-function by using Polyak averaging. The Q-learning in DDPG is performed by minimizing the MSBE loss with stochastic gradient decent given by

$$\mathcal{L}(\phi, \mathcal{D}) = \mathbb{E}_{(s,a,r,s',d)\sim\mathcal{D}}\left[\left(Q_\phi(s,a) - \left(r + \gamma(1-d)Q_{\phi_{\text{targ}}}\left(s', \mu_{\theta_{\text{targ}}}(s')\right)\right)\right)^2\right], \tag{3.40}$$

where ,$\mu_{\theta_{\text{targ}}}$ is the target policy [117]. The implementation of the DDPG model for CF-mMIMO environment is shown in algorithm 2.

3.4.2.1 State

The State Space, S, for DDPG is defined as $S = [g_{mnlt}, A_{ml_{t-1}}]$ where g_{mnl} are the channel gains, $A_{ml_{t-1}}$ is user association matrix and t simply indicates the time-step.

3.4.2.2 Action

The DDPG Action Space, A, is defined as $A = [P, U]$ where $P \triangleq \rho_{mnl}^{downlink}$ is the power allocation vector and $U \triangleq A_{ml}^{D}$ is the user association vector. The DDPG model is designed to generate continuous actions, making it well-suited for tasks like power allocation. However, it is not naturally equipped to handle binary or discrete actions, such as those involved in user association. As a result, DDPG cannot effectively perform hybrid action tasks without modifications [95]. In order to use DDPG for hybrid action space, we modify the DDPG in the following ways: First the DDPG performs action and produces a set of continuous actions between 0 and 1 which is expected when using the sigmoid output layer, we then split the actions into two groups. We divide the actions into two groups: one for user association and another for power allocation. The action values for user association are then discretized using a threshold of 0.5, where values greater than

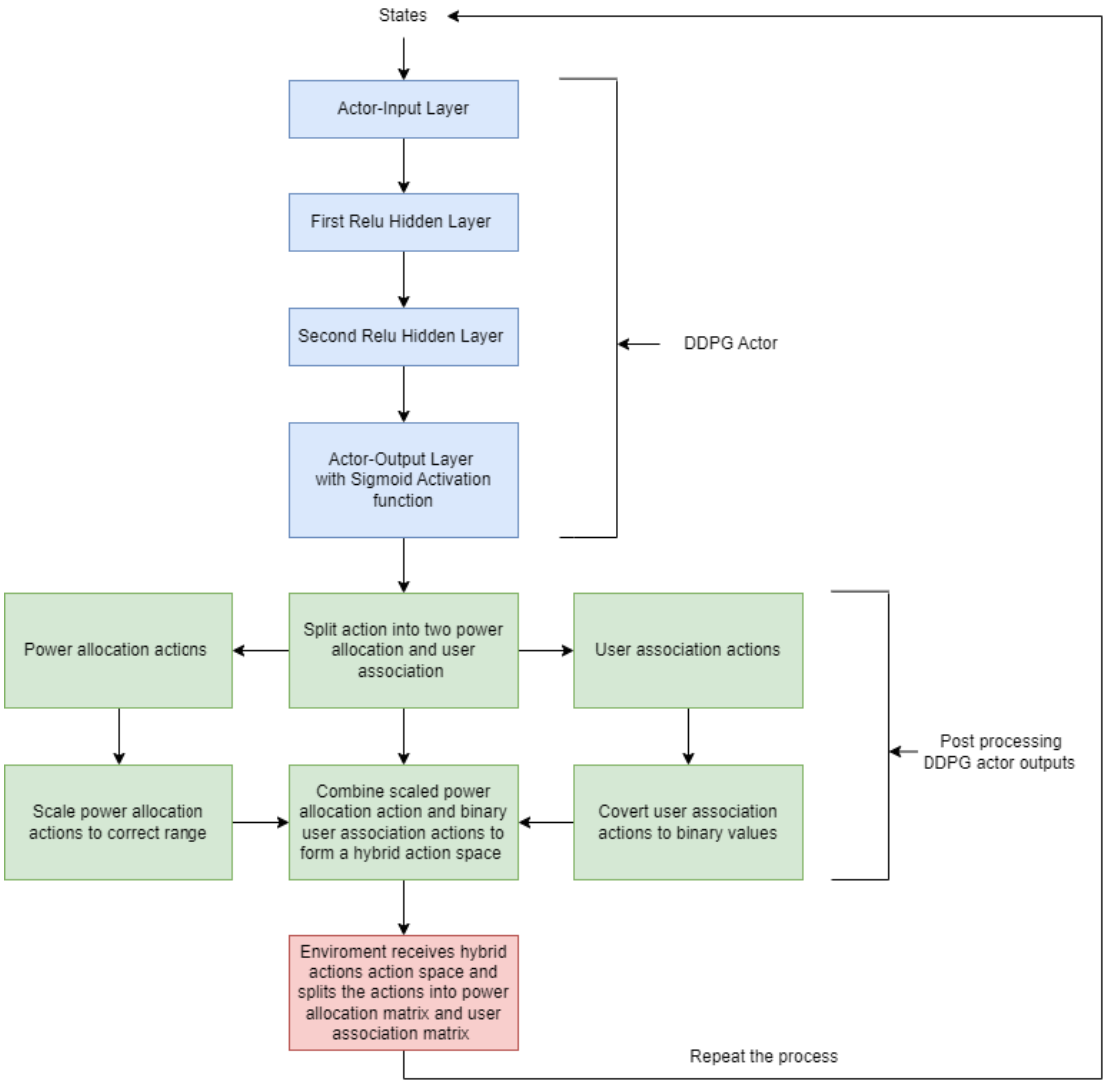


FIGURE 3.3: Post processing actions from the actor network.

or equal to 0.5 are mapped to 1 (associated), and values below 0.5 are mapped to 0 (not associated)

The actions of power allocation are scaled up to appreciate range which it determined by maximum transmit of the AP. Once we transform the actions for user association and power allocation, we combine the actions in one action space with two vectors corresponding to user actions and power allocation, respectively. The DDPG as mentioned before has a critic network and an actor network, this does not change. However, how you configure the critic and actor networks has big impact on model performance. By configure we refer the number of hidden layers,

the size of each layer, the types of hidden layers used and in the case of actor networks the activation layer used for the outputs.

3.4.2.3 Actor and Critic Network Architectures

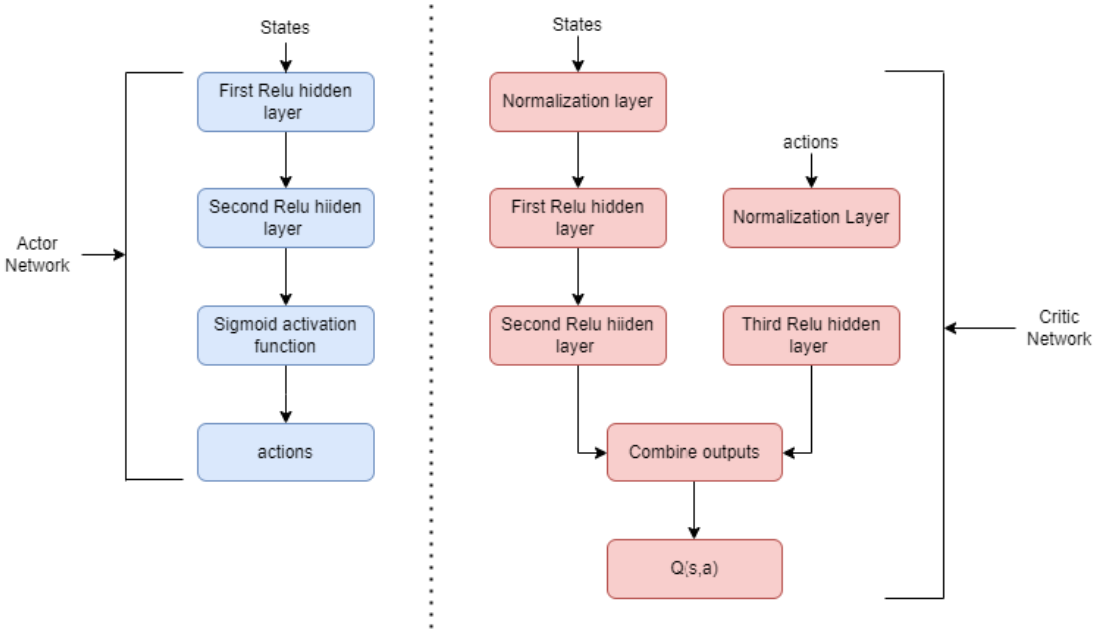


FIGURE 3.4: Actor and critic network structure.

The DDPG algorithm, as mentioned before, consists of a critic and actor networks; this fundamental architecture does not change. However, the way the critic and actor networks are configured has a significant impact on the model's performance. By configuration, we refer to aspects such as the number of hidden layers, the size of each layer, the types of layers used, and, for actor networks specifically, the activation function applied to the output layer.

In the proposed DDPG model, we found the best configuration for the critic networks to be three ReLU hidden layers with normalization layers in between, as illustrated in Figure 3.4. For the proposed model, the best configuration for the actor networks was two ReLU hidden layers with a Sigmoid activation function at the output, also shown in Figure 3.4.

Notice that we do not explicitly mention the input and output layers because they are determined by the state space and, for the actor network, by the number of actions required. Although the critic network does not produce any action, its output layer has a single output representing the Q-value. The three hidden ReLU layers for the critic network are sized $[512 \times \delta, 256 \times \delta, 256 \times \delta]$, where $\delta = \frac{K}{10}$ is a scaling factor used to scale up the critic network size depending on K, the number of UE on which the model is being trained on. Similarly, the two hidden ReLU layers for the actor networks are sized $[512 \times \delta, 256 \times \delta]$.

3.4.2.4 Reward

The reward function for the proposed DDPG model is given by

$$R_{sum}^{downlink} + w_1 \log_2 (0.1 + \frac{{DL_{min}}^2}{\beta}) + w_2 \log_2 (0.1 + \frac{{UL_{min}}^2}{\beta}), \tag{3.41}$$

where DL_{min} and UL_{min} are the number of UE that meet the minimum required SE for DL and UL respectively, K is the total number of UE, w_1 and w_2 are positive valued hyper-parameters used for scaling and $\beta = \left(\frac{K}{2}\right)^2$. $\bar{R}_{nl}^{D}$ and $\bar{R}_{nl}^{U}$ are the minimum required UE SE for DL and UL transmission, respectively, with each set to 0.1bps/Hz. This is different from the work done literature. In [78],[94] and [101], the reward function is simply the achievable DL sum rate without any penalties, this is the same as their objective function. In [89], the reward function is UL sum rate with a penalty associated with minimum required UE UL SE. In [95], the reward function is the EE with a penalty associated with minimum required UE DL SE. In [96], the reward function is the EE with penalty associated with SSE constraints in their objective function. We note, red that none of the papers incorporated a penalty associated with minimum required UE SE for both DL and UL transmission as shown in (3.41).

The reward function can be broken down into four distinct polices relating to each component that makes up the function.

1. Policy-1: is simply the reward function as defined in (3.41). It is used to show the effect of having reward function with components related to the number of UE that meet the minimum required DL and UL SE in addition to DL SSE.

2. Policy-2: is the reward function as defined in (3.41) with w_2 =0. It is used to show the effect of having a reward function, with only components related to DL transmission, namely, the number of UE that meet the minimum required DL SE and DL SSE in the reward function.

3. Policy-3: is the reward function as defined in (3.41) with w_1=0. It is used to show the effect of having a reward function with components related to number of UE that meet the minimum required UL SE and DL SSE.

4. Policy-4: is the reward function as defined in (3.41) with both w_1=0 and w_2=0. It is used to show the effect of having reward function entirely composed of DL SSE.

3.5 Complexity Analysis

When calculating the computational complexity of the DDPG model we consider the inferencing phase. The factors that determine the inferencing computational complexity include the size of the NN, and the complexity of the environment. To calculate the inferencing complexity of the DDPG model, we must consider the size of the input layers, hidden layers and output layers for both the actor and critic networks. Typically, the inference performance is calculated in terms of Floating Operations Per Second (FLOPS). The number of FLOPs for the DDPG policy and value networks can be calculated respectively as,

$$FLOPS_{POLICY} = |S|\,|H_j| + \sum_{J} |H_{j-1}H_j| + |A|\,|H_J|, \tag{3.42}$$

and

$$FLOPS_{value} = |S|\,|H_j| + \sum_{J} |H_{j-1}H_j| + |A|\,|H_v| + |H_J| + |H_v|, \tag{3.43}$$

where $|S|$, $|A|$, $|H_j|$ and $|H_v|$ are the number of elements in state, actions, n-th hidden layer in policy and value networks and the hidden layer for action input branch. From Figure 3.4, the critic has two input layers, one for the states and one for the actions. Input state size is defined by $S = [g_{mnlt}, A_{ml_{t-1}}]$,where is $g_{mnlt} \in \mathbb{C}^{M\times K}$ and $A_{ml_{t-1}} \in \mathbb{N}^{M\times L.}$ The state input layer is proceeded by two hidden layers of size $[512 \times \partial, 256 \times \partial]$ where $\partial = \frac{K}{10}$. Therefore, the FLOPS of the state input layer and its two hidden layers is $(M \times K + M \times L) \times 512 \times \partial + \partial(512\times 256+256)$. The actions input layer is proceeded by one hidden layer of size $256 \times \partial$. Therefore, the FLOPS of the action input layer and its one hidden layer is $(M \times K + M \times L) \times 256 \times \partial + 256 \times \partial$. As shown in Figure 3.4, the last hidden layers from the state input branch and the action input branch are concatenated. The resulting FLOPS is computed as $\partial(256 + 256) \times 1$, where the 1 represents the single output neuron of the critic network corresponding to the estimated Q-value. Therefore, the total FLOPS for the critic is $\partial\,(M \times (K + L) \times 768 + 132096)$. The actor network consists of the input layer, two hidden layers and an output layer. The actor network has input layer consisting of $S = [g_{mnlt}, A_{ml_{t-1}}]$ defined by $M \times K + M \times L$ proceeded by two hidden layers that a have a size of $[512 \times \partial, 256 \times \partial]$, respectively, where $\partial = \frac{K}{10}$. The output layer of the actor is number actions defined by $M \times (K + L)$). The total FLOPS for the actor network is $768 \times \partial \times (M\,(K + L) + 131072)$. It is important to note that for inferencing we only use the actor network FLOPS, once the DDPG model has been trained there is no need for the critic.

3.6 Simulation and Training Parameters

We evaluate CF-mMIMO network with a 1 km $\times$ 1 km square area with default parameters shown in Table 3.2. During training, the model is trained on 1000 episodes with each episodes having 50-time steps per episode. At the end each

TABLE 3.2: System model and simulation parameters

Parameter	**Value**
Number of AP	20
Number of UE	10
Number of UE in a cluster	2
UL pilot power	100mW
UE circuitry power consumption	100mW
Maximum AP instantaneous transmit power	2W
Coherence time interval	1ms
Number of symbols per coherence time interval	200
System bandwidth	20MHz
Carrier frequency	2Ghz
Power spectral density of noise	-174dbm/Hz
Noise at the receiver	9dbm
Energy harvesting efficiency coefficient	0.8
Time allocated for energy harvesting	250 microseconds
Successive interference cancellation coefficient	0.01
UAV operating height	15m
UAV maximum velocity	1m/s
UE maximum velocity	50m/s
Path loss exponent for LoS (α_1)	1.2
Path loss exponent for NLoS (α_2)	2.1
Environment constants b_1 and b_2	11.95 and 0.136
Actor learning rate	1×10^{-4}
Critic learning rate	1×10^{-4}
Buffer size	20000
Discount value	0.99
Batch Size	32
Ornstein noise standard deviation	0.1
Soft Update	0.01
Critic network RELU hidden layers	(512,256,256)
Actor network RELU hidden layers	(512,256)

episode it is assumed that both the UE and UAVs can move in any direction within the confines of the stipulated area at velocity of 1m/s and 50m/s, respectively [83]. The operating height of the UAVs is fixed at 15m [9]. Therefore, during each episode, the large fading coefficient of the channel gains are static. However, the small-scale fading coefficients for the channel gains vary at each time-step, with each time-step corresponding to a coherence interval of 1 ms. To ensure the DRL model is trained under diverse conditions accounting for different initial positions of UE and UAVs, the starting positions of both UE and UAVs are uniformly randomized every 10 episodes. Thus, both the large-scale and small-scale channel gain coefficients change. The simulations were conducted using the HP Elite-book 860 16-inch G9 Notebook with 32 GB of ram and 12th Gen Intel(R) Core (TM) i7-1255U 1.70 GHz CPU. The simulation and training environment was developed in Python, utilizing Tensor-flow for model construction and OpenAI Gym for creating and interacting with our custom environment.

3.7 Results and Analysis

In this section, we discuss and analyse the results of the proposed DDPG model. We first analyse the impact of hyper-parameter tuning on the DDPG model's performance. We then present and analyse the results showing the impact of EH time allocation on the trained DDPG's model performance for different transmit AP powers. We also present and analyse the results showing the impact of UAV's operating height and velocity on the trained DDPG model's performance. Lastly, the results are presented comparing the performance of the trained DDPG with random power allocation and user association schemes.

3.7.1 Hyper-parameter Tuning on Training Performance

When we compare the impact of each hyper-parameter, we only change the hyper-parameter of concern whilst keeping the rest of hyper-parameters with their default values shown in Table 3.2. This was done to ensure we isolated the impact of each individual hyper-parameter without any dependences on other hyper-parameters. We also generate testing dataset from the environment, that has been used throughout the duration of the hyper-parameter testing. This was done to ensure that there were no performance gains or disadvantages, due to random variations in channel gains.

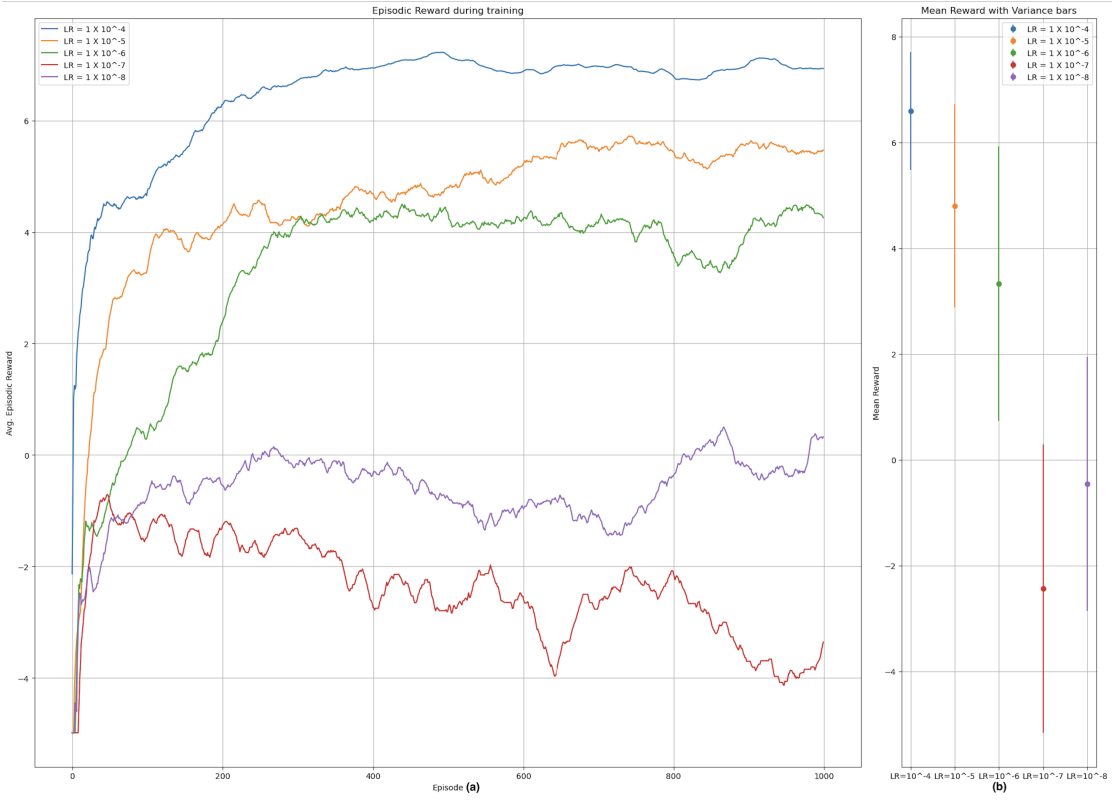


FIGURE 3.5: The effect of learning rate on DDPG training performance.

Figure 3.5 shows the effect of Learning Rate (LR) on the model's performance during training. In gradient descent, the learning rate is used to determine the magnitude of the of the step size taken during each iteration of the gradient descent process. If the learning rate is too small, it can lead to slow convergence and if the learning rate is large it can lead to faster convergence at risk of overshooting the

minimum. The results reveal that there is a clear correlation on model performance and learning rate. The results reveal that a learning rate of 1×10^{-4} clearly performs the best by a significant margin in terms of convergence and overall episodic rewards after convergence when compared to other learning rates. The results reveal that by decreasing the learning rates by one order of magnitude, subsequent learning rates perform worse than the previous learning rate during training, in terms of convergence and episodic rewards after convergence. One notable exception to this trend is when the LR $= 1 \times 10^{-7}$ and LR $= 1 \times 10^{-8}$. Where we observe a LR $= 1 \times 10^{-8}$ performs better than a of LR $= 1 \times 10^{-7}$. This highlights the unpredictable nature of hyper-parameter tuning. Another observation is that, as the learning rates decrease, there are greater fluctuations in episodic rewards after convergence. This is reflected in the mean and variance bars of the learning rates shown in Figure 3.5(b). We can conclude from the results, that there is clear correlation between learning rate and model training performance.

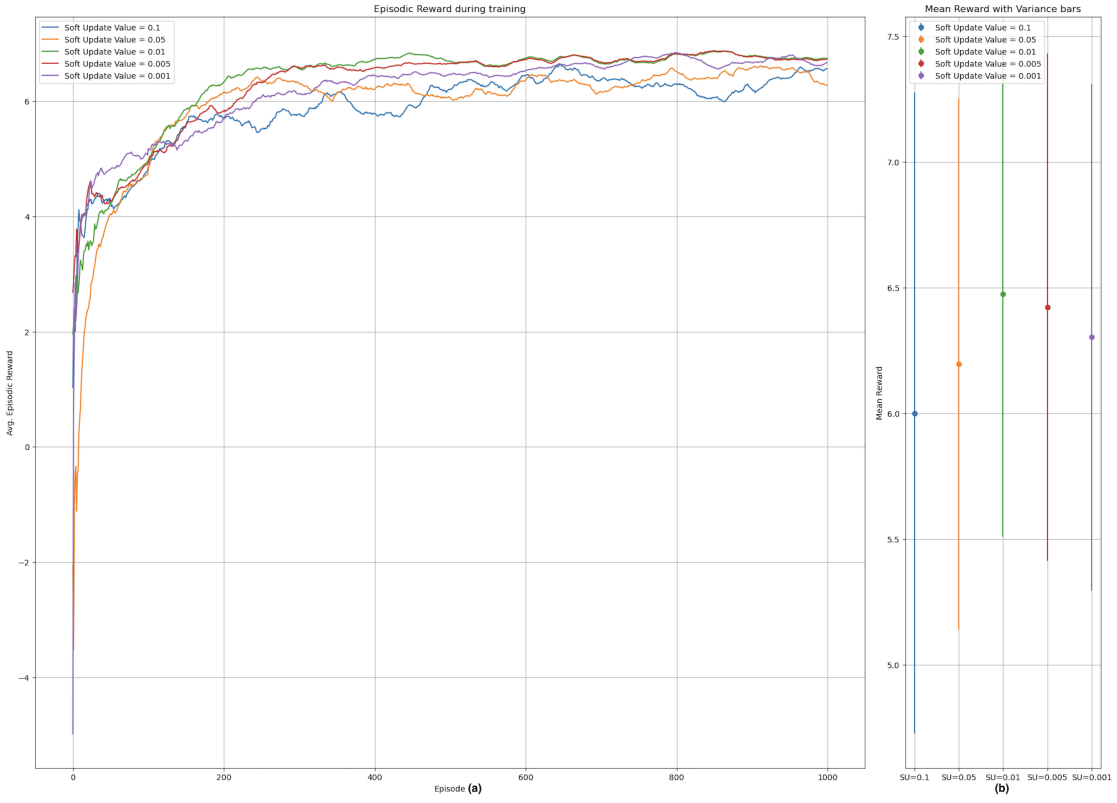


FIGURE 3.6: The effect of soft update size on training performance.

Figure 3.6 shows the impact of the soft update value on the model's training performance. By observing the mean reward, shown in figure 3.6(b), during training. The results reveal, a soft update value of 0.01 performs the best, closely followed by a soft update value of 0.005 and then a soft update value of 0.001. The soft update value of 0.1 and soft update value of 0.05 perform the worst. After the 400th episode when the model converges for all soft update values. The results reveal, soft update values of 0.01, 0.005 and 0.001 consistently perform better than the soft update values 0.1 and 0.05 as shown in Figure 3.6(a). We note there is no correlation between soft update value size and training performance as best performing soft update value lies between the largest and smallest soft update values. This implies that if the soft update value is too small or too large, the training performance may be affected. We also observe that the best performing hyper-parameter values tend to have less variance during training as shown in Figure 3.6(b), a similar trend to that of the results on the impact of learning rate on model's training performance. It also important to note that the best soft update value is highly dependent on the environment. This is also reflected in literature where in [101], [79] and [91] soft update values of 0.001, 0.006, and 0.005 were found to perform the best for their environments, respectively.

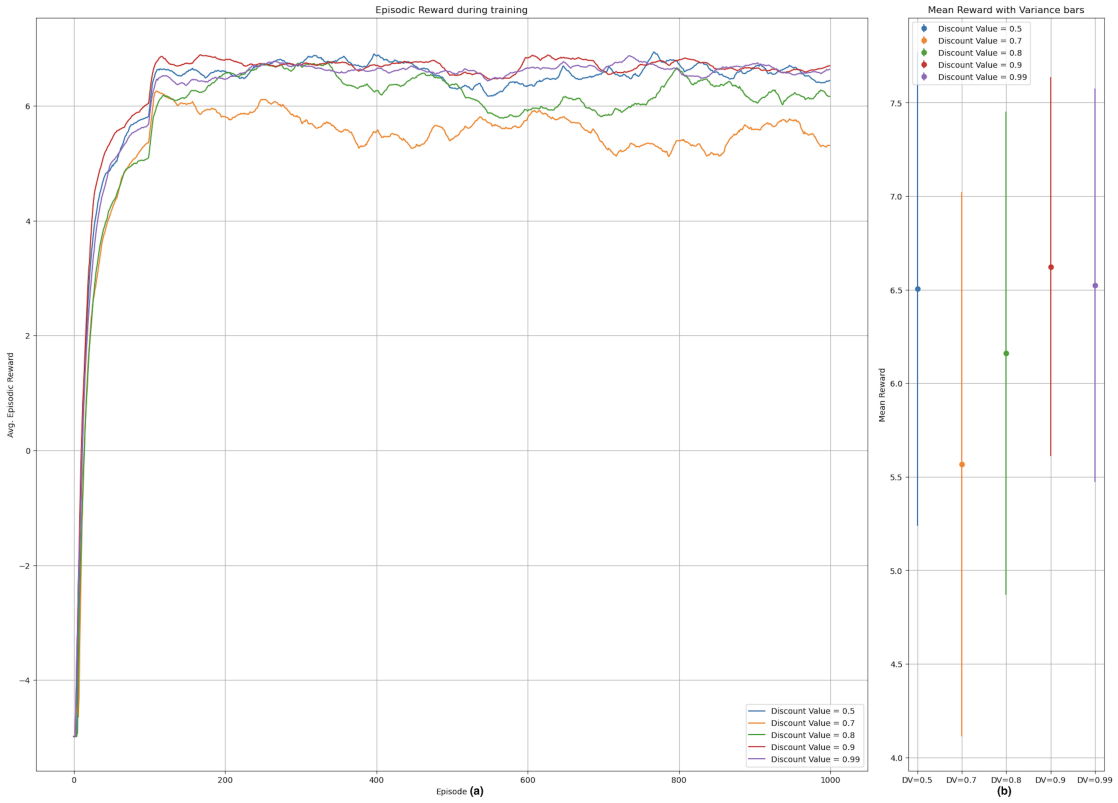


FIGURE 3.7: The effect of discount value on training performance

Figure 3.7 shows the effect of discount factor on model training performance, by observing the mean reward during training, the results reveal that a discount factor of 0.9 performs the best, closely followed by a discount factor 0.99 and a discount factor of 0.5 as shown in Figure 3.7(b). The worst performing discount factor is 0.7 followed by a discount factor of 0.8. This is somewhat surprising as a larger discount value close to 1, encourages the agent to consider the long-term consequences of its actions, promoting strategies that may yield higher rewards in the future whereas a discount value close to 0 encourages the agent to focus on immediate actions. Therefore, one would expect to see the performance increase as the discount value increases. However, as shown the results in Figure 3.7(a) there is no clear correlation between discount value size and model training performance. The reason why a discount factor of 0.9 yields the best performance is that it achieves an optimal balance between immediate and future rewards. In the context of the system model used, this suggests that the discount factor effectively weighs the trade-off between maximising the DL SSE which reflects short-term

rewards and the additional gains associated with the number of UE meeting the minimum required DL and UL SE, which represent long-term rewards. In contrast, other discount values may fail to strike this balance, resulting in suboptimal optimization of both short-term and long-term reward. The results also reveal that the convergence performance of all the discount factors are relatively the same, with all of them converging around the 100th training episode. The results also reveal the discount factor of 0.7 and discount factor of 0.8 have greater fluctuations after convergence when compared to other discount factors. From the results shown, we conclude there is no correlation between the discount factor size and training performance. The best discount factor is highly dependent on the environment, this is also reflected in literature where in [91],[79] and [100] discount factors of 0.5, 0.7 and 0.99 were found to respectively perform the best for their environments.

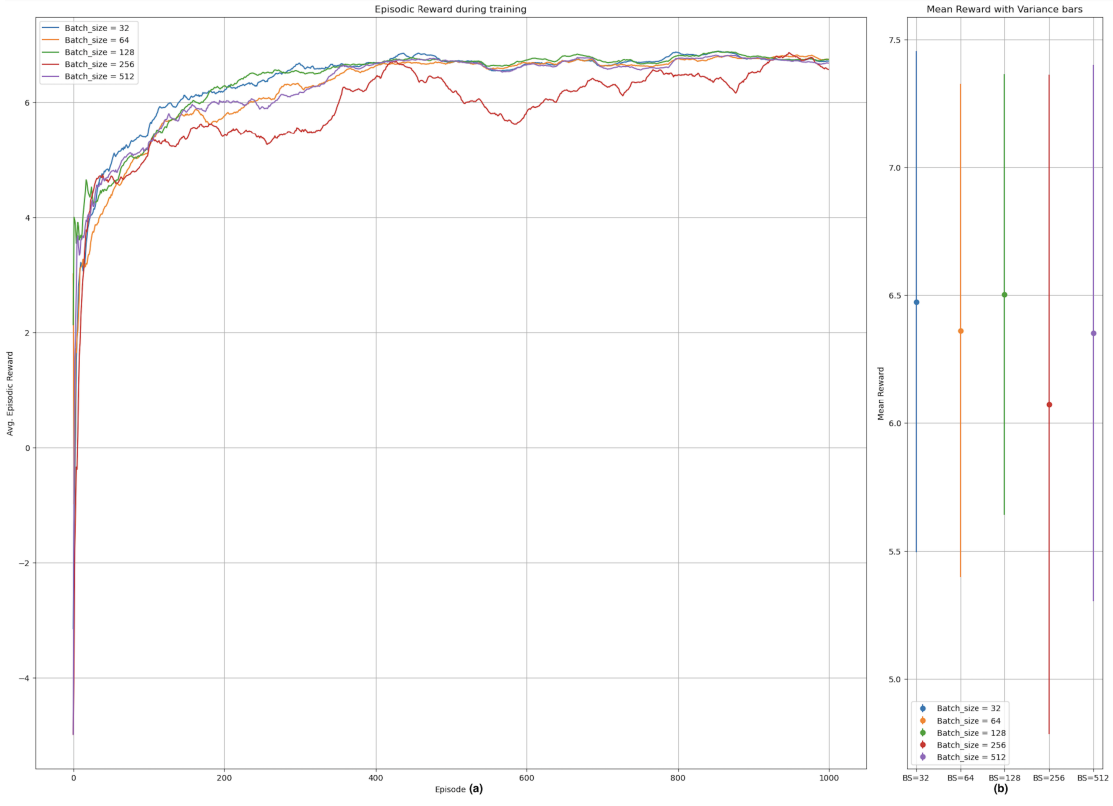


FIGURE 3.8: DDPG training performance after hyper-parameter tuning.

Figure 3.8 shows the effect of batch size on model training performance. By observing the mean reward during training, a batch size of 128 performs the best closely followed by a batch of 32, then a batch size of 64, followed by a batch

size of 512 and lastly, a batch size of 256 performs the worst as shown in Figure 3.8(b). However, the results indicate that adjusting the batch size did not lead to significant performance improvements in terms of average reward, as all batch sizes demonstrated relatively similar performance based on their mean rewards during training as shown in Figure 3.8(b). We notice there is no correlation between batch size and performance as we observe that the middle batch size performs the best. However, the results reveal that batch size influences how quickly the model converges during training, where a batch size of 32 and 128 led to faster convergence when compared to other batch sizes as shown in Figure 3.8(a). During training, we also observe that a batch size of 256 has greater reward fluctuations after convergence. This is also reflected by the mean and variance bars, where a batch size of 256 has the greatest variance when compared to other batch sizes. We can conclude that the batch size has an impact on the convergence of the of DDPG model during training. It is important to note that the effect of a batch is highly dependent on the environment; this is also reflected in the literature where in [79], [90] and [100] a batch size of 32, 100, and 128 was found to perform the best for their respective environments.

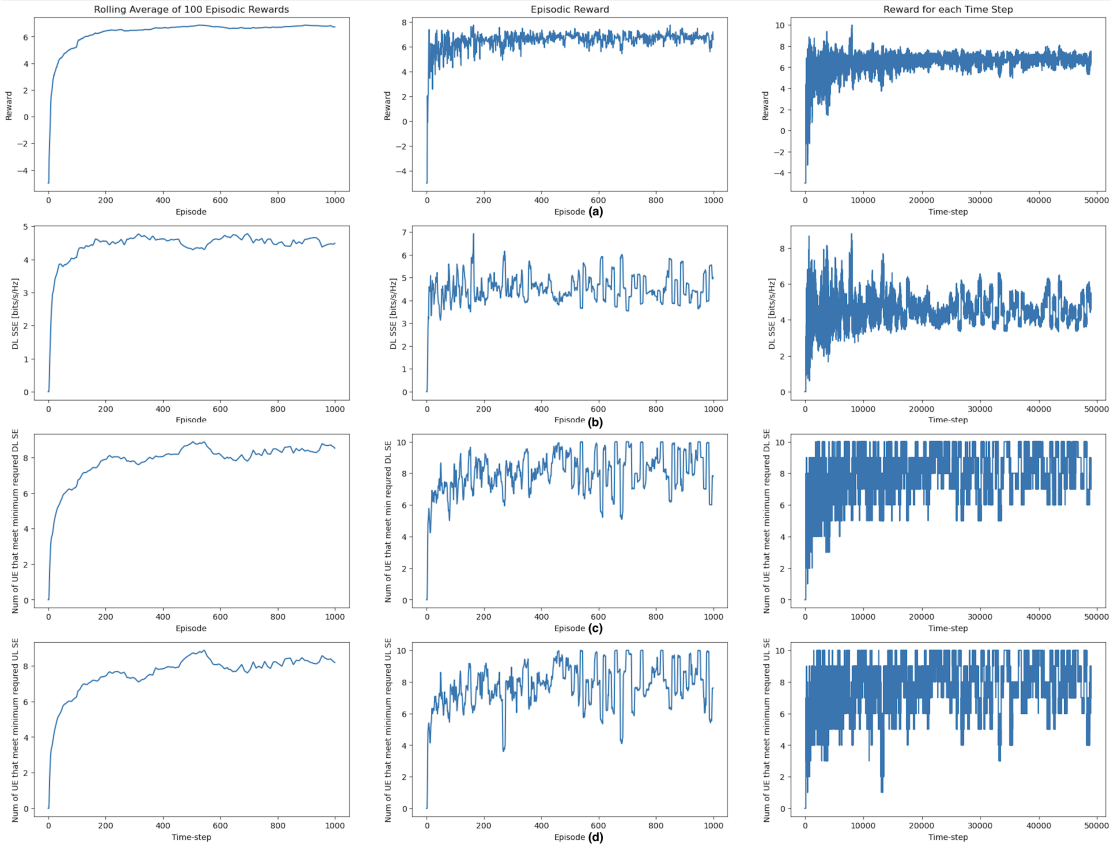


FIGURE 3.9: DDPG training performance after hyper-parameter tuning.

Figure 3.9 shows the performance of the model training after hyper-parameter tuning for Policy-1. From left to right, Figure 3.9(a) shows the rolling average reward, episodic average reward and the reward at each time step. Figure 3.9(b) shows the rolling average for DL SSE, episodic average DL SSE and DL SSE at each time step. Figure 3.9(c) shows the rolling average of the number of UE that meet the minimum required DL SE, the episodic average of the number of UE that meet the minimum required DL SE and the number of UE that meet the minimum required DL SE at each time step, from left to right. Figure 3.9(d) shows the rolling average of the number of UE that meet the minimum required UL UE, the episodic average of the number of UE that meet the minimum required UL SE and the number of UE that meet the minimum required UL SE at each time step, from left to right. The results reveal the model behaves as expected during training, focusing on the first column, as the average reward increases, the results show an increase in the DL SSE and the number of UE that meet the

minimum required DL and UL SE. From (3.41), we know the reward determined by DL SSE and the number of UE that meet the minimum required DL and UL SE.

3.7.2 Impact of Energy Harvesting Time Allocation

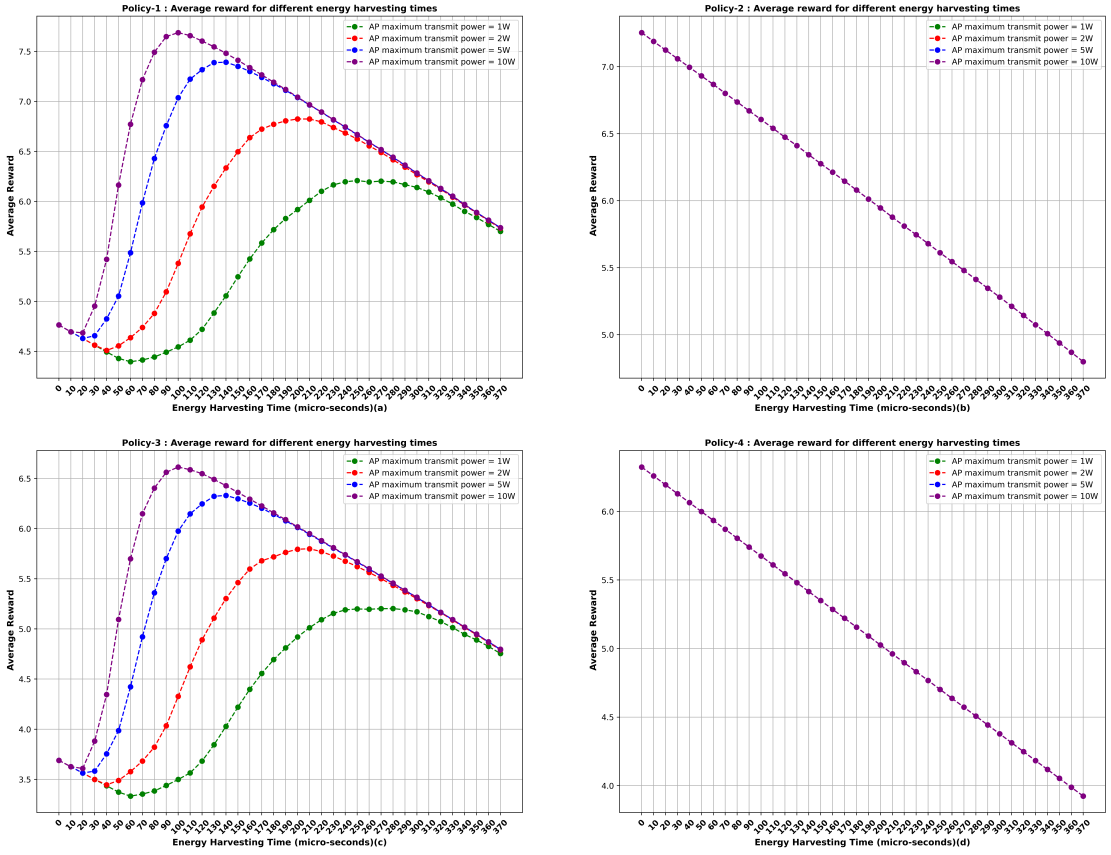


FIGURE 3.10: Impact for energy harvesting time on policy performance.

Figure 3.10 shows the impact of time allocated for EH has on the average reward for different policies with various AP instantaneous transmit powers. The results reveal that for Policy-1 and Policy-3, both the AP transmit power and energy harvesting time play an important role in maximising policy performance. The results reveal that there is clear correlation between the peak achievable average reward and transmit powers for Policy-1 and Policy-3 as shown in Figure 3.10(a) and 3.10(c), respectively. The results also show clear correlation between the amount energy harvesting time required to achieve peak policy performance and the AP

transmit powers. An increase in the AP transmit powers leads to less time being required for energy harvesting to achieve peak average reward performance for Policy-1 and Policy-3. For Policy-1 and Policy-3, optimal energy harvesting times for peak achievable reward performance are nearly the same across the different AP transmit powers as shown in Table 3.3. It is important to note that there is a non-linear response for Policy-1 and Policy-3 as we increase the time allocated to energy harvesting. The reward performance quickly increases when more time is allocated for energy harvesting until peak achievable reward is reached, then there is gradual decrease in performance. This contrasts with the linear response revealed by the results for Policy-2 and Policy-4 where increasing time allocated for energy harvesting leads to a linear decline in performance as hown in Figure 3.10(b) and 3.10(d), respectively. Recall that Policy-2 and Policy-4 only focus on DL, specifically maximising the DL SSE and in the case of Policy-2 also maximising the number UE that meet minimum required DL SE. When considering only DL, to maximise the average reward, the amount of time allocated for DL is the only factor that matters. From Figure 3.2, and OP1 constraint C5, we know as more time is allocated for energy harvesting in the coherence time interval, there is less time allocated for DL and UL data transmission. This explains the linear relationship shown by the results, between the average reward and the amount time allocated for energy harvesting for Policy-2 and Policy-4. When the time allocated for energy harvesting is zero, both Policy-2 and Policy-4 achieve peak reward performance for all transmit powers as their reward functions only have components related to DL transmission. The results show for Policy-2 and Policy-4 the transmit powers have no effect on performance, this can be attributed to the model being trained with the AP transmit power set to 2W. It may also indicate that the transmit powers are not the limiting factor in the system model, 1W maybe near or above system AP transmit power saturation point for the system model. In CF-mMIMO with NOMA increasing the AP power has very little effect beyond as certain threshold, which we call this the AP transmit power saturation point, this shown in [7], where a power increase of from 32 dBm (1.58W) to 40dBm (10W) shared across 32 APs, had a small increase in DL SSE from 30bits/Hz/s to

TABLE 3.3: Optimal energy harvesting time for different AP transmit powers

AP transmit power (watts)	Policy-1 (μs)	Policy-2 (μs)	Policy-3 (μs)	Policy-4 (μs)
1	250	0	270	0
2	200	0	210	0
5	140	0	140	0
10	100	0	140	0

about 35bits/Hz/s in CF mMIMO system with 32 APs and 20 UE.

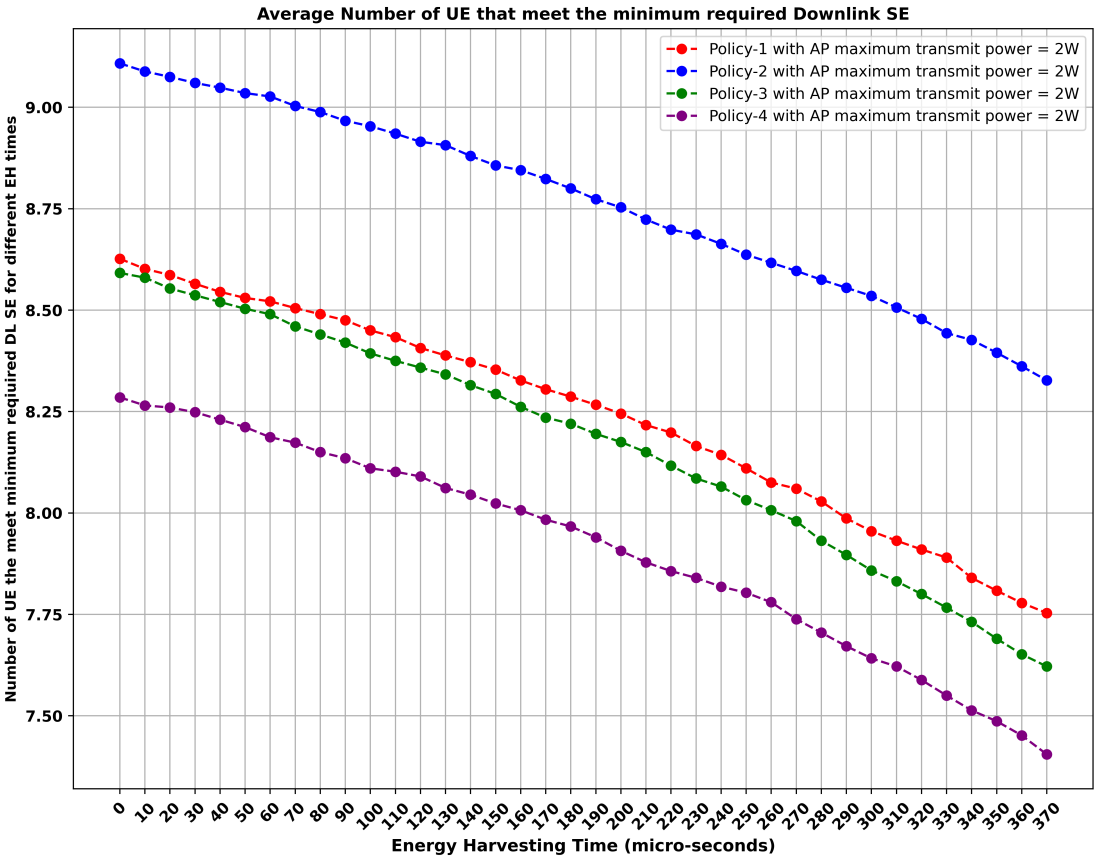


FIGURE 3.11: Policy performance for the number of UE that meet minimum required DL SE.

Figure 3.11 shows the effect of varying the time allocated for EH has on the number of UE that meet the minimum required DL SE for different policies. The results reveal Policy-2 performs best followed by Policy-1 then Policy-3 and lastly, Policy-4 with the worst performance. It was expected that Policy-2 and Policy-1 would perform the best as their reward functions have a component related to the number of UE that meet the minimum required DL SE whereas Policy-3 and

Policy-4 do not have such a component in their reward functions. The results reveal that Policy-3 performs better than Policy-4, this implies that by having a component related to the number of UE that meet minimum required UL SE in the reward function indirectly improves the performance of policy when trying to maximise the number of UE that meet the minimum required DL SE.

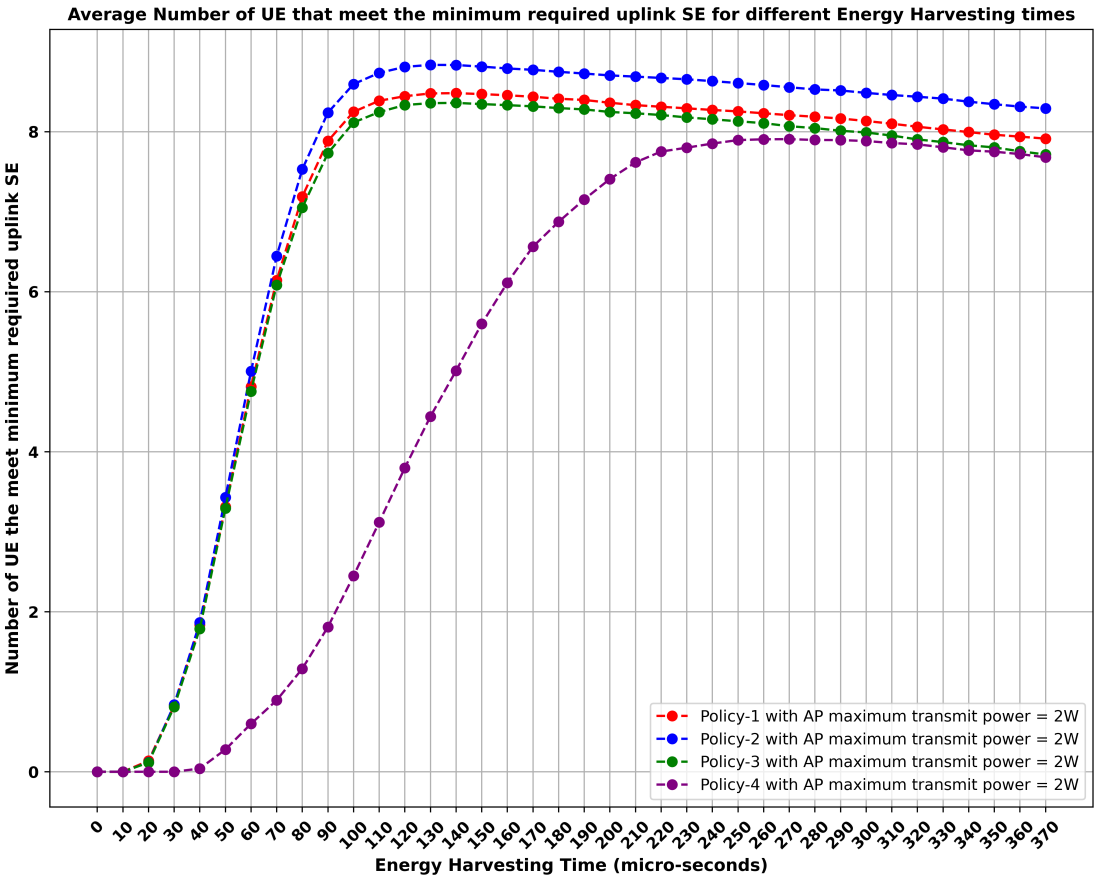


FIGURE 3.12: Policy performance for the number of UE that meet minimum required UL SE.

Figure 3.12 shows the effect of varying the time allocated for EH has on the number of UE that meet the minimum required UL SE for different policies. The results reveal Policy-2 performs the best followed by Policy-1, followed by Policy-3 and lastly Policy-4. This is a surprising result, as Policy-3 would be expected to perform the best closely followed by Policy-1 as they have component related to the number of UE that meet the minimum required UL SE in their reward functions, yet the results reveal Policy-2 performs the best. This contrasts the results from Figure 3.11 for the number of UE that meet the minimum required DL SE, where

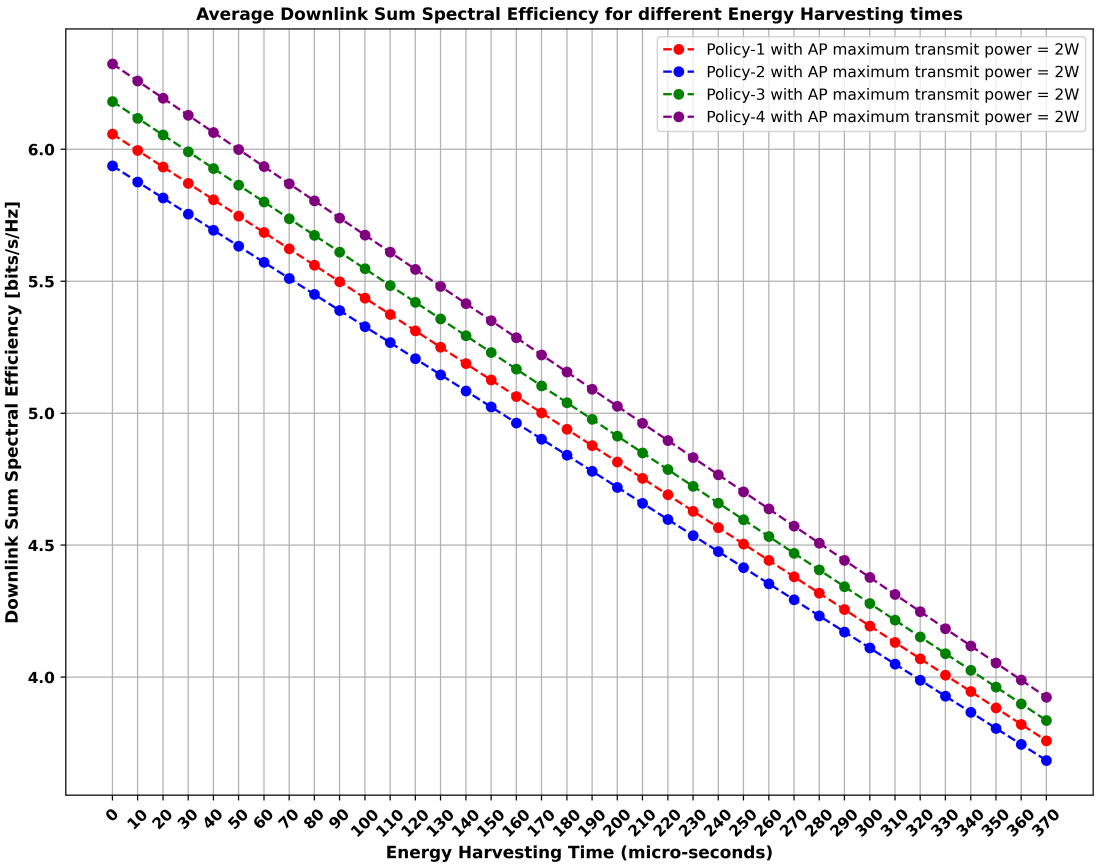


FIGURE 3.13: DL SSE performance for different policies.

the results showed having a component relating to the metric being evaluated yields better performance. This implies, by maximising the number of UE that meet minimum required DL SE indirectly maximises the number UE that meet the minimum required UL SE. This might explain why the results revealed, Policy-1 performing better than Policy-3. However, we cannot ignore the impact of having a component linked to the number of number UE that meet the minimum required as Policy-1 and Policy-3 have a clear advantage over Policy-4. It is important to note that we observe similar results for different AP maximum instantaneous transmit powers.

Figure 3.13 shows the effect of varying the time allocated for EH has on the DL SSE performance for different polices. The results reveal that Policy-4 performs the best, followed by Policy-3 followed by Policy-1 and lastly Policy-2 performs the worst. It was expected that Policy-4 would have the best performance as its reward function is entirely dependent on the DL SSE. A surprising observation

is that Policy-3 performs better than Policy-1 and Policy-4. This implies there is an inverse relationship between DL SSE performance and the number of UE that meet the minimum required DL and UL SE, as the results in Figure 3.11 and Figure 3.12 revealed Policy-2 had the best performance and Policy-4 had the worst performance yet for the DL SSE, the opposite is true.

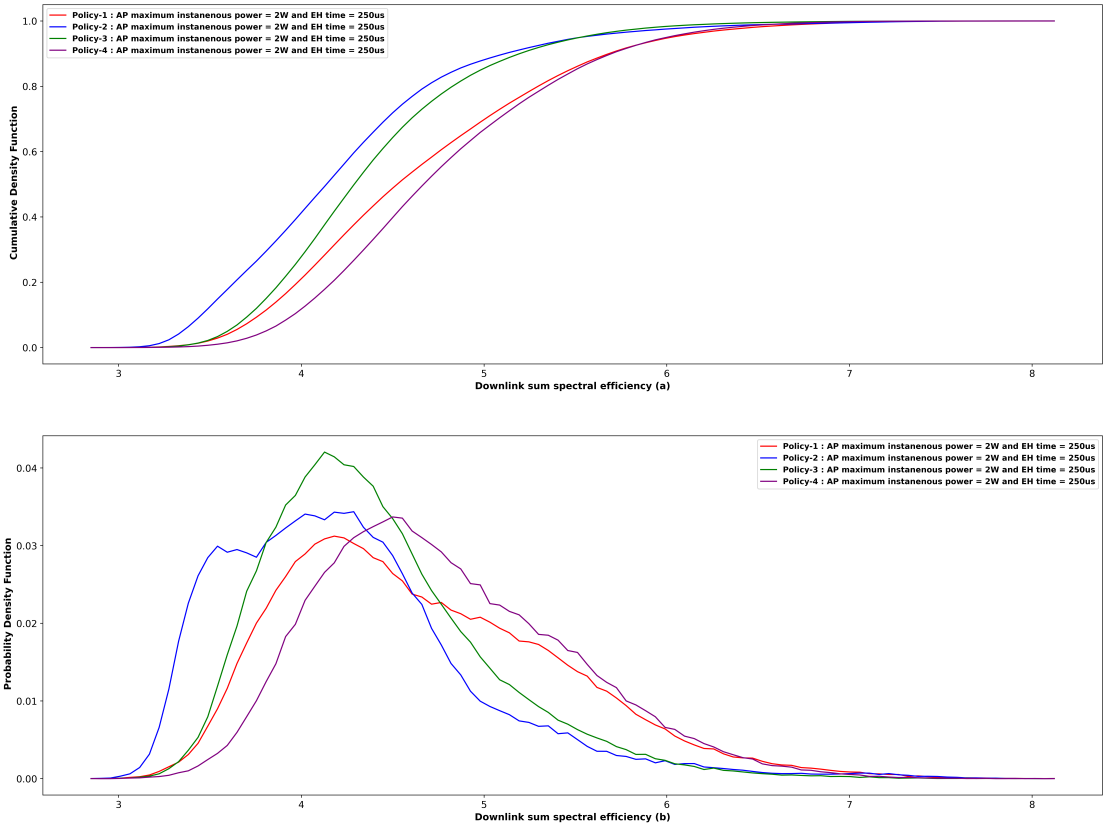


FIGURE 3.14: CDF and PDF of DL SSE.

Figure 3.14 shows Cumulative Density Function (CDF) and Probability Density Function (PDF) for DL SSE. The CDF and PDF results serve to confirm the results in Figure 3.11, the results reveal Policy-4 is more likely to achieve greater DL SSE when compared to other polices and Policy-2 more likely to achieve lower DL SSE when compared the other polices.

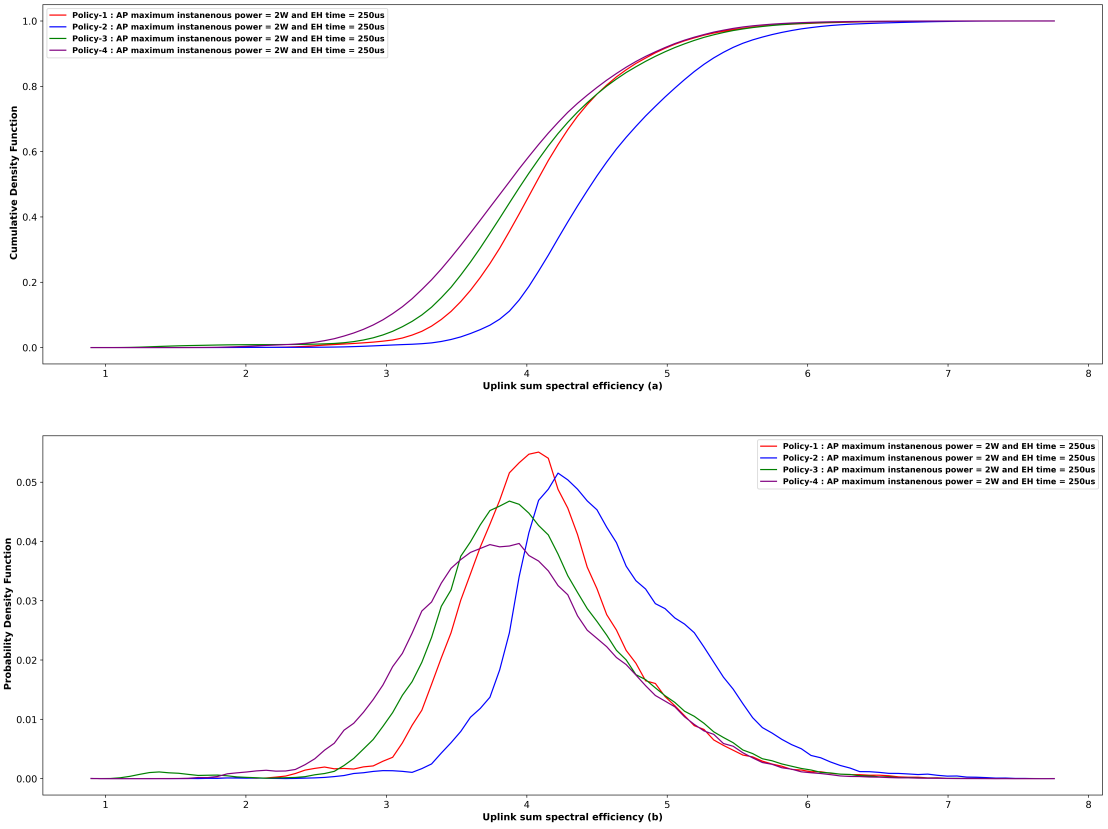


FIGURE 3.15: CDF and PDF UL SSE.

Figure 3.15 shows the CDF and PDF for UL SSE. The inverse result from Figure 3.14 is shown, where Policy-2 is more likely to achieve greater UL SSE when compared to other polices and Policy-4 is more likely to achieve the lowest UL SSE.

Figure 3.16 shows the Probability Mass Function (PMF) for the number of UE that meet the minimum required DL SE. The results reveal that Policy-2 has the highest probability, by a significant margin, of having all the UE meet the minimum required DL SE when compared to the other policies. This contrasts the performance of Policy-4 where the results reveal that it has the lowest probability of having all the UE meet the minimum required DL SE when compared to the other polices. The results also reveal that for Policy-2, the most likely outcome for number of UE that meet the minimum required DL SE is 10, as this outcome has the highest probability. For Policy-1, Policy-3 and Policy-4 the most likely outcome for the number UE that meet minimum required DL SE is 8, as this

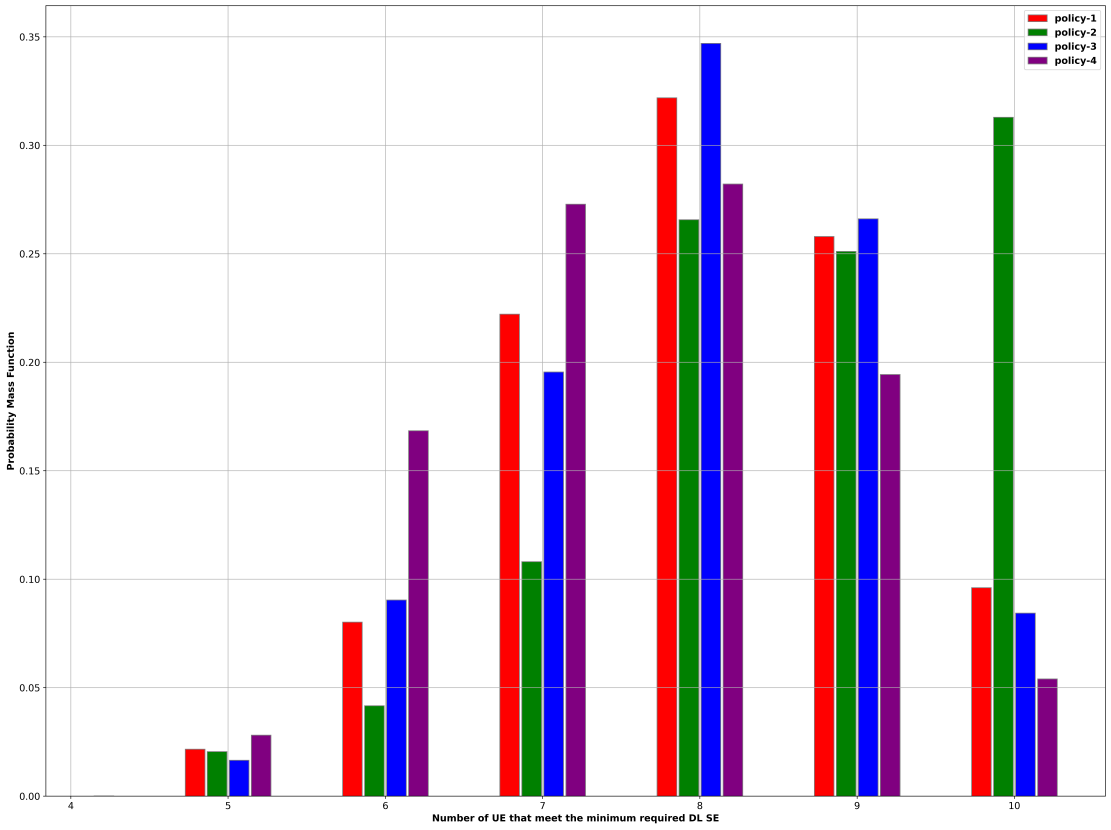


FIGURE 3.16: PMF of the number UE that meet minimum required DL SE.

outcome has the highest probability for the three polices. The results from Figure 3.16 also validate the results from Figure 3.11.

Figure 3.17 shows the PMF for the number of UE that meet the minimum required UL SE. The results reveal that Policy-2 has the highest probability, by significant margin, of having all the UE meet the minimum required UL SE when compared to the other policies. This contrasts the performance of Policy-4 where the results reveal that it has the lowest probability of having all the UE meet the minimum required UL SE when compared to the other polices. The results also reveal that for Policy-2, the most likely outcome for number of UE that meet the minimum required UL SE is 10, as this outcome has the highest probability. For Policy-1, Policy-3 and Policy-4 the most likely outcome for the number UE that meet minimum required DL SE is 8, as this outcome has the highest probability for the three polices. The results from Figure 3.16 also validate the results from Figure 3.12.

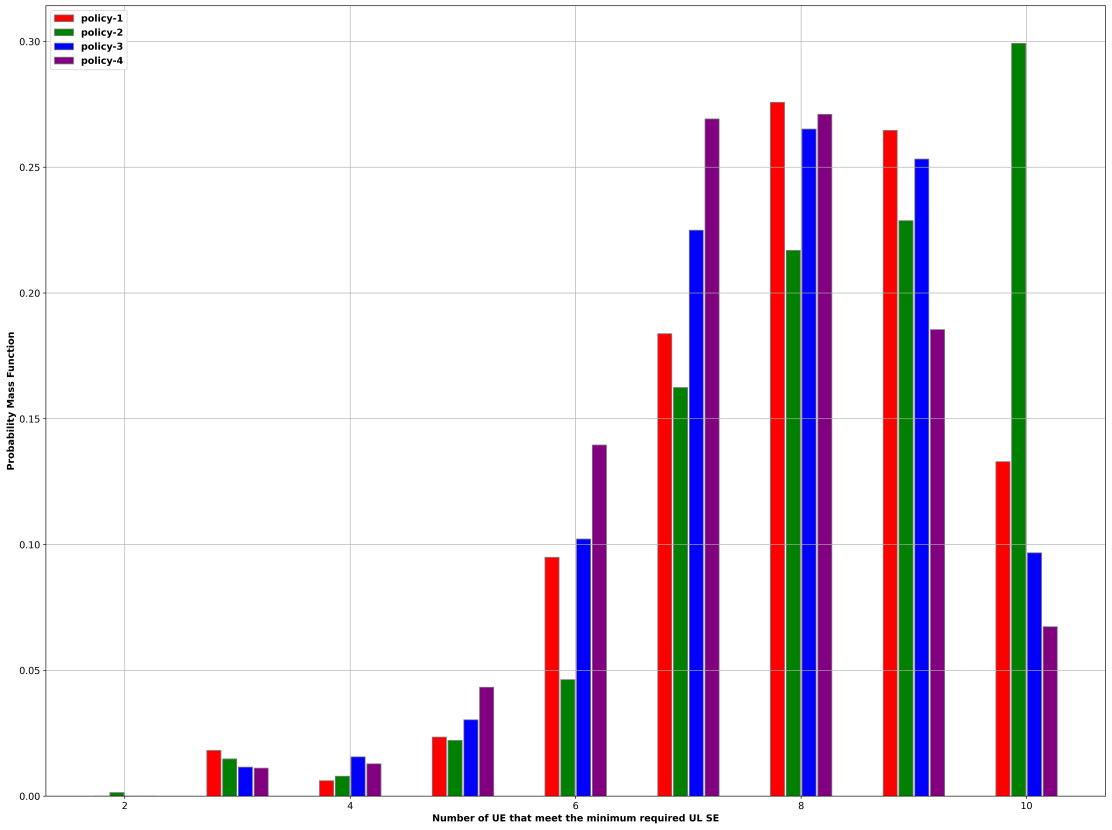


FIGURE 3.17: PMF of the number UE that meet minimum required UL SE.

Figure 3.18 shows the effect of changing the UAVs operating height has on the trained DDPG model's performance; with a particular focus on the number of UE that meet the minimum UL SE requirements. We know from (3.2) and (3.5), that the operating height of UAVs plays an important role in the PLoS which is used to determine large scale coefficient of the system model channel gains in (3.1). In [9], [119] and [120], the effects of a UAV's operating height have been studied and shown to have significant impact in maximising the performance of UAV aided wireless communication systems. However, the work in [9], [119] and [120] do not focus on the impact a UAV's operating height has on energy harvesting of UE. In literature, the UAV operating height ranges from 15m on the low end to 300m on the high end as indicated in [9] and [119], respectively. In Figure 3.18, with the number of UE set to 10 and AP maximum instantaneous transmit power set to 1W, the results reveal there is a clear correlation between the number of UE that minimum required UL SE and the operating height of UAVs. As the operating height of UAV increases, the number UE that meet minimum required

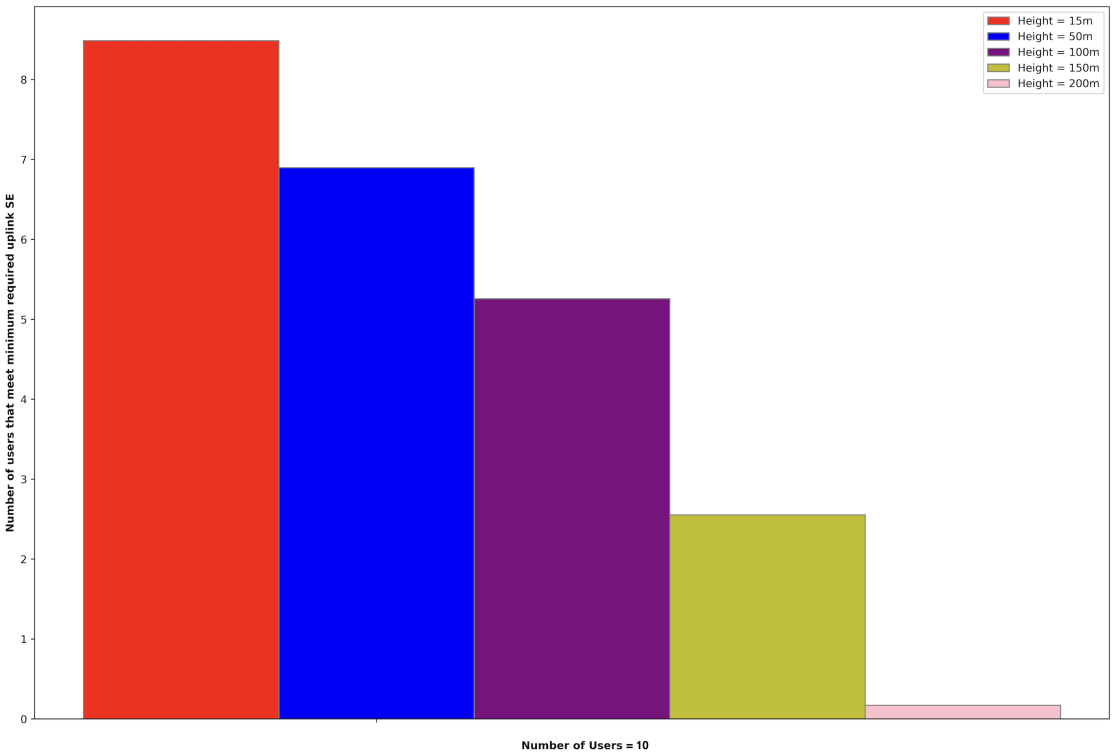


FIGURE 3.18: The impact of UAV operating height on trained DDPG model performance.

UL SE decrease. We can conclude that PLoS does not have a big impact when considering the amount of energy harvested by the UE for UL data transmission. However, we can conclude that the biggest factor in determining the number of UE that meet the minimum required SE is the 3D distance between the UE and UAV. From (3.5), height plays an important role in determining the 3D distance between UAVs and UE, as the UAV operating height increases, the greater the 3D distance between the UAV and UE. From [108], we know the energy harvested is determined by the distance between energy harvesting UE and wireless power source.

Figure 3.19 shows the effect of changing the UAV's operating velocity has on the trained DDPG model's performance, with a particular focus on the number of UE that meet the minimum required DL SE. The model was trained with UAVs velocity set 50m/s. Therefore, we use UAV velocity of 50m/s as the benchmark for comparison. The velocities 10m/s, 25m/s and 50m/s, chosen to test the impact UAVs operating velocity were taken from [119], [9] and [83], respectively. In [119]

and [83], the UAV operating velocities range from 10m/s at the low end to 50m/s at the high end, respectively. The results reveal the model performance for three UAV velocities is in line with the training performance we observed during training when the model converged. Although, the results reveal when UAV's velocity is set to 10m/s, there is a slight performance advantage when compared to when UAV velocity is set to 50m/s and 25m/s, respectively. The performance difference between the three UAVs velocities is minimal and can be attributed to slight variations in UAVs and UE positions. Therefore, we can conclude for the range of UAV operating velocities we tested, changing the UAV's velocity has very little to no effect on the model's performance. This result can be attributed to the deterministic nature of DDPG models, where once the model is trained, when given a state the actions are predictable. In other words, the DDPG model does not consider possible future states once it's been trained, each action-state pair is viewed in isolation. This is opposite of stochastic DRL models that incorporate randomness and uncertainty leading to a variety of actions when given the same state.

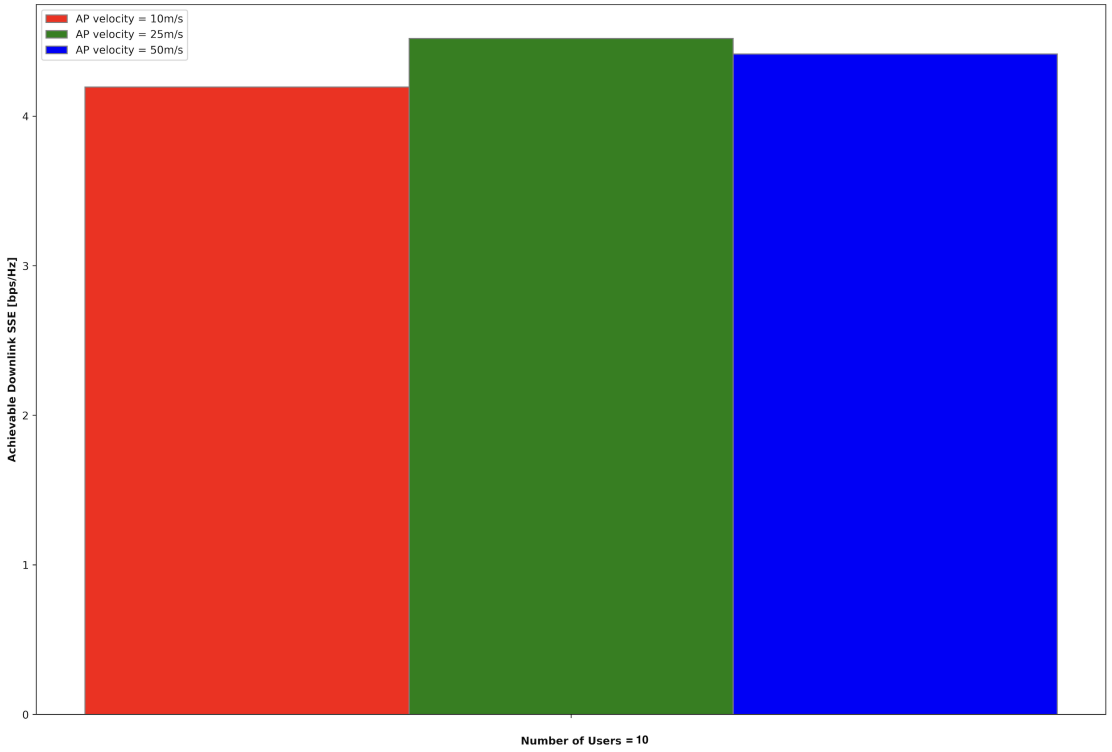


FIGURE 3.19: The impact of UAV operating velocity on trained DDPG model performance.

3.7.3 DDPG Inference Performance

In Figures 3.20, 3.21 and 3.22, we evaluate the inference performance of the trained DDPG model by comparing it to CF-mMIMO Network with random user association and power allocation and CF-mMIMO network with random power allocation and with no user association. When referring to CF-mMIMO with no user association, it means all the APs simultaneously serve all the UE in the defined system model subject to power constraint (C1) outlined in OP1. Note for each simulation a run total of 1000 episodes were used and to ensure fairness during testing, the same dataset was used to test the performance the of DDPG model and the random user association and power allocation schemes.

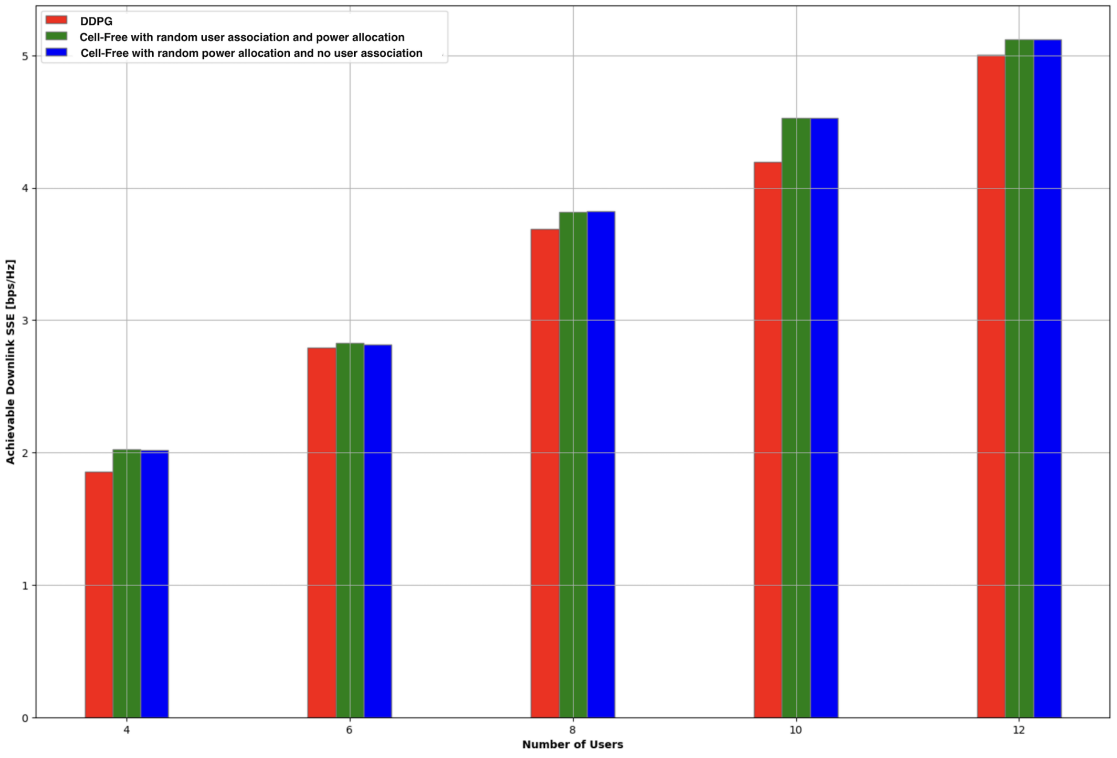


FIGURE 3.20: Achievable DL SSE performance of DDPG model.

In Figure 3.20, we clearly observe that the DDPG model performs the worst with it having the lowest DL SSE. But from (OP1), we know the objective is not only to maximise the DL SSE but to also consider the number of UE that meet the minimum required SE for both DL and UL. It is important to note we expected the DDPG model to have the worst performance in terms of DL SSE as the results

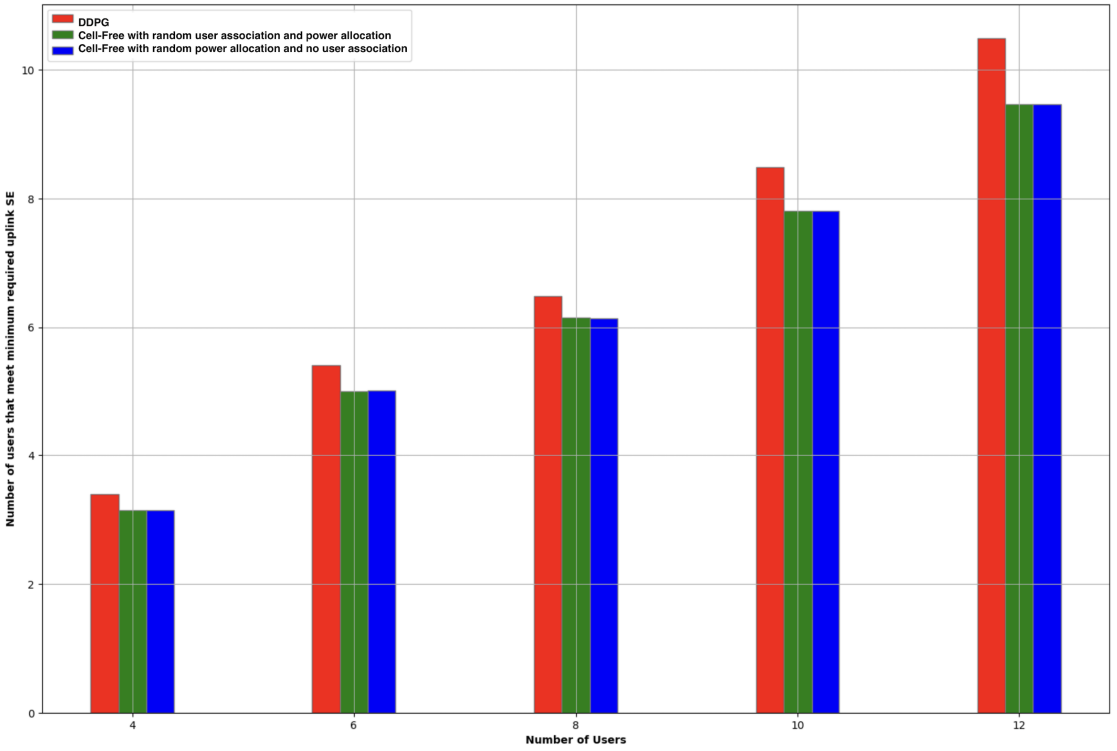


FIGURE 3.21: Achievable number of UE that meet the minimum required DL SE.

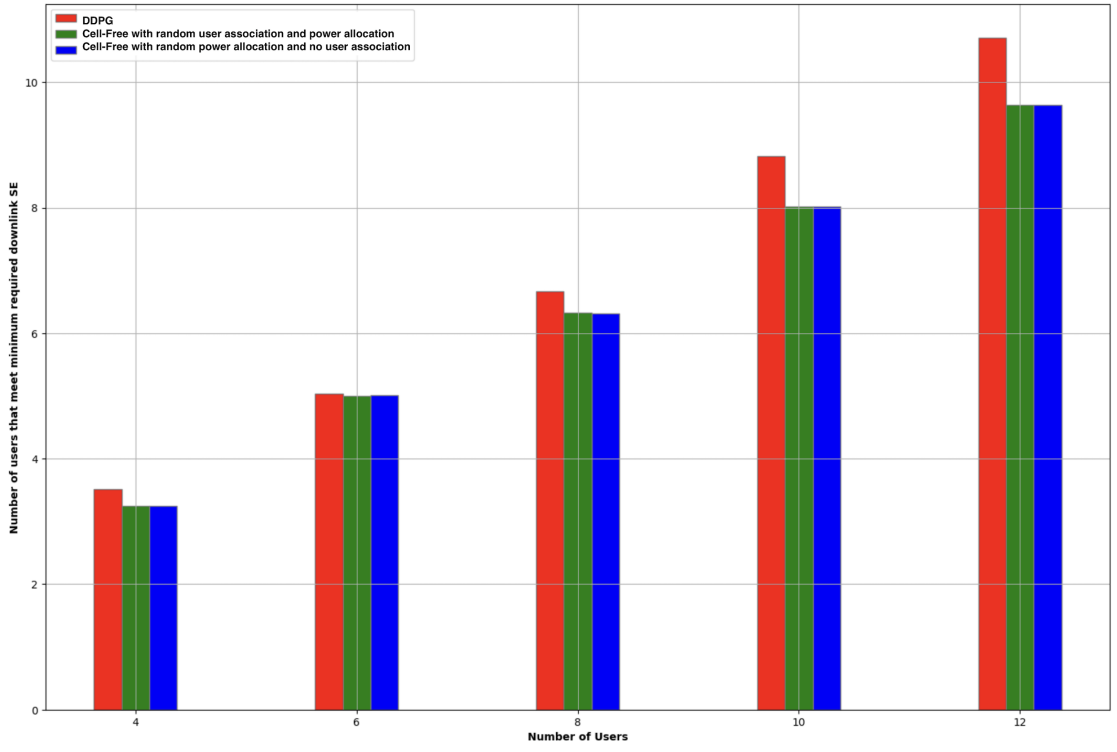


FIGURE 3.22: Achievable number of UE that meet the minimum required UL SE.

from Figures 3.11, 3.12 and 3.13 showed there is an inverse relationship between the DL SSE and the number UE that meet the minimum required DL and UL SE.

In Figure 3.21, we clearly observe that the DDPG performs the best when considering the number of UE that meet the minimum required SE for UL. An interesting observation is that as the number of UE increases in the system model, the bigger the advantage the DDPG model has in terms of comparing the number of UE that meet the minimum required DL and UL SE. This implies as the number UE increases, the more important user association and power allocation powers become. In Figure 3.22, we observe a similar trend for the number of UE that meet the minimum required SE for DL. The results show the DDPG performs the best when considering the number of UE that meet minimum required SE for DL. However, the computational complexity of the DDPG model must be taken into account. It is important to note that random user association and power allocation schemes have a complexity of $\mathcal{O}(n)$, indicating linear-time execution. In comparison, the inference phase of the DDPG model is computationally more intensive, making the random schemes faster in this regard. Additionally, DDPG requires significant computational resources and power during the training phase, although inference consumes considerably less power. When weighing the performance gains against the computational and energy overheads, it becomes evident that employing the DDPG model is more justifiable in scenarios where the system contains a large number of UE.

3.8 Conclusion

In this chapter, we introduced a CF-mMIMO NOMA network that incorporates SWIPT, with machine learning-aided user association and power allocation. We formulated an objective function aimed at maximising the DL SSE, subject to minimum SE requirements for both DL and UL transmissions, as well as AP transmit power constraints.

To solve this optimisation problem, we employed a DRL framework based on a MDP, using the DDPG algorithm. The results demonstrate that the DDPG model outperforms baseline schemes such as random power allocation without user association and random power allocation with random user association. Specifically, DDPG achieves a higher number of UE meeting both the DL and UL SE requirements, although it slightly underperforms in terms of DL SSE compared to the random baseline schemes. However, the improved performance comes at the cost of significantly higher computational complexity compared to the random baseline schemes.

We also analysed the impact of varying the time allocated for energy harvesting on the four different polices of the reward function at different AP transmit power levels. A heuristic approach was used to determine the optimal energy harvesting durations for each transmit power level. The results show that the UAV operating velocity has little to no impact on the performance of the trained model. In contrast, the UAV operating height has a significant impact: as height increases, the amount of energy harvested by UE decreases, which negatively affects the number of UE meeting the minimum UL SE requirements.

Hyper-parameter tuning was conducted to improve the performance of the DDPG model during training. We found that the learning rate had the most substantial impact. The optimal configuration included a learning rate of 10^{-4} for both the actor and critic networks, a batch size of 128, a discount factor of 0.9, and a soft update rate of 0.01, which yielded the best training performance.

CHAPTER 4

A Comparative Analysis of Bootstrapped DDPGs

4.1 Introduction

Recent advancements in DRL training techniques have demonstrated improved performance across established baseline models such as DQN, DDPG, and TD3 [121]-[131]. However, these improvements have predominantly been validated in relatively simple environments, such as Atari games and OpenAI Gym tasks (e.g., Hopper, Cheetah, Walker, and Pong). These environments are typically characterized by low-dimensional state and action spaces, which stand in stark contrast to the high-dimensional, complex nature of real-world problems such as those encountered in wireless communication systems, including CF-mMIMO networks with intricate resource allocation challenges. As a result, there is no clear guarantee that the gains observed in these simplified settings will directly translate to more complex domains, such as the environment introduced in Chapter 3.

To address this gap, this chapter provides a comprehensive review of training techniques that have been shown to enhance DRL performance. Drawing inspiration from the work in [121], we explore the use of various bootstrapping methods to improve the performance of the DDPG model discussed in Chapter 3. Additionally, we introduce a novel variant of DDPG, termed the Multi-Head-Elastic-Step (MH-ES) model. This new approach is evaluated against both the baseline DDPG and several bootstrapped DDPG variants, focusing on performance during both training and inference phases.

The contributions can be summarized as follows:

- In this work, we implement and evaluate the performance of several bootstrapped DDPG variants within a complex CF-mMIMO environment. This approach differs from the study conducted in [121], where performance enhancements were applied to the DQN algorithm and assessed within relatively simple game-based environments. The key contribution lies in extending and adapting these bootstrapping techniques to the DDPG framework and demonstrating that such enhancements can lead to significant performance improvements in a high-dimensional, complex environment. This validates the applicability of these methods beyond simplified benchmarks and highlights their potential for real-world, large-scale wireless communication systems.
- We introduce a novel DDPG variant where we combine Multi-head DDPG and DDPG Elastic step to form Multi-Head-Elastic-Step-DDPG.

The rest of the chapter is structured as follows: Section 4.2 covers the background and literature review on bootstrapping techniques. In Section 4.3, the inference computational complexity of the different DDPG variants is presented, followed by an evaluation of their performance in Section 4.4. Finally, Section 4.5 summarizes the work presented in the chapter.

4.2 Background and Literature Review

4.2.1 Bootstrapping and Thompson Sampling

Bootstrapping in statistics is a metric that uses random sampling with replacement, it involves resampling from a single data set to create a multitude of simulated samples. Those samples are used to calculated standard errors, confidence intervals and prediction errors [130].

The basic principles of Bootstrapping are:

1. Each data point in the original data sample has an equal chance of being chosen and resampled into a simulated sample.

2. Because data points are returned to the original sample after being selected, a data point may be chosen more than once for the same simulated sample.

3. Each resampled or simulated sample is the same size as the original sample

Thompson Sampling (TS), also known as posterior sampling or probability matching, is a heuristic algorithm used for choosing actions that address the exploration-exploitation dilemma in the multi-armed bandit problem [131]. TS balances exploration and exploitation with the goal of selecting actions that maximise the cumulative reward.

Exploration involves performing an action multiple times to gather information. The results of exploration (rewards or penalties) guide the selection of future actions with the aim of maximising rewards and improving performance. One advantage of TS is that as more information is gathered, the algorithm reduces its search, focusing more on exploitation; conversely, when less information is available, it favours exploration.

In RL, TS explores to resolve uncertainty when it is likely that such resolution will help the agent identify the optimal action, but it avoids probing in situations where feedback would not be beneficial. Thus, TS effectively balances the trade-off between exploration and exploitation. The set of past experiences $\mathcal{D}$ consists of actions and rewards. The rewards are modelled using a parametric likelihood function $P(r \mid a, \theta)$, where θ is a set of parameters. In some problems, sampling from or computing the posterior $P(\theta \mid \mathcal{D})$ can be performed using Bayes' rule [131]. To choose the best action from the posterior $P(\theta \mid \mathcal{D})$ according to the probability that the action is optimal is given by

$$\int 1\big[E(r \mid a, \theta) = \max_{a'}\big] \, P(\theta \mid D) \, d\theta. \tag{4.1}$$

In DRL, bootstrapping and Thompson sampling have been used to encourage deep exploration and have been shown to improve DRL models particularly in DQN model as shown in [121],[124],[127]. Although bootstrapping has been used in [121]-[128], there are different ways of bootstrapping a DRL model. In [123], the authors purpose using a Self-Adaptive Double Bootstrapping (SOUP) DDPG to address the inefficient exploration and unstable training that be observed when using baseline DDPG. The SOUP DDPG was shown to improve exploration efficiency by using multiple actor heads for capturing more potential actions and multiple critic heads for evaluating more reasonable Q-values collaboratively. The double bootstrapping in SOUP refers to bootstrapping both the critics and actors. The self adaptive refers to using self-adaptive confidence strategies to address the Double Bootstrapped (DB)-DDPG training oscillations caused by using multiple critic heads. In [123], the authors demonstrate their Bootstrapped DQN can combine deep exploration with DNN for exponentially faster learning compared to traditional dithering-based methods. They use multiple actors with a shared network. The authors in [124], also provide key insights highlighting the differences between bootstrapped DQN, Thompson DQN and Ensemble DQN. The difference between bootstrapped DQN and Thompson DQN, is that Thompson DQN resamples at every time step and the bootstrapped DQN does not. Both Ensemble DQN and bootstrapped DQN use the same architecture with the difference between being that Ensemble DQN uses an ensemble policy. In [125], the authors use a TS approach where they keep multiple hypothesis of the posterior knowledge by introducing multiple actor heads. They maintain the distribution over the potential field-based penalty function. They use TD3 based Thompson sampling where multiple actor heads are adopted. They apply Thompson sampling by randomly drawing an actor from a pool of actors before each episode, representing a hypothesis based on the posterior knowledge. In [126], the authors propose a Multi-Critic-DDPG (MCDDPG) for reliable learning. They cite the training sensitivity and instability caused by relaying on single critic in a base DDPG model. In [127], the authors use a Bootstrapped and Bayesian Deep Q-Network (B2DQN) algorithm based on bootstrapping and TS for joint power control and

beam-forming in mmWave networks. The bootstrapping induces diversity to prevent exploration from being trapped in local optimal and TS allows for targeted exploration of actions with higher estimated returns with higher probability.

4.2.2 N-Step DDPG

Algorithm 3 NS-DDPG For Joint Power Allocation and User Association

Initialise policy parameters θ, Q-function parameters, empty replay buffer D, M is the max number of episodes, T the is max number of time-steps, γ is the discount factor, N =0 is the bootstrap step counter, N-Step is step size for bootstrapping and $G_T = 0$ is the accumulated bootstrapped reward
Initialise environment
for *episode* $\leftarrow$ 1 to M **do**
    Observe states from environment S = $[g_{mnlt}, A_{ml_{t-1}}]$
    Select appropriate action A = $[P, U]$
    Store (s, a, r, s') in reply buffer D;
    for *episode* $\leftarrow$ 1 to T **do**
        if s' *is terminal* **then**
            Reset environment states
        else
            if *step module* $N - Step == 0$ **then**
                for $K \leftarrow 1$ to *updates* **do**
                    Sample mini-batch B from reply buffer D
                    Compute targets;
                    $y(r', s') = r + \gamma Q_{\emptyset targ}(s', \mu_{\emptyset targ}(s))$
                    Update Q-function by one step of gradient decent;
                    $\nabla_{\emptyset} \frac{1}{|B|} \sum_{(s,a,r,s\prime) \in B} \left(Q_{\emptyset}\left(s, a\right) - y(r', s')^2\right)$
                    Update policy by one step of gradient accent;
                    $\nabla_{\theta} \frac{1}{|B|} \sum_{(s,a,r,s\prime) \in B} \left(Q_{\emptyset}\left(s, \mu_{\theta}(s)\right)\right)$
                    Update target networks;
                    $\emptyset_{targ} \leftarrow \rho \emptyset_{targ} + (1 - \rho)\emptyset$
                    $\theta_{targ} \leftarrow \rho \theta_{targ} + (1 - \rho)\theta$
                end for
            else
                Increment N
                $G_t += \gamma^N r_{t+N+1}$
            end if
        end if
    end for
end for

Multi-step also known as N-Step (NS), refers to methods where n-steps are to be bootstrapped during training by the agent, where n refers to the number samples

an agent bootstraps during training. The Q-learning accumulates single reward and then uses the greedy action at the next step to bootstrap [121]. A forward view multi-step target can used as an alternative, the truncated n-step return from a given state can be expressed as

$$R_t^n \equiv \sum_{k=0}^{n-1} \gamma_t^k R_{t+k+1}, \tag{4.2}$$

where R_{t+k+1} is the reward after k steps, n is the number of steps to be bootstrapped and γ_t^k is the discount factor at step k.

In NS-DDPG, when $n > 1$, the bellman optimality equation for the Q^* function is given by

$$Q^*(s_t, a_t) = \mathbb{E}_{\pi^*}\left[R_t^{*(n)} + \gamma^n max_a Q^*(s_{t+n}, a)\right], \tag{4.3}$$

where π^* and n are the policy and bootstrapping step size, respectively [127]. In NS-DDPG, when $n > 1$, the bellman optimality equation for the V^* function is given by

$$V^*(s_t) = \mathbb{E}_{\pi^*}\left[R_t^{*(n)} + \gamma^n V^*(s_{t+n})\right]. \tag{4.4}$$

The baseline DDPG stores its $(s_t, a_t, r_t, s_{t+1}, d_t)$ in a replay buffer to be sampled later for training. In NS-DDPG multi-step experiences are sampled from the replay buffer and used to calculate the direct accumulated reward which are added to the bootstrapped Q-value [127]. The experiences at t are sampled into a mini batch, the n consecutive experiences form t to $t+n+1$ are treated as single multi-time step sample. Therefore, the n-step bootstrapped estimated action value of each sample in the n-step mini-batch $(s_t, a_t, r_t, \cdots s_{t+n}, d_{t+n})$ with size N is given by

$$\hat{Q}_t^{(n)} \begin{cases} \sum_{i=0}^{n-1} \gamma^i r_{t+i} + \gamma^n max_a Q_{\theta Q^-}(s_{t+n}, a) \;\; if \;\; \forall \;\; k \in [1, \cdots, n] \;\; and \;\; d_{t+k} \neq 1 \\ \\ \sum_{i=0}^{k-1} \gamma^i r_{t+i}, \quad if \;\; \exists \;\; k \in [1, \cdots, n] \;\; and \;\; d_{t+k} = 1 \end{cases}, \tag{4.5}$$

where $d_{t+k} = 1$ indicated if the episode is done, otherwise $d_{t+k} = 0$. The Q-value function of NS-DDPG is updated by minimizing the objective is given by

$$L_{Q^Q} = \mathbb{E}_{(s_t, a_t, r_t, \cdots s_{t+n}, d_{t+n})\ D}\left[\hat{Q}_t^{(n)} - Q_{\theta^Q}(s_t, a_t)\right]. \tag{4.6}$$

An example is, if n=2 the agent will bootstrap over two-time steps and when n=5 the agent will bootstrap over five-time steps during training, hence the name n-step. Multi-step methods aim to address high variance problem suffered by DRL model due to overestimation during training, especially when solving complex control problems [124]. The multi-step method has been shown not only to reduce variance during training but also to improve training speed, as demonstrated in [121],[127]-[128]. In [128], the authors show that when n-step methods are applied to DQNs, they are highly effective in improving training convergence and overall training performance. Similarly, the authors in [129], show multi-step methods when applied to DDPG can greatly improve convergence and training performance. The authors in [129] take this further by demonstrating that multi-step DDPG achieves comparable or even better performance than TD3 and SAC. This raises the question: *"Why not simply use a TD3 model instead of applying the multi-step approach to DDPG?"*. Although the question is valid and completely reasonable, it fails to address the differences in architecture between DDPG and TD3. TD3 extends and builds on top of DDPG by introducing an extra critic to address the over estimation issue found in DDPG. This naturally leads to more computational complexity during training which may lead to a slower convergence, longer training time and more computation resources when compared to multi-step DDPG [127].

Algorithm 3 shows the implementation of NS-DDPG. In the NS-DDPG a step size of 5 was shown to perform the best. In terms of implementation differences between the baseline DDPG and NS-DDPG, the N-Step Q and value functions are updated using equations (4.3) and (4.4). There are no structural differences between the baseline DDPG and NS-DDPG actor and critic networks.

4.2.3 Elastic Step DDPG

However, multi-step methods are not perfect and performance is highly dependent on choosing the correct step size. If the step size is too small or too large, the model's performance will be greatly affected as shown in [128],[129]. Therefore, it is imperative that the correct step size be chosen. However, the process of finding the correct step size can be laborious. To address this issue the authors in [128] and [129] introduce an elastic step mechanism to both DQN and DDPG, respectively. The elastic step refers to the automation of the number of steps to be bootstrapped during training by the agent. In [129], the authors introduce the concept of the elastic step by applying it to DQN, demonstrating that it outperforms both the baseline DQN and the non-automated multi-step DQN. It is important to note that the elastic step continues to outperform multi-step DQN, even when the latter uses an appropriately chosen fixed step size. This is because, unlike the static step size used in multi-step DQN, the elastic step in elastic DQN continuously adapts its step size throughout training [129]. The authors in [128] build on the work done in [129] and extend the elastic step to DDPG. Elastic Step DDPG is shown to perform better when compared to baseline DDPG [128].

The performance of Bootstrapped DDPG can be further enhanced by incorporating the elastic step mechanism. Unlike the conventional fixed-step approach, the elastic step dynamically determines the number of steps n to be bootstrapped during training. This automation allows the agent to accumulate experiences deemed similar into a single bootstrapped update, while handling dissimilar experiences separately, thereby improving learning efficiency. At the beginning of training, the agent initializes a storage buffer B and a step counter v. During each time-step, the raw state is stored. A sub-sample is selected based on a sampling value u, and passed through the critic network. The hidden layer output is standardized and its dimensionality is reduced using PCA. The reduced features are then clustered using HDBSCAN. Cluster labels are assigned to both the current state s_t and the next state s_{t+1}. If both states fall into the same cluster, the agent accumulates the reward using:

TABLE 4.1: HDBSCAN Hyper-parameters

Minimum cluster size	5
Cluster selection epsilon	0
Distance metric	Euclidean
Alpha	1
Leaf size	40
Cluster selection epsilon	leaf

$$G_t += \gamma^N r_{t+N+1}, \tag{4.7}$$

where G_t is the accumulated reward, γ is the discount factor, and r is the reward at step $t + N + 1$. The step counter v is then incremented. If the cluster labels differ, the current experience is considered distinct and is added to the replay buffer D for separate training.

Algorithm 4 shows the implementation of the ES-DDPG. The implementation of the ES-DDPG is the same as NS-DDPG apart from an addition of clustering algorithm, used to determine when to bootstrap and additional buffer B, used to store the states. The clustering algorithm used for ES-DDPG is the HDBSCAN. Initially states are stored in B, when the frequency update value is reached the with sample size U, states are randomly retrieved from buffer B and then the subsample is passed to the first critic hidden layer, $critic_h$. The output is then standardized and PCA is applied to the standardized output. After PCA is applied to the outputs, the outputs are then used to train the clustering algorithm. Once the clustering algorithm is trained, when given two consecutive states as input, if the states have the same cluster label the algorithm will bootstrap the reward if cluster labels are different the algorithm will store (s, a, G_t, s') in reply buffer D, where G_t is the bootstrapped reward. The parameters used for HDBSCAN clustering algorithm are shown in Table 4.1. The buffer size for B was 750 and random state sample size U was 250. The Cluster update frequency was set to 5 episodes.

Algorithm 4 ES-DDPG For Joint Power Allocation and User Association

Initialise policy parameters θ, Q-function parameters, empty replay buffer D, M is the max number of episodes, T is the max number of time-steps, γ is the discount factor, B is the state memory bank with capacity H, N is the bootstrap step counter and $G_t = 0$ is the accumulated bootstrapped reward
Initialise environment
for *episode* $\leftarrow 1$ to M **do**
    Observe states from environment S $= [g_{mnlt}, A_{ml_{t-1}}]$
    Select appropriate action A $= [P, U]$
    Observe next state, s' and reward, r
    Store s' in B
    for *step* $\leftarrow 1$ to T **do**
        if s' *is terminal* **then**
            Reset environment states
        else
            if Cluster update frequency is reached **then**
                With sample u, randomly retrieve sub sample
                from B Generate s_h by passing sub sample
                into $critic_h$ Standardise s_h and apply
                PCA Train or retrain Clustering algorithm on s_h
                if cluster labels are not equal, **then**
                    Sample mini-batch B from reply buffer D
                    Compute targets;
                    $y(r', s') = r + \gamma Q_{\emptyset targ}(s', \mu_{\emptyset targ}(s))$
                    Update Q-function by one step of gradient decent;
                    $\nabla_{\emptyset} \frac{1}{|B|} \sum_{(s,a,r,s\prime) \in B} \left(Q_{\emptyset}\left(s, a\right) - y(r', s')^2\right)$
                    Update policy by one step of gradient accent;
                    $\nabla_{\theta} \frac{1}{|B|} \sum_{(s,a,r,s\prime) \in B} \left(Q_{\emptyset}\left(s, \mu_{\theta}(s)\right)\right)$
                    Update target networks;
                    $\emptyset_{targ} \leftarrow \rho \emptyset_{targ} + (1 - \rho)\emptyset$
                    $\theta_{targ} \leftarrow \rho \theta_{targ} + (1 - \rho)\theta$
                else
                    Increment N
                    $G_t += \gamma^N r_{t+N+1}$
                end if
            end if
        end if
    end for
end for

4.2.4 Multi-Head DDPG

In the proposed Bootstrapped MH-ES-DDPG, the actor network is bootstrapped, which has been shown to enhance exploration sampling efficiency. This is achieved by employing multiple actor heads, allowing the model to capture a broader range of potential actions. Given K number of actors with a random set parameter, θ_i, where i $\in K$, drawn from the posterior distribution $P(\theta)$. Each actor head represents a hypothesis and form the distribution $P(\theta|D)$ which is dependent on the prior distribution $P(\theta)$. To maintain the distribution $P(\theta|D)$ each actor head is updated separately based on the this subset, D_i, defined by the mask, mt, drawn from the Bernoulli distribution. The Bernoulli distribution is defined by the masking probability p. The mask mt has the size K and indicates for each actor whether the specific sample is part of its subset or not, this results in the dataset D, which is the replay buffer. Where D is defined by the collection of tuples (s, a, r, s', mt). Each actor head only depends on its own subset D_i. However, the losses from all actor heads are combined to update the shared network [124]. TS is applied by drawing randomly from a pool of actor heads, representing a hypothesis based on the posterior knowledge.

In terms of structural modifications to the baseline DDPG, the critic network remains unchanged, while the actor network is extended by adding multiple actor heads. This approach should not be confused with the use of separate, independent actor networks as in [125]. Instead, the architecture maintains a shared backbone for all actor heads, with multiple instances of the final hidden layer and corresponding output layers one for each defined actor head. Given an input state, the actor network generates multiple action candidates, one from each head. During training, a random actor head is selected and followed for the entire episode, prioritizing exploration over optimality. However, during evaluation, the policy becomes greedy, selecting the actor head that maximises the Q-value [124].

The optimal actor head at each time-step is selected using:

$$\theta^* = \arg\max_{\theta_i} Q(s_t, \pi(s_t, \theta_i)). \tag{4.8}$$

Accordingly, the target for the action-value function is given by

$$y_t = r_t + \gamma \max_i Q'_i(s_{t+1}, \pi'_i(s_{t+1})). \tag{4.9}$$

Algorithm 5 shows the implementation of MH-DDPG. In the MH-DDPG implementation, the multi-head refers to the actor heads. In terms of structural difference between the baseline DDPG and MH-DDPG the critic network remains the same and difference lie in the actor network, with the MH-DDPG having additional actor heads. To avoid confusion when referring to the actor heads, we do not refer to multiple actor networks. In the implementation the actors have a shared body the last hidden layer and the out-layer count as an actor head. The DDPG presented in Chapter 3, has an actor network with two hidden layers with size of [**512**,256], the bold 512 refers to the shared body. To ensure diverse actions from the actor heads, we initialise the actor heads with different random weights. The number of actor heads used in the MH-DDPG were five. It is important to note at the beginning of each episode a random actor head is chosen and in addition the s, a, r, and s' a Bernoulli masking probability is stored in the replay buffer. We set the Bernoulli masking probability in the MH-DDPG to p=0.5.

4.2.5 Multi-Head Elastic-Step DDPG

Algorithm 6 shows the implementation of MH-ES-DDPG. The implementation of the MH-ES-DDPG is essentially an amalgamation of the implementation of the MH-DDPG and ES-DDPG. The same Elastic Step algorithm used in ES-DDPG is applied to MH-DDPG to form MH-ES-DDPG. Therefore, the parameters used for the MH-DDPG and the ES-DDPG are the same parameters used for MH-ES-DDPG.

Algorithm 5 MH-DDPG For Joint Power Allocation and User Association

Initialise policy parameters $\theta_{i\in K}$, Q-function parameters, empty replay buffer D, M is the max number of episodes, T is the max number of time-steps, γ is the discount factor, p is the masking probability
Initialise environment
for *episode* $\leftarrow 1$ to M **do**
    Observe states from environment S = $[g_{mnl_t}, A_{ml_{t-1}}]$
    Pick actor k $\sim$ uniform K
    Select appropriate action A = $[P, U]$
    Observe next state, s' and reward, r
    Sample mask $m \sim$ Bernoulli p
    Store (s, a, r, s', m) in reply buffer D;
    for *step* $\leftarrow 1$ to T **do**
        if s' *is terminal* **then**
            Reset environment states
        else
            for $J \leftarrow 1$ to *updates* **do**
                Sample mini-batch B from reply buffer D
                Compute targets;
                $y(r', s') = r + \gamma Q_{\emptyset targ}(s', \mu_{\emptyset targ}(s))$
                Update Q-function by one step of gradient decent;
                $\nabla_{\emptyset} \frac{1}{|B|} \sum_{(s,a,r,s\prime)\in B} \left(Q_{\emptyset}\left(s, a\right) - y(r', s')^2\right)$
                Update policy by one step of gradient accent;
                $\nabla_{\theta} \frac{1}{|B|} \sum_{(s,a,r,s\prime)\in B} \left(Q_{\emptyset}\left(s, \mu_{\theta}(s)\right)\right)$
                Update target networks;
                $\emptyset_{targ} \leftarrow \rho\emptyset_{targ} + (1-\rho)\emptyset$
                $\theta_{targ} \leftarrow \rho\theta_{targ} + (1-\rho)\theta$
            end for
        end if
    end for
end for

4.3 Complexity Analysis

The inferencing performance of the ES-DDPG and NS-DDPG in terms FLOPS can be calculated using (3.42) and (3.43) from Section 3.5 in Chapter 3. Similar to the baseline DDPG during inferencing for ES-DDPG and NS-DDPG, we only consider the FLOPS for the actor network since the critic is not used during inferencing. However, for the MH-ES-DDPG and MH-DDPG both the critic and actor networks are used during inferencing. To calculate the FLOPS for the policy and value networks MH-ES-DDPG and MH-DDPG we adapt the (3.42) and (3.43)

Algorithm 6 MH-ES-DDPG For Joint Power Allocation and User Association

Initialise policy parameters $\theta_{i\in K}$, Q-function parameters, empty replay buffer D, M is the max number of episodes, T is the max number of time-steps, γ is the discount factor, B is the state memory bank with capacity H, N is the bootstrap step counter, p is the masking probability and $G_T = 0$ is the accumulated bootstrapped reward
Initialise environment
for *episode* $\leftarrow 1$ to M **do**
    Observe states from environment S = $[g_{mnlt}, A_{ml_{t-1}}]$
    Pick actor k $\sim$ uniform K
    Select appropriate action A = $[P, U]$
    Observe next state, s' and reward, r
    Sample mask $m \sim$ Bernoulli p
    Store (s, a, r, s', m) in reply buffer D;
    for *step* $\leftarrow 1$ to T **do**
        if s' *is terminal* **then**
            Reset environment states
        else
            if Cluster update frequency is reached **then**
                With sample u, randomly retrieve sub sample
                from B Generate s_h by passing sub sample
                into $critic_h$ Standardise s_h and apply
                PCA Train or retrain Clustering algorithm on s_h
                if cluster labels are not equal, **then**
                    Sample mini-batch B from reply buffer D
                    Compute targets;
                    $y(r', s') = r + \gamma Q_{\emptyset targ}(s', \mu_{\emptyset targ}(s))$
                    Update Q-function by one step of gradient decent;
                    $\nabla_{\emptyset} \frac{1}{|B|} \sum_{(s,a,r,s\prime)\in B} \left(Q_{\emptyset}\left(s, a\right) - y(r', s')^2\right)$
                    Update policy by one step of gradient accent;
                    $\nabla_{\theta} \frac{1}{|B|} \sum_{(s,a,r,s\prime)\in B} \left(Q_{\emptyset}\left(s, \mu_{\theta}(s)\right)\right)$
                    Update target networks;
                    $\emptyset_{targ} \leftarrow \rho\emptyset_{targ} + (1-\rho)\emptyset$
                    $\theta_{targ} \leftarrow \rho\theta_{targ} + (1-\rho)\theta$
                else
                    Increment N
                    $G_t += \gamma^N r_{t+N+1}$
                end if
            end if
        end if
    end for
end for=0

TABLE 4.2: The number of total FLOPS required during inferencing

DDPG variant	Actor FLOPS	Critic FLOPS	Total FLOPS considered during inferencing
DDPG	361472	362496	361472
NS-DDPG	361472	362496	361472
ES-DDPG	361472	362496	361472
MH-DDPG	591872	1449934	2041806
MH-ES-DDPG	591872	1449934	2041806

to include the additions of extra actor heads. The policy and value networks for MH-ES-DDPG and MH-DDPG is given by

$$\text{FLOPS}_{\text{Policy-MH}} = |S||H_j| + \sum_j \left(|H_{j-1}H_j| + |A|\right)|H_j| \times F \tag{4.10}$$

and

$$\text{FLOPS}_{\text{value-MH}} = \left(|S||H_j| + \sum_j \left(|H_{j-1}H_j| + |A||H_v| + |H_j| + |H_v|\right)\right) \times F \tag{4.11}$$

where $|S|$, $|A|$, $|H_j|$, $|H_v|$, and F are the number of elements in the state, actions, the j-th hidden layer in the policy and value networks, the hidden layer for the action input branch, and the number of actor heads, respectively. The total FLOPS for the DDPG variants are shown in Table 4.2.

4.4 Results and Analysis

We evaluate Cell-Free-mMIMO with a 1 km $\times$ 1 km square area with default parameters shown in Table 4.3. During training, the model is trained on 1000 episodes with each episodes having 50-time steps per episode. At the end each episode it is assumed that both the UE and UAVs can move in any direction within the confines of the stipulated area at velocity of 1m/s and 50m/s, respectively

Table 4.3: System model and simulation parameters

Parameter	**Value**
Number of AP	20
Number of UE	10
Number of UE in a cluster	2
Uplink pilot power	100mW
UE circuitry power consumption	100mW
Maximum AP instantaneous transmit power	2W
Coherence time interval	1ms
Number of symbols per coherence time interval	200
System bandwidth	20MHz
Carrier frequency	2Ghz
Power spectral density of noise	-174dbm/Hz
Noise at the receiver	9dbm
Energy harvesting efficiency coefficient	0.8
Time allocated for energy harvesting	250 microseconds
Successive interference cancellation coefficient	0.01
UAV operating height	15m
UAV maximum velocity	1m/s
UE maximum velocity	50m/s
Path loss exponent for LoS (α_1)	1.2
Path loss exponent for NLoS (α_2)	2.1
Environment constants b_1 and b_2	11.95 and 0.136
Actor learning rate	1×10^{-4}
Critic learning rate	1×10^{-4}
Buffer size	20000
Discount value	0.99
Batch Size	32
Ornstein noise standard deviation	0.1
Soft Update	0.01
Critic network RELU hidden layers	(512,256,256)
Actor network RELU hidden layers	(512,256)

[81]. The operating height of the UAVs is fixed at 15m [9]. Therefore, during each episode, the large fading coefficient of the channel gains are static. However, the small-scale fading coefficients for the channel gains vary at each time-step, with each time-step corresponding to a coherence interval of 1 ms. To ensure the DRL model is trained under diverse conditions accounting for different initial positions of UE and UAVs, the starting positions of both UE and UAVs are uniformly randomized every 10 episodes. Thus, both the large-scale and small-scale channel gain coefficients change. The simulation and training environment was developed in Python, utilizing Tensor-flow for model construction and OpenAI Gym for creating and interacting with the custom environment. It should be noted the system model presented in Chapter 3.1 is the same system model that is used in Chapter 4.

Figure 4.1 and Figure 4.2 show the training performance and training times of NS-DDPG, ES-DDPG, MH-DDPG, and MH-ES-DDPG, respectively. In Figure 4.1, we observe that ES-DDPG and NS-DDPG converge faster than the baseline DDPG, which is consistent with the findings in [127] and [129]. We also observe that MH-DDPG and MH-ES-DDPG converge more slowly than the baseline DDPG, due to the increased computational complexity introduced by having multiple actor heads. This observation aligns with findings in [124] and [125]. Despite the added complexity from the clustering algorithm, ES-DDPG converges at a rate similar to NS-DDPG and exhibits fewer reward fluctuations after convergence. This suggests that the additional computational overhead of Elastic Step DDPG is offset by more efficient bootstrapping, compared to the baseline DDPG and NS-DDPG.

However, this effect is not observed when comparing MH-DDPG with MH-ES-DDPG. In this case, MH-ES-DDPG converges more slowly than MH-DDPG, indicating that the improved bootstrapping efficiency of Elastic Step cannot overcome the increased computational complexity introduced by multiple actor heads. In fact, it slows down convergence relative to MH-DDPG. Nevertheless, once MH-ES-DDPG converges, it demonstrates fewer reward fluctuations compared to MH-DDPG. Furthermore, after the 410th episode, MH-ES-DDPG performs as well as

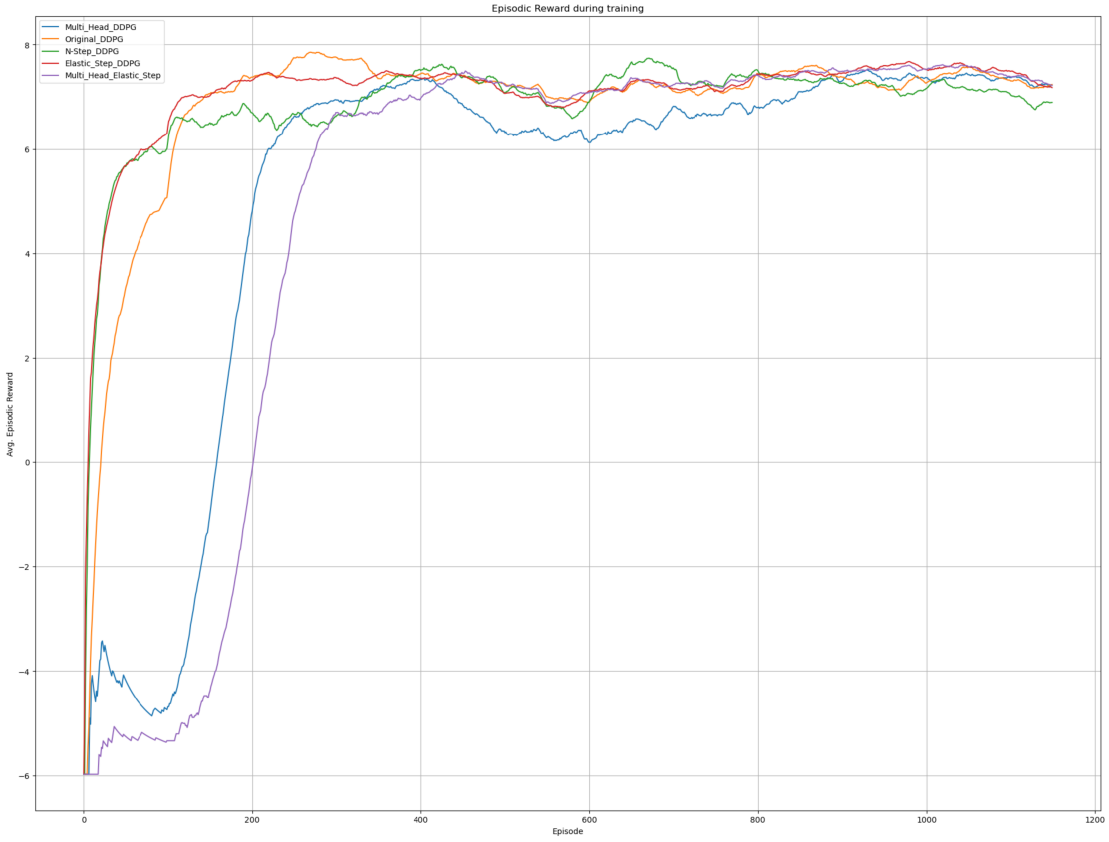


FIGURE 4.1: DDPG variants training performance.

or better than the MH-DDPG.

In Figure 4.2, we study the effect of ES-DDPG, NS-DDPG, MH-DDPG, and MH-ES-DDPG on model training times. We observe that NS-DDPG has the fastest training time, closely followed by the baseline DDPG with ES-DDPG having the third fastest training time. In contrast, MH-DDPG and MH-ES-DDPG take considerably more time to train due to the additional computational complexity introduced by the use of multiple actor heads, when compared to the other DDPG variants. It is important to note that, although this is not directly reflected in the presented training times, NS-DDPG requires significantly more time overall to achieve optimal performance. This is because selecting an appropriate bootstrapping step size is critical for NS-DDPG. If the step size is too small or too large, the model's performance is negatively affected. As a result, we had to run NS-DDPG multiple times to determine the optimal step size, which we found to be 5.

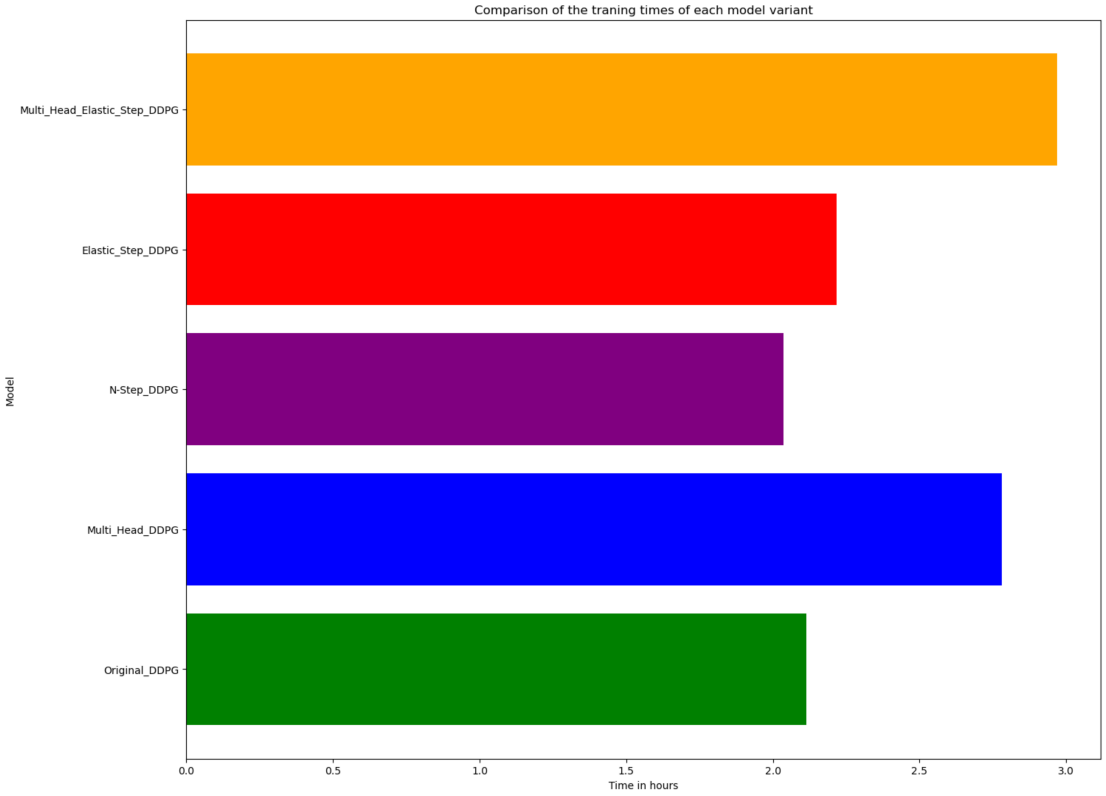


FIGURE 4.2: Comparison of DDPG variants training times.

Additionally, during the process of tuning the bootstrapping step size for NS-DDPG, we occasionally observed a phenomenon known as catastrophic forgetting, where the model completely or significantly forgets previously learned knowledge or skills.

In this section, we evaluate the inference performance of the baseline DDPG, NS-DDPG, ES-DDPG, MH-DDPG, and MH-ES-DDPG. The comparison includes the average achievable SSE, the average number of UE that meet the minimum uplink and downlink SE requirements, the average inference time, and finally, the overall performance of the DDPG variants. To ensure fairness in the evaluation, all DDPG variants are tested using the same dataset generated from the simulation environment. In other words, each variant is evaluated on identical data to allow for a fair and consistent performance comparison.

In Figure 4.3, we observe the inference performance of the DDPG variants in terms of achievable DL SSE. MH-DDPG performs the best, followed by MH-ES-DDPG, then ES-DDPG, then baseline DDPG, and finally NS-DDPG, which performs the

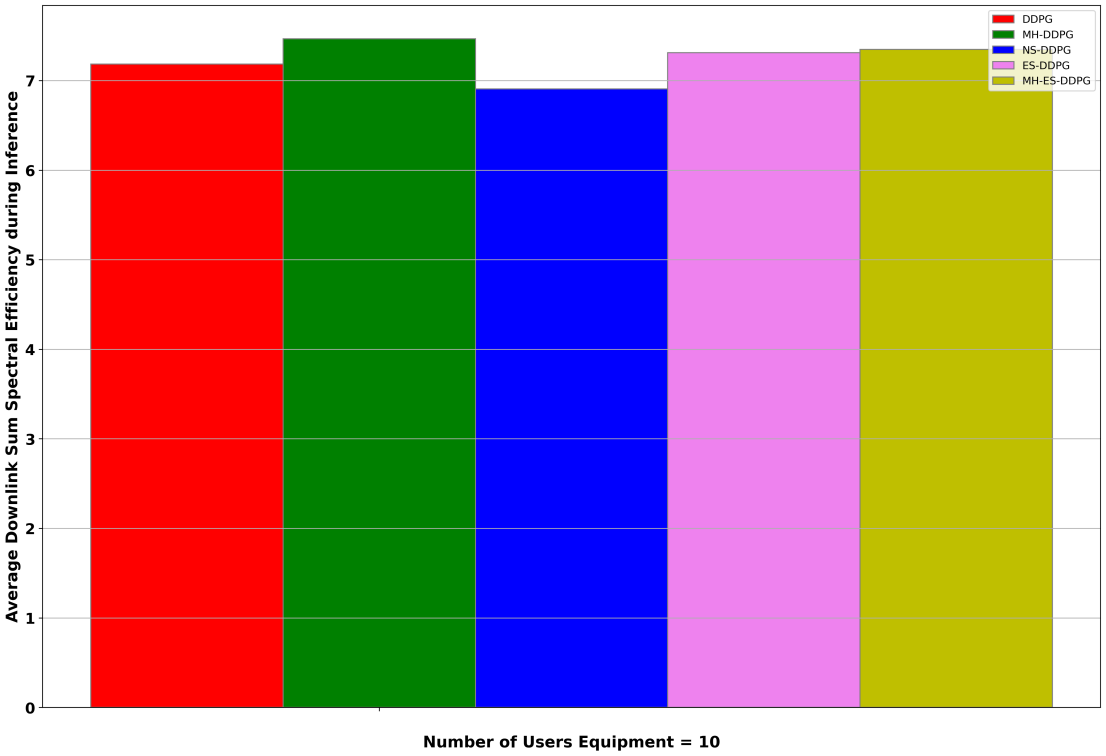


FIGURE 4.3: Comparison of the achievable DL SSE performance of DDPG variants.

worst. However, it is important to note that the performance differences among the DDPG variants are relatively small. The DDPG variants demonstrate similar inference performance, with only minimal variation in DL SSE.

In Figure 4.4, we compare the inference performance of the DDPG variants in terms of the average number of UE that meet the minimum required DL SE. We observe that MH-ES-DDPG, ES-DDPG, and MH-DDPG show a clear and significant performance advantage compared to the baseline DDPG and NS-DDPG. MH-ES-DDPG performs the best, followed closely by ES-DDPG and MH-DDPG, then the baseline DDPG, while NS-DDPG performs the worst by a significant margin. These results suggest that when a fixed bootstrapping step size is used, the model may focus excessively on maximising the overall DL SSE, potentially at the expense of meeting individual UE SE requirements for DL transmission.

In Figure 4.5, we compare the inference performance of the DDPG variants in terms of the average number of UE that meet the minimum required UL SE. We observe that MH-ES-DDPG and MH-DDPG show a clear performance advantage

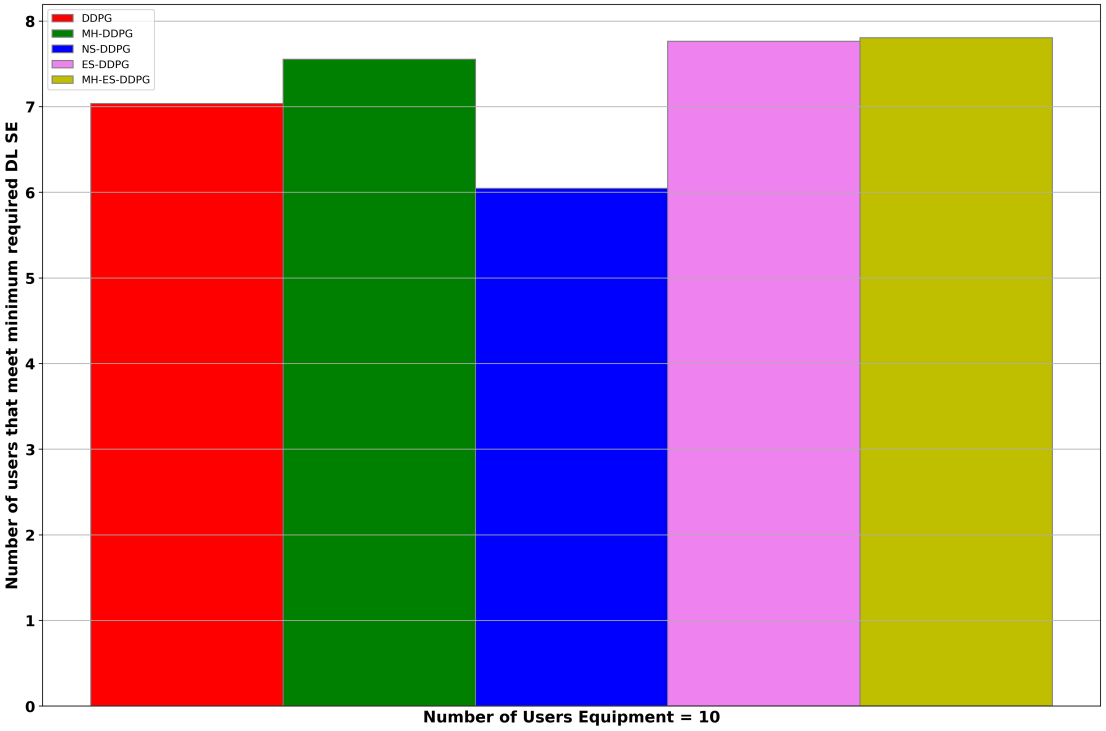


FIGURE 4.4: Comparison of the achievable number of UE that meet the minimum required DL SE.

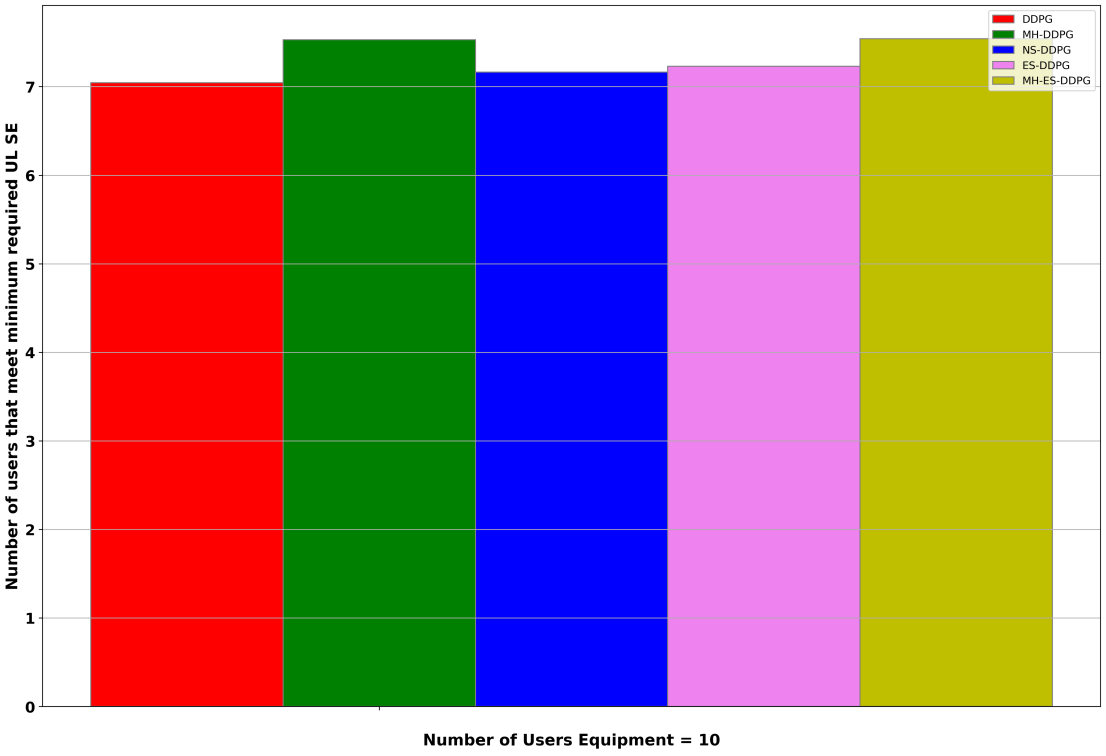


FIGURE 4.5: Comparison of the achievable number of UE that meet the minimum required UL SE.

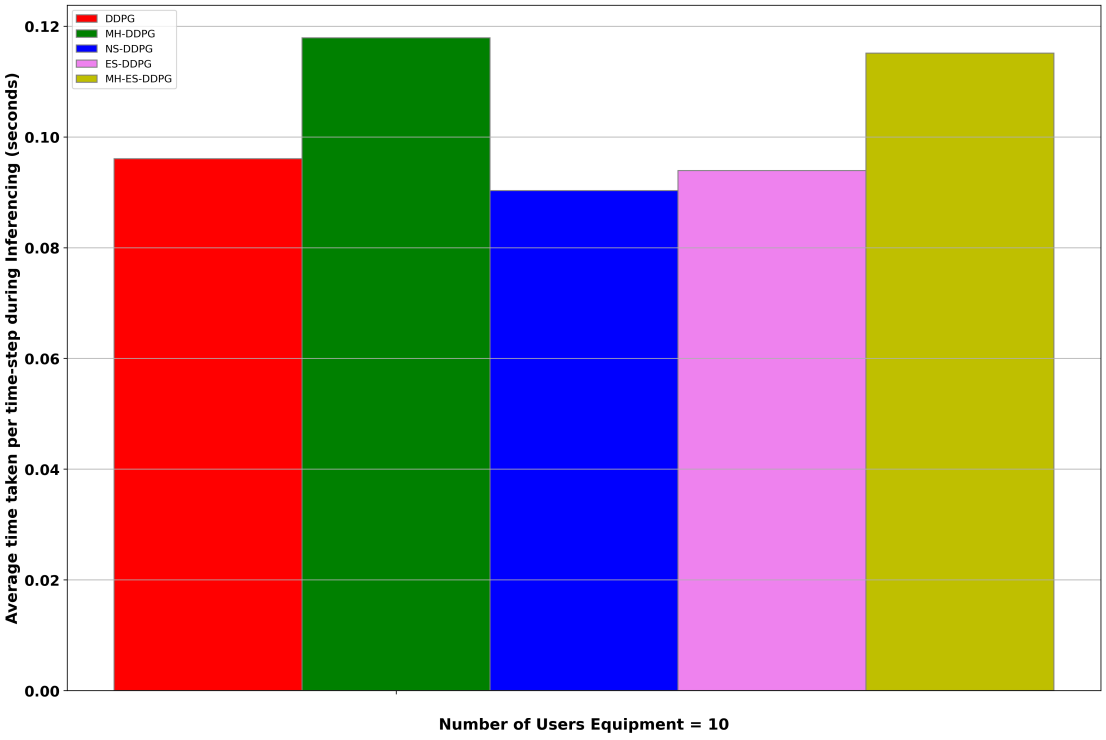


FIGURE 4.6: Comparison of the DDPG variants' reward during inferencing.

over ES-DDPG, the baseline DDPG and NS-DDPG. However, this advantage is less significant compared to the results observed in Figure 4.4. MH-DDPG performs the best, followed closely by MH-ES-DDPG, then ES-DDPG, followed by NS-DDPG, and finally the baseline DDPG, which shows the worst performance. It is important to note that the performance of baseline DDPG, ES-DDPG, and NS-DDPG are relatively similar, with only minor differences between them.

Figure 4.6 shows the inference speed performance of the DDPG variants. On average, NS-DDPG exhibits the fastest inference speed, closely followed by ES-DDPG and the baseline DDPG. In contrast, MH-DDPG and MH-ES-DDPG demonstrate significantly lower inference speed performance compared to the other DDPG variants. This is expected, as both MH-DDPG and MH-ES-DDPG include additional actor heads, five in total. As a result, they generate five different actions, one for each actor head.

In addition to generating multiple actions, both MH-DDPG and MH-ES-DDPG require a critic to evaluate each of the five actions in order to select the one that maximises the reward. The increased computational complexity of these models

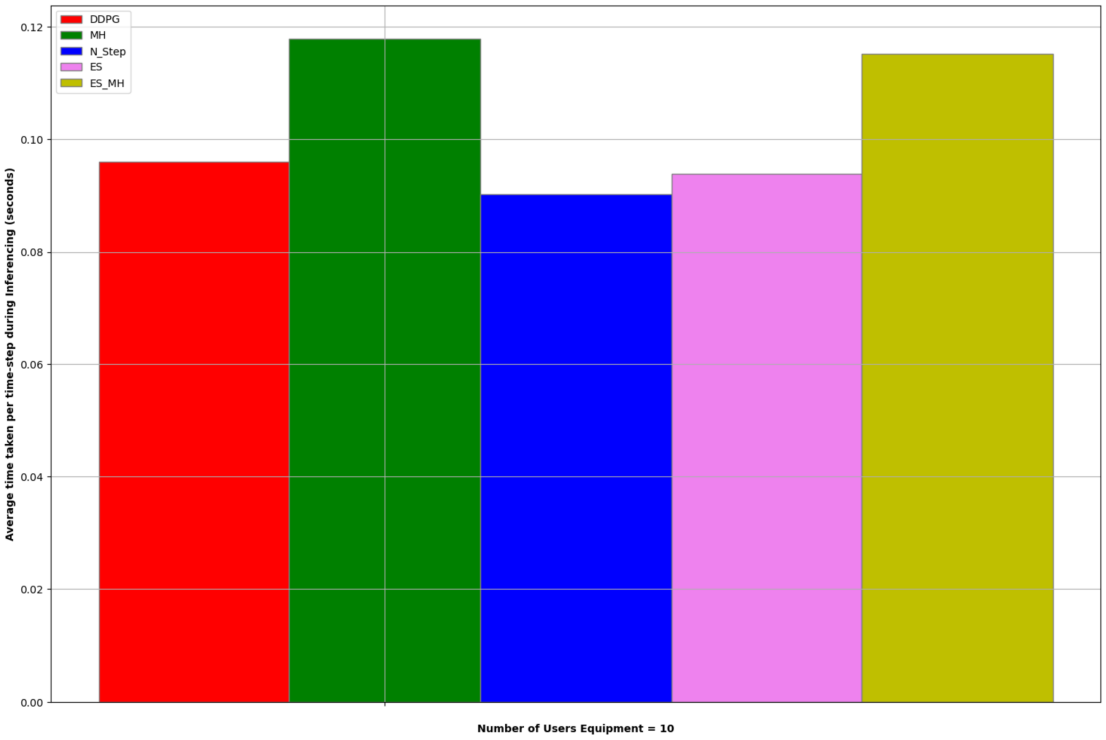


FIGURE 4.7: Comparison of DDPG variants' inferencing speeds.

is reflected in Table 4.2, which shows that MH-DDPG and MH-ES-DDPG require significantly more FLOPS during inference compared to the other DDPG variants. In contrast, baseline DDPG, NS-DDPG, and ES-DDPG each use only a single actor head and therefore, do not rely on a critic during inference.

Another important point to note is that, although ES-DDPG introduces greater computational complexity during training (compared to DDPG and NS-DDPG) due to the clustering algorithm, the inference complexity remains the same as that of the baseline DDPG and NS-DDPG. This is because the clustering algorithm is used only during training to improve performance and is not required during inference, resulting in reduced computational overhead post-training.

In Figure 4.7, we compare the reward performance of the DDPG variants. The reward metric encapsulates the overall system performance, and therefore, it can be used to determine which DDPG variant performs best across all metrics related to the reward function.

Based on the results presented in Figure 4.7, MH-DDPG achieves the highest overall performance, followed closely by MH-ES-DDPG and ES-DDPG. All three of these DDPG variants outperform the unmodified baseline DDPG. However, NS-DDPG fails to outperform or even match the performance of the baseline DDPG. This suggests that NS-DDPG struggles to learn the critical features of the environment due to its dynamic and complex nature, resulting in poor generalisation to unseen states during inference.

In contrast, ES-DDPG performs better because of its ability to adaptively adjust the bootstrapping step size during training. This allows it to more effectively learn important features from the state space. We also observe that applying the Elastic-Step approach to MH-DDPG (i.e., MH-ES-DDPG) offers no significant performance advantage, as the reward performance of MH-DDPG and MH-ES-DDPG is relatively similar.

Overall, MH-DDPG and MH-ES-DDPG demonstrate the best performance across all evaluation metrics, including downlink SSE and the number of UE that meet the minimum required SE for both uplink and downlink transmissions. The performance improvements achieved by the MH-DDPG and MH-ES-DDPG models come at the cost of significantly higher computational complexity, particularly in terms of FLOPs during inference. This increased complexity results in slower inference times compared to the other DDPG variants. Additionally, the power consumption during both training and inference is expected to be higher for MH-DDPG and MH-ES-DDPG due to their multi-head architecture. When considering the trade-offs and advantages between training duration, inference speed and power efficiency, a strong case can be made for ES-DDPG as the most balanced and efficient variant in overall.

4.5 Conclusion

In this chapter we improve the inference and training performance of the DDPG introduced in Chapter 3. We combined the DDPG with various bootstrapping

techniques to form DDPG variants, namely, NS-DDPG, ES-DDPG, MH-DDPG and a novel DDPG variant we named Multi-Head-Elastic-Step DDPG. We show the training performance of the DDPG can be improved in terms of convergence and model training time, with NS-DDPG and ES-DDPG converging faster during training and in the case of NS-DDPG also resulting in faster training times. We show that during inferencing the MH-DDPG, ES-DDPG and MH-ES-DDPG perform better than the baseline DDPG introduced in Chapter 3 in every category related the reward function. This is particularly evident when comparing the number of UE that meet the minimum required DL SE, where the MH-DDPG and MH-ES-DDPG have a significant performance advantage over the baseline DDPG. We conclusively show that the performance of the DDPG can be improved by bootstrapping. We can conclude the MH-DDPG and the MH-ES-DDPG have the best overall performance. However, the performance gains come with cost of additional computational complexity during inferencing as shown in Table 4.2.

CHAPTER 5

Conclusion and Future Work

This chapter summaries the work that was done in this research including the key findings. The chapter also discusses areas of potential improvement as well as potential areas of future research.

5.1 Conclusion

This section concludes the work done in this dissertation.

- In Chapter 1, we outlined the overall scope of the dissertation and provided key insights into the motivation and significance of the research. The chapter also presented the problem statement, described the research methodology, and highlighted the main contributions of the study. Additionally, we introduced the central research question: *'How can machine learning be applied to power allocation and user association in UAV-aided CF-mMIMO systems with NOMA and SWIPT, while accounting for the effects of imperfect SIC and ensuring minimum SE requirements for both UL and DL data transmission?'*

- In Chapter 2, an extensive literature review was conducted and a background of key concepts vital to the system model presented in Chapter 3, was done. We provided a detailed background of different CF-mMIMO architectures, NOMA with a particular focus on PD-NOMA, SWIFT and its two architectures, namely TS-SWIPT and PD-SWIFT, on the use of UAVs in wireless communication systems and DRL. In the case NOMA, SWIPT,

UAVs and DRL, we provide an extensive review of how they have been used in a CF-mMIMO network architecture. From the literature reviewed, a gap was identified where no existing system model integrates CF-mMIMO with SWIPT, NOMA, and UAVs as mobile APs, aiming to maximise DL SSE through joint power allocation and user association optimization using DRL, while accounting for imperfect SIC and minimum SE requirements for both UL and DL transmissions. A gap also existed where no SWIPT frame optimization was done CF-mMIMO.

- In Chapter 3, a CF-mMIMO-NOMA network system model incorporating SWIPT was developed. An objective function was formulated to maximise the DL SSE while satisfying both UL and DL QoS requirements. To solve this problem, a DRL model based on the DDPG algorithm was proposed. The DDPG agent, operating in a hybrid action space, was capable of performing joint user association and power allocation. A reward function aligned with the objective was also designed. Simulation results demonstrated that the DDPG outperformed a random user association and power allocation baseline in the proposed system model. Additionally, a heuristic method was used to determine the optimal energy harvesting time allocation under different UAV transmit power levels. The chapter also investigated the effects of UAV velocity and operating height on the performance of the trained DDPG model. It was shown that UAV velocity, within the tested range, had minimal impact on the model's effectiveness. In contrast, the UAV operating height significantly influenced the number of UE meeting the minimum required (SE) in the UL. As the 3D distance between UAVs and UE increases, the number of UE satisfying the minimum UL SE requirement decreases. Furthermore, a hyper-parameter sensitivity analysis revealed that the learning rate had the most significant impact on training performance compared to other hyper-parameters.

- In Chapter 4, we address the question: *"How can the training and inference performance of the chosen DRL model be improved?"* This was explored through the application of various bootstrapping techniques integrated with

the DDPG model presented in Chapter 3. A brief literature and background review of the bootstrapping methods was first provided, followed by detailed implementations of ES-DDPG, NS-DDPG, MH-DDPG, and the combined MH-ES-DDPG. The results in Chapter 4 demonstrate that both training and inference performance of the baseline DDPG can be improved through bootstrapping mechanisms. In particular, training convergence was significantly enhanced using multi-step methods, as evidenced by the performance of ES-DDPG and NS-DDPG. Furthermore, inference performance was also improved, as shown by the results of ES-DDPG, MH-DDPG, and MH-ES-DDPG. Overall, we conclude that MH-DDPG and MH-ES-DDPG deliver the best performance among the tested variants. However, these performance improvements come at the cost of increased computational complexity during inference.

5.2 Future Work

- In Chapter 3, we operated under the assumption that UAVs possess unlimited power, allowing them to hover and transmit indefinitely. However, in practical scenarios, UAVs are constrained by finite energy resources. A natural extension of the work presented in Chapter 3 would be to incorporate a more realistic system model by introducing a battery and energy consumption framework for the UAVs. It is important to note that in Chapter 3, only the UE harvest energy from radio RF signals, while the UAVs do not engage in any form of energy harvesting. To further enhance the realism and sustainability of the system, we propose extending the model by enabling energy harvesting for the UAVs as well. Rather than relying on RF energy, UAVs can be equipped with solar panels to harvest solar energy. This modification aims to prolong the operational lifetime of UAVs and reduce their dependence on fixed energy reserves.

- In Chapter 3, both the UAVs and UE are modelled to move in random directions. While this is a reasonable assumption for UE since it reflects the

inherent randomness in user mobility in real-world scenarios it is inefficient for UAVs. Random movement of UAVs can lead to suboptimal system performance. Existing literature has shown that optimizing UAV trajectories can significantly enhance network efficiency and overall system performance. Therefore, a valuable extension to the work presented in Chapter 3 would be to incorporate UAV trajectory optimization. By intelligently planning UAV flight paths based on network conditions and user locations, the system can achieve improved coverage, energy efficiency, and communication quality.

- Another potential extension to the work presented in Chapters 3 and 4 is to evaluate the performance of the proposed DRL models under varying UL and DL SE requirements. This would allow us to assess the adaptability and robustness of the DRL models introduced in these chapters. In the current setup, both UL and DL SE requirements are assumed to be fixed and equal. However, in practical scenarios, UL and DL SE demands often vary depending on the specific application and the type of UE involved. By varying these requirements, we can gain deeper insights into how well the models generalize to different communication needs and system configurations.

- There has been growing research interest in the integration of DRL with GNNs. A promising extension of the current work would be to evaluate the performance of the DDPG model when combined with a GNN in the context of the CF-mMIMO system model. GNNs are well-suited for extracting meaningful representations from unstructured data, particularly in systems that can be naturally modelled as graphs where nodes represent entities and edges capture the relationships between them. In our case, the CF-mMIMO network can be represented as a graph where APs and UE are modelled as nodes, and the edges represent associations or communication links between them. By using a GNN to handle the user association task, the DRL model can then focus solely on optimizing power allocation based on the user associations inferred by the GNN. This division of tasks has the potential to significantly reduce the complexity of the DRL problem by narrowing the action space, leading to improved learning efficiency and scalability.

References

[1] Huawei. (n.d.). ITU-R WP5D Completed the Recommendation Framework for IMT-2030 (Global 6G Vision). [online] Available at: https://www.huawei.com/en/huaweitech/future-technologies/itu-r-wp5d-completed-recommendation-framework-imt-2030 (Accessed 23 Jul. 2023).

[2] J. Malmodin, and D. Lundén, 2018. The electricity consumption and operational Report from the KTH Centre for Sustainable Communications Stockholm, Sweden 2018 carbon emissions of ICT network operators. KTH Centre for Sustainable Communications.

[3] Capacity Media. (2011). Environmental issues in telecoms. [online] Available at: https://www.capacitymedia.com/article/29ot4oe12l25efyhle328/top-story/environmental-issues-in-telecoms (Accessed 21 Jul. 2023)

[4] V. Ranasinghe, N. Rajatheva and M. Latva-aho, "Graph Neural Network Based Access Point Selection for Cell-Free Massive MIMO Systems," 2021 IEEE Global Communications Conference (GLOBECOM), Madrid, Spain, 2021, pp. 01-06.

[5] M. Zaher, Ö. Tuğfe Demir, E. Björnson and M. Petrova, "Distributed DNN Power Allocation in Cell-Free Massive MIMO," 2021 55th Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, USA, 2021, pp. 722-726.

[6] Z. Wu, Y. Jiang, Y. Huang, F. -C. Zheng and P. Zhu, "Energy-Efficient Joint AP Selection and Power Control in Cell-Free Massive MIMO Systems: A Hybrid Action Space-DRL Approach," in IEEE Communications Letters, vol. 28, no. 9, pp. 2086-2090, Sept. 2024.

[7] X. T. Dang, M. T. P. Le, H. V. Nguyen, S. Chatzinotas and O. -S. Shin, "Optimal User Pairing Approach for NOMA-Based Cell-Free Massive MIMO Systems," in IEEE Transactions on Vehicular Technology, vol. 72, no. 4, pp. 4751-4765, April 2023.

[8] L. Wang and Q. Zhang, "Cell-Free Massive MIMO with UAV Access Points: UAV Location Optimization," 2022 IEEE/CIC International Conference on Communications in China (ICCC), Sanshui, Foshan, China, 2022, pp. 262-267.

[9] J. Zheng, J. Zhang and B. Ai, "UAV Communications With WPT-Aided Cell-Free Massive MIMO Systems," in IEEE Journal on Selected Areas in Communications, vol. 39, no. 10, pp. 3114-3128, Oct. 2021.

[10] J. Zong, Y. Liu, H. Liu, Q. Wang and P. Chen, "6G Cell-Free Network Architecture," 2022 IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI), Changchun, China, 2022, pp. 421-425.

[11] E. Björnson and L. Sanguinetti, "Scalable Cell-Free Massive MIMO Systems," in IEEE Transactions on Communications, vol. 68, no. 7, pp. 4247-4261, July 2020.

[12] H. A. Ammar, R. Adve, S. Shahbazpanahi, G. Boudreau and K. V. Srinivas, "User-Centric Cell-Free Massive MIMO Networks: A Survey of Opportunities, Challenges and Solutions," in IEEE Communications Surveys Tutorials, vol. 24, no. 1, pp. 611-652, Firstquarter 2022.

[13] I. P. Keramidi, J. S. Vardakas, K. Ramantas, I. Moscholios and C. Verikoukis, "Traffic Management in Cell-Free-Based 6G Networks," 2022 13th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP), Porto, Portugal, 2022, pp. 710-714.

[14] D. Yu, H. Lee, S. -E. Hong and S. -H. Park, "Weighted MMSE Optimization of Conjugate Beamforming for Cell-Free Massive MIMO," 2021 Twelfth

International Conference on Ubiquitous and Future Networks (ICUFN), Jeju Island, Korea, Republic of, 2021, pp. 261-263.

[15] Z. Wang, M. Li, R. Liu and Q. Liu, "Joint User Association and Hybrid Beamforming Designs for Cell-Free mmWave MIMO Communications," in IEEE Transactions on Communications, vol. 70, no. 11, pp. 7307-7321, Nov. 2022.

[16] S. Cho, I. -K. Choi, D. Gwak and S. -E. Hong, "A Delete-based Beam Selection Algorithm for Cell-free Networks," 2021 International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, Korea, Republic of, 2021, pp. 1620-1624.

[17] Y. Al-Eryani and E. Hossain, "Self-Organizing mmWave MIMO Cell-Free Networks With Hybrid Beamforming: A Hierarchical DRL-Based Design," in IEEE Transactions on Communications, vol. 70, no. 5, pp. 3169-3185, May 2022.

[18] F. Fredj, Y. Al-Eryani, S. Maghsudi, M. Akrout and E. Hossain, "Distributed Beamforming Techniques for Cell-Free Wireless Networks Using Deep Reinforcement Learning," in IEEE Transactions on Cognitive Communications and Networking, vol. 8, no. 2, pp. 1186-1201, June 2022.

[19] S. -E. Hong and J. -H. Na, "Joint Access Point Beamforming and Switch On/Off Scheme for Energy Efficient Cell-Free mmWave massive MIMO," 2022 13th International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, Korea, Republic of, 2022, pp. 730-732.

[20] S. -E. Hong, "Comparison Study of RF Beamforming Schemes in Hybrid Precoding for Cell-Free MmWave Massive MIMO UDN System," 2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall), Victoria, BC, Canada, 2020, pp. 1-5.

[21] S. Huang, Y. Ye, M. Xiao, H. V. Poor and M. Skoglund, "Decentralized Beamforming Design for Intelligent Reflecting Surface-Enhanced Cell-Free

Networks," in IEEE Wireless Communications Letters, vol. 10, no. 3, pp. 673-677, March 2021.

[22] Z. Yang and Y. Zhang, "Beamforming optimization for RIS-aided SWIPT in cell-free MIMO networks," in China Communications, vol. 18, no. 9, pp. 175-191, Sept. 2021.

[23] X. Gan, C. Zhong, C. Huang, Z. Yang and Z. Zhang, "Multiple RISs Assisted Cell-Free Networks With Two-Timescale CSI: Performance Analysis and System Design," in IEEE Transactions on Communications, vol. 70, no. 11, pp. 7696-7710, Nov. 2022.

[24] B. Makki, K. Chitti, A. Behravan and M. -S. Alouini, "A Survey of NOMA: Current Status and Open Research Challenges," in IEEE Open Journal of the Communications Society, vol. 1, pp. 179-189, 2020.

[25] L. Dai, B. Wang, Z. Ding, Z. Wang, S. Chen and L. Hanzo, "A Survey of Non-Orthogonal Multiple Access for 5G," in IEEE Communications Surveys Tutorials, vol. 20, no. 3, pp. 2294-2323, thirdquarter 2018.

[26] O. Maraqa, A. S. Rajasekaran, S. Al-Ahmadi, H. Yanikomeroglu and S. M. Sait, "A Survey of Rate-Optimal Power Domain NOMA With Enabling Technologies of Future Wireless Networks," in IEEE Communications Surveys Tutorials, vol. 22, no. 4, pp. 2192-2235, Fourthquarter 2020,

[27] Z. Ding, X. Lei, G. K. Karagiannidis, R. Schober, J. Yuan and V. K. Bhargava, "A Survey on Non-Orthogonal Multiple Access for 5G Networks: Research Challenges and Future Trends," in IEEE Journal on Selected Areas in Communications, vol. 35, no. 10, pp. 2181-2195, Oct. 2017.

[28] J. Zeng et al., "Investigation on Evolving Single-Carrier NOMA Into Multi-Carrier NOMA in 5G," in IEEE Access, vol. 6, pp. 48268-48288, 2018.

[29] N. M. Balasubramanya, A. Gupta and M. Sellathurai, "Combining Code-Domain and Power-Domain NOMA for Supporting Higher Number of Users,"

2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 2018, pp. 1-6.

[30] A. Khansa, Y. Yin, G. Gui and H. Sari, "Power-Domain NOMA or NOMA-2000?," 2019 25th Asia-Pacific Conference on Communications (APCC), Ho Chi Minh City, Vietnam, 2019, pp. 336-341.

[31] P. N. Thakre and S. B. Pokle, "A survey on Power Allocation in PD-NOMA for 5G Wireless Communication Systems," 2022 10th International Conference on Emerging Trends in Engineering and Technology - Signal and Information Processing (ICETET-SIP-22), Nagpur, India, 2022, pp. 1-5.

[32] S. M. R. Islam, N. Avazov, O. A. Dobre and K. -s. Kwak, "Power-Domain Non-Orthogonal Multiple Access (NOMA) in 5G Systems: Potentials and Challenges," in IEEE Communications Surveys Tutorials, vol. 19, no. 2, pp. 721-742, Secondquarter 2017.

[33] X. Zhang and Q. Zhu, "NOMA and User-Centric Based Cell-Free Massive MIMO Over 6G Big-Data Mobile Wireless Networks," GLOBECOM 2020 - 2020 IEEE Global Communications Conference, Taipei, Taiwan, 2020, pp. 1-6.

[34] F. Rezaei, A. R. Heidarpour, C. Tellambura and A. Tadaion, "Underlaid Spectrum Sharing for Cell-Free Massive MIMO-NOMA," in IEEE Communications Letters, vol. 24, no. 4, pp. 907-911, April 2020.

[35] Z. Wang, D. Zhang, K. Xu, W. Xie, J. Xv and X. Li, "NOMA in Cell-Free mMIMO Systems with AP Selection," 2020 International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, China, 2020, pp. 430-435.

[36] A. A. Ohashi et al., "Cell-Free Massive MIMO-NOMA Systems With Imperfect SIC and Non-Reciprocal Channels," in IEEE Wireless Communications Letters, vol. 10, no. 6, pp. 1329-1333, June 2021.

[37] Y. Li and G. A. Aruma Baduge, "NOMA-Aided Cell-Free Massive MIMO Systems," in IEEE Wireless Communications Letters, vol. 7, no. 6, pp. 950-953, Dec. 2018.

[38] T. K. Nguyen, H. H. Nguyen and H. D. Tuan, "Cell-Free Massive MIMO-NOMA with Optimal Backhaul Combining," 2020 IEEE Eighth International Conference on Communications and Electronics (ICCE), Phu Quoc Island, Vietnam, 2021, pp. 455-460.

[39] A. Rafieifar, H. Ahmadinejad and A. Falahati, "IRS-aided NOMA in a Cell Free Massive MIMO System," 2022 30th International Conference on Electrical Engineering (ICEE), Tehran, Iran, Islamic Republic of, 2022, pp. 874-879.

[40] J. Zhang, J. Fan, J. Zhang, D. W. K. Ng, Q. Sun and B. Ai, "Performance Analysis and Optimization of NOMA-Based Cell-Free Massive MIMO for IoT," in IEEE Internet of Things Journal, vol. 9, no. 12, pp. 9625-9639, 15 June15, 2022.

[41] X. Xue, G. Meng and J. Cao, "Secrecy Information and Energy Transmission Optimization of SWIPT System," 2020 9th Asia-Pacific Conference on Antennas and Propagation (APCAP), Xiamen, China, 2020, pp. 1-2.

[42] F. Jameel, Faisal, M. A. A. Haider and A. A. Butt, "A technical review of simultaneous wireless information and power transfer (SWIPT)," 2017 International Symposium onRecent Advances in Electrical Engineering (RAEE), Islamabad, Pakistan, 2017, pp. 1-6.

[43] B. Talukdar, D. Kumar and W. Arif, "Performance Analysis of a SWIPT enabled Cognitive Radio Sensor Network using TS protocol," 2020 Advanced Communication Technologies and Signal Processing (ACTS), Silchar, India, 2020, pp. 1-5.

[44] M. Abedi, H. Masoumi and M. J. Emadi, "Power Splitting-Based SWIPT Systems With Decoding Cost," in IEEE Wireless Communications Letters, vol. 8, no. 2, pp. 432-435, April 2019.

[45] S. Zargari, H. Ahmadinejad, B. Abolhassani and A. Falahati, "SWIPT-NOMA in Cell-Free Massive MIMO," 2020 28th Iranian Conference on Electrical Engineering (ICEE), Tabriz, Iran, 2020, pp. 1-6.

[46] F. Tan, P. Wu, Y. -C. Wu and M. Xia, "Energy-Efficient Non-Orthogonal Multicast and Unicast Transmission of Cell-Free Massive MIMO Systems With SWIPT," in IEEE Journal on Selected Areas in Communications, vol. 39, no. 4, pp. 949-968, April 2021.

[47] S. Kusaladharma, W. -P. Zhu, W. Ajib and G. A. A. Baduge, "Stochastic Geometry Based Performance Characterization of SWIPT in Cell-Free Massive MIMO," in IEEE Transactions on Vehicular Technology, vol. 69, no. 11, pp. 13357-13370, Nov. 2020.

[48] X. Zhang, J. Wang and H. V. Poor, "Statistical Delay and Error-Rate Bounded QoS Provisioning for SWIPT Over CF M-MIMO Based IoT Networks Using FBC," 2021 55th Annual Conference on Information Sciences and Systems (CISS), Baltimore, MD, USA, 2021, pp. 1-6.

[49] Q. Tuan Do, D. Shumeye Lakew, A. Tien Tran, D. Thien Hua and S. Cho, "A Review on Recent Approaches in mmWave UAV-aided Communication Networks and Open Issues," 2023 International Conference on Information Networking (ICOIN), Bangkok, Thailand, 2023, pp. 728-731.

[50] H. Kurunathan, H. Huang, K. Li, W. Ni and E. Hossain, "Machine Learning-Aided Operations and Communications of Unmanned Aerial Vehicles: A Contemporary Survey," in IEEE Communications Surveys Tutorials, vol. 26, no. 1, pp. 496-533.

[51] Y. Bai, H. Zhao, X. Zhang, Z. Chang, R. Jäntti and K. Yang, "Toward Autonomous Multi-UAV Wireless Network: A Survey of Reinforcement Learning-Based Approaches," in IEEE Communications Surveys Tutorials, vol. 25, no. 4, pp. 3038-3067.

[52] Z. Wei et al., "UAV-Assisted Data Collection for Internet of Things: A Survey," in IEEE Internet of Things Journal, vol. 9, no. 17, pp. 15460-15483, 1 Sept.1, 2022.

[53] L. Zhang et al., "A Survey on 5G Millimeter Wave Communications for UAV-Assisted Wireless Networks," in IEEE Access, vol. 7, pp. 117460-117504, 2019.

[54] S. A. Owaid, A. H. Miry and T. M. Salman, "A Survey on UAV-Assisted Wireless Communications: Challenges, Technologies, and Application," 2024 11th International Conference on Electrical and Electronics Engineering (ICEEE), Marmaris, Turkiye, 2024, pp. 333-340.

[55] P. Lohan, B. Kantarci, M. Amine Ferrag, N. Tihanyi and Y. Shi, "From 5G to 6G Networks: A Survey on AI-Based Jamming and Interference Detection and Mitigation," in IEEE Open Journal of the Communications Society, vol. 5, pp. 3920-3974, 2024.

[56] A. Ahmed, C. Chaieb, W. Ajib, H. Elbiaze and R. Glitho, "Resource Allocation and UAVs Placement in Cell-free Wireless Networks," GLOBECOM 2022 - 2022 IEEE Global Communications Conference, Rio de Janeiro, Brazil, 2022, pp. 5995-6000.

[57] C. D'Andrea, A. Garcia-Rodriguez, G. Geraci, L. G. Giordano and S. Buzzi, "Cell-Free Massive MIMO for UAV Communications," 2019 IEEE International Conference on Communications Workshops (ICC Workshops), Shanghai, China, 2019, pp. 1-6.

[58] M. Elwekeil, A. Zappone and S. Buzzi, "Power Control in Cell-Free Massive MIMO Networks for UAVs URLLC Under the Finite Blocklength Regime," in IEEE Transactions on Communications, vol. 71, no. 2, pp. 1126-1140, Feb. 2023.

[59] O. Abbasi and H. Yanikomeroglu, "A Cell-Free Scheme for UAV Base Stations with HAPS-Assisted Backhauling in Terahertz Band," ICC 2022 - IEEE International Conference on Communications, Seoul, Korea, Republic of, 2022, pp. 249-254.

[60] C. Liu, W. Feng, Y. Chen, C. -X. Wang and N. Ge, "Cell-Free Satellite-UAV Networks for 6G Wide-Area Internet of Things," in IEEE Journal on Selected Areas in Communications, vol. 39, no. 4, pp. 1116-1131, April 2021.

[61] L. A. Iliadis, Z. D. Zaharis, S. P. Sotiroudis, P. Sarigiannidis, G. K. Karagiannidis and S. K. Goudos, "Towards 6G: Deep Learning in Cell-Free Massive MIMO," 2022 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom), Sofia, Bulgaria, 2022, pp. 269-273.

[62] W. Chen, X. Qiu, T. Cai, H. -N. Dai, Z. Zheng and Y. Zhang, "Deep Reinforcement Learning for Internet of Things: A Comprehensive Survey," in IEEE Communications Surveys Tutorials, vol. 23, no. 3, pp. 1659-1692

[63] N. C. Luong et al., "Applications of Deep Reinforcement Learning in Communications and Networking: A Survey," in IEEE Communications Surveys Tutorials, vol. 21, no. 4, pp. 3133-3174, Fourthquarter 2019.

[64] Q. Mao, F. Hu and Q. Hao, "Deep Learning for Intelligent Wireless Networks: A Comprehensive Survey," in IEEE Communications Surveys Tutorials, vol. 20, no. 4, pp. 2595-2621, Fourthquarter 2018.

[65] C. Zhang, P. Patras and H. Haddadi, "Deep Learning in Mobile and Wireless Networking: A Survey," in IEEE Communications Surveys Tutorials, vol. 21, no. 3, pp. 2224-2287, thirdquarter 2019.

[66] K. Arulkumaran, M. P. Deisenroth, M. Brundage and A. A. Bharath, "Deep Reinforcement Learning: A Brief Survey," in IEEE Signal Processing Magazine, vol. 34, no. 6, pp. 26-38, Nov. 2017.

[67] T. Nithya, V. Nivas Kumar, G. S, S. Deepa, V. C. M and R. Siva Subramanian, "A Comprehensive Survey of Machine Learning: Advancements, Applications, and Challenges," 2023 Second International Conference on Augmented Intelligence and Sustainable Systems (ICAISS), Trichy, India, 2023, pp. 354-361.

[68] T. R. N and R. Gupta, "A Survey on Machine Learning Approaches and Its Techniques:," 2020 IEEE International Students' Conference on Electrical,Electronics and Computer Science (SCEECS), Bhopal, India, 2020, pp. 1-6.

[69] Z. Zhu and H. Zhao, "A Survey of Deep RL and IL for Autonomous Driving Policy Learning," in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 9, pp. 14043-14065, Sept. 2022.

[70] GeeksforGeeks (2024). Differences between Modelfree and Model-based Reinforcement Learning. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/differences-between-model-free-and-model-based-reinforcement-learning/ (Accessed 15 Dec. 2024)

[71] Z. Zhu, K. Lin, A. K. Jain and J. Zhou, "Transfer Learning in Deep Reinforcement Learning: A Survey," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 11, pp. 13344-13362, 1 Nov. 2023.

[72] Y. Lin et al., "A Survey on Reinforcement Learning for Recommender Systems," in IEEE Transactions on Neural Networks and Learning Systems, vol. 35, no. 10, pp. 13164-13184, Oct. 2024.

[73] S. Munikoti, D. Agarwal, L. Das, M. Halappanavar and B. Natarajan, "Challenges and Opportunities in Deep Reinforcement Learning With Graph Neural Networks: A Comprehensive Review of Algorithms and Applications," in IEEE Transactions on Neural Networks and Learning Systems, vol. 35, no. 11, pp. 15051-15071, Nov. 2024.

[74] T. T. Nguyen, N. D. Nguyen and S. Nahavandi, "Deep Reinforcement Learning for Multiagent Systems: A Review of Challenges, Solutions, and Applications," in IEEE Transactions on Cybernetics, vol. 50, no. 9, pp. 3826-3839, Sept. 2020.

[75] A. Feriani and E. Hossain, "Single and Multi-Agent Deep Reinforcement Learning for AI-Enabled Wireless Networks: A Tutorial," in IEEE Communications Surveys Tutorials, vol. 23, no. 2, pp. 1226-1252, Second quarter 2021.

[76] B. Narottama and T. Q. Duong, "Quantum Neural Networks for Optimal Resource Allocation in Cell-Free MIMO Systems," GLOBECOM 2022 - 2022 IEEE Global Communications Conference, Rio de Janeiro, Brazil, 2022, pp. 2444-2449.

[77] Y. Al-Eryani, M. Akrout and E. Hossain, "Multiple Access in Cell-Free Networks: Outage Performance, Dynamic Clustering, and Deep Reinforcement Learning-Based Design," in IEEE Journal on Selected Areas in Communications, vol. 39, no. 4, pp. 1028-1042, April 2021.

[78] F. D. Tilahun, A. T. Abebe and C. G. Kang, "DRL-based Distributed Resource Allocation for Edge Computing in Cell-Free Massive MIMO Network," GLOBECOM 2022 - 2022 IEEE Global Communications Conference, Rio de Janeiro, Brazil, 2022, pp. 3845-3850.

[79] W. Li, W. Ni, H. Tian and M. Hua, "Deep Reinforcement Learning for Energy-Efficient Beamforming Design in Cell-Free Networks," 2021 IEEE Wireless Communications and Networking Conference Workshops (WCNCW), Nanjing, China, 2021, pp. 1-6.

[80] C. D'Andrea, A. Zappone, S. Buzzi and M. Debbah, "Uplink Power Control in Cell-Free Massive MIMO via Deep Learning," 2019 IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), Le gosier, Guadeloupe, 2019, pp. 554-558.

[81] S. Biswas and P. Vijayakumar, "AP selection in Cell-Free Massive MIMO system using Machine Learning Algorithm," 2021 Sixth International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), Chennai, India, 2021, pp. 158-161.

[82] M. Samir, D. Ebrahimi, C. Assi, S. Sharafeddine and A. Ghrayeb, "Leveraging UAVs for Coverage in Cell-Free Vehicular Networks: A Deep Reinforcement Learning Approach," in IEEE Transactions on Mobile Computing, vol. 20, no. 9, pp. 2835-2847, 1 Sept. 2021.

[83] B. Zhang, Y. Jiang, Y. Huang and P. Zhu, "A DRL Scheme for Resource Allocation in the MEC-Empowered CF-mMIMO System," 2023 IEEE 23rd International Conference on Communication Technology (ICCT), Wuxi, China, 2023, pp. 495-500.

[84] N. Ghafouri, J. S. Vardakas, K. Ramantas and C. Verikoukis, "A Multi-Level Deep RL-Based Network Slicing and Resource Management for O-RAN-Based 6G Cell-Free Networks," in IEEE Transactions on Vehicular Technology, vol. 73, no. 11, pp. 17472-17484, Nov. 2024.

[85] A. K. Kocharlakota, S. A. Vorobyov and R. W. Heath, "Attention Neural Network for Downlink Cell-Free Massive MIMO Power Control," 2022 56th Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, USA, 2022, pp. 738-743.

[86] N. Rajapaksha, K. B. Shashika Manosha, N. Rajatheva and M. Latva-Aho, "Deep Learning-based Power Control for Cell-Free Massive MIMO Networks," ICC 2021 - IEEE International Conference on Communications, Montreal, QC, Canada, 2021, pp. 1-7.

[87] C. Huang, G. Chen, Y. Wen, Z. Lin, Y. Xiao and P. Xiao, "Deep Learning-Based Resource Allocation in UAV-RIS-Aided Cell-Free Hybrid NOMA/OMA Networks," GLOBECOM 2023 - 2023 IEEE Global Communications Conference, Kuala Lumpur, Malaysia, 2023, pp. 5641-5646.

[88] Y. -C. Chuang, W. -Y. Chiu, R. Y. Chang and Y. -C. Lai, "Deep Reinforcement Learning for Energy Efficiency Maximization in Cache-Enabled Cell-Free Massive MIMO Networks: Single- and Multi-Agent Approaches," in IEEE Transactions on Vehicular Technology, vol. 72, no. 8, pp. 10826-10839, Aug. 2023.

[89] M. Rahmani, M. Bashar, M. J. Dehghani, P. Xiao, R. Tafazolli and M. Debbah, "Deep Reinforcement Learning-based Power Allocation in Uplink Cell-Free Massive MIMO," 2022 IEEE Wireless Communications and Networking Conference (WCNC), Austin, TX, USA, 2022, pp. 459-464.

[90] X. Zhang, M. Kaneko, V. An Le and Y. Ji, "Deep Reinforcement Learning-based Uplink Power Control in Cell-Free Massive MIMO," 2023 IEEE 20th Consumer Communications Networking Conference (CCNC), Las Vegas, NV, USA, 2023, pp. 567-572.

[91] F. Fredj, Y. Al-Eryani, S. Maghsudi, M. Akrout and E. Hossain, "Distributed Beamforming Techniques for Cell-Free Wireless Networks Using Deep Reinforcement Learning," in IEEE Transactions on Cognitive Communications and Networking, vol. 8, no. 2, pp. 1186-1201, June 2022.

[92] L. Luo, J. Zhang, S. Chen, X. Zhang, B. Ai and D. W. K. Ng, "Downlink Power Control for Cell-Free Massive MIMO With Deep Reinforcement Learning," in IEEE Transactions on Vehicular Technology, vol. 71, no. 6, pp. 6772-6777, June 2022.

[93] Y. Zhao, I. G. Niemegeers and S. M. H. De Groot, "Dynamic Power Allocation for Cell-Free Massive MIMO: Deep Reinforcement Learning Methods," in IEEE Access, vol. 9, pp. 102953-102965, 2021.

[94] N. Ghiasi, S. Mashhadi, S. Farahmand, S. M. Razavizadeh and I. Lee, "Energy Efficient AP Selection for Cell-Free Massive MIMO Systems: Deep Reinforcement Learning Approach," in IEEE Transactions on Green Communications and Networking, vol. 7, no. 1, pp. 29-41, March 2023.

[95] Z. Wu, Y. Jiang, Y. Huang, F. -C. Zheng and P. Zhu, "Energy-Efficient Joint AP Selection and Power Control in Cell-Free Massive MIMO Systems: A Hybrid Action Space-DRL Approach," in IEEE Communications Letters, vol. 28, no. 9, pp. 2086-2090, Sept. 2024.

[96] Y. Huang, Y. Jiang, F. -C. Zheng, P. Zhu, D. Wang and X. You, "Enhancing Energy-Efficient URLLC in Cell-Free mMIMO Systems With Transceiver

Impairments: An RSMA-DRL Based Approach," in IEEE Wireless Communications Letters, vol. 13, no. 5, pp. 1443-1447, May 2024.

[97] H. Li, W. Sun, Q. Song, L. Guo and A. Jamalipour, "Exploiting RIS's Role for Energy Efficiency and Security in Cell-Free Massive MIMO Networks," in IEEE Communications Letters, vol. 28, no. 5, pp. 1087-1091, May 2024.

[98] D. Deng et al., "Flexible User Duplexing in Cell-Free Massive MIMO: A Deep Reinforcement Learning Approach," 2022 IEEE/CIC International Conference on Communications in China (ICCC), Sanshui, Foshan, China, 2022, pp. 296-301.

[99] H. Suh, S. Kang and T. Hwang, "Intelligent AP Control and Computation Offloading for Cell-Free Massive MIMO MEC Networks," 2024 International Conference on Electronics, Information, and Communication (ICEIC), Taipei, Taiwan, 2024, pp. 1-3.

[100] M. Du, X. Sun, Y. Zhang, J. Wang and P. Liu, "Joint Cooperation Clustering and Downlink Power Control for Cell-Free Massive MIMO with Deep Reinforcement Learning," 2023 IEEE 23rd International Conference on Communication Technology (ICCT), Wuxi, China, 2023, pp. 618-623.

[101] D. Deng, C. Wang, et al., "Joint flexible duplexing and power allocation with deep reinforcement learning in cell-free massive mimo system," China Communications, vol. 20, no. 4, pp. 73-85, 2023.

[102] Q. Wang, Q. Chen, Y. Zhang and H. Wang, "Learning-Based Intelligent Reflecting Surface-Aided Cell-Free Massive MIMO Systems," in IEEE Transactions on Vehicular Technology, vol. 72, no. 9, pp. 12338-12342, Sept. 2023.

[103] F. D. Tilahun, A. T. Abebe and C. G. Kang, "Multi-Agent Reinforcement Learning for Distributed Resource Allocation in Cell-Free Massive MIMO-Enabled Mobile Edge Computing Network," in IEEE Transactions on Vehicular Technology, vol. 72, no. 12, pp. 16454-16468, Dec. 2023.

[104] H. Li, Y. Jiang, Y. Huang, F. -C.Zheng and G. Wu, "A Multi-Agent DRL Method for Distributed Energy-Efficient Association and Hybrid Precoding in mmWave Cell-Free Massive MIMO Systems," in IEEE Communication-sLetters, vol. 29, no. 1, pp. 70-74, Jan. 2025

[105] X. Liao, Y. Zhang, P. Liu and Y.Fu, "Optimizing Energy Efficiency in Cell-Free Massive MIMO Systems: A Distributed DNN-Driven Method," in IEEE Wireless Communications Letters, 2025.

[106] X. Liao et al., "DNN-Based Distributed Downlink Power Control in User-Centric Cell-Free Massive MIMO Systems," 2025 IEEE 26th International Workshop on Signal Processing and Artificial Intelligence for Wireless Communications (SPAWC), Surrey, United Kingdom, 2025, pp. 1-5.

[107] M. Sande and G. Binini, "A deep learning-based channel estimation scheme for cell-free massive MIMO systems,"in SAIEE Africa Research Journal, vol. 116, no. 4, pp. 160-168, Dec. 2025.

[108] O. Kenneth, E. Mwangi and D. B. O. Konditi, "Optimizing UAV Location for Deployment in Cell-Free Massive MIMO Networks Using a Soft Actor-Critic Reinforcement Learning," in IEEE Access, vol. 13, pp. 139680-139695, 2025.

[109] J. Xu et al., "Joint AP Scheduling and Power Allocation Based on Synergistic DRL for Cell-Free Massive MIMO," in IEEE Communications Letters, vol.29, no. 5, pp. 1082-1086, May 2025.

[110] J. Seo et al., "Reinforcement Learning-Based Backhaul Routing for APs in Cell-Free Massive MIMO Networks," 2025 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), Fukuoka, Japan, 2025, pp. 1037-1042.

[111] J. Jiang et al., "Spatio-Temporal GNN-Based Cell-Free Massive MIMO Network With Maximal Benefit-Cost Ratio," in IEEE Transactions on Vehicular Technology, vol. 74, no. 7, pp. 11490-11494, July 2025.

[112] S. Mishra, L. Salaün and H.Yang, "Bigraph GNN-Aided Energy Efficiency Maximization for Cell-Free Massive MIMO," 2025 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN), Barcelona, Spain, 2025, pp. 1-7.

[113] K. D. Christelle Adje, O. Habachi, G. Chalhoub, A. B. Letaifa and M. Haddad,"FedEst: A Federated K-Means Based Load Balancing for Channel Estimation in Cell-Free Massive MIMO Networks," 2025 International Wireless Communications and Mobile Computing (IWCMC), Abu Dhabi, United Arab Emirates, 2025, pp.1235-1240.

[114] Y. Liu et al., "AP Selection in Uplink Cell-Free Massive MIMO: An Unsupervised Heterogeneous GNN Approach," 2025 IEEE Wireless Communications and Networking Conference (WCNC), Milan, Italy, 2025, pp. 1-6.

[115] J. Guo, C. -K. Wen, X. Li and S. Jin, "Deep Learning-based Position-domain Channel Extrapolation for Cell-Free Massive MIMO," in IEEE Transactions on Wireless Communications, 2025.

[116] W. Zhao, H. Wang and R. Song, "Two User Clustering Schemes for Cell-Free Massive MIMO-NOMA System," 2021 13th International Conference on Wireless Communications and Signal Processing (WCSP), Changsha, China, 2021, pp. 1-5.

[117] Openai.com. (2014). Deep Deterministic Policy Gradient — Spinning Up documentation. [online] Available at: https://spinningup.openai.com/en/latest/algorithms/ddpg.html.

[118] C. Diaz-Vilor, A. Lozano and H. Jafarkhani, "Cell-Free UAV Networks: Asymptotic Analysis and Deployment Optimization," in IEEE Transactions on Wireless Communications, vol. 22, no. 5, pp. 3055-3070, May 2023.

[119] A. L. Imoize, H. I. Obakhena, F. I. Anyasi, M. A. Adelabu, K. V. N. Kavitha and N. Faruk, "Spectral Efficiency Bounds of Cell-Free Massive MIMO Assisted UAV Cellular Communication," 2022 IEEE Nigeria 4th International

Conference on Disruptive Technologies for Sustainable Development (NIGERCON), Lagos, Nigeria, 2022, pp. 1-5.

[120] S. Refas et al., "Analysis of Communication Distance and Energy Harvesting for Vehicular VLC Using Commercial Taillights," 2023 5th International Conference on Computer Communication and the Internet (ICCCI), Fujisawa, Japan, 2023, pp. 162-167.

[121] M. Hessel et al., "Rainbow: Combining Improvements in Deep Reinforcement Learning," arXiv.org, 2017. https://arxiv.org/abs/1710.02298

[122] Zheng, Z., Yuan, C., Lin, Z., Cheng, Y. and Wu, H. (2018). Self-Adaptive Double Bootstrapped DDPG.

[123] I. Osband, C. Blundell, A. Pritzel, and B. Van Roy, "Deep Exploration via Bootstrapped DQN," arXiv.org, Feb. 15, 2016.

[124] C. Gebauer and M. Bennewitz, "Penalized Bootstrapping for Reinforcement Learning in Robot Control," 7th International Conference on Computer Science, Engineering and Information Technology (CSEIT 2020), vol. 2020. CS IT, no. CSCP 2020, pp. 155–165, Sep. 2020.

[125] J. Wu, R. Wang, R. Li, H. Zhang and X. Hu, "Multi-critic DDPG Method and Double Experience Replay," 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Miyazaki, Japan, 2018, pp. 165-171.

[126] B. Zeng, Y. Zhong and X. Niu, "Efficient Exploration Through Bootstrapped and Bayesian Deep Q-Networks for Joint Power Control and Beamforming in mmWave Networks," in IEEE Communications Letters, vol. 27, no. 2, pp. 566-570, Feb. 2023.

[127] L. Meng, R. Gorbet and D. Kulić, "The Effect of Multi-step Methods on Overestimation in Deep Reinforcement Learning," 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 2021, pp. 347-353.

[128] A. Ly, R. Dazeley, P. Vamplew, F. Cruz and S. Aryal, "Elastic step DDPG: Multi-step reinforcement learning for improved sample efficiency," 2023 International Joint Conference on Neural Networks (IJCNN), Gold Coast, Australia, 2023, pp. 01-06.

[129] A. Ly, R. Dazeley, P. Vamplew, F. Cruz, and S. Aryal, "Elastic step DQN: A novel multi-step algorithm to alleviate overestimation in Deep Q-Networks," Neurocomputing, pp. 127170–127170, Dec. 2023.

[130] P. Joshi, "What is Bootstrap Sampling in Statistics and Machine Learning?," Medium, Feb. 12, 2020. https://medium.com/analytics-vidhya/what-is-bootstrap-sampling-in-statistics-and-machine-learning-4bb510fa4a8c (accessed Mar. 24, 2025).

[131] D. Eckles and M. Kaptein, "Thompson sampling with the online bootstrap," arXiv (Cornell University), Oct. 2014.