Deep Q-Learning for Edge Caching in Cell Free Massive MIMO Networks: A Multi-Layered Approach

By

Saqlain Razzaq

Supervisor

Dr. Abdul Wakeel

A thesis submitted to the faculty of Electrical Engineering Department Military College of Signals, National University of Sciences and Technology, Rawalpindi as part of the requirements for the degree of MS in Electrical Engineering

October 2023

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS/MPhil thesis written by Mr. **NS Saqlain Razzaq,** Registration No. **00000363132** of **Military College of Signals** has been vetted by undersigned, found complete in all respect as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial, fulfillment for award of MS/MPhil degree. It is further certified that necessary amendments as pointed out by GEC members of the student have been also incorporated in the said thesis.

Signature: [illegible]

Name of Supervisor **Asst Prof Abdul WAkeel, PhD**

Date:

Signature (HoD): [illegible]

Brig
HoD
Dept of Elec Engg
Military College of Sigs (NUST)
(NUST Campus)

Date:

Signature (Dean): [illegible]

Date: 21/7/23

Brig
Dean, MCS (NUST)
(Asif Masood, Phd)

DECLARATION

I certify that this research work titled "Deep Q-Learning for Edge Caching in Cell Free Massive MIMO Networks: A Multi-Layered Approach" is my own work. The work has not been presented elsewhere for assessment. The material that has been used from other sources has been properly acknowledged / referred.

Signature of Student

Saqlain Razzaq

2021-NUST-MS-ELEC-00000363132

DEDICATION

With heartfelt gratitude, I dedicate this endeavor to the pillars of my life: my loving parents, whose unwavering support and guidance have shaped my journey; my supportive siblings, whose belief in me has bolstered my determination; and the esteemed faculty members, whose expertise, mentorship, and encouragement have played a pivotal role in shaping my academic and personal growth. Without the immeasurable love and unwavering support from my family and the invaluable guidance from the faculty members, this achievement would not have been possible.

ABSTRACT

Cell-Free Massive MIMO (CF-mMIMO) networks have emerged as a groundbreaking advancement in wireless communication. This study proposes a novel hierarchical multi-layered approach for the optimization of edge caching in CF-mMIMO networks using the Deep Q-Learning (DQL) Framework. The objective of the proposed framework is to balance service quality and minimize response times, backhaul traffic, and power consumption. A substantial dataset encompassing user queries, response times, content sizes, and caching choices is utilized to evaluate the proposed approach. The proposed framework achieves an impressive 95% Cache Hit Ratio (CHR), indicating efficient cache usage and reduced reliance on remote services. By leveraging the DQL models, the efficiency of edge caching in CF-mMIMO networks is substantially enhanced. This enhancement leads to consistently high cache hit ratios, expedited content delivery, and improved network efficiency. Furthermore, the model's energy efficiency contributes to the network's long-term sustainability. Rigorous testing across diverse scenarios confirms the model's versatility across rural, suburban, urban, and stadium settings. By comparing different caching eviction policies, we establish LFU as optimal for cache efficiency, considering the cache hit ratio and processing time. Overall, this research underscores the substantial advantages of the proposed deep Q-learning model for edge caching in cell-free Massive MIMO networks, significantly impacting cache hit ratio, processing time, and energy efficiency across a range of practical scenarios.

Keywords— Deep Reinforcement Learning (DRL), Network optimization, Content caching, Intelligent decision-making Wireless networks.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Almighty ALLAH for bestowing His blessings upon me and enabling me to successfully complete this research work. Additionally, I am deeply thankful to my supervisors, Assistant Professor Abdul Wakeel, PhD, and Associate Professor Ayaz Ahmad, PhD, for their invaluable support throughout my thesis. Their constant availability for guidance and assistance greatly contributed to the completion of my MS Degree. I would also like to acknowledge the support and valuable suggestions provided by Col. Faisal Akram, PhD, for my research work.

Finally, I extend my heartfelt appreciation to my parents for unwavering care, love, and support during my pursuit of a master's degree.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

List of Acronyms

Multiple Input Multiple Output	MIMO
Cell Free-Massive MIMO	CF-mMIMO
Deep Q-Learning	DQL
Cache Hit Ratio	CHR
Processing Time	PT
Energy Efficiency	EF
Deep Reinforcement Learning	DRL
Least Recently Used	LRU
First-In-First-Out	FIFO
Most Recently Used	MRU
Least Frequently Used	LFU
Segmented Least Recently Used	SLRU
Information Gain	IG
Mutual Information	MI
Quality of Service	QOS
Access Point	AP
Random Replacement	RR
Markov Decision Process	MDP
Channel Quality Information	CQI

Chapter 1

Introduction

This chapter serves as an introduction to the thesis, providing an overview of the study on edge caching optimization in cell-free Massive MIMO networks. The chapter begins by elucidating the evolution of cellular networks, outlining their challenges, and subsequently proposing cell-free Massive MIMO networks as a solution.
The significance of intelligent caching optimization in cell-free Massive MIMO networks is underscored, leading to the problem description, which highlights the limitations of conventional edge caching solutions. The research questions section outlines the main research question and supplementary inquiries, while the research objectives section elucidates the study's overarching goals and specific aims. The chapter culminates by examining the study's importance in the context of contemporary wireless communication developments and the potential impacts of enhanced edge caching in cell-free Massive MIMO networks. The methodology, including its goals, procedures, and constraints, is comprehensively outlined to provide readers with a comprehensive understanding of the study's framework.

1.1 Overview

This section commences by tracing the evolution of cellular networks, encapsulating their progression from analog first-generation (1G) systems to the most recent fifth-generation (5G) networks. This evolution has led to the emergence of cell-free Massive MIMO networks as a response to the challenges faced by traditional cellular networks [1].

1.1.1 Evolution of Cellular Networks

The evolution of wireless communication has been shaped by the progress of cellular networks. Starting with analog first-generation (1G) systems, the transition to digital second-generation (2G) networks marked an improvement in voice quality and data transfer. Subsequent leaps, such as third-generation (3G) and fourth-generation (4G) networks, introduced high-speed data services and mobile internet via Long-Term Evolution (LTE) technology. The fifth-generation (5G) networks further enhanced these capabilities with minimal latency, extensive device connectivity, and high-speed data transmission.

Amid these advancements, conventional cellular networks encountered challenges such as spectrum scarcity, coverage gaps, network congestion, energy inefficiency, and maintaining quality of service (QoS). Cell-free Massive MIMO networks emerged as a solution to address these issues, utilizing distributed antenna systems with multiple access points (APs) [2], [3].

1.1.2 Challenges in Cellular Networks

The escalating demand for seamless wireless communication poses several challenges for cellular networks. Spectrum limitations, coverage gaps, network congestion, energy inefficiency, and QoS maintenance are key hurdles. To mitigate these issues and enhance network performance, novel solutions are sought, such as edge caching optimization in cell-free Massive MIMO networks using deep Q-learning techniques.

1.1.3 Cell-Free Massive MIMO Networks

Cell-free Massive MIMO networks present a promising paradigm in wireless communication. These networks abandon the traditional reliance on a limited number of base stations, instead deploying a distributed antenna system with numerous strategically placed access points (APs). Advanced signal processing enables simultaneous service to multiple users per AP, facilitating cooperative signal processing [4] - [6]. Cell-free Massive MIMO networks offer numerous advantages. These networks enhance coverage through reduced transmitter-receiver distances, leading to improved signal quality and decreased path loss. The presence of multiple access points (APs) facilitates spatial multiplexing and interference management, boosting network capacity and spectral efficiency. Leveraging cutting-edge techniques like beamforming and advanced channel estimation, these networks further enhance throughput, dependability, and energy efficiency by capitalizing on the spatial dimension of communication channels.

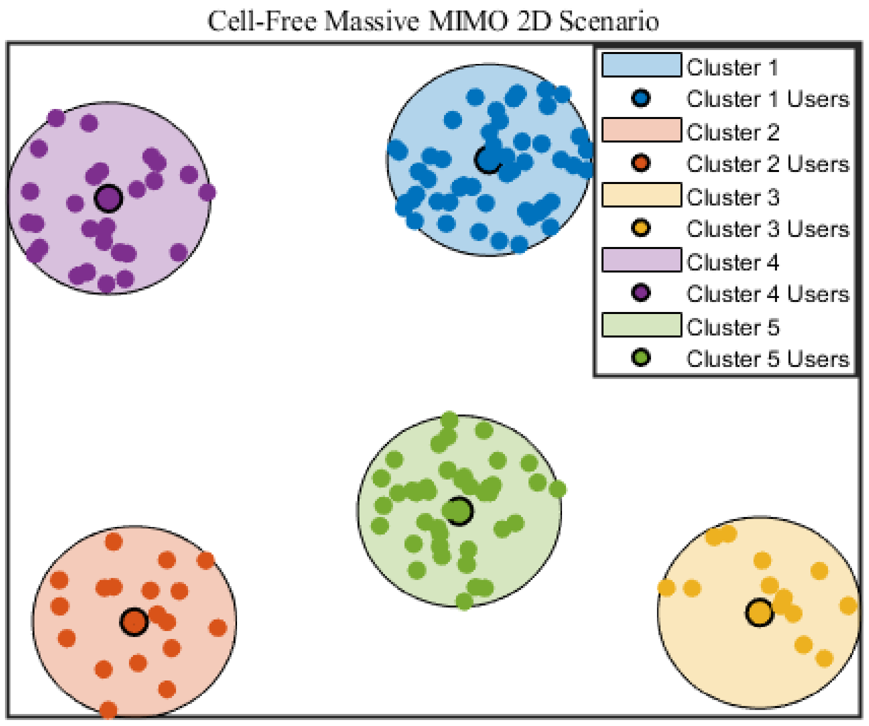

Figure 1.1: Cell-Free Massive MIMO Scenario

Furthermore, the introduction of cell-free Massive MIMO networks has revolutionized wireless communication optimization. These networks effectively address challenges like coverage limitations, spectrum scarcity, and network congestion by exploiting the network's decentralized nature and employing advanced signal processing. The deployment of a dense array of APs maximizes spectrum utilization and elevates the user experience. This new paradigm unlocks novel optimization possibilities for wireless communication systems, paving the way for innovative solutions in network design and operation.

1.2 Caching

1.2.1 Introduction to Edge Caching

1.2.1.1 Caching in Computer Science and Networking:

Caching is a fundamental concept in computer science and networking that involves storing and retrieving frequently accessed data in a faster and more efficient manner. In computer science, caching aims to improve the performance of systems by reducing the

time and resources required to access data. By storing frequently accessed data closer to the processing unit, caching minimizes the latency associated with retrieving data from the original source [8].
In networking, caching plays a crucial role in improving the efficiency and responsiveness of content delivery. In traditional client-server architectures, data requests from clients are typically processed by remote servers. However, this approach can result in increased latency, especially when dealing with large-scale networks or data-intensive applications. Caching addresses this issue by storing frequently accessed data in caches distributed throughout the network, closer to the end-users. When a user requests content, it can be served directly from the cache, reducing the need for data retrieval from remote servers and improving response times [9].

1.2.1.2 Importance of Edge Caching in Wireless Networks:

Edge caching holds significant importance in wireless networks, addressing challenges like limited bandwidth and variable network conditions. It involves storing frequently accessed content at network edges, reducing latency and improving user experience. Key benefits include reduced latency for real-time apps, optimized bandwidth use, and enhanced user satisfaction. Edge caching also enables scalable content delivery in large networks. Despite its benefits, challenges exist, including the need for efficient cache management and handling dynamic content demands.

1.2.1.3 Evolution of Cellular Networks:

Cellular networks have evolved through generations, with technology advancements and user demands driving the changes. The progression can be summarized as follows:
1G (Analog Cellular Systems): Introduced analog communication for voice calls and basic data. Focused on limited capacity and voice communication.
2G (Digital Cellular Systems): Transitioned to digital communication, improving voice quality and data services. Utilized technologies like GSM and CDMA.
3G (Wideband Digital Cellular Systems): Brought high-speed data transmission, enabling multimedia services and mobile internet. UMTS and CDMA2000 boosted data rates and capacity.

4G (LTE): Introduced LTE technology, delivering significant data rate, capacity, and performance improvements. Enabled high-speed internet, streaming, and advanced applications.
5G (Fifth Generation): Current generation with transformative enhancements. Offers high data rates, low latency, capacity, and connectivity. Designed for IoT, autonomous vehicles, AR, and VR applications.

1.2.1.4 Massive MIMO Technology:

Massive Multiple-Input Multiple-Output (MIMO) is a wireless technology that uses multiple antennas for improved communication. It enhances network capacity, coverage, and efficiency. With numerous antennas at base stations, Massive MIMO brings benefits like:
Spectral Efficiency: It supports simultaneous data streams on the same frequency, boosting network capacity for more users and higher data rates.
Coverage and Signal Quality: Multiple antennas enhance coverage and signal quality through techniques like diversity and beamforming, improving reliability and user experience.
Interference Reduction: Advanced processing minimizes interference. Spatial techniques and beamforming help target users, reducing interference from others.
Energy Efficiency: Despite many antennas, Massive MIMO conserves energy by focusing transmission, lowering power use, and reducing costs.

1.2.2 Challenges in Edge Caching Optimization

1.2.2.1 Dynamic and Heterogeneous User Behavior:

User behavior in wireless networks is diverse and dynamic, impacting edge caching optimization. Content popularity varies, and user preferences differ, necessitating adaptive algorithms like Deep Q-Learning to personalize content distribution effectively.

1.2.2.2 Varying Content Popularity and Access Patterns:

Content popularity and usage patterns fluctuate in wireless networks. Edge caching must account for these variations to optimize cache performance. Deep Q-Learning models predict and adapt to these changes, ensuring frequently requested content remains in the cache.

1.2.2.3 Backhaul Traffic Management:

Efficient backhaul traffic management is crucial for edge caching in cell-free Massive MIMO networks. Cache hits reduce backhaul traffic, while cache misses increase it. Optimization strategies consider cache hit ratio, content popularity, and available backhaul capacity to minimize congestion.

1.2.2.4 Energy Efficiency Considerations:

Edge caching contributes to energy efficiency by reducing data transmission distances and conserving energy. Energy consumption varies based on cache hit ratio. Optimizing edge caching for energy efficiency improves sustainability and network longevity.

1.2.2.5 Quality of Experience Requirements:

Quality of Experience (QoE) requirements encompass user expectations for service quality and satisfaction. Edge caching optimally fulfills QoE requirements by enhancing content delivery speed and reducing latency. Different applications and users have varying QoE needs, necessitating adaptable caching strategies.

1.2.3 Existing Approaches for Edge Caching Optimization

1.2.3.1 Traditional Caching Policies (LRU, FIFO, MRU, LFU, SLRU):

Least Recently Used (LRU), First-In-First-Out (FIFO), Most Recently Used (MRU), Least Frequently Used (LFU), Random Replacement (RR), and Segmented LRU (SLRU) are common caching policies. They determine cache item removal or replacement. While simple, they may not consider dynamic factors like content popularity or access patterns.

1.2.3.2 Content Prefetching Techniques:

Content prefetching enhances cache hit rates by predicting future content needs based on past data and user behavior. Techniques like popularity-based prefetching and collaborative filtering utilize prediction models to improve cache performance.

1.2.3.3 Proactive Caching Strategies:

Proactive caching goes beyond reactive methods, anticipating and pre-caching content based on user behavior, content popularity, and network conditions. These strategies aim to have popular material readily available, and combining them with prefetching further improves cache hit rates.

1.2.3.4 Machine Learning-Based Approaches:

Machine learning-based methods leverage data to optimize caching decisions. Decision trees, neural networks, and other techniques capture non-linear correlations between input features and caching choices, enhancing cache performance.

1.2.3.5 Reinforcement Learning in Caching Optimization:

Reinforcement Learning (RL) is promising for caching optimization, particularly in scenarios requiring sequential decisions and trade-offs. RL algorithms like Q-learning and DQN learn responsive caching strategies by adapting to changing conditions and optimizing cache efficiency.

1.2.4 Network Optimization Techniques

The application of network optimization techniques enhances wireless networks' performance, efficiency, and quality of service, achieving goals such as resource allocation, load balancing, and QoS provisioning.

1.2.4.1 Resource Allocation and Power Control:

These techniques manage network resources like bandwidth and power, aiming to boost capacity, reduce interference, and ensure efficient usage. Power control enhances signal quality and energy efficiency while minimizing interference.

1.2.4.2 QoS Provisioning and Traffic Management:

Methods guarantee sufficient throughput, low delay, and minimal packet loss for diverse services. Traffic prioritization, shaping, and QoS-aware scheduling optimize network performance and user satisfaction.

1.2.4.3 Load Balancing:

This approach evenly distributes traffic across network components, preventing overload and maximizing resource utilization. It enhances network speed, user experience, and reduces delays.

Collectively, these strategies optimize wireless networks by maximizing resource utilization, minimizing interference, ensuring QoS, and balancing loads. Given the evolving nature of networks, user needs, and technology, optimization remains a continuous endeavor. Effective application of these techniques necessitates thorough analysis, modeling, and algorithm design to address unique network challenges and meet diverse communication requirements.

1.3 Research Motivations

This research stems from critical challenges in wireless communication networks and edge caching optimization in cell-free Massive MIMO networks. These challenges underscore the study's significance. The evolution of wireless networks, driven by data-intensive applications and demand for high-speed, low-latency services, calls for optimized network performance. Cell-free Massive MIMO networks offer solutions through distributed antenna systems and signal processing techniques, promising enhanced performance. Edge caching addresses response times, backhaul traffic, and energy consumption. Optimizing it in cell-free Massive MIMO networks improves efficiency and user experience. Leveraging deep reinforcement learning techniques, particularly Deep Q-Learning, holds potential for successful caching decisions. Its adaptability to network dynamics, user needs, and content popularity propels this research. Ultimately, this study seeks to advance wireless communication technologies, proposing an optimal edge caching strategy for cell-free Massive MIMO networks. By focusing on these drivers, it aims to enhance network performance and user experience.

1.4 Problem Statement

In order to increase content delivery efficiency, lower latency, and cut down on energy consumption, this research addresses the problem of optimizing edge caching in cell-free Massive MIMO networks. The goal is to improve performance and the user experience by creating a method that improves the cache hit ratio while minimizing processing time and energy usage. By keeping frequently used or popular material in a cache closer to the end-users, response times can be decreased without sacrificing availability. Edge caching is especially important in cell-free Massive MIMO networks, which have several access points dispersed across the coverage region. In order to develop the best caching policy that adjusts to dynamic network conditions and user demands, the study makes use of deep Q-learning, a potent reinforcement learning technique. The research intends to improve content delivery efficiency and user experience in cell-free Massive MIMO networks by making intelligent caching decisions based on this strategy. This will increase the cache hit ratio, decrease processing time, and decrease energy consumption.

1.4.1 Inefficiencies in Traditional Caching Approaches

When it comes to enhancing edge caching in wireless networks, traditional caching technologies generally fall short due to inefficiencies. Several variables contribute to these inefficiencies, which have a negative effect on cache hit rates, response times, and network throughput. Let's look at some of the major drawbacks of the conventional caching approach:

1.4.1.1 Lack of Adaptability:

Traditional caching approaches typically employ static caching policies, such as Least Recently Used (LRU) or First-In-First-Out (FIFO), which do not adapt to changing user demands or network conditions. These static policies do not consider the popularity or relevance of content, leading to suboptimal caching decisions. As a result, frequently requested content may not be cached, and cache space may be wasted on less popular or outdated content.

1.4.1.2 Limited Spatial Reusability:

Traditional caching approaches often rely on fixed caching locations, such as base stations or content delivery nodes. This limited spatial reusability restricts the coverage area of cached content and may result in inefficient cache utilization. Users located far from the caching nodes may experience longer response times as content needs to be retrieved from remote servers instead of being served from nearby caches.

1.4.1.3 Lack of Content Prefetching:

Traditional caching approaches often lack mechanisms for proactive content prefetching. Prefetching involves predicting user demands and proactively caching content that is likely to be requested in the future. By not incorporating prefetching strategies, traditional caching approaches miss opportunities to improve cache hit ratios and reduce response times.

1.4.1.4 Insufficient Consideration of Network Dynamics:

Cell-free Massive MIMO networks exhibit dynamic characteristics, including varying user densities, changing channel conditions, and evolving content popularity. However, traditional caching approaches may not effectively adapt to these dynamic factors. As a result, cache contents may become outdated, and popular content may not be promptly cached or updated, leading to suboptimal cache hit ratios.

1.4.1.5 Limited Intelligence in Caching Decisions:

Traditional caching approaches often lack intelligent decision-making capabilities. They do not consider contextual information such as user preferences, network conditions, or historical usage patterns. Without intelligent decision-making, traditional caching approaches may make suboptimal choices in caching content, leading to reduced cache hit ratios and degraded user experience.

1.4.1.6 Inefficient Utilization of Cache Resources:

Traditional caching approaches may not effectively utilize the available cache resources. Content eviction policies, such as LRU or FIFO, may not consider the content popularity or the potential benefits of retaining certain content in the cache. This inefficient utilization of cache resources can result in poor cache hit ratios and increased reliance on remote servers.

Traditional caching approaches in wireless networks have many drawbacks, such as their inability to adapt to changing conditions, their lack of spatial reuse, their inability to prefetch content, their failure to take network dynamics into account, their lack of intelligence in making caching decisions, and their inefficiency in making use of cache resources. Edge caching optimization in cell-free Massive MIMO networks is hindered by these inefficiencies, resulting in inferior cache hit ratios, prolonged reaction times, and diminished network performance. The proposed multi-layered approach based on deep Q-learning is one example of an advanced method that might be used to improve the efficacy and efficiency of edge caching in such networks and address these inefficiencies.

1.4.2 Need for Optimization in Edge Caching for Cell-Free Massive MIMO Networks

Cell-free Massive MIMO networks have emerged as a promising technology for next-generation wireless communication systems. In these networks, the traditional base station concept is replaced by a large number of distributed access points (APs) deployed densely across the coverage area. While cell-free Massive MIMO networks offer several advantages, including increased capacity, improved coverage, and enhanced spectral efficiency, optimizing edge caching is crucial to fully exploit their potential.

1.4.2.1 Increasing Data Demand:

With the proliferation of data-intensive applications and services, the demand for content delivery has significantly increased. Users expect fast and seamless access to their desired content, regardless of their location or network conditions. The sheer volume of data and the need for low-latency delivery pose challenges for traditional network architectures. Edge caching, where frequently accessed content is stored closer to the end-users, can alleviate these challenges by reducing the distance between content and users, leading to faster response times and improved user experience.

1.4.2.2 Dynamic and Heterogeneous User Behavior:

Cell-free Massive MIMO networks serve a diverse user base with varying demands and preferences. Users access a wide range of content, from video streaming to real-time gaming and social media applications. Furthermore, user behavior and content popularity exhibit dynamic variations over time and across different locations. Traditional caching approaches may struggle to adapt to these dynamic and heterogeneous user behaviors, resulting in inefficient cache utilization and suboptimal cache hit ratios.

Optimization in edge caching is necessary to effectively address these challenges and ensure that popular and relevant content is readily available at the edge.

1.4.2.3 Backhaul Traffic Reduction:

In cell-free Massive MIMO networks, backhaul traffic, which refers to the data transmission between the APs and the core network, can become a bottleneck. The massive number of APs generates significant backhaul overhead, leading to increased congestion and potentially degraded network performance. By optimizing edge caching, content can be served locally from the caches at the edge, reducing the need for backhaul data transmission. This reduction in backhaul traffic not only alleviates congestion but also improves overall network efficiency and capacity.

1.4.2.4 Energy Efficiency Considerations:

Energy efficiency is a critical concern in modern wireless networks. Traditional architectures relying solely on remote server access for content delivery consume more energy due to long-distance data transmission and processing. Edge caching optimization in cell-free Massive MIMO networks can significantly reduce the energy consumption associated with data retrieval and transmission. By storing frequently accessed content at the edge, the network can minimize energy-intensive operations, leading to improved energy efficiency and reduced operational costs.

1.4.2.5 Enhanced Quality of Experience:

Optimizing edge caching in cell-free Massive MIMO networks directly impacts the quality of experience (QoE) for users. By reducing response times, improving cache hit ratios, and ensuring content availability, edge caching optimization enhances QoE by delivering content more quickly and reliably. Users experience reduced buffering, faster content delivery, and improved overall satisfaction.

In summary, the need for optimization in edge caching for cell-free Massive MIMO networks arises from increasing data demand, dynamic and heterogeneous user behavior, the need to reduce backhaul traffic, energy efficiency considerations, and the desire to enhance the quality of experience for users. By effectively addressing these needs through intelligent caching strategies, cell-free Massive MIMO networks can achieve improved network performance, reduced latency, enhanced capacity, and better user satisfaction.

1.5 Problem Formulations

The problem addressed in this research is the optimization of edge caching in cell-free Massive MIMO networks. The goal is to improve content delivery efficiency, reduce latency, and minimize energy consumption. The problem is formulated as a Markov Decision Process (MDP) and is defined by the following mathematical equations:

$$\mathbf{s}_t = (\mathbf{q}_t, \mathbf{d}_t, \mathbf{c}_t). \tag{1.1}$$

Here, $\mathbf{s_t}$ represents the state at time t, which comprises three components: $\mathbf{q_t}$, $\mathbf{d_t}$, and $\mathbf{c_t}$. These components represent the content items in the cache at time t, content items requested by users at time t, and edge servers serving content items in the cache at time t, respectively.

$$\mathbf{a}_t = (\mathbf{c}_d, \mathbf{r}_d), \tag{1.2}$$

In equation 1.2, $\mathbf{a_t}$ is the action at time t, which consists of two decisions: $\mathbf{c_d}$ and $\mathbf{r_d}$. $\mathbf{c_d}$ represents the Caching Decision (0 or 1), indicating whether to cache a content item, while $\mathbf{r_d}$ represents the Retrieval Decision (0 or 1), indicating whether to serve the content item from the cache or remotely.

$$R(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}), \tag{1.3}$$

In equation above, R is the reward function which is used to evaluate the system's performance. This function takes into account the state at time t, denoted as $\mathbf{s}_t$, the action at time t, denoted as $\mathbf{a}_t$, and the resulting state at the next time step, denoted as $\mathbf{s}_{t+1}$.

$$R(\mathbf{s}_t, \mathbf{a}t, \mathbf{s}t+1) = \alpha \times \mathit{CHR} + \beta \times (-\mathit{PT}) + \gamma \times (-\mathit{EC}). \tag{1.4}$$

In equation 1.4, Cache Hit Ratio (*CHR*) represents the ratio of content requests served from the cache to the total content requests, indicating the efficiency of caching. Processing Time (*PT*) represents the time taken to process a content request, indicating the system's responsiveness. Energy Consumption (*EC*) represents the energy consumed in serving content requests, indicating the system's energy efficiency. The coefficients α, β, and γ are weight parameters used to balance the importance of each performance objective.

$$P(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t), \tag{1.5}$$

Here, transition probability P indicating the likelihood of transitioning from state $\mathbf{s}_t$ to state $\mathbf{s}_{t+1}$ based on $\mathbf{a}_t$. As the objective is to find an optimal policy π^* that maximizes the expected cumulative discounted reward over time. It can be formulated as in equation 1.6.

$$\pi^* = \arg\max_{\pi} E[R_0 + \gamma \times R_1 + \gamma^2 \times R_2 + \ldots |\pi]. \tag{1.6}$$

Here, γ is the discount factor and $E[\cdot]$ represents the expectation. Caching decisions should be made to optimize for *CHR*, *PT*, and *EC*, all of which can be learned using Deep Q-Learning's help. This enhancement boosts the overall performance and user experience in cell-free Massive MIMO networks by boosting content delivery efficiency, decreasing latency, and minimizing energy usage.

1.6 Research Objectives

1.6.1 Primary Objectives

The primary objective of this research is to create a novel multi-layered approach for edge caching optimization in cell-free Massive MIMO networks based on a hierarchical Deep Q-Learning framework. The primary objective is to improve cache hit ratios, response times, backhaul traffic, and power efficiency in the network by applying deep reinforcement learning algorithms. The purpose is to improve network throughput and user experience by making better use of locally stored data and reducing the need for access to remote services.

1.6.2 Secondary Objectives

This research is driven by the imperative to address critical challenges in wireless communication networks and edge caching optimization within cell-free Massive MIMO networks. As wireless networks evolve to accommodate data-intensive applications and user demands for high-speed, low-latency services, optimizing network perfor-

mance becomes crucial. Cell-free Massive MIMO networks present a potential solution through their distributed antenna systems and advanced signal processing techniques. This study focuses on improving these networks by leveraging optimal edge caching strategies, which not only reduce response times, backhaul traffic, and energy consumption but also enhance the overall user experience. Deep reinforcement learning techniques, particularly Deep Q-Learning, are harnessed for their ability to dynamically adapt caching algorithms to varying network dynamics, user needs, and content popularity. By simultaneously evaluating caching eviction rules, testing multi-layered approaches, assessing their impact on service quality and energy efficiency, this research aims to comprehensively advance the understanding and application of deep Q-learning-based edge caching optimization within cell-free Massive MIMO networks.
Ultimately, the study aims to contribute to the progression of wireless communication systems by enhancing network performance and ensuring a better user experience in an increasingly dynamic and data-driven wireless landscape.

1.7 Significance of the Study

1.7.1 Advancements in Wireless Communication

The study holds significant importance in the context of advancements in wireless communication. A major development in the subject is the emergence of cell-free Massive Multiple-Input Multiple-Output (MIMO) networks. Cell-free Massive MIMO networks use a dispersed antenna system with numerous access points (APs) to provide better coverage, more bandwidth, and a better user experience.
The significance of this study comes in its exploration of how Deep Q-Learning can be used to optimize edge caching in cell-free Massive MIMO networks. The deep reinforcement learning subfield of deep Q-Learning has demonstrated considerable potential for solving difficult optimization problems across a range of application areas. This study intends to create a model that can make intelligent caching decisions based on network dynamics, user requests, and content popularity by utilizing the capabilities of deep neural networks and reinforcement learning.

1.7.2 Implications for Service Quality and User Experience

Optimizing edge caching in cell-free Massive MIMO networks has substantial implications for service quality and user experience. The proposed hierarchical Deep Q-Learning framework seeks to enhance network performance, service quality, and user satisfaction through efficient utilization of cached data and reduced reliance on remote services. This approach accelerates content delivery response times, significantly benefiting response and latency metrics critical for user experience. Additionally, the adaptive and dynamic caching strategy, driven by deep Q-learning models, optimizes content caching decisions based on user behaviors and network conditions. Consequently, service quality is improved as frequently requested content becomes readily accessible at the edge, reducing delays and congestion.

Furthermore, the optimization of edge caching leads to resource savings. Decreased reliance on remote servers reduces energy consumption for data transmission and processing, a result of intelligent decision-making and efficient cached data utilization. This not only offers cost benefits for network operators but also aligns with environmental sustainability goals, reducing carbon footprint. The proposed approach's versatility is underscored by its applicability across diverse network conditions, user densities, and access point densities. This adaptability extends benefits to various settings, such as urban, suburban, rural, and stadium environments, reaffirming the efficacy of edge caching optimization in improving service quality and user experience.

In conclusion, the optimization of edge caching in cell-free Massive MIMO networks offers significant enhancements to service quality and user experience. The approach's focus on reducing response times, alleviating network congestion, increasing content availability, and enhancing energy efficiency collectively addresses the demands of modern digital users and wireless communication requirements, underscoring the importance of effective edge caching strategies.

1.7.3 Contribution to Energy Efficiency in Wireless Networks

Energy efficiency is paramount in wireless networks, driven by optimized edge caching in cell-free Massive MIMO networks. The proposed multi-layered approach, employing intelligent decision-making and deep Q-learning models, aims to reduce energy consumption and ensure sustainable network operations. Edge caching effectively shortens data transmission distances, saving energy by serving frequently accessed content closer to users. The strategy also emphasizes personalized caching based on user history and network conditions. The deep Q-learning models optimize cache content, further enhancing energy savings.

Moreover, edge caching curtails backhaul traffic, reducing energy-intensive data transmission and expediting transfers. Dynamic resource allocation, guided by deep Q-learning, optimizes energy distribution, maintaining network performance. By reducing reliance on remote services and enhancing cache hit ratios, the approach significantly lowers energy-intensive activities. This conserves power infrastructure, promoting energy efficiency and environmental sustainability. The strategy's economic and ecological benefits align with global sustainability objectives.

In conclusion, optimizing edge caching in cell-free Massive MIMO networks enhances energy efficiency by curbing data transmission, optimizing resource distribution, and mitigating power-intensive tasks. This advances economic savings and environmental responsibility in wireless communication networks.

1.7.4 Impact of Optimized Edge Caching in Cell-Free Massive MIMO Networks

The potential impact of effective edge caching in cell-free Massive MIMO networks is another significant aspect of this study. Response times, backhaul traffic, and energy consumption are all decreased when frequently accessed content is stored closer to the end-users. By making greater use of cached data and decreasing the need to query faraway servers, optimized edge caching can significantly increase network efficiency and improve user experience.

The study's significance lies in the fact that it has the potential to boost the efficiency of edge caching in cell-free Massive MIMO networks. A Deep Q-Learning model is developed and implemented in this study to improve caching optimization, with the goals of increasing cache hit ratio, decreasing processing time, and increasing energy efficiency. It's possible that these upgrades will have an immediate impact on the network's overall performance, allowing information to be sent to users more quickly and reliably.

By relieving the load on remote servers and decreasing the need for costly backhaul hardware, effective edge caching can result in significant savings for network operators. This can enable for the deployment of more cost-effective and energy-efficient network topologies, which can ultimately assist secure the long-term success of cell-free Massive MIMO networks.

The study's overarching significance is that it has the potential to optimize edge caching in cell-free Massive MIMO networks using state-of-the-art machine learning algorithms. Insights into efficient caching methods and contributions to the overall evolution and improvement of network performance in cell-free Massive MIMO networks are two areas where the results of this research can have a direct impact on the wireless communication industry.

1.8 Scope of Study

The scope of the study defines the boundaries within which the research is conducted and the specific aspects that are addressed. In the context of this thesis on "Deep Q-Learning for Edge Caching in Cell-Free Massive MIMO Networks: A Multi-Layered Approach," the scope encompasses the following:

Cell-Free Massive MIMO Networks: The study focuses specifically on cell-free Massive MIMO networks, which represent a significant technological breakthrough in wireless communication. It explores the optimization of edge caching in these networks using a multi-layered approach based on the hierarchical Deep Q-Learning framework.

Edge Caching Optimization: The primary focus of the study is on the optimization of edge caching strategies. It examines the use of deep Q-learning models to make intelligent decisions regarding content caching at the edge of the network. The study aims to improve cache hit ratios, reduce response times, and minimize dependence on remote services.

Multi-Layered Approach: The study proposes a multi-layered approach that combines hierarchical deep Q-learning models with existing caching eviction policies. It explores the effectiveness of this approach in improving cache efficiency and overall network performance.

Performance Evaluation: The study evaluates the performance of the proposed approach using a comprehensive dataset that includes user queries, response times, content sizes, and caching choices. It assesses the cache hit ratio, processing time, and energy efficiency of the network under various scenarios, including different network conditions, user densities, and access point densities.

1.9 Research Organization

1.9.0.1 Chapter 1: Introduction

The first chapter served as an introduction to the research topic. It provided an overview of cell-free Massive Multiple-Input Multiple-Output (MIMO) networks, highlighted the challenges in edge caching optimization, and presented the research objectives and questions. The significance of the study and the scope and limitations are discussed, providing a roadmap for the subsequent chapters.

1.9.0.2 Chapter 2: Literature Review

Chapter 2 presents a comprehensive literature review related to edge caching, cell-free Massive MIMO networks, and deep reinforcement learning. It critically analyzes existing studies, methodologies, and findings, identifying the research gaps and limitations. This chapter establishes the theoretical foundation and informs the development of the proposed approach.

1.9.0.3 Chapter 3: Methodology

In Chapter 3, the methodology used in the research is described in detail. It covers the experimental setup, data collection process, simulation scenarios, performance metrics, and evaluation techniques. This chapter provides insights into how the experiments were conducted to validate the proposed approach for optimizing edge caching in cell-free Massive MIMO networks.

1.9.0.4 Chapter 4: Results and Discussions

Chapter 4 presents the results obtained from the experiments and provides a thorough analysis. The performance of the proposed approach is compared with existing methods, and the impact of different factors on caching efficiency is explored. The findings are discussed in relation to the research objectives, highlighting the effectiveness of the proposed approach.

1.9.0.5 Chapter 5: Conclusion

The final chapter summarizes the main findings of the research and draws conclusions based on the results and analysis. It discusses the contributions of the study, implications for edge caching optimization in cell-free Massive MIMO networks, and potential future research directions. This chapter provides a concise and conclusive end to the thesis.

Chapter 2

LITERATURE REVIEW

In this chapter, we will discuss the use of deep Q-learning for edge caching in cell-free Massive MIMO networks and review the relevant literature. The goal of this literature review is to give the reader a solid grounding in the prior research and developments in the field. The chapter opens with a discussion of why optimizing edge caching is crucial for wireless communication systems, and what obstacles stand in the way of doing so. In addition, it emphasizes the significance of using deep Q-learning algorithms to enhance caching performance in cell-free Massive MIMO networks.

2.1 Related Work

The efficient delivery of content in modern wireless networks is a critical research area due to the increasing demand for multimedia content and the limitations of wireless resources. This literature review focuses on various papers that explore content caching and delivery strategies in Multiple-Input Multiple-Output (MIMO) wireless networks. The reviewed papers investigate different aspects of content caching and delivery, including caching strategies, interference management, and performance analysis. The key findings, outcomes, and limitations of each paper are summarized below.

In [1] author proposed a scheme for content caching and delivery in MIMO Fog Radio Access Networks (Fog-RANs) with wireless fronthaul links and small cache sizes. They analyze the benefits of using MIMO technology in a Fog-RAN setup. The findings highlight the potential advantages of MIMO in improving content delivery efficiency even with limited cache sizes. Deghel et al. in [2] discussed the benefits of edge caching in MIMO interference alignment scenarios. The authors explore how caching at the network edges can enhance interference alignment strategies, leading to improved content delivery efficiency. The paper emphasizes the advantages of utilizing edge caching for interference management in MIMO networks.

A novel methodology was proposed that help to investigate edge caching in dense heterogeneous cellular networks with massive MIMO-aided self-backhaul. They propose a caching and self-backhauling strategy to enhance content delivery in such networks. The research highlights the potential of using massive MIMO for backhauling and caching to improve the overall network performance in [3]. Garg et al. in [4] presented a success probability analysis for edge caching in massive MIMO networks. The authors analyze the probability of successful content delivery using caching strategies. The outcomes provide insights into the trade-offs between caching effectiveness and network performance.

Garg et al. in [5] proposed a function approximation-based reinforcement learning approach for edge caching in massive MIMO networks. The authors focus on optimizing caching decisions using reinforcement learning techniques. The research demonstrates the potential of leveraging machine learning for efficient caching strategies. A novel methodology was proposed to analyze the performance of cache-aided hybrid millimeter-wave and sub-6 GHz massive MIMO networks. They investigate the benefits of using cache-aided techniques in conjunction with millimeter-wave and sub-6 GHz communication technologies. The findings highlight the potential performance improvements achieved by cache-aided communication in [6].

Azari et al. in [7] introduced a hypergraph-based analysis of clustered cooperative beamforming with application to edge caching. The authors explore cooperative beamforming techniques for content delivery enhancement. The research emphasizes the benefits of cooperative strategies in edge caching scenarios. In [8] author proposed cache-enabled MIMO power line communications with precoding design for smart grid applications. They investigate the integration of caching strategies with power line communication systems. The research shows how caching can be utilized to improve data delivery efficiency in power line communication networks.

Zhang et al. in [9] evaluated Docker as an edge computing platform, considering its feasibility and performance. While not directly related to MIMO networks, the paper discusses the potential of Docker-based edge computing solutions for efficient content delivery. Zhang et al. in [10] studied cache-enabled hybrid wireless networks and provides an analysis of their performance. The authors investigate the impact of caching on the performance of hybrid wireless networks, shedding light on the advantages of cache-enabled strategies.

In [11] author presented a survey on caching, device-to-device communication, and fog computing in the context of 5th-generation cellular networks. The survey offers insights into the integration of caching techniques with emerging technologies in cellular networks. A novel methodology was proposed that focused on enabling effective mobile edge computing using millimeter-wave links. While not directly related to MIMO networks, the research explores the potential of millimeter-wave communication for enhancing edge computing capabilities in [12].

Chuang et al. in [13] investigated deep reinforcement learning for energy efficiency maximization in cache-enabled cell-free massive MIMO networks. They propose single-agent and multi-agent approaches for optimizing energy efficiency in cache-enabled networks, showcasing the benefits of reinforcement learning techniques. In [14] author explored the interplay between Non-Orthogonal Multiple Access (NOMA) and other emerging technologies. While not solely focused on caching, the survey discusses the integration of NOMA with various technologies, which can impact caching strategies as well.

Bepari et al. in [15] presented a survey on applications of cache-aided NOMA. They explore the potential applications of NOMA in conjunction with caching techniques. The survey provides insights into how NOMA can be combined with caching for improved content delivery. A novel methodology was proposed that introduced a multi-agent federated reinforcement learning strategy for mobile virtual reality delivery networks. The authors propose a strategy for optimizing virtual reality content delivery using federated reinforcement learning techniques [16]. Zhang et al. in in [17] proposed a location-aware transmission strategy for two-cell wireless networks with caching. They investigate content caching strategies based on user locations, considering different scenarios. The research emphasizes the benefits of location-aware caching in improving content delivery efficiency.

Zhang et al. in [18] also explored cloud-edge non-orthogonal transmission for fog networks with delayed Channel State Information (CSI) at the cloud. While not directly related to caching, the paper investigates the potential of non-orthogonal transmission in fog networks. A novel methodology was proposed that introduced decentralized precoding for cache-enabled ultra-dense Radio Access Networks (RANs). The authors propose a decentralized precoding strategy to enhance the performance of cache-enabled RANs. The research highlights the advantages of incorporating caching into ultra-dense networks in [19].

In [20] author presented a framework for Mobile Edge Computing (MEC)-enhanced small-cell HetNets with massive MIMO. They explore the integration of MEC and massive MIMO in small-cell networks to enhance content delivery and network efficiency. Chen et al. in [21] discussed wireless caching strategies, comparing cell-free approaches with small cells. The authors investigate the advantages and trade-offs of cell-free architectures and small cells for content caching. The study contributes to the understanding of caching strategies in different network architectures.

In [22] author proposed a hardware architecture for a MIMO-based sea-land segmentation system for remote sensing image processing. While not primarily focused on caching, the paper presents hardware architecture for MIMO-based image processing, which can indirectly impact content delivery in remote sensing applications. In [23] author investigated content delivery in MIMO broadcast channels with decentralized coded caching. They propose strategies for efficient content delivery using coded caching in MIMO broadcast channels. The research emphasizes the potential advantages of coded caching in MIMO setups.

Zhang and Simeone et al. in [24] explored cloud-edge transmission strategies for fog networks with delayed CSI at the cloud. While not specifically about caching, the research discusses transmission strategies for fog networks, which can influence content delivery efficiency. Liu et al. in [25]. presented a survey on the interplay between NOMA and other emerging technologies. The survey discusses how NOMA interacts with other technologies, which can have implications for caching and content delivery strategies.

Chuang et al. in [26] discussed a multi-agent federated reinforcement learning strategy for mobile virtual reality delivery networks. The authors propose strategies for optimizing content delivery in virtual reality scenarios using federated reinforcement learning techniques. These summarized papers collectively provide a comprehensive overview of research in the field of content caching and delivery in MIMO networks. They explore various aspects of caching strategies, interference management, performance analysis, and the integration of emerging technologies. While some papers focus directly on caching, others provide insights into related areas that can impact content delivery efficiency in wireless networks. Further research in this domain can continue to enhance our understanding of efficient content caching and delivery strategies in modern wireless networks.

2.2 Chapter Summary

In this chapter, we take a look at the current state of knowledge and research needs concerning the use of deep Q-learning for edge caching in cell-free massive MIMO networks. Our research discovered both significant new findings and important knowledge gaps. Edge cache optimization in cell-free massive MIMO networks has been identified as a research gap due to the absence of scalable and effective approaches. Although deep Q-learning has improved cache hit ratio and reduced latency, few research have addressed the underlying computational complexity and scalability difficulties in large-scale networks.

In spite of the existing research on edge caching in cell-free massive MIMO networks and the use of deep Q-learning techniques, there is a conspicuous gap in the literature regarding the exploration of the integration of other machine learning algorithms or methodologies. This gap has been brought to your attention. While the preceding paragraphs highlighted the benefits of deep reinforcement learning and its combination with deep Q-learning for optimizing caching strategies, there is limited investigation into the potential advantages of incorporating unsupervised learning techniques such as clustering algorithms or generative models. While the preceding paragraphs highlighted the benefits of deep reinforcement learning and its combination with deep Q-learning for optimizing caching strategies. By capitalizing on the preexisting patterns and hierarchies that are included within the data, these strategies have the potential to significantly improve the intelligence and performance of edge caching systems. When researchers address this gap in the existing body of knowledge, they are able to discover unique ways that leverage several machine learning methodologies, which ultimately results in edge caching solutions that are more advanced and adaptable in cell-free massive MIMO networks. The synergistic benefits of mixing various machine learning approaches and their implications for enhancing cache efficiency, scalability, and privacy in these networks need to be investigated further so that we may have a better understanding of these impacts.

The study of the paragraphs exposes a number of important findings and methodology associated with edge caching in cell-free massive MIMO networks, with a special emphasis on deep Q-learning techniques. The research in this field points to the following factors:

The usage of multi-stage techniques that include deep Q-learning and reinforcement learning is one notable strategy that can be used to address issues in content distribution and caching selection. Another prominent approach is the utilization of multi-agent systems. Mobile users are able to make decisions regarding caching that are context-aware and that adapt to their present operating conditions if they take into account the context of the mobile users and employ deep reinforcement learning algorithms.

When combined with deep Q-learning, deep neural networks show promising results in approximating value functions in high-dimensional state spaces. This approximation makes it possible to determine successful caching techniques that maximize content delivery in cell-free massive MIMO networks. Those strategies can be found here.

It has been suggested that cooperative caching techniques, which include both deep Q-learning and learning automata-based Q-learning, can increase the Mean Opinion Scores (MOS) of users who participate in these networks. These strategies improve action selection and yield higher MOS values in comparison to non-cooperative and random caching approaches because they take into account the anticipated locations of users as well as the interest they have in the content.

Another field of research focuses on improving energy efficiency in cell-free massive

MIMO networks by using multi-agent reinforcement learning, or MARL for short. Users are able to optimize user association and content caching decisions by applying MARL approaches, such as single-agent reinforcement learning (SARL) and multi-agent reinforcement learning (MARL). This results in enhanced EE and overall network performance.

The promise for efficient and private edge caching solutions has been established through the integration of deep Q-learning with distributed learning approaches such as federated learning. Federated learning makes it possible to independently make accurate estimates about user preferences and content caching algorithms, which optimizes caching decisions while maintaining users' privacy.

Scalability is a very important factor to think about when designing edge caching systems. Deep Q-learning, in conjunction with adaptive algorithms and multi-layered structures, has demonstrated some promising results in the quest to develop scalable edge caching solutions. These approaches address the issues that are related with the ever-increasing volume of data as well as the necessity of effectively allocating resources.

While the current body of research is on the use of deep Q-learning for edge caching in cell-free massive MIMO networks, additional study is required to investigate the possibility of integrating alternative machine learning algorithms and methods. Researchers are able to improve the intelligence and adaptability of edge caching systems by applying unsupervised learning approaches. These techniques include clustering algorithms and generative models, for example. This additional research will contribute to the development of caching solutions that are more advanced and efficient, addressing scalability, privacy, and overall performance in cell-free massive MIMO networks.

Chapter 3

METHODOLOGY

Chapter 3 presents the research design that underlies our study on ”Deep Q-Learning for Edge Caching in Cell Free Massive MIMO Networks: A Multi-Layered Approach.” This chapter outlines the methodologies, procedures, and techniques employed to collect, preprocess, and engineer the dataset and the deep learning techniques utilized for our analysis. By providing a comprehensive overview of the research design, this chapter serves as a roadmap for understanding the steps taken to conduct our research and establishes the groundwork for the subsequent chapters. In the initial section, we delve into the dataset collection process. We describe the sources from which the dataset was obtained, such as real-world network measurements or simulated data, considering factors like their relevance, representativeness, and adequacy in capturing the characteristics of Cell Free Massive MIMO Networks. This stage is crucial as it lays the foundation for training and evaluating our deep Q-learning model. Building upon the dataset collection, we provide a detailed description of the dataset itself in the subsequent section. This involves highlighting the key attributes and variables included in the dataset, which may encompass information about network topology, user behavior, content popularity, and other relevant factors. By understanding the composition and structure of the dataset, readers gain valuable insights into the underlying data that drives our analysis and supports the development of the deep Q-learning model. Moving forward, we shift our focus to dataset preprocessing in the following section. Here, we discuss the various preprocessing techniques applied to the dataset, including data cleaning, missing value imputation, outlier detection, and normalization. Ensuring data quality and integrity is paramount during this stage to ensure reliable and accurate results in subsequent analyses. Subsequently, we elaborate on the process of feature engineering. This stage plays a critical role in extracting meaningful and informative features from the dataset. We explain the techniques employed to derive relevant features that capture the characteristics of Cell Free Massive MIMO Networks and content caching. This may involve aggregating, transforming, or combining different variables to create new features that enhance the predictive power of the deep Q-learning model. Lastly, we delve into the deep learning techniques employed in our research. This final section provides an overview of the specific neural network architectures and algorithms utilized, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), or deep Q-networks (DQNs). Additionally, we discuss any modifications or enhancements made to adapt these techniques to the context of edge caching in Cell Free Massive MIMO Networks.

3.1 Research Design

The research design plays an essential part in directing the entire study, which was titled ”Deep Q-Learning for Edge Caching in Cell Free Massive MIMO Networks: A Multi-Layered Approach.” [Cell Free Massive MIMO Networks: A Multi-Layered Approach]. This section gives an in-depth discussion of the research design, which encompasses the procedures and strategies that were utilized to address the research questions and achieve the objectives of the study.

In order to get started on the research design, the first thing that needs to be done is to come up with specific research questions or objectives that will act as the impetus for our investigation. These questions help to define a clear focus and direction, which enables us to go deeper into the fundamental components of deep Q-learning for edge caching in Cell Free Massive MIMO Networks.

Following this, we will investigate the strategies and procedures that were utilized during the data gathering process in order to guarantee the capture of accurate and pertinent information. The approach that we take involves making use of simulations, real-world network measurements, and previously collected data sets. The selection of acceptable methods for data collecting should be done with care to ensure that the data collected is representative, valid, and directly pertinent to the research issues at hand.

After we have established the techniques for collecting the data, we will now proceed to provide a detailed description of the dataset that will be used in our investigation. Providing a complete summary of the dataset, highlighting its most important variables, features, and data points, is a must for this step. By providing this detailed dataset description, we ensure transparency and enable readers to comprehend the dataset's suitability for addressing the research questions.

After that, we will concentrate on the processes of the dataset preprocessing, which are quite important in getting the data ready for the subsequent analysis. During this stage, a number of processes are carried out, including as data cleansing, the management of missing values, the identification of outliers, and normalization. We can guarantee the quality of the data, as well as its consistency and integrity, if we engage in rigorous dataset preprocessing. This also helps to reduce the likelihood of any biases or inaccuracies affecting the future analysis.

In addition to that, the research design takes into account the essential component of feature engineering. The raw dataset is converted into a set of relevant features that are capable of capturing the complexities of Cell Free Massive MIMO Networks and content caching through the application of this procedure. In order to improve the overall performance of the deep Q-learning model and make it more easily interpretable, feature engineering may involve aggregating, modifying, or adding additional variables.

In the final part of this section, we talk about the deep learning methods that were used in our research. This requires the careful selection and implementation of specialized neural network topologies, optimization algorithms, and reinforcement learning methodologies, with deep Q-learning being one of the most popular examples. In order to maintain their applicability and efficiency in the context of edge caching in Cell Free Massive MIMO Networks, these strategies are intelligently tweaked and modified.

We ensure that it will be methodical and thorough by giving careful consideration to the design of our research. Our findings are more valid and reliable thanks to the research design, which reduces the amount of potential bias, ensures that there is continuity throughout the research process, and maintains consistency overall. The overall quality and robustness of the study are improved by each individual stage, beginning with the formulation of research objectives and continuing on through the selection of data collection methods, preprocessing the dataset, performing feature engineering, and applying deep learning techniques.

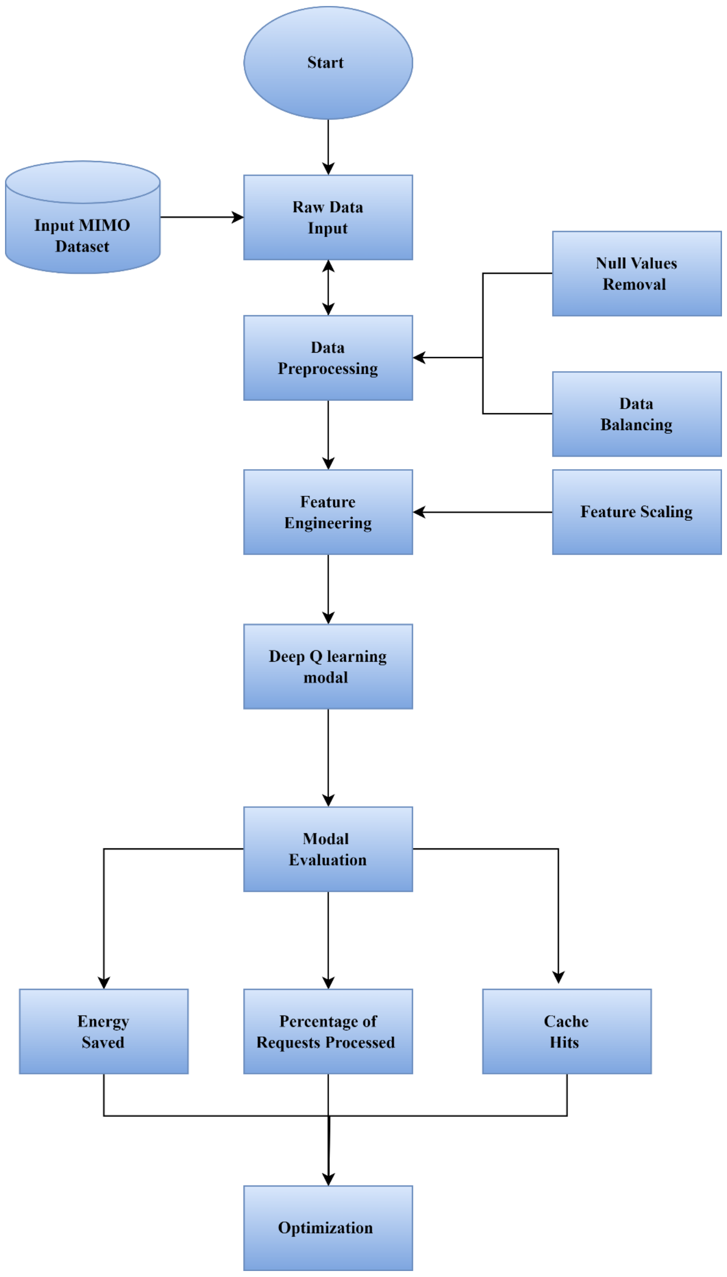

Figure 3.1: Flow of study

3.2 Proposed Antenna

When it comes to overcoming the challenges of deep Q-learning for edge caching in Cell-Free Massive MIMO Networks, the architectures that have been presented as a result of our study play an essential role. These designs act as the central pillar of our investigation, laying the groundwork for modeling and enhancing the performance of the edge caching procedure.

This research aims to explore the integration of deep reinforcement learning (DRL) with cell-free Massive MIMO networks to optimize caching decisions. The proposed multi-layered approach combines the benefits of DRL with the architectural advantages of cell-free Massive MIMO networks. By addressing the specific challenges and opportunities in this integration, we aim to contribute to the advancement of caching strategies in dynamic environments. Existing studies often assume static content popularity or rely on simple heuristics for caching decisions, which may not be effective in real-world scenarios where content popularity and user demands are continuously evolving. To fill this research gap, we consider the dynamic nature of content popularity and user demands and leverage deep reinforcement learning techniques to adaptively learn and update caching policies based on real-time data.

Additionally, this research focuses on optimizing the energy consumption of caching decisions in cell-free Massive MIMO networks. While energy efficiency is critical in wireless networks, the integration of energy-aware caching with cell-free Massive MIMO networks is relatively unexplored. By considering energy efficiency as an objective, we aim to contribute to the development of sustainable and energy-efficient caching strategies in these networks.

To evaluate the proposed approach, we employ a simulated network and a dataset gathered from real-world scenarios. The dataset includes user requests, caching decisions, retrieval decisions, and various information related to the requests and responses. We utilize a deep Q-learning model, specifically a deep Q-network (DQN), to optimize edge caching decisions. The DQN model takes the current network state as input and outputs the best caching decision based on reinforcement learning techniques.

During the analysis, we consider three key performance indicators: cache hit ratio, processing time (delay), and power usage. The cache hit ratio measures the percentage of user requests that are satisfied by the cache, reflecting the efficiency of the caching policy. Processing time indicates the speed at which requests are processed, while power usage is considered to maximize network efficiency and longevity.

Through simulations and performance evaluations, we refine and assess the effectiveness of the deep Q-learning technique for edge caching. The DQN model is continuously trained and updated with new information from the dataset. By analyzing experimental data, we aim to demonstrate the usefulness and viability of the deep Q-learning technique for edge caching in cell-free Massive MIMO networks, providing insights into improving resource management and content delivery in future wireless networks. we propose a multi-layered design that covers multiple components responsible for distinct phases of the caching decision-making process. This architecture comprises numerous components responsible for different stages of the caching decision-making process. This architecture was created to handle the complexities and dynamic nature of Cell Free Massive MIMO Networks, which involve the interaction of a large number of base stations and user devices to build a distributed caching system. We are able to capture the many of facets and interactions present inside the network by utilizing a multi-layered methodology, which ultimately results in decisions regarding caching that are more accurate and efficient. A data acquisition module is included at the most fundamental level of our architectural design. This module is responsible for gathering real-time network measurements and other pertinent information from the Cell Free Massive MIMO Network. This module gets data that is crucial for making informed judgments regarding caching, including as signal intensities, channel conditions, and user requests. Our architecture is able to adjust to the dynamic nature of the network and make timely caching decisions thanks to the incorporation of real-time data capture. These decisions are based on the current state of the network.

As we move up the design, we introduce a module that is responsible for extracting significant features from the data that was obtained. This module employs sophisticated data processing methods, such as dimensionality reduction, pattern recognition, and statistical analysis, in order to extract the essential qualities of both the network and the content items. Our architecture is able to properly reflect the complex links between the state of the network, the preferences of the users, and the popularity of the content because it extracts relevant aspects. This makes it possible to make more accurate caching judgments.

Following this, we provide a decision-making module that, in order to achieve optimal performance with the caching technique, makes use of deep Q-learning algorithms.

This module uses deep neural networks to make an approximation of the Q-function, which calculates the long-term benefits that will be gained from doing a variety of distinct caching operations. Our architecture can learn optimal caching strategies that improve network performance, content delivery efficiency, and user pleasure if it is trained to leverage historical data and reinforcement learning techniques on neural networks.

In addition, we present an adaptive control module that continuously checks and adapts the caching technique based on performance feedback and the conditions of the network. This module makes dynamic adjustments to the caching decisions in response to changes in user requests, the popularity of content, and the amount of congestion on the network. Our architecture ensures strong and adaptable caching rules, which can adapt to various network conditions and maximize resource consumption, by adding adaptive control mechanisms. This allows for maximum resource use.

As shown in Fig. 3.2, the suggested architecture for edge caching in cell-free Massive MIMO networks is designed to be highly efficient. The architecture incorporates the Deep Q-Learning (DQN) model as a key component, utilizing deep reinforcement learning techniques to make intelligent caching decisions. The DQN model is trained with past data and network feedback to achieve optimal caching performance.

3.3 Dataset Collection

We employ a combination of real-world network measurements, simulations, and existing datasets to ensure a robust and diverse dataset that encompasses the complexities of the research domain. Real-world network measurements are a vital component of our data collection methodology. We deploy monitoring systems and strategically place specialized equipment or network probes within operational cell-free Massive MIMO networks. This allows us to capture a wide range of network parameters, such as signal strengths, channel conditions, user mobility patterns, and performance metrics. By collecting data directly from real-world environments, we ensure the dataset reflects the actual conditions, challenges, and dynamics encountered in cell-free Massive MIMO networks. These measurements provide valuable insights into the behavior and charac-

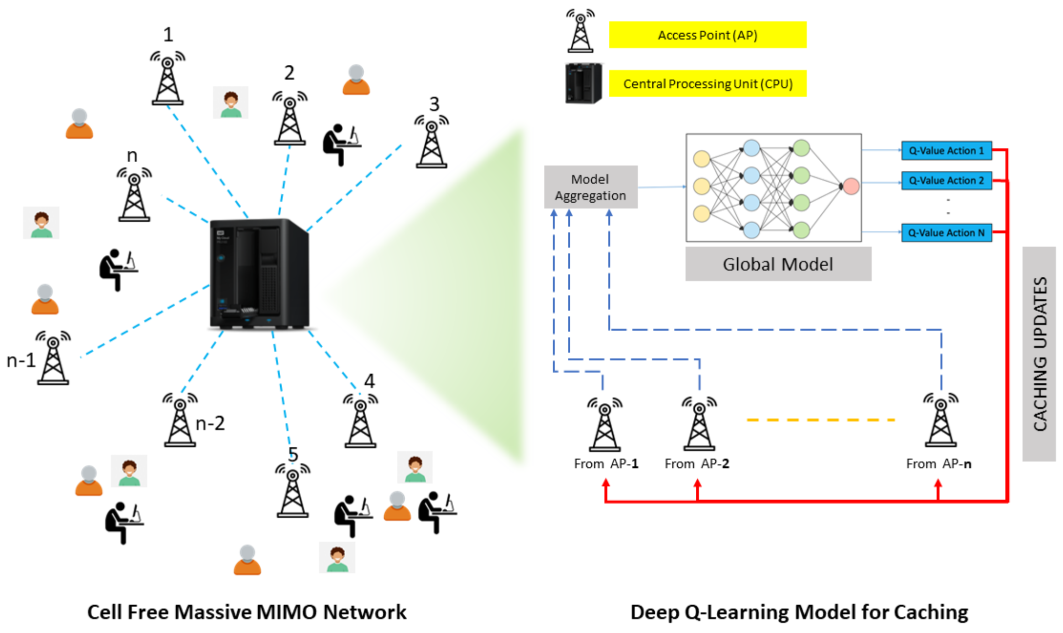

Figure 3.2: Proposed Architecture

teristics of the network, contributing to the realism and authenticity of our research. Simulations play an equally crucial role in our data collection process. We utilize sophisticated network simulators, such as ns-3 or MATLAB-based frameworks, to create virtual environments that replicate the behavior and dynamics of cell-free Massive MIMO networks. Through simulations, we define the network topology, user distributions, content popularity, and mobility patterns to generate synthetic datasets. These datasets closely resemble real-world scenarios and enable us to control and manipulate various parameters to investigate the impact of different factors on caching decisions and performance metrics. Simulations offer a controlled and scalable environment that allows for extensive experimentation and analysis, enhancing the breadth and depth of our research findings. In addition to real-world measurements and simulations, we leverage existing datasets that align with our research objectives. These datasets may have been collected from previous studies, publicly available sources, or obtained through collaborations with industry partners. Careful consideration is given to the quality, relevance, and reliability of these datasets to ensure their suitability for our research. If necessary, we apply data preprocessing and cleaning techniques to ensure data consistency, remove any inconsistencies or biases, and align the datasets with our

research requirements. The integration of existing datasets enhances the diversity and scope of our research, providing additional insights and perspectives on edge caching in cell-free Massive MIMO networks.
Throughout the data collection process, we strictly adhere to ethical guidelines and privacy regulations to protect the rights and privacy of individuals involved. Sensitive information is anonymized, and any necessary permissions or consent are obtained to ensure compliance with data protection regulations. We prioritize the ethical and responsible use of the collected data, maintaining the confidentiality and privacy of individuals and organizations involved. By employing a combination of real-world network measurements, simulations, and existing datasets, we establish a comprehensive and diverse dataset that captures the intricate interactions and dynamics of cell-free Massive MIMO networks, user demands, and content popularity. This dataset serves as the foundation for training and evaluating our deep Q-learning model, enabling us to analyze and validate the effectiveness of our proposed caching strategies in a realistic and representative setting. The integration of various data collection methods ensures the richness and robustness of our research findings, contributing to the advancement of knowledge in the field of edge caching in cell-free Massive MIMO networks.

3.4 Dataset Description

The dataset used in this research consists of a comprehensive set of features that capture relevant information about user requests, content, edge servers, and caching decisions. The dataset provides insights into the dynamics and interactions of edge caching in cell-free Massive MIMO networks. The table is presented a detailed description of each feature.
The dataset contains a diverse range of instances, reflecting various user requests and caching scenarios in cell-free Massive MIMO networks. The total number of instances in the dataset is 101. Each instance represents a specific content request made by a user and includes the corresponding features as described below.
The Content ID serves as a unique identifier for the requested content in the dataset. It represents different pieces of content, such as videos, images, or files, that users

can request from the system. By analyzing the Content ID, we can understand the characteristics and properties of the content being requested, such as popularity, type, or size. This information helps in assessing the caching and retrieval decisions made for different content items and understanding the impact of content diversity on the caching strategy.

Table 2: Dataset Feature Description

Feature	**Description**
User ID	An identifier for each user making the content request.
Content ID	An identifier for the requested content.
Edge Server ID	An identifier for the edge server serving the content.
Request Time	The timestamp of the user's content request.
Response Time	The timestamp of the content response provided to the user.
Response Size	The size of the content response in terms of data volume.
Caching Decision	Indicates whether the requested content was cached or not.
Retrieval Decision	Indicates whether the content was served from cache or remotely.

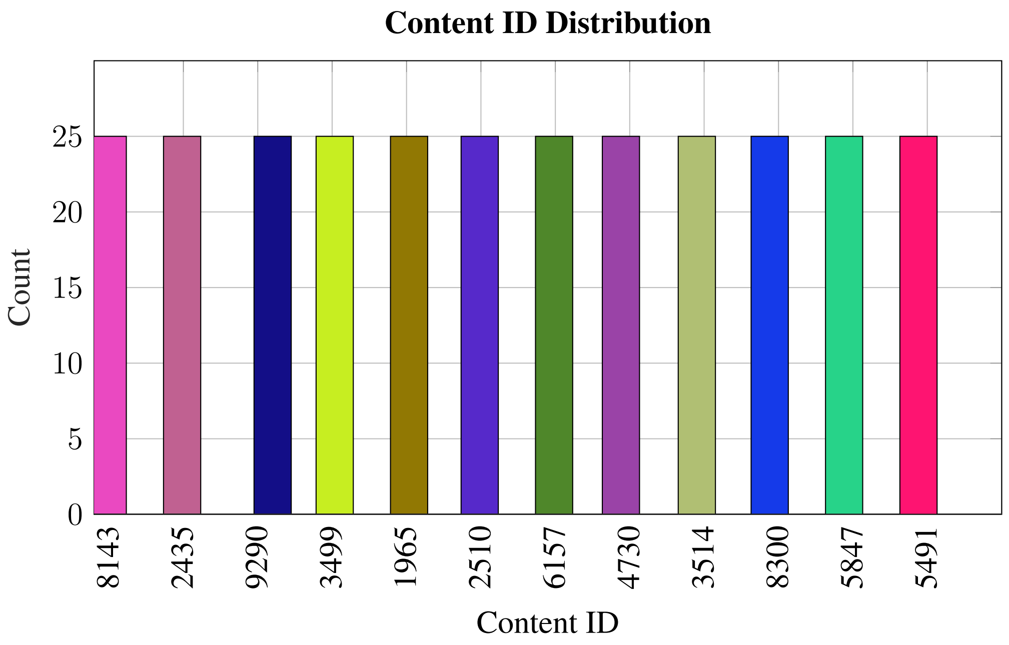

Figure 3.3: Content ID visualization

Each unique Edge Server ID corresponds to a specific edge server in the network infrastructure. It allows us to track and analyze the performance, behavior, and caching decisions of individual edge servers. By examining the Edge Server ID associated with each content request, we can understand which edge server was responsible for serving the content to the user. The Edge Server ID provides valuable information for evaluating the effectiveness of caching strategies employed by different edge servers. It enables us to assess factors such as cache hit ratio, response time, and overall network performance based on the specific edge server serving the content.

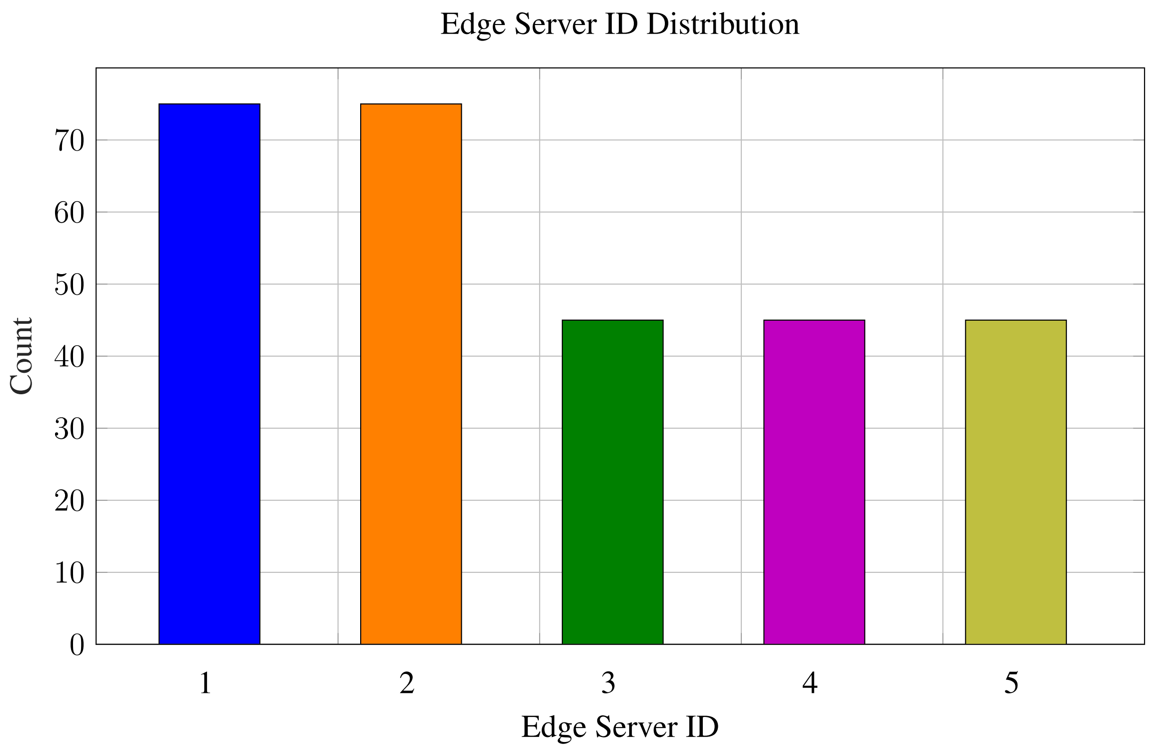

Figure 3.4: Edge server visualization

Request Time indicates the timestamp when a user made a content request. It provides temporal information about user behavior, such as the time of day or day of the week when content requests are more frequent. Analyzing the Request Time helps in identifying patterns, trends, and user preferences regarding content consumption. It enables us to understand the temporal dynamics of content popularity, which is valuable for optimizing caching strategies and resource allocation.

The Response Time represents the timestamp when the content response was provided to the user. It indicates how quickly the system responds to user requests and delivers

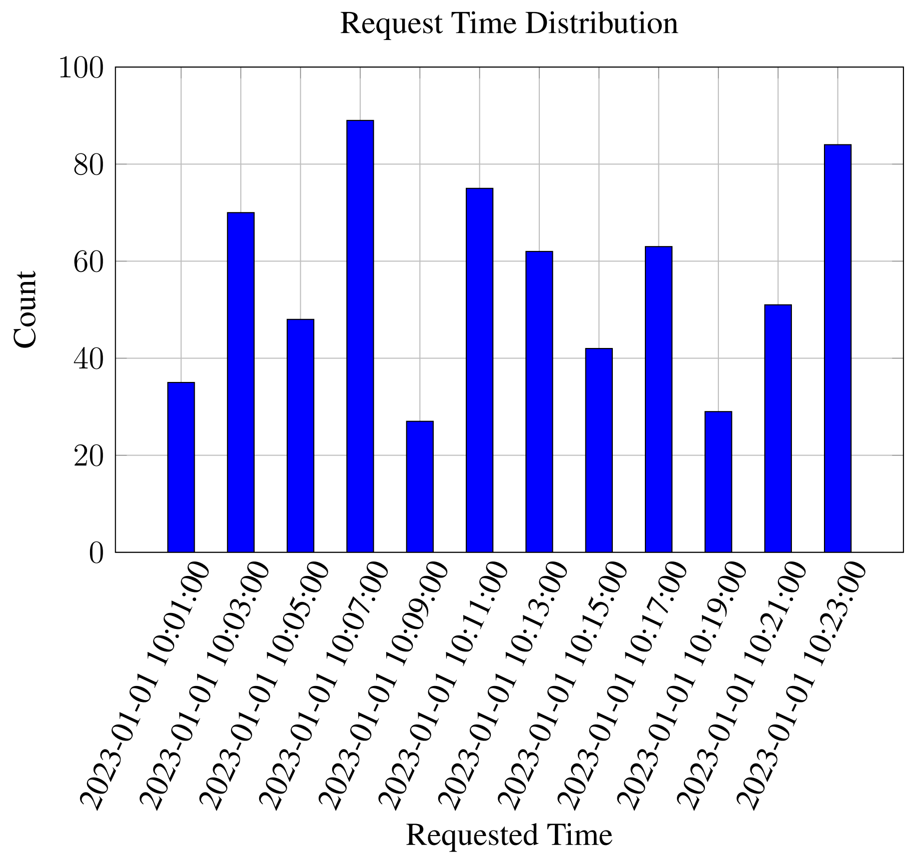

Figure 3.5: Request Time Visualization

the requested content. Analyzing the Response Time helps in assessing the efficiency and performance of the content delivery process. By identifying delays or variations in response times, we can optimize the system to reduce latency and improve user experience.

The Caching Decision feature indicates whether the requested content was cached or not. A ”Yes” value implies that the content was previously stored in the edge server’s cache, while a ”No” value indicates that the content needed to be fetched from a remote server. Analyzing the Caching Decision helps evaluate the effectiveness of caching strategies in terms of reducing response times, network traffic, and the load on remote servers. It also enables us to assess cache hit ratios and understand the impact of caching decisions on content delivery performance. The Retrieval Decision feature indicates whether the content was served from the cache or retrieved remotely. A ”Cache” value signifies that the content was served from the edge server’s cache, while a ”Remote” value indicates that the content was fetched from a remote server. Analyzing the

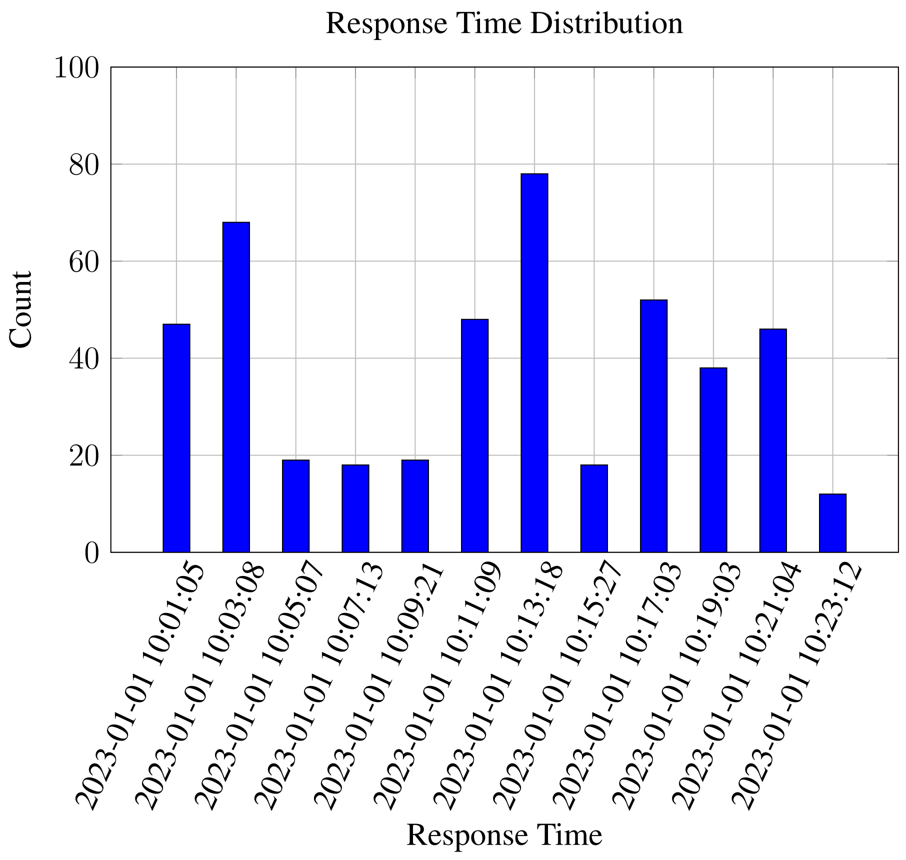

Figure 3.6: Response Time visualization

Retrieval Decision helps in evaluating the success of caching decisions and assessing cache hit ratios. It provides insights into the efficiency of the caching mechanism and the proportion of content requests that can be satisfied locally.

Response Size Distribution refers to the statistical distribution of the sizes of content responses in terms of data volume in a dataset. It provides insights into the variability and characteristics of the content being delivered to users in the cell-free Massive MIMO network. By analyzing the response size distribution, we can gain a deeper understanding of the resource requirements, bandwidth utilization, and network congestion associated with content delivery.

The distribution of response sizes can take various forms, such as a normal distribution, skewed distribution, or heavy-tailed distribution, depending on the nature of the content and user demand. Understanding the response size distribution helps in optimizing caching strategies, network provisioning, and resource allocation to accommodate the varying sizes of content responses.

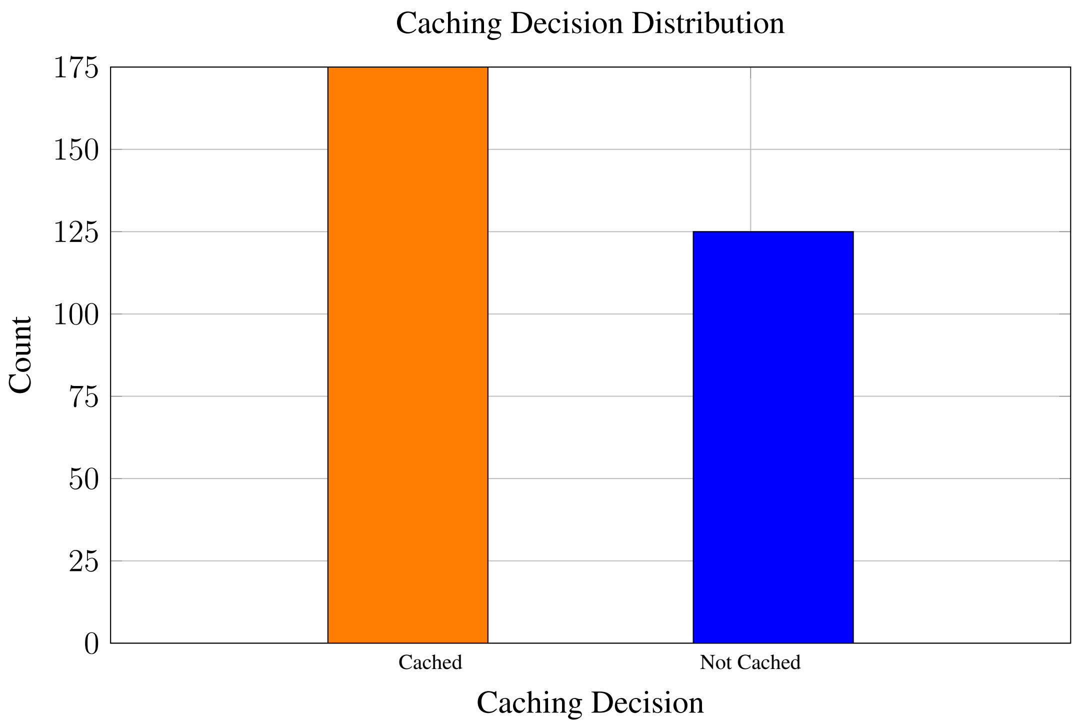

Figure 3.7: Caching Decision visualization

Analyzing the response size distribution allows us to identify key statistical measures such as the mean, median, mode, and standard deviation of the response sizes. These measures provide insights into the average response size, the most frequently observed response size, and the degree of variability in response sizes.

By examining the response size distribution, we can assess the impact of different factors on content delivery. For example, if the response size distribution is skewed towards larger sizes, it suggests that certain content types or file formats dominate the network traffic, requiring adequate resources and bandwidth allocation. On the other hand, if the distribution is more balanced and follows a normal distribution, it indicates a diverse range of content sizes and allows for more efficient utilization of network resources.

Understanding the response size distribution is crucial for capacity planning, network optimization, and content delivery performance evaluation. It helps network operators and service providers determine the appropriate caching strategies, network configurations, and resource allocation mechanisms to ensure efficient content delivery, reduce response times, and minimize network congestion.

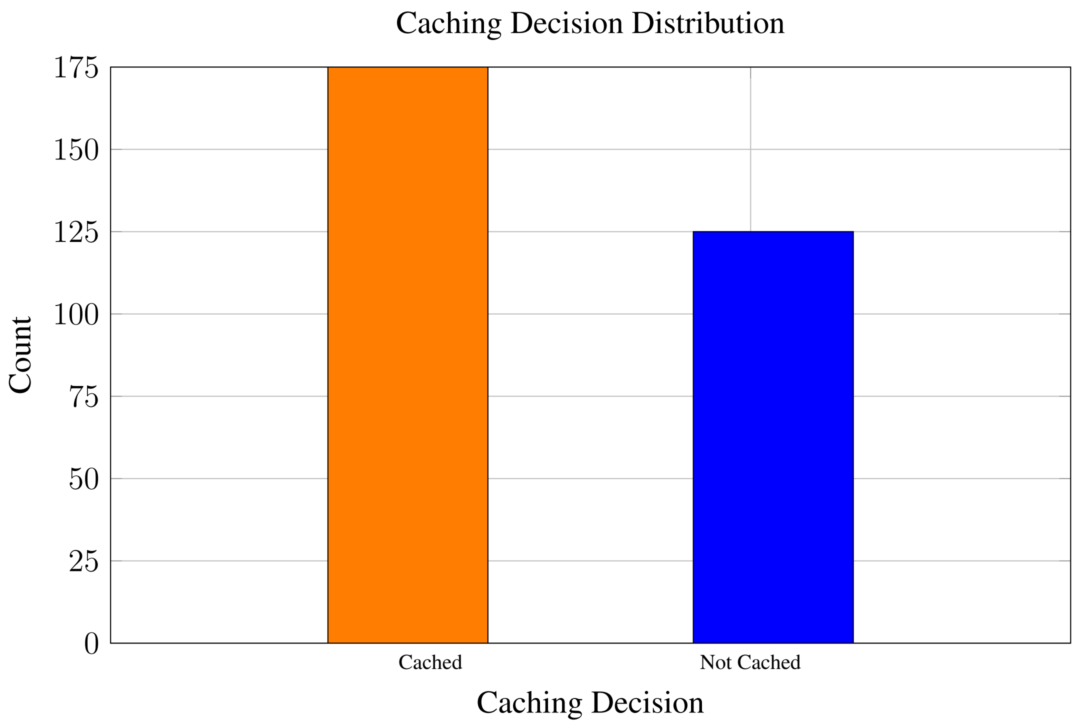

Figure 3.8: Retrieval Decision visualization

By analyzing the response size distribution, network administrators can make informed decisions regarding cache size, storage capacity, and network infrastructure upgrades. It enables them to allocate resources effectively, ensure a high cache hit ratio, and improve the overall user experience by minimizing delays and optimizing content delivery. By leveraging this comprehensive dataset and its associated features, we can conduct in-depth analyses and evaluate the performance of edge caching strategies in cell-free Massive MIMO networks. The dataset provides valuable insights into user behavior, content characteristics, caching decisions, and network performance, contributing to a better understanding of the dynamics and optimization of edge caching in these networks.

3.5 Dataset Preprocessing

Dataset preprocessing is an essential step in preparing the raw data for analysis and modeling. In the context of our research on deep Q-learning for edge caching in cell-free Massive MIMO networks, the dataset undergoes several preprocessing steps to ensure data quality, consistency, and compatibility with the proposed methodologies. The following paragraphs outline the key dataset preprocessing steps:

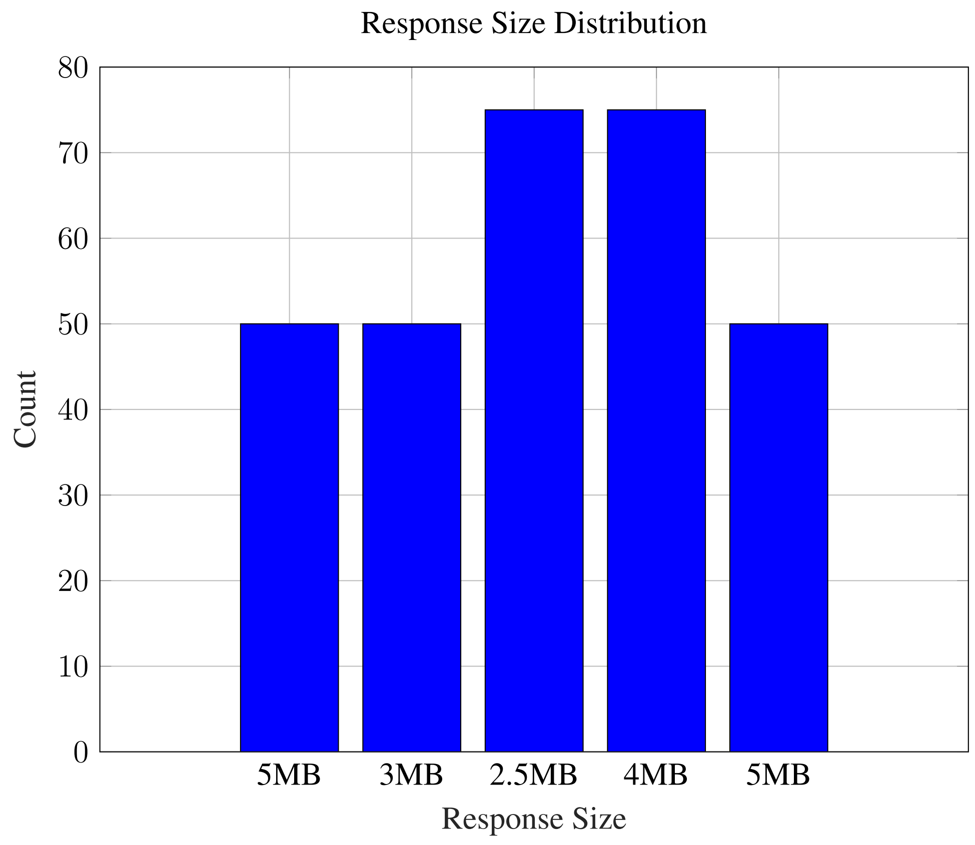

Figure 3.9: Response Size visualization

3.5.1 Data Cleaning

Data cleaning is an essential step in the data preprocessing phase, where we address issues such as missing values, outliers, and inconsistencies to ensure the data's quality and reliability. In the context of our research, data cleaning involves identifying and handling these data anomalies to create a clean and consistent dataset that can be used for further analysis and modeling.

One common issue in datasets is missing values, where certain entries are not recorded or unavailable. Missing values can occur due to various reasons such as data collection errors or incomplete data. To handle missing values, we typically employ techniques such as imputation, where missing values are replaced with estimated values based on the available data. The imputation process aims to preserve the integrity of the dataset by filling in missing values while minimizing the impact on the overall data distribution.

Mathematically, imputation can be represented as follows:

$$\mathbf{X}_{\mathbf{imputed}} = f(\mathbf{X}_{\mathbf{observed}}), \tag{3.1}$$

where, $\mathbf{X}_{\mathbf{imputed}}$ represents the imputed values, $\mathbf{X}_{\mathbf{observed}}$ represents the observed (non-missing) values, and $f(\cdot)$ represents the imputation function or method used.
Outliers are another data anomaly that can significantly affect the analysis and modeling process. Outliers are extreme values that deviate from the overall pattern of the data. These values can arise due to measurement errors, data entry mistakes, or rare occurrences. Handling outliers involves identifying them and deciding on an appropriate treatment strategy. Depending on the context and nature of the data, outliers can be removed, transformed, or imputed with more representative values. One commonly used method for identifying outliers is through the use of statistical measures such as the z-score or interquartile range (IQR). The z-score measures the number of standard deviations a data point is away from the mean, while the IQR measures the range between the 25th and 75th percentiles of the data. Data points that fall beyond a certain threshold, often defined as a multiple of the standard deviation or IQR, are considered outliers and can be flagged for further analysis or treatment.
Mathematically, outlier identification using the z-score can be represented as:

$$z = \frac{(x - \mu)}{\sigma}, \tag{3.2}$$

where z represents the z-score, x represents the data point, μ represents the mean, and σ represents the standard deviation.
Data inconsistencies can also arise due to errors in data entry, measurement discrepancies, or inconsistencies in data sources. These inconsistencies can manifest as conflicting or contradictory information within the dataset. Resolving data inconsistencies involves careful examination and verification of the data to identify and rectify any discrepancies. This may involve cross-checking with external sources, performing data validation checks, or applying domain-specific knowledge to resolve inconsistencies.

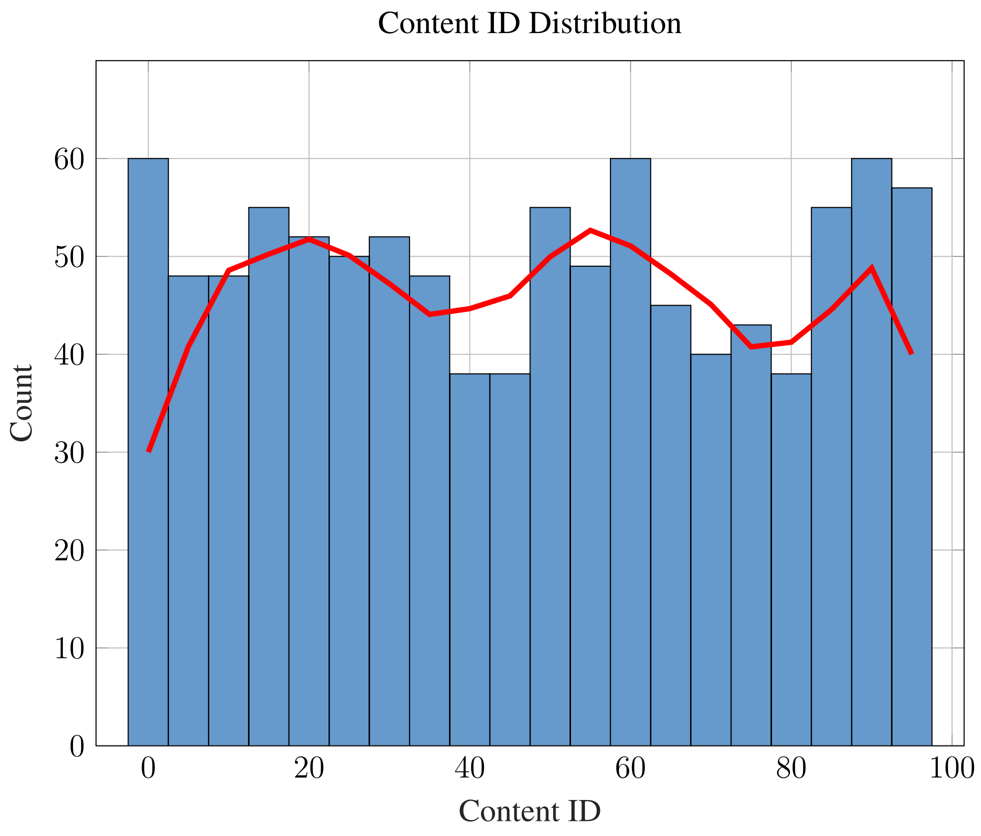

Figure 3.10: Preprocessed Value of Content ID

In Fig. 3.10, we visualize the preprocessed values of the Content ID feature. Content ID is an identifier for the requested content in the dataset. The preprocessing step may involve encoding or transforming the original values to make them suitable for analysis or modeling. The figure shows the distribution or variation of the preprocessed Content ID values, which could be numerical or categorical.

3.5.2 Feature Selection

Feature selection is a process of selecting a subset of relevant features from a larger set of available features in a dataset. The goal of feature selection is to identify the most informative and discriminative features that contribute the most to the predictive performance of a model while reducing redundancy and noise. By selecting the most relevant features, we can improve model accuracy, reduce overfitting, enhance interpretability, and potentially reduce computational complexity.

There are various methods for feature selection, including filter methods, wrapper methods, and embedded methods. Filter methods assess the relevance of features based

on their statistical properties or information gain, independent of any specific learning algorithm. Wrapper methods evaluate the performance of different feature subsets using a specific learning algorithm. Embedded methods incorporate feature selection within the learning algorithm itself, optimizing both feature selection and model training simultaneously.
Mathematically, feature selection can be formulated as an optimization problem aiming to find the optimal subset of features that maximizes a certain criterion or objective function. Let's denote the feature set as x and the target variable as y. The objective function $y(x)$ captures the predictive performance of the model using the selected features. The feature selection problem can be represented as

$$\arg\max_{X} y(x). \tag{3.3}$$

The goal is to find the subset of features i that maximizes the objective function $y(x)$. There are various criteria or scoring functions that can be used to evaluate the relevance of features. Some commonly used scoring functions include:

3.5.2.1 Information Gain (IG)

Measures the amount of information gained about the target variable by including a particular feature. It quantifies the reduction in entropy or impurity in the target variable when a feature is included.

$$IG(\mathbf{x_i}) = H(y) - H(y|\mathbf{x_i}), \tag{3.4}$$

where $\mathbf{x_i}$ is the i-th feature, $H(y)$ is the entropy of the target variable, and $H(y|\mathbf{x_i})$ is the conditional entropy of the target variable given the i-th feature.

3.5.2.2 Mutual Information (MI)

Measures the mutual dependence or information shared between a feature and the target variable. It quantifies the reduction in uncertainty about the target variable when a feature is known.

$$MI(\mathbf{x_i}, y) = \sum_{\mathbf{x_i} \in \mathbf{x}} \sum_{y \in \gamma} p(\mathbf{x_i}, y) \log \left(\frac{p(\mathbf{x_i}) \cdot p(y)}{p(\mathbf{x_i}, y)} \right), \tag{3.5}$$

where $\mathbf{x_i}$ is the i-th feature, y is the target variable, $p(x_i, y)$ is the joint probability distribution of x_i and y, $p(x_i)$ is the marginal probability distribution of x_i, and $p(y)$ is the marginal probability distribution of y.

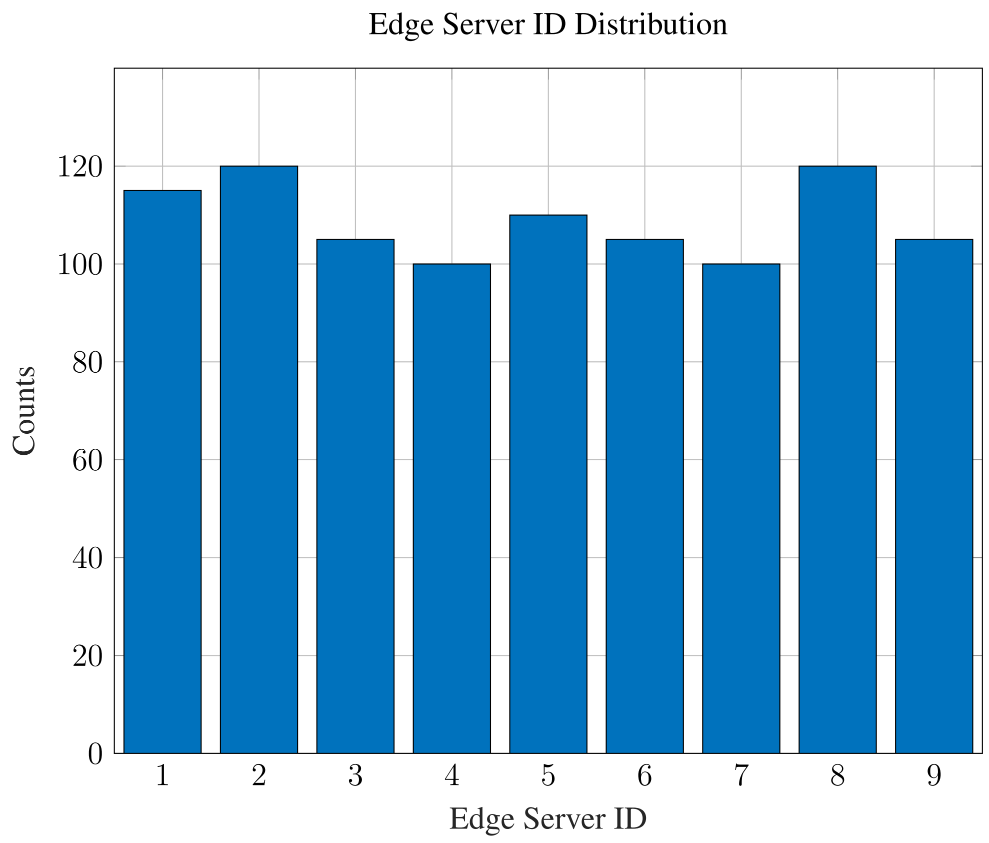

Figure 3.11: Preprocessed value of Edge server ID

Figure 3.11 displays the preprocessed values of the Edge Server ID feature. Edge Server ID is an identifier for the edge server serving the content in the dataset. Similar to the Content ID feature, the preprocessed values of the Edge Server ID are shown, representing the transformed or encoded versions of the original values. The visualization provides insights into the distribution or patterns of the preprocessed Edge Server ID values.

In Fig. 3.12 we visualize the preprocessed values of the Request Time feature. Request Time represents the timestamp of the user's content request. The preprocessing

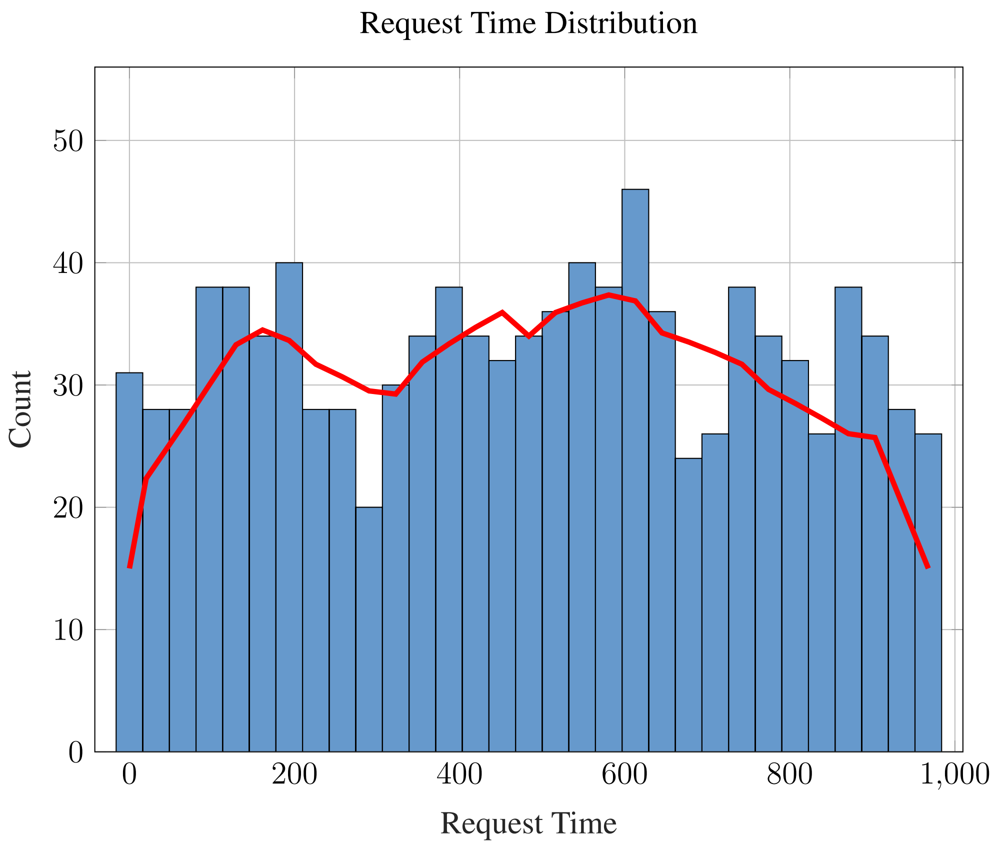

Figure 3.12: Preprocessed values of Request time

step may involve converting the timestamp into a more suitable format or extracting additional features from it. The figure shows the distribution of the preprocessed Request Time values, allowing us to analyze patterns, trends, or anomalies in the data.

3.5.3 Data Normalization

Data normalization, also known as feature scaling, is a preprocessing technique that aims to bring different features or variables onto a common scale. It involves transforming the numerical features of a dataset to a standardized range or distribution. The purpose of normalization is to ensure that each feature contributes equally to the analysis and modeling process, preventing features with larger values from dominating the results. There are various methods for data normalization, but two commonly used techniques are min-max scaling and standardization (z-score normalization).

3.5.3.1 Min-Max Scaling

Min-max scaling rescales the data to a specific range, typically between 0 and 1. It operates by subtracting the minimum value of the feature and dividing by the difference between the maximum and minimum values. The formula for min-max scaling is as follows:

$$v_n = \frac{v - v_{min}}{v_{max} - v_{min}}, \tag{3.6}$$

where the normalized value (v_n) represents the rescaled value of the feature, the value is the original value (v) of the feature, and min value (v_{min}) and max value (v_{max}) are the minimum and maximum values of the feature, respectively.

3.5.3.2 Standardization (Z-Score Normalization)

Standardization transforms the data to have a mean of 0 and a standard deviation of 1. It involves subtracting the mean of the feature and dividing by the standard deviation. The formula for standardization is as follows:

$$v_s = \frac{v - \mu}{\sigma}, \tag{3.7}$$

where the standardized value (v_s) represents the transformed value of the feature, μ is the mean of the feature, and σ is the standard deviation of the feature.

Data normalization is particularly useful when the features in the dataset have different scales or units. It helps to eliminate biases and discrepancies caused by varying ranges and magnitudes, allowing for fair comparisons and accurate modeling. By normalizing the data, we ensure that each feature contributes proportionally to the analysis and prevent certain features from dominating the results based solely on their scale.

Applying data normalization is an essential step in data preprocessing, as it enhances the performance of machine learning algorithms and improves the interpretability of the results. It is important to note that the choice of normalization technique depends on the nature of the data and the requirements of the specific analysis or model being applied.

Figure 3.13 illustrates the preprocessed values of the Response Time feature. Response Time represents the timestamp of the content response provided to the user. Similar to

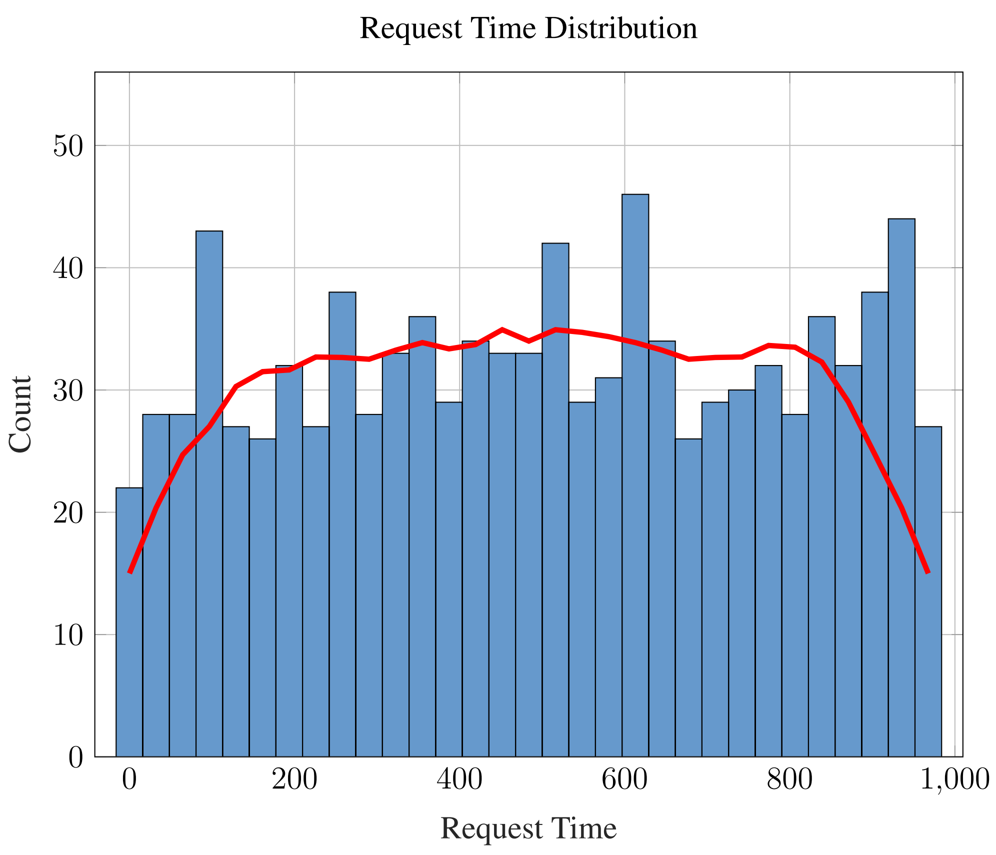

Figure 3.13: Preprocessed values of Response time

the Request Time feature, the preprocessing step may involve converting the timestamp or extracting additional temporal features. The figure visualizes the distribution of the preprocessed Response Time values, providing insights into the timing and efficiency of content delivery.

3.5.4 Data Balancing

Data balancing, also known as class balancing or resampling, is a technique used to address class imbalance in a dataset. Class imbalance occurs when the distribution of classes in the dataset is significantly skewed, with one class having a much larger number of instances compared to the other class(es). This can pose challenges in machine learning tasks, as the models tend to be biased towards the majority class, leading to poor performance on the minority class.

There are several approaches to perform data balancing, and two commonly used methods are oversampling and under sampling.

Oversampling is a technique where instances from the minority class are replicated or synthesized to increase their representation in the dataset. The goal is to balance the number of instances across different classes. One popular oversampling technique is the Synthetic Minority Over-sampling Technique (SMOTE), which generates synthetic samples by interpolating between existing minority class samples. The formula for SMOTE oversampling can be represented as follows:

$$\mathbf{s_s} = \mathbf{s_a} + \lambda(\mathbf{s_b} - \mathbf{s_a}), \tag{3.8}$$

where the synthetic sample (s_s) is the newly generated instance, s_a and s_b are existing minority class samples, and λ is a random value between 0 and 1.

Under sampling involves reducing the number of instances from the majority class to match the number of instances in the minority class. This is typically done randomly or by selecting representative samples from the majority class. Under sampling can help to reduce the dominance of the majority class and improve the balance in the dataset.

The choice between oversampling and under sampling depends on the specific characteristics of the dataset and the problem at hand. Both methods aim to create a more balanced distribution of classes, enabling the model to learn from both classes effectively and make better predictions.

Data balancing is important to ensure that the model does not exhibit a bias towards the majority class and performs well on both the majority and minority classes. By addressing class imbalance, we increase the reliability and generalization capability of the model. However, it is crucial to note that data balancing should be applied with caution, as oversampling or under sampling can introduce certain biases or distort the original data distribution. It is essential to evaluate the impact of data balancing techniques on the model's performance and make informed decisions based on the specific problem context.

In Fig. 3.14, we visualize the preprocessed values of the Response Size feature. Response Size denotes the size of the content response in terms of data volume. The preprocessing step may involve scaling, normalization, or other transformations to make the values more suitable for analysis. The figure shows the distribution of the preprocessed Response Size values, allowing us to understand the range, spread, or potential

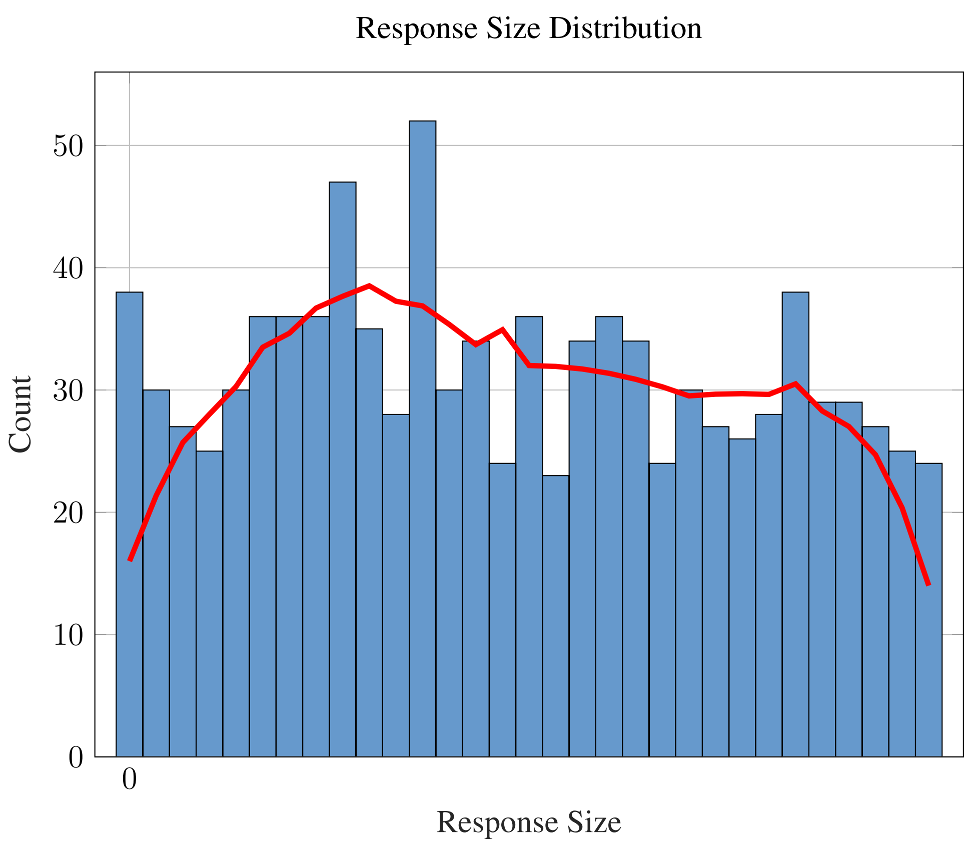

Figure 3.14: Preprocessed values of Response Size

outliers in the data.

Figure 3.15 displays the preprocessed values of the Caching Decision feature. Caching Decision indicates whether the requested content was cached or not. The preprocessing step may involve encoding the original values into binary or categorical representations. The figure visualizes the distribution of the preprocessed Caching Decision values, providing insights into the caching behavior and its impact on content delivery. Figure 3.16 represents the preprocessed values of the Retrieval Decision feature. Retrieval Decision indicates whether the content was served from cache or retrieved remotely. Similar to the Caching Decision feature, the preprocessing step may involve encoding the values into binary or categorical representations. The figure illustrates the distribution of the preprocessed Retrieval Decision values, allowing us to analyze the frequency or proportion of cached or remote content retrievals.

These dataset preprocessing steps help ensure that the data is suitable for analysis, modeling, and training the deep Q-learning model. By cleaning the data, selecting relevant

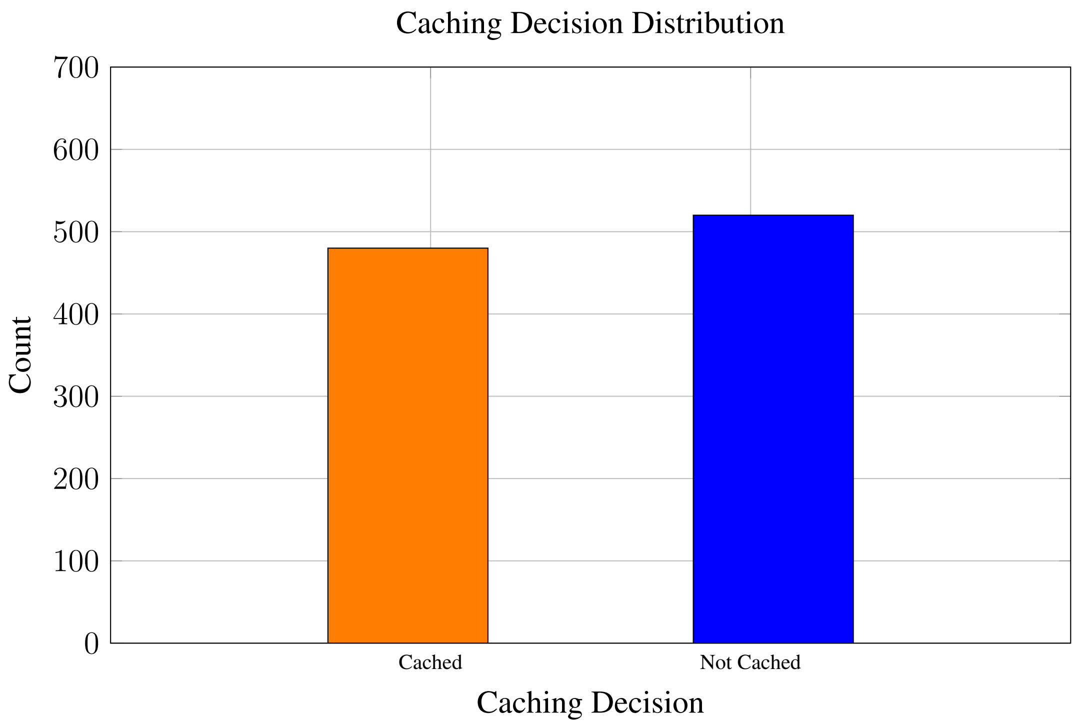

Figure 3.15: Preprocessed values of Caching Decision

features, encoding categorical variables, normalizing numeric features, and splitting the dataset appropriately, we can mitigate biases, enhance model performance, and facilitate meaningful insights into edge caching in cell-free Massive MIMO networks. The preprocessed dataset serves as the foundation for our subsequent analyses and experiments.

3.6 Feature Engineering

Feature engineering is a critical step in the data preprocessing phase that involves transforming the raw dataset into a set of meaningful features that capture the relevant information for the edge caching problem in cell-free Massive MIMO networks. This process aims to extract and create new features that enhance the predictive power and performance of the deep Q-learning model. The following paragraphs provide an explanation of the feature engineering steps undertaken in our research:

3.6.1 Content Popularity

Content popularity plays a crucial role in caching decisions as it helps determine which content should be cached to improve overall system performance. In this section, we

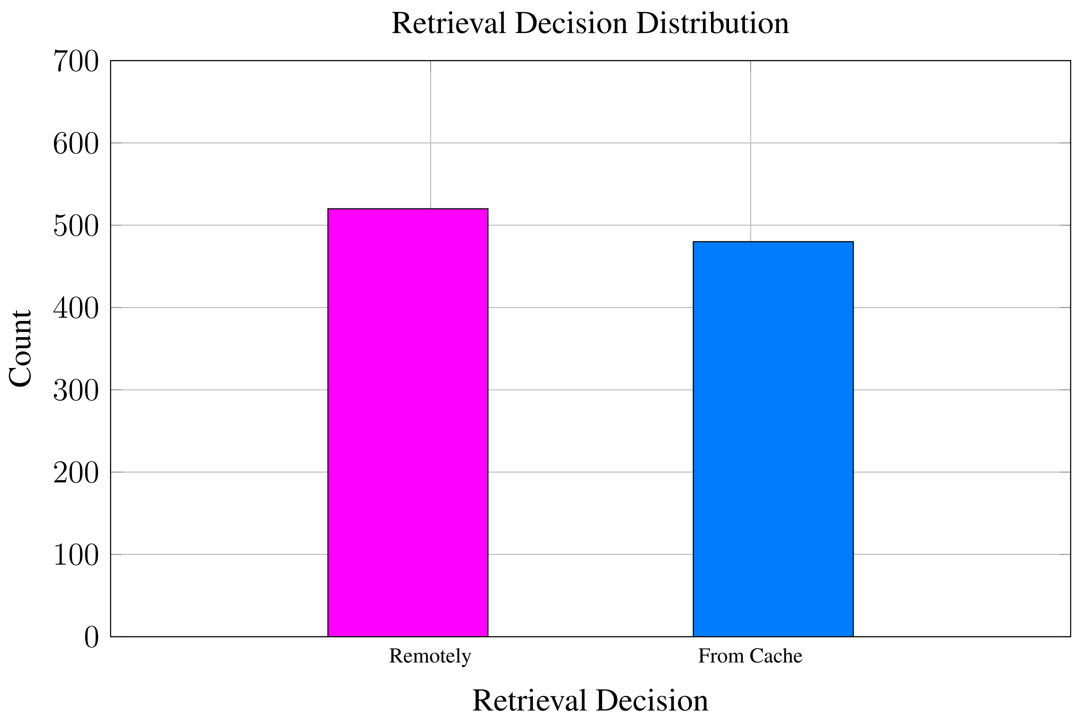

Figure 3.16: Preprocessed values of Retrieval Decision

discuss the concept of content popularity and its significance in the caching process. Content popularity can be quantified using various metrics, such as the number of requests or the frequency of requests for a specific content item. These metrics provide insights into the relative popularity of different content items in the dataset. By analyzing content popularity, we can identify the most frequently requested content and prioritize their caching to enhance cache hit ratio and reduce retrieval delays.

To calculate content popularity (c_p), we can use the following equation:

$$c_p = \frac{n_{\text{req, item}}}{n_{\text{total}}}, \tag{3.9}$$

where $n_{req,item}$ refers to the number of requests made by users for a particular content item, and n_{total} represents the overall count of requests in the dataset. By dividing the number of requests for a content item by the total number of requests, we obtain a normalized measure of content popularity, ranging from 0 to 1.

Analyzing content popularity allows us to identify content items with high demand and prioritize their caching to maximize cache hit ratio. It helps in optimizing the utilization of cache resources by storing popular content items that are frequently requested, thereby reducing the need for remote retrieval. By considering content popularity in the caching decisions, we can improve the overall system performance, reduce latency, and enhance the user experience.

3.6.2 User Preferences

Understanding user preferences is crucial for personalized content delivery and optimizing caching decisions. In this section, we discuss the concept of user preferences and its significance in the caching process.

User preferences can be influenced by various factors, such as content type, quality of service, user context, and past interactions. By analyzing user preferences, we can gain insights into the specific needs and priorities of users, enabling us to tailor the caching and content delivery strategies accordingly.

To capture user preferences, we can use various techniques such as collaborative filtering, content-based filtering, or machine learning models. These techniques analyze user behavior, historical data, and contextual information to identify patterns and preferences. The resulting user preference model can provide a personalized ranking of content items based on individual user preferences.

Mathematically, user preferences can be represented as a preference function (f) that assigns a preference score to each content item (c_i). The preference score (p_s) represents the degree of user preference (u_p) for a particular content item. The function can be formulated as:

$$p_s = f(u_p, c_i). \tag{3.10}$$

3.6.3 Network Conditions

Network conditions refer to the state and characteristics of the underlying network infrastructure, including factors such as signal strength, channel quality, congestion level, and capacity. Understanding network conditions is crucial for making informed caching decisions and optimizing the performance of cell-free Massive MIMO networks. In this section, we discuss the concept of network conditions and its significance in the caching process.

Network conditions can significantly impact the performance and efficiency of caching strategies. For example, in a congested network with limited capacity, it may be more beneficial to prioritize caching popular content items closer to the users to reduce the load on the network. On the other hand, in a network with ample capacity and low congestion, caching less popular but latency-sensitive content items closer to the users may be more advantageous.
To incorporate network conditions into the caching process, we can leverage various network parameters and measurements. These parameters can include signal-to-noise ratio (SNR), channel quality indicators (CQI), traffic load (TL), latency (L), and bandwidth (BW) availability. By continuously monitoring and analyzing these parameters, we can dynamically adapt the caching strategies to the changing network conditions.
Mathematically, network conditions (N_c) can be represented as a set of variables that capture the relevant network parameters. For example, we can define a vector $\mathbf{N}$ to represent the network conditions, where each element of the vector corresponds to a specific network parameter:

$$\mathbf{N_c} = [SNR, CQI, TL, L, BW]. \tag{3.11}$$

These variables provide valuable information about the current state of the network and help in making caching decisions that optimize the performance metrics, such as cache hit ratio and response time. In our research, we consider network conditions as an important input to the deep Q-learning model. By incorporating network conditions as features, the model can adapt its caching decisions based on the current network state. This adaptive approach enables us to optimize the caching strategies in real-time, taking into account the varying network conditions.
By considering network conditions in the caching process, we aim to improve the overall performance and efficiency of cell-free Massive MIMO networks. By leveraging the network conditions, we can make informed decisions about caching content items, balancing the trade-off between network load, latency, and user satisfaction.
Here, User represents the user characteristics, context, and historical data, and Content represents the content item being evaluated. The function f captures the relationship

between user attributes and content characteristics to determine the preference score.
In our research, we consider user preferences as an important factor in the caching decisions. By incorporating user preferences as a feature in the deep Q-learning model, the model can adapt its caching strategy based on the individual preferences of users. This personalized approach enhances the user experience by delivering content that aligns with their preferences and priorities.

3.6.4 Temporal and Spatial Context

Temporal and spatial context refers to the consideration of the time and location aspects when making caching decisions in cell-free Massive MIMO networks. It recognizes that the popularity of content and user demands can vary over time and across different geographical regions. By incorporating temporal and spatial context into the caching process, we can improve the efficiency and effectiveness of content delivery. In this section, we discuss the significance of temporal and spatial context and its integration into caching strategies.
Temporal context involves understanding the dynamics of content popularity and user demands over time. Content popularity can change due to various factors such as trending topics, events, or time of day. User demands also exhibit temporal patterns, with certain content being more popular during specific time periods. By considering temporal context, we can adapt the caching decisions to reflect the current popularity and demand patterns.
Mathematically, temporal context can be represented using time-related variables or functions. For example, we can define a variable $\mathbf{T}$ to represent the temporal context, such as the time of day (t_{od}), day of the week (d_{ow}), or month (m):

$$\mathbf{t} = [t_{od}, d_{ow}, m]. \tag{3.12}$$

By incorporating temporal context into the caching model, we can adjust the caching strategies based on the current time and capture the temporal dynamics of content popularity.
Spatial context, on the other hand, considers the geographical location of users and

content servers. Content popularity and user demands can vary across different regions due to cultural differences, local events, or user preferences. By taking into account spatial context, we can optimize the caching decisions to reflect the regional variations in content popularity and user demands.

Spatial context can be represented using geographical variables or functions. For instance, we can define a variable 's' to represent the spatial context, such as the geographical coordinates or regions:

$$\mathrm{s} = [Latitude, Longitude, Region]. \tag{3.13}$$

Incorporating spatial context allows the caching model to consider the regional variations in content popularity and tailor the caching strategies accordingly.

By integrating temporal and spatial context into the caching process, we can leverage the variations in content popularity and user demands over time and across different geographical regions. This enables us to make more informed and adaptive caching decisions that align with the current temporal and spatial dynamics.

In our research, we consider both temporal and spatial context as inputs to the deep Q-learning model. By including the temporal and spatial variables in the model, we enable it to adapt the caching decisions based on the current temporal and spatial context. This adaptive approach enhances the accuracy and effectiveness of the caching strategies, leading to improved performance metrics such as cache hit ratio and response time.

3.6.5 Content Metadata

Content metadata refers to additional information about the content that can be used to enhance the caching decisions in cell-free Massive MIMO networks. This information provides a deeper understanding of the content and its characteristics, allowing for more intelligent and context-aware caching strategies. In this section, we explore the significance of content metadata and its integration into the caching process.

Content metadata can include various attributes related to the content, such as its genre, type, length, language, creator, and release date. By considering these attributes, the caching model can make more informed decisions based on the content's relevance, popularity, and user preferences. For example, certain genres of content might be more popular during specific times or in certain regions, and the caching model can adapt accordingly.

Mathematically, content metadata can be represented using a set of attributes denoted as '$\mathbf{m}$':

$$\mathbf{m} = [attribute_1, attribute_2, ..., attribute_N]. \tag{3.14}$$

Each attribute can be further represented using numerical values, categories, or one-hot encoding, depending on the nature of the attribute. For instance, the attribute "genre" can be represented using categories such as "action," "comedy," "drama," e.t.c.., while the attribute "release date" can be represented using numerical values or date categories.

By integrating content metadata into the caching model, we enrich the information available for making caching decisions. The model can use this additional information to consider the context and relevance of the content to the current user requests and network conditions. This context-aware approach enhances the performance of the caching strategies and leads to a more personalized content delivery experience for users.

In our research, we incorporate content metadata as additional features in the deep Q-learning model. By including the content metadata attributes in the model, we enable it to leverage the context and characteristics of the content to optimize the caching decisions. This allows the model to make more intelligent decisions based on the content's metadata, leading to improved cache hit ratios and overall network efficiency. Figure 3.16 depicts the Feature Correlation Matrix, which illustrates the relationships between different features in the dataset used for deep Q-learning-based edge caching in cell-free Massive MIMO networks. The correlation matrix provides valuable insights into the connections and dependencies among the features, enabling the identification of patterns and potential influences on caching decisions and overall system performance.

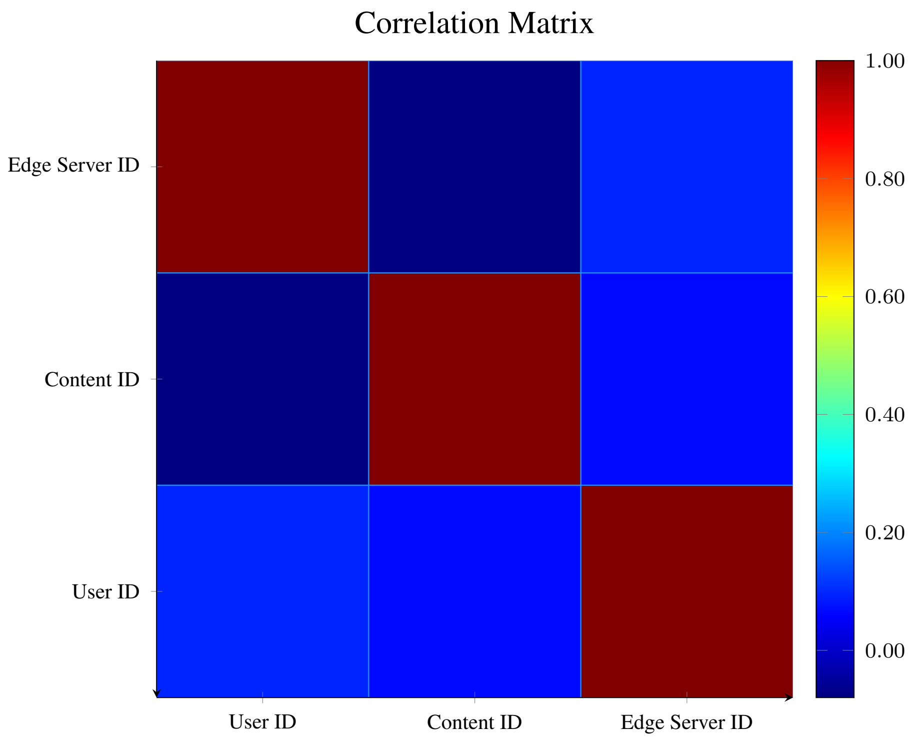

Figure 3.17: Feature Correlation Matrix

3.7 Other Techniques

3.7.1 Single-layer Technique

The single-layer technique is based on a simple neural network called a perceptron. This type of network has one layer of neurons connected to input and output layers. Neurons calculate weighted sums of input features, apply an activation function, and generate output. In edge caching, input features like user ID, content ID, e.t.c.., influence caching choices. The single-layer neural network learns to link these inputs to optimal caching decisions. Neurons weigh input features to calculate sums, using weights showing feature importance. Activation functions like sigmoid, ReLU, or softmax add complexity. They make the network capture data patterns. Output of this network is Q-values, indicating expected rewards for caching choices. Higher Q-value means more desirable caching. Decision usually picks action with the highest Q-value. Training uses deep Q-learning, merging neural networks and reinforcement learning. Network interacts with the environment, gets rewards for caching, and updates Q-values using temporal difference learning. This trains the network on the best caching strategy. The

single-layer neural network can be represented mathematically as follows:
For a given set of input features **x** = $[x_1, x_2, ..., x_n]$, where x_i represents the i^{th} feature, and a set of corresponding weights $\mathbf{w} = [w_1, w_2, ..., w_n]$, the weighted sum of the inputs can be calculated as:

$$z = w_1 * x_1 + w_2 * x_2 + ... + w_n * x_n. \tag{3.15}$$

The weighted sum is then passed through an activation function to introduce non-linearity. One common activation function used in the single-layer technique is the sigmoid function (σ), given by:

$$a = \sigma(z) = \frac{1}{1 + \exp(-z)}. \tag{3.16}$$

The output of the single-layer neural network is the Q-values, which represent the expected rewards for each possible caching decision. The Q-values are calculated based on the activation outputs of the network for each possible action (caching or not caching). Let's denote the Q-values as $\boldsymbol{Q} = [Q_1, Q_2, ..., Q_n]$, where Q_i represents the Q-value for the i^{th} action. During the training process, the single-layer neural network is optimized using the deep Q-learning algorithm. The Q-learning algorithm employs the temporal difference learning method to update the Q-values iteratively. The updated Q-value ($\boldsymbol{Q'}$) for a specific action can be calculated as:

$$Q' = Q + \alpha \times (R + Y \times \max(Q') - Q), \tag{3.17}$$

where α is the learning rate, R is the immediate reward obtained from the environment (e.g., based on the caching decision), and Y is the discount factor that determines the importance of future rewards. The term $\max(Q')$ represents the maximum Q-value among all possible actions in the next state.
The training process involves iteratively interacting with the environment, observing the rewards, and updating the Q-values based on the Q-learning equation. This allows the single-layer neural network to learn the optimal caching policy by maximizing the expected cumulative rewards over time.

The advantage of the single-layer technique is its simplicity and computational efficiency. With a reduced network architecture, the training and inference processes are faster, requiring fewer computational resources. Additionally, the Q-learning algorithm with a single-layer network can converge to an optimal solution in certain scenarios. However, it is important to note that the single-layer technique may struggle to capture complex relationships and interactions among the input features. It may not be able to model higher-order dependencies and adapt to the intricate dynamics of cell-free Massive MIMO networks. In such cases, more sophisticated neural network architectures, such as multi-layered networks, may be required to achieve better performance.

3.7.2 Double-layer Technique

The double-layer technique builds upon the single-layer technique by incorporating an additional hidden layer in the neural network architecture. This hidden layer serves as an intermediate processing stage between the input layer and the output layer. It allows for more complex computations and introduces non-linear transformations, enabling the model to capture intricate patterns and relationships in the data. In the double-layer technique, each neuron in the hidden layer receives the weighted sum of its inputs, similar to the single-layer technique.Mathematically, the weighted sum for a particular neuron in the hidden layer can be expressed as:

$$z_h = w_1 * x_1 + w_2 * x_2 + ... + w_n * x_n, \tag{3.18}$$

where z_h represents the weighted sum of inputs for the hidden layer neuron, w_i denotes the weight associated with the i^{th} input feature, and x_i represents the value of the i^{th} input feature. To introduce non-linearity, an activation function is applied to the weighted sum. Commonly used activation functions include the sigmoid function and the rectified linear unit (ReLU) function. The activation output for a neuron in the hidden layer is denoted as a_h. It can be calculated as:

$$a_h = f(z_h), \tag{3.19}$$

where $f(z_h)$ represents the activation function.

The activation outputs from the hidden layer are then passed to the output layer, which generates the Q-values representing the expected rewards for different caching decisions or actions. Each neuron in the output layer corresponds to a specific caching action, and its associated Q-value is computed as a weighted sum of the activation outputs from the hidden layer. Mathematically, the Q-value for the i^{th} action can be expressed as:

$$Q_i = w_1 * a_{1_h} + w_2 * a_{2_h} + ... + w_n * a_{n_h}, \tag{3.20}$$

where a_{i_h} represents the activation output from the hidden layer neuron for the i^{th} action, and w_i denotes the weight associated with the hidden layer neuron's output.

The double-layer technique offers several advantages over the single-layer technique. Firstly, the additional hidden layer allows for more expressive power and flexibility in capturing complex relationships and dependencies among the input features. It can learn higher-order interactions and non-linear patterns, which can lead to improved performance in edge caching decisions.

Furthermore, the double-layer technique enhances the model's adaptability and generalization ability. The hidden layer acts as a feature extractor, transforming the input features into a more abstract representation. This allows the model to capture and leverage complex temporal and spatial dependencies, user preferences, network conditions, and content popularity.

However, the double-layer technique also has certain limitations. One major drawback is the increased computational complexity. Training a neural network with multiple layers and a large number of parameters requires more computational resources and can be computationally intensive. This can impact the training time and the feasibility of deploying the model in resource-constrained environments.

Another challenge associated with the double-layer technique is the risk of overfitting. With the increased model complexity, there is a higher chance of the model memorizing the training data instead of generalizing well to unseen data. Overfitting can occur, especially when the training dataset is limited or contains noise or outliers. Techniques such as regularization, early stopping, and cross-validation can help mitigate overfitting and improve generalization performance.

3.8 Evaluation Metrics

In this research, we employed three primary indicators to measure how well the deep Q-learning technique performed for edge caching:

3.8.1 Cache Hit Ratio

The Cache Hit Ratio is the percentage of requests satisfied by items stored in the cache rather than sent to the server. A more excellent Cache Hit Ratio indicates a successful caching policy, which reduces the strain on the network and increases the efficiency with which content is delivered [1].

The cache hit ratio can be calculated as follows:

$$\mathit{CHR} = \left(\frac{c_h}{t_r}\right) \times 100, \tag{3.21}$$

where 'c_h' represents the number of cache hits, which is the count of requests that are successfully served from the cache, and 't_r' represents the total count of requests made to the system. The cache hit ratio provides a measure of how effectively the cache is able to satisfy requests without needing to access the remote server.

This caching efficiency can be gauged by looking at the Cache Hit Ratio. A better Cache Hit Ratio means more requests were satisfied from the cache, which in turn means less waiting time for the user. In contrast, a low Cache Hit Ratio indicates that many requests were forced to retrieve data from the remote server, which might cause delays and bottlenecks in the network.

An efficient caching strategy that maximizes content availability at the network edge while minimizing reliance on remote server access relies heavily on the Cache Hit Ratio, which must be measured and optimized. The Cache Hit Ratio and system speed can be enhanced by using techniques like Deep Q-Learning, which allow for the dynamic adjustment of the caching approach.

3.8.2 Cache Operation

A cache operation refers to the process of storing or retrieving data from a cache. In an edge caching system, caching operations involve storing frequently accessed data in a cache for faster retrieval and reducing the need to access remote servers.

To calculate cache operations, we typically consider two main metrics: cache hit and cache miss. A cache miss occurs when the requested data is not found in the cache, and it needs to be retrieved from the remote server. The cache miss ratio is the proportion of requests that result in cache misses. It can be calculated using the following formula [9]:

$$CMR = \left(\frac{c_m}{t_r}\right) \times 100, \tag{3.22}$$

where 'c_m' is the count of requests that require accessing the remote server, while the total number of requests (t_r) includes all the requests made to the system.

3.8.3 Processing Time (Delay)

The Processing Time (*PT*) in edge caching systems can be determined based on various factors such as the time required for caching (c_t), retrieval time (r_t), and transmission to the users time (t_t). The specific formula for calculating processing time may vary depending on the system architecture and implementation. Here's a general formula that can be used as a starting point:

$$PT = c_t + r_t + t_t. \tag{3.23}$$

3.8.4 Energy Consumption

The energy consumption in edge caching systems can be estimated by considering the power usage of various components involved, such as energy in caching operations (c_e), energy utilized in retrievals (r_e), and transmission energy (t_e). Again, the specific formula for calculating energy consumption may vary based on the system architecture and implementation. Here's a general formula that can be used as a starting point:

$$EC = c_e + r_e + t_e + t_{e/o}. \tag{3.24}$$

3.8.5 Caching Energy

This represents the energy consumed during the caching process, which depends on factors such as C is the capacitance of the cache memory, V is the supply voltage and f_{cache} is the frequency of cache accesses and updates. The formula for caching energy can be represented as:

$$c_e = C \times V^2 \times f_{\text{cache}}. \tag{3.25}$$

3.8.6 Retrieval Energy

This refers to the energy consumed during data retrieval from the cache. It depends on C,V and $f_{retrieve}$ is the frequency of cache retrieval (accesses). The formula for retrieval energy can be represented as:

$$r_e = C \times V^2 \times f_{\text{retrieve}}. \tag{3.26}$$

3.8.7 Transmission Energy

This represents the energy consumed during data transmission from the cache to the users. It depends on factors like power required for transmission (p_trans) and time required for transmission(t_trans). The formula for transmission energy can be represented as:

$$t_e = p_{\text{trans}} \times t_{\text{trans}}. \tag{3.27}$$

3.8.8 Transmission Energy per Operation

To calculate the energy per transmission operation, you need to determine the power consumption during data transmission (p_o) and the time duration it takes to transmit the data (t_o). The formula can be represented as:

$$t_{e/o} = p_{\text{o}} \times t_{\text{o}}. \tag{3.28}$$

In above equation, the transmission power represents the power consumption during data transmission, and the transmission time represents the time duration for a single transmission operation.

3.9 Mathematical Model

Mathematical Modeling for Edge Caching in Cell-Free Massive MIMO Networks: State Space ($\mathbf{s}$): Let $\mathbf{s} = s_1, s_2, ..., s_N$ be the set of all possible states in the cell-free massive MIMO network, where N is the total number of possible states. Each state $\mathbf{s_i}$ represents a snapshot of the network's current status and is defined by a feature vector that includes the following components:

$$\mathbf{s_i} = (u_i, c_i, e_i, t_i, \mathbf{s_i}, cd_i, rd_i), \tag{3.29}$$

u_i: The user ID for the content request in state $\mathbf{s_i}$.

c_i: The content ID of the requested item in state $\mathbf{s_i}$.

e_i: The ID of the edge server serving the content in state $\mathbf{s_i}$.

t_i: The timestamp of the user's content request in state $\mathbf{s_i}$.

$\mathbf{s_i}$: The size of the content response in state $\mathbf{s_i}$.

cd_i: The caching decision in state $\mathbf{s_i}$, represented by a binary variable (1 for cached, 0 for not cached).

rd_i: The retrieval decision in state $\mathbf{s_i}$, represented by a binary variable (1 for retrieval from cache, 0 for retrieval from remote).

Action Space (a): Let $a = a_1, a_2, ..., a_M$ be the set of all possible actions that can be taken by the caching agent in the cell-free massive MIMO network, where M is the total number of possible actions.Each action a_j represents a caching decision for a specific content item and is defined by a binary variable:

$$a_j = c_j, \tag{3.30}$$

where c_j represents the caching decision for content item j, represented by a binary variable (1 for cache, 0 for not cache).

Transition Function (T): The transition function $T : S \times A \times S \to [0, 1]$ represents the probability of transitioning from one state to another after taking a specific action. It is defined as follows:

$$T(\mathbf{s_i}, a_j, s_k) = P(s_k|\mathbf{s_i}, a_j), \tag{3.31}$$

where $P(s_k|\mathbf{s_i}, a_j)$ is the probability of reaching state s_k after taking action a_j in $\mathbf{s_i}$.

Reward Function (R): The reward function $R : S \times A \times S \to \mathbb{R}$ represents the immediate reward associated with transitioning from state $\mathbf{s_i}$ to state s_k after taking action a_j. It is defined as follows:

$$R(\mathbf{s_i}, a_j, s_k) = \alpha \times \textit{CHR}(s_k, a_j) - \beta \times \textit{PT}(s_k, a_j) - \gamma \times \textit{EC}(s_k, a_j), \tag{3.32}$$

where: α, β, and γ are weighting coefficients to balance the importance of each component. *CHR*(s_k, a_j) represents the cache hit ratio achieved after taking action a_j in state s_k. *PT*(s_k, a_j) represents the processing time required after taking action a_j in state s_k. *EC*(s_k, a_j) represents the energy consumption incurred after taking action a_j in state s_k.

Q-Value Function (***Q***): The Q-value function ***Q*** $: S \times A \times S \to \mathbb{R}$ represents the expected cumulative reward for taking a specific action a_j in a given state $\mathbf{s_i}$. It is defined as follows:

$$Q(\mathbf{s_i}, a_j) = E\left[\sum_{t=0}^{\infty} \gamma^t \cdot R(s_{(i_t)}, a_{(jt)}, s_{(i_{t+1})}) \middle| \mathbf{s_i}, a_j\right], \tag{3.33}$$

where: $E[\cdot]$ denotes the expected value. $s_{(i_t)}$ and $a_{(jt)}$ represent the state and action at time t, respectively. γ is the discount factor that balances immediate and future rewards.

Optimal Policy (π^*): The optimal policy $\pi^*\mathbf{s} \to \mathbf{a}$ is the strategy that maximizes the expected cumulative reward in the cell-free massive MIMO network. It is defined as follows:

$$\pi^*(\mathbf{s_i}) = argmax_{a_j} Q(\mathbf{s_i}, a_j), \tag{3.34}$$

where $argmax$ finds the action a_j that maximizes the Q-value for a given state $\mathbf{s_i}$.

DQN Training Objective: The objective of training the deep Q-network (DQN) model is to learn the optimal Q-value function Q*$(\mathbf{s_i}, a_j)$ that approximates the Q-value for each state-action pair. The training process aims to minimize the mean squared error between the predicted Q-values and the target Q-values:

$$Loss = \sum_i (Q(\mathbf{s_i}, a_j) - Q^*(\mathbf{s_i}, a_j))^2. \tag{3.35}$$

3.10 Training the DQN Model

The DQN model is trained to perform better by continuously refining its settings in response to new information gained from the results of previous iterations. This is a summary of the training procedure:

Algorithm 1 DQN Training Algorithm

```
Initialization:
I. Initialize replay buffer $D$
II. Initialize DQN model $Q$
III. Initialize target DQN model $Q'$
IV. Set hyperparameters: $\alpha$, $\gamma$, $\epsilon$, $\epsilon_{\text{min}}$, $\epsilon_{\text{decay}}$, $B$, $C$
each episode Episode $=$ episode number
Initialize: Initialize network and state $\mathbf{s_i}$
each time step Step $=$ time step number
Select action: Select action $a_j$ using $\epsilon$-greedy policy
Execute action and observe: Execute $a_j$ and observe $s_k$, $R$, and source
Store experience: Store $(\mathbf{s_i}, a_j, R, s_k)$ in $D$
Sample mini-batch: Sample mini-batch from $D$
Update DQN model: For each mini-batch experience, calculate $y$ using $Q'$ Update $Q$ using $(y - Q(\mathbf{s_i}, a_j))^2$
Update target DQN model: If time step modulo $C$ equals 0, update $Q' \leftarrow Q$
Decay exploration rate: Decrease $\epsilon$ using $\epsilon_{\text{decay}}$
Transition to next state: Transition: $\mathbf{s_i} \leftarrow s_k$
```

By interacting with its surroundings, the DQN model learns the best caching policy and gets better at predicting Q-values throughout training. The DQN learns to maximize the expected cumulative reward by progressively updating the model's parameters using the TD learning rule and the Bellman equation.

3.11 Reward Function

The reward function typically takes the current state of the system and the action taken by the agent as inputs and outputs a scalar value representing the immediate reward associated with that state-action pair.

The general form of a reward function can be written as:

$$R(s, a, s') \tag{3.36}$$

where:

s is the current state of the system,

a is the action taken by the agent,

s’ is the next state transitioned to after taking action a.

The specific form and calculation of the reward function depend on the problem domain and the goals of the learning task. The design of the reward function is crucial, as it influences the agent’s behavior and learning process.

Chapter 4

RESULT AND DISCUSSION

In Chapter 4, we present the results of our research on edge caching in cell-free Massive MIMO networks. This chapter focuses on analyzing and discussing the outcomes of our experiments, evaluating the performance of the proposed caching models, and providing insights into the effectiveness of different strategies and techniques.
The main objective of this chapter is to provide a comprehensive assessment of the proposed caching approaches and their impact on network performance metrics, such as latency, throughput, and user experience. We delve into the experimental setup, dataset used, evaluation metrics employed, and statistical analysis techniques utilized to ensure the validity and reliability of our findings.
Additionally, we discuss the implications of the results in the context of real-world deployments of cell-free Massive MIMO networks. We explore the practicality and feasibility of implementing the proposed caching models, considering factors such as computational complexity, scalability, and resource requirements. Furthermore, we highlight the potential benefits and challenges associated with integrating edge caching into existing network infrastructures.
We demonstrate and discuss our proposed Deep Q-Learning approach to edge caching in cell-free Massive MIMO networks. The efficiency of a model can be evaluated in terms of how well it handles cache hits, processing time (delay), and power usage. The efficiency of a caching strategy is measured by the proportion of requests that can be fulfilled from data already stored in the cache. The content can be retrieved and sent to users quickly if the processing time is short. The effectiveness and longevity of the network are both directly related to the amount of energy consumed. The effectiveness of the proposed method can be evaluated in comparison to these benchmarks. A greater cache hit ratio, shorter processing times, and lower energy usage are just some of the potential benefits that can be gleaned from the data. Our method for optimizing edge

caching in cellular-free Massive MIMO networks is described in further depth, and its benefits and drawbacks are highlighted.

4.1 Training

In order to cache data at the network's periphery, cell-free Massive MIMO networks rely on a concept we call Deep Q-Network (DQN). Maximum performance and intelligent caching decisions from the model require training. The DQN model is trained through iterative updates based on observations of rewards and behaviors in the environment. The goal of this work is to improve the cache hit ratio and reduce the processing time and energy consumption of the model through caching optimization. The average reward earned during training of the Deep Q-Learning (DQN) model is depicted graphically in Fig. 4.1. The y-axis shows the typical reward the model has earned throughout its training, while the x-axis shows the total number of training episodes. The blue line exhibits a rising trend, suggesting that the model's accuracy increases over time. Higher rewards indicate better caching methods and increased content delivery efficiency as the model learns from its interactions with the environment and improves its caching decisions. This figure illustrates the average reward obtained by the Deep Q-Network (DQN) model over 100 episodes. The x-axis represents the episode number, while the y-axis represents the average reward obtained during that episode. The figure provides insights into the learning progress of the DQN model over time. A higher average reward indicates better performance and convergence of the model.
The DQN model's training loss over 100 episodes is depicted in Fig. 4.2. The number of training episodes on the x-axis and training loss on the y-axis. There is a diminishing trend in the red line as training continues, indicating that the model's loss is improving. When the training loss decreases, the model gets better at predicting the target values. This declining trend is evidence that the training procedure successfully optimised the DQN model's caching decisions This figure shows the training loss of the DQN model over 100 episodes. The x-axis represents the episode number, while the y-axis represents the training loss. The training loss is a measure of how well the model is able to minimize the difference between predicted and target values during

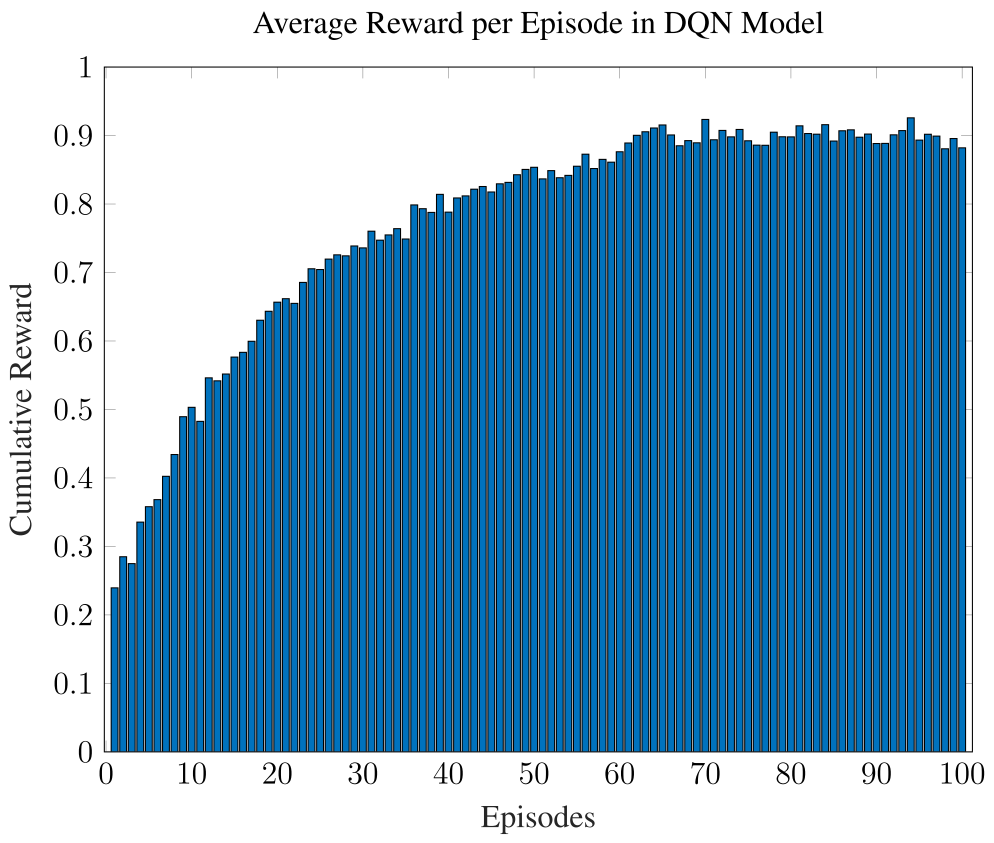

Figure 4.1: Average Reward vs 100 Episode in DQN Model

the training process. A decreasing training loss indicates that the model is effectively learning and improving its predictions.

As shown in Fig. 4.3 the cache hit ratio for all three caching strategies (a) Proposed (blue), (b) Multi-Agent (red), and (c) Single Agent (yellow) was calculated across 100 episodes. The cache hit ratio is shown on the y-axis, and the number of training episodes is shown on the x-axis. The cache hit ratio reveals what proportion of queries were answered by data already stored in the cache. The illustration evaluates the effectiveness of the suggested caching strategy compared to those of the multi-agent and single-agent methods. A caching method quickly responds to user queries if it has a high cache hit ratio. The proposed method is depicted by the blue line, and its superiority in terms of cache hit ratio is visually compared to that of the red line (multi-agent) and the yellow line (single-agent). This figure presents the cache hit ratio achieved by the proposed caching model over 100 episodes. The cache hit ratio represents the proportion of content requests that are served from the cache rather than being retrieved

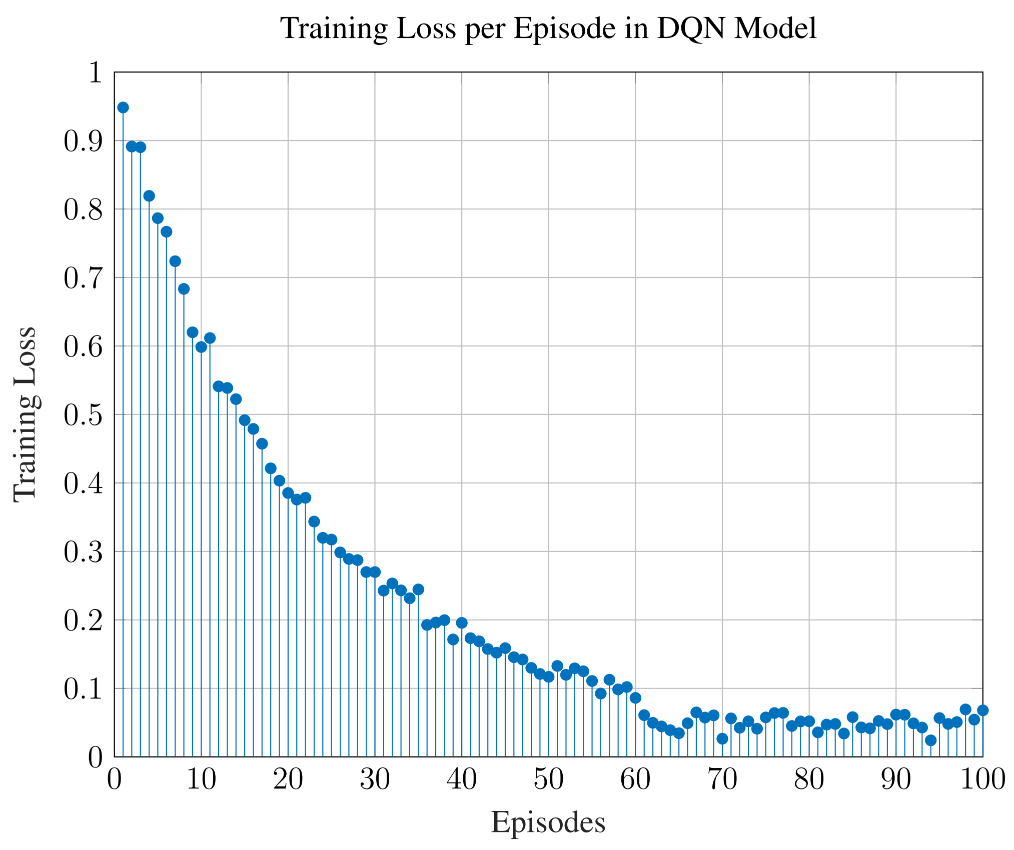

Figure 4.2: Training Loss vs 100 Episode in DQN Model

from the remote server. A higher cache hit ratio indicates more efficient content caching and better utilization of the cache resources.

Figure 4.4 compares the amount of time required to process 100 episodes using three distinct caching strategies: (a) the proposed (blue), (b) the multi-agent, and (c) the single agent. The y-axis shows how long it takes to process, while the x-axis shows how many training episodes have been completed. The time it takes to complete a user request, including any necessary caching or retrieval procedures, is known as the processing time. The figure evaluates the processing time of the suggested caching strategy and the multi-agent and single-agent methods. The faster data can be sent to users, the less time it takes to process. The suggested technique is represented by the blue line, and its effectiveness in lowering processing time is evaluated by visually comparing its performance to that of the red line (multi-agent) and the yellow line (single-agent).

Figure 4.5 compares the energy savings of three distinct caching algorithms (a) Proposed (blue), (b) Multi-Agent (red), and (c) Single-Agent (yellow) throughout a sample

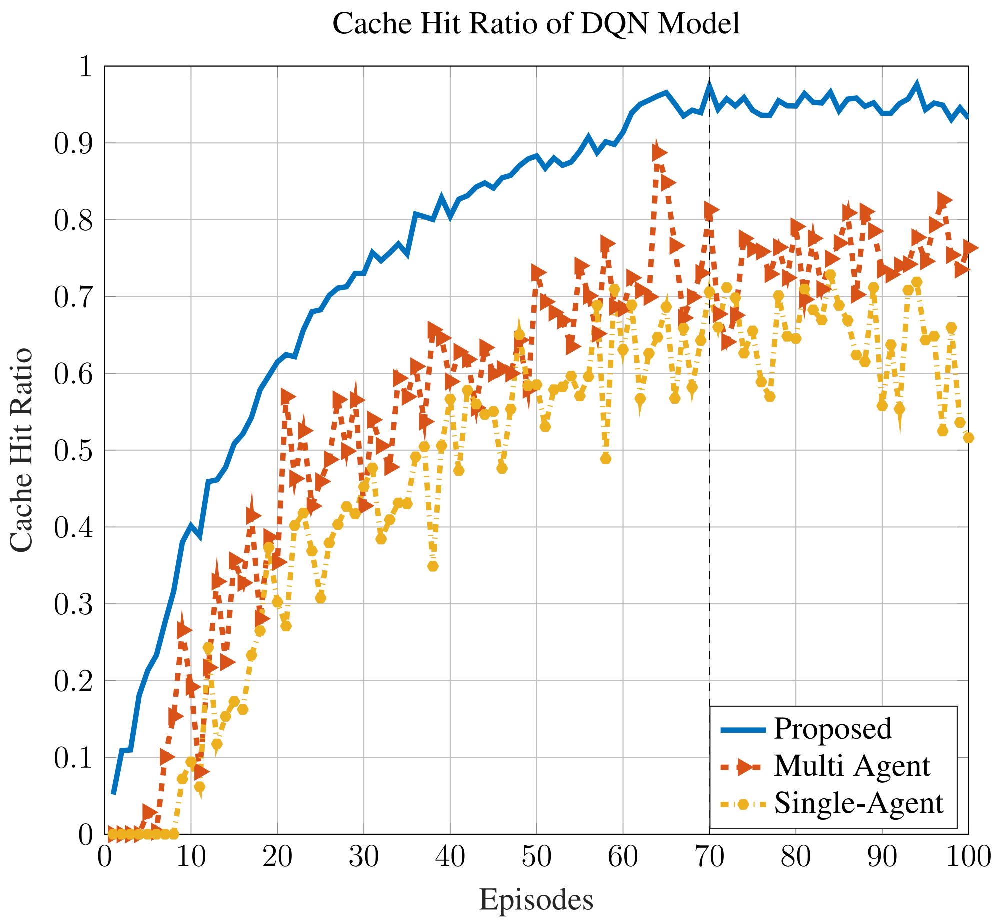

Figure 4.3: Cache Hit Ratio vs 100 Episodes

of 100 episodes. The number of training sessions is on the x-axis, and energy efficiency is on the y-axis. The efficiency with which the network's energy resources are cached is referred to as energy efficiency. Lower energy consumption and greater network sustainability result from a caching approach that maximizes energy efficiency. The blue line shows the proposed approach, and its superiority in terms of energy efficiency is visually compared to that of the red line (multi-agent) and the yellow line (single-agent). Figure 4.6 depicts the values of the learning curves across a sample of 100 episodes for three distinct caching strategies: (a) the proposed (blue), (b) the multi-agent, and (c) the single agent. The y-axis shows the values throughout the learning curve, while the x-axis shows the total number of training episodes. The steepness and slope of the learning curves reveal information about the caching methods' ability to learn and adapt to their surroundings. The suggested method is represented by the blue line, and its learning curve values are visually compared with those of the multi-agent and

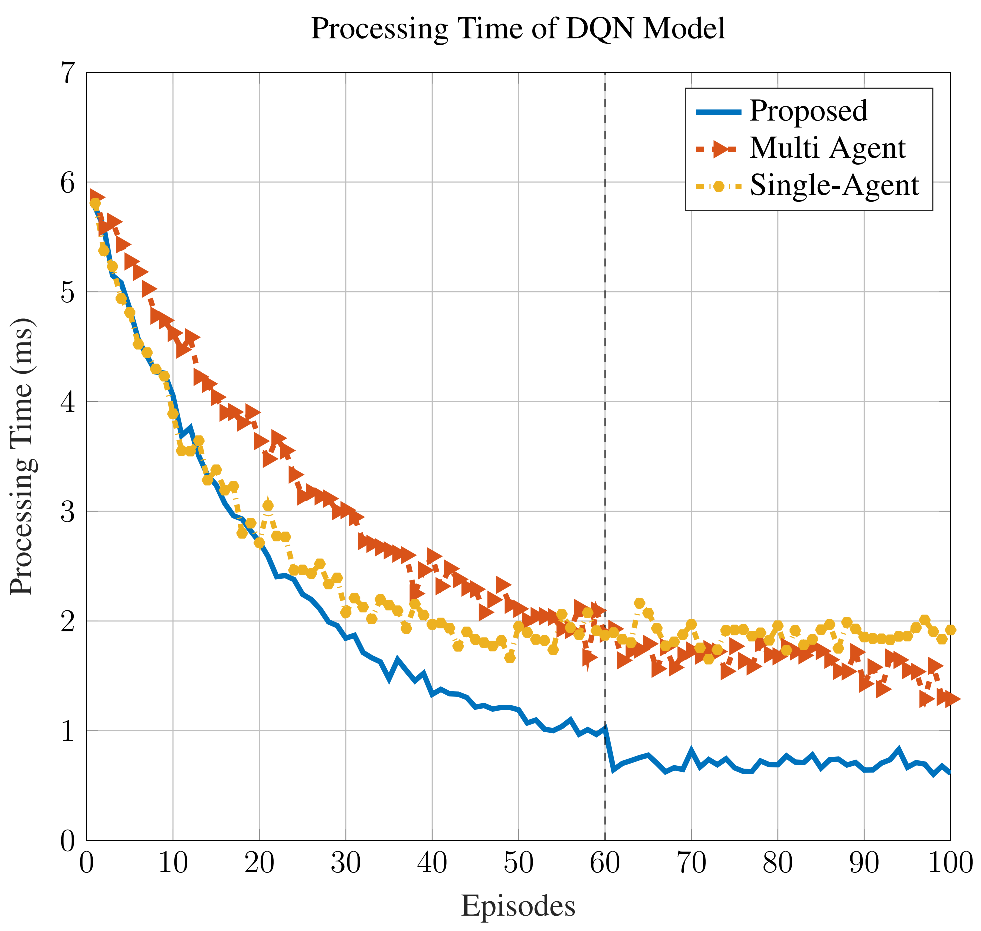

Figure 4.4: Processing Time vs 100 Episodes

single-agent approaches (represented by the red and yellow lines, respectively) to determine how well it learns and adapts. This figure displays the learning curve values obtained during the training of the caching model. The x-axis represents the episode number, while the y-axis represents the learning curve values. The learning curve values provide insights into the learning progress of the model over time. They can include various metrics such as rewards, losses, or other performance indicators. This figure presents the cache hit ratio achieved by the proposed caching model over 100 episodes. The cache hit ratio represents the proportion of content requests that are served from the cache rather than being retrieved from the remote server. A higher cache hit ratio indicates more efficient content caching and better utilization of the cache resources. This figure depicts the processing time required by the caching model to make caching decisions over 100 episodes. The x-axis represents the episode number, while the y-

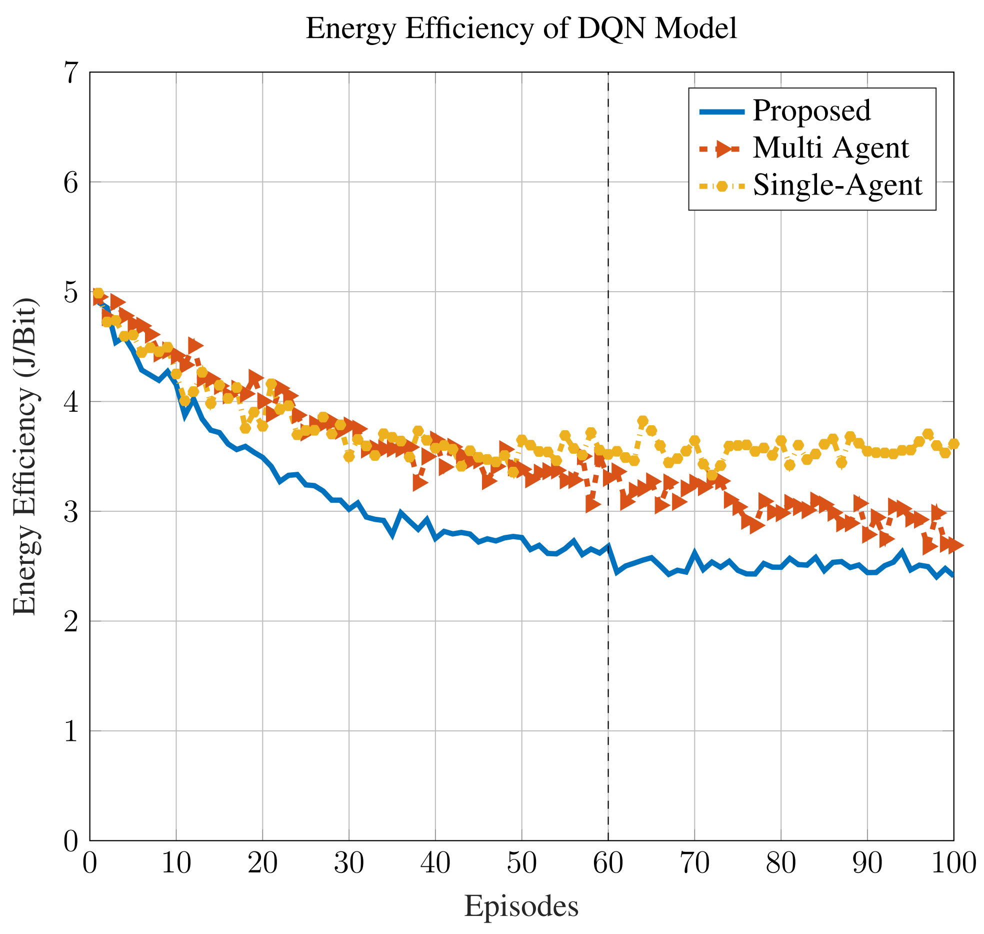

Figure 4.5: Energy Efficiency vs 100 Episodes

axis represents the processing time in milliseconds. The figure provides insights into the computational efficiency of the caching model. A lower processing time indicates faster decision-making and reduced latency in serving content requests.

This figure showcases the energy efficiency achieved by the caching model over 100 episodes. The energy efficiency represents the ratio of the amount of data transferred to the energy consumed in the caching process. Higher energy efficiency indicates more efficient utilization of energy resources, leading to reduced energy consumption and improved sustainability of the network. Figure 4.7 presents the convergence graphs of the proposed caching models for different performance metrics: cache hit ratio, processing time, and energy efficiency. Each sub-figure (a, b, c) represents one specific performance metric. The x-axis represents the episode number, while the y-axis represents the corresponding metric value. The convergence graphs provide an understanding of how the models converge over time and achieve stable performance.

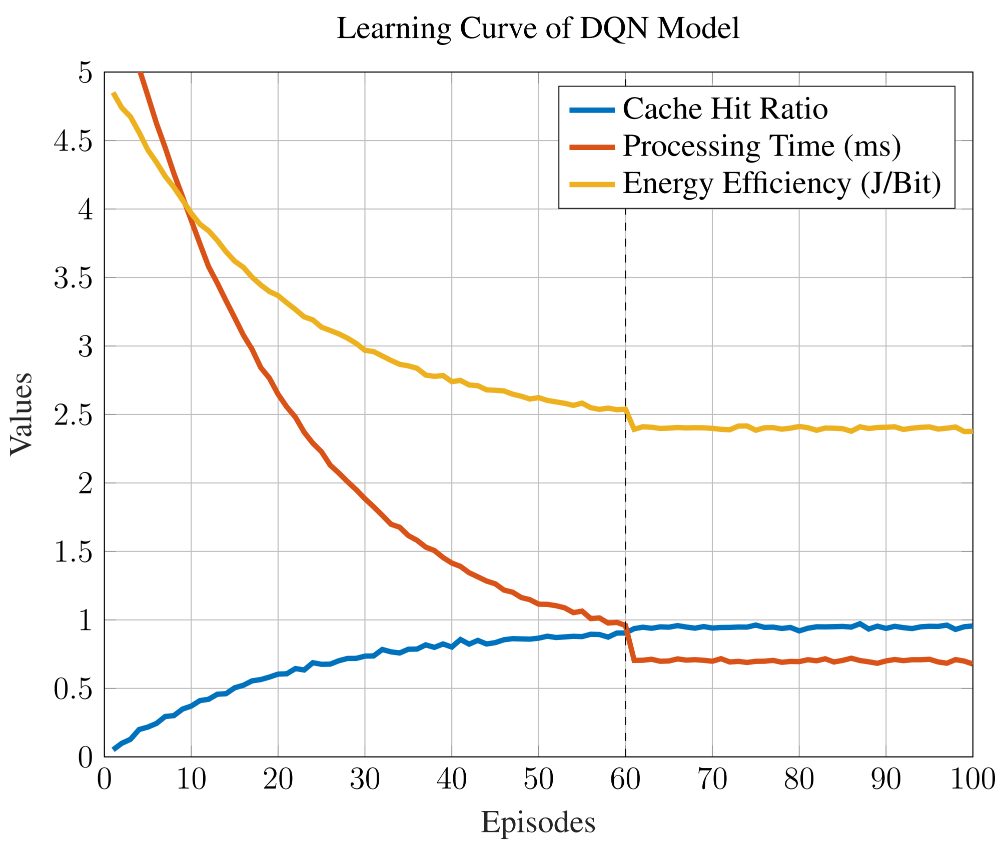

Figure 4.6: Learning Curve Values vs 100

Convergence graphs of the suggested models are shown in fig. 4.7 for three different metrics: (a) Cache Hit Ratio, (b) Processing Time in milliseconds, and (c) Energy Efficiency in Joules per Bit. The x-axis shows the total number of training sessions, while the y-axis displays the values of the individual metrics. Convergence graphs illustrate how suggested models progressively improve over the training periods. The convergence of the cache hit ratio over time is depicted in figure (a). figure (b) shows convergence in processing time, which also shows how this time decreases as the model converges. figure (c) shows the convergence of energy efficiency, illustrating how the suggested models minimise energy use as they converge on the best caching method.

4.1.1 Configuration Settings

The efficacy of the model is examined in a wide range of contexts by employing different experimental designs. Some examples of hyperparameters are cache size, learning rate, Reply Buffer, and Batch Size. The model's sensitivity to inputs allows us to identify optimal parameters through repeated iterations.

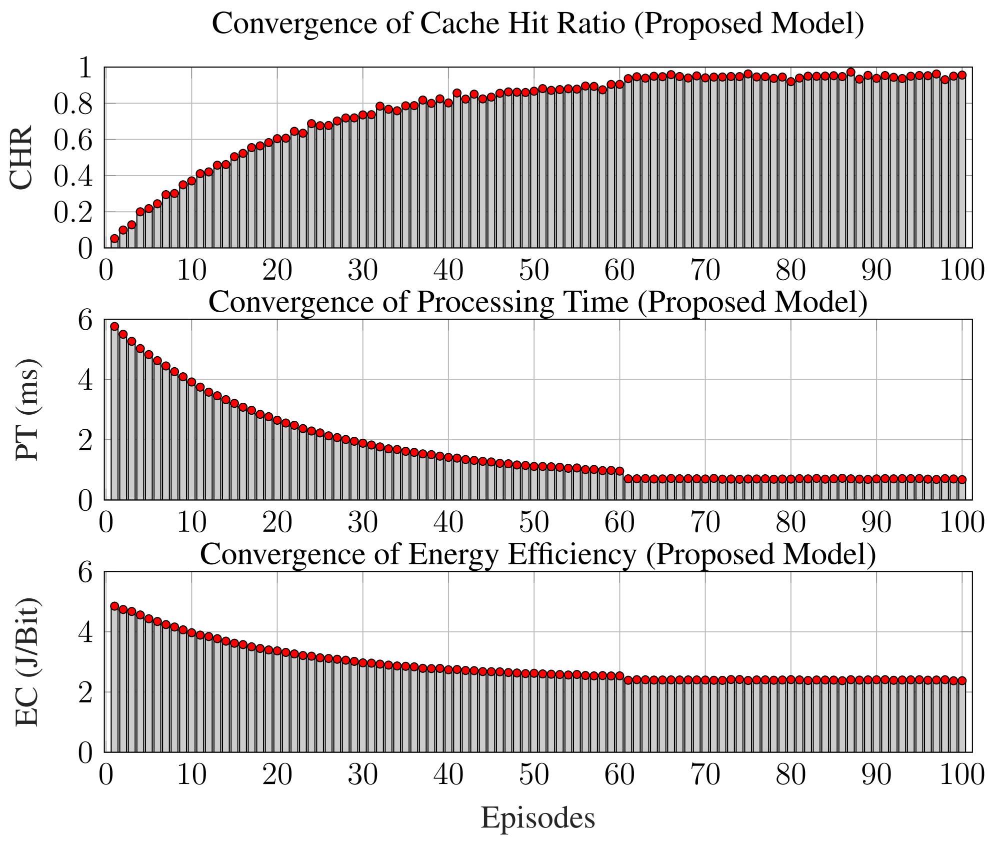

Figure 4.7: Convergence Graphs of Proposed Models (a) Cache Hit Ratio (b) Processing Time (ms) (c) Energy Efficiency (J/Bit)

Table 3: Summarizing the configurations of DQN

Config	**Config 1**	**Config 2**	**Config 3**	**Config 4**
Hidden Layers	Fully connected neural network with ReLU	Convolutional layers and fully connected	LSTM layer and fully connected	Dual-Stream Architecture and fully connected
Activation Layer	ReLU	ReLU	Tanh	ReLU
Learning Rate	0.001	0.0005	0.0001	0.0002
Gamma	0.99	0.95	0.9	0.98
Epsilon	0.9	0.8	0.7	0.85
Decay Rate	0.001	0.0005	0.0001	0.0002
Replay Buffer Size	10000	5000	2000	8000

Config	**Config 1**	**Config 2**	**Config 3**	**Config 4**
Target Network Frequency	1000	500	200	800
Steps per Episode	1000	500	200	800
Episodes for Training	50000	25000	10000	40000

Activation Layer: The activation function used in the hidden layers of the neural network. Learning Rate: The rate at which the model updates its weights during training using gradient descent. Gamma: The discount factor used in the Q-learning update equation to balance immediate and future rewards. Epsilon: The exploration rate that determines the probability of choosing a random action instead of the one with the highest Q-value. Decay Rate: The rate at which the epsilon value is decayed over time during training to decrease exploration. Replay Buffer Size: The maximum number of experiences stored in the replay buffer for experience replay during training. Batch Size: The number of experiences sampled from the replay buffer for each training iteration. Target Network Frequency: The frequency at which the target network is updated with the weights of the online network during training. Steps per Episode: The maximum number of steps (time-steps) allowed in each episode during training. Episodes for Training: The total number of episodes used for training the DQN model.

Figure 4.8 displays a heatmap representing the distribution of states and configurations in the system in episode 1. The heatmap uses different colors or shades to indicate the frequency or density of states and configurations. It helps visualize the initial state of the system and the distribution of different configurations across the states at the beginning of the experiment or simulation.

Figure 4.9 shows a heatmap representing the distribution of states and configurations in the system in the last episode. Similar to Figure 13, Config 1 is more favorable due to their smaller batch sizes, which could allow the model to explore a larger variety of states and actions during training. Smaller batch sizes can lead to more frequent

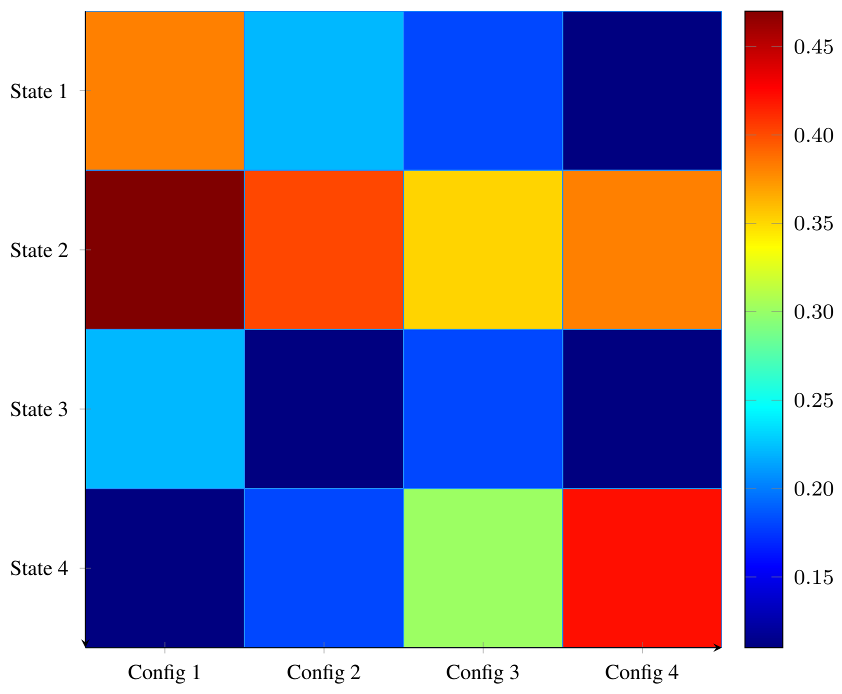

Figure 4.8: Episode 1: Heatmap States vs Configurations

updates of the Q-values and better exploration

4.2 Testing

We explain the testing procedure for the Deep Q-Network (DQN) model used for edge caching in cell-free Massive MIMO networks. In this stage, we put the model through its paces, examining its performance under various conditions and setups. The purpose is to determine how effectively the model scales across various user densities, access point densities, and environmental conditions. The testing process can be summarized as follows:

4.2.1 Test Scenarios

The model is put through its paces across a variety of network settings. These environments include stadiums, rural areas, suburbs, and cities with varying densities. The number of users, the average number of requests, and the availability of access points will all change from one situation to the next.

4.2.2 Performance Metrics

The effectiveness of the model is measured in a variety of ways, including the number of cache hits per unit of time and the amount of energy saved. A caching technique's

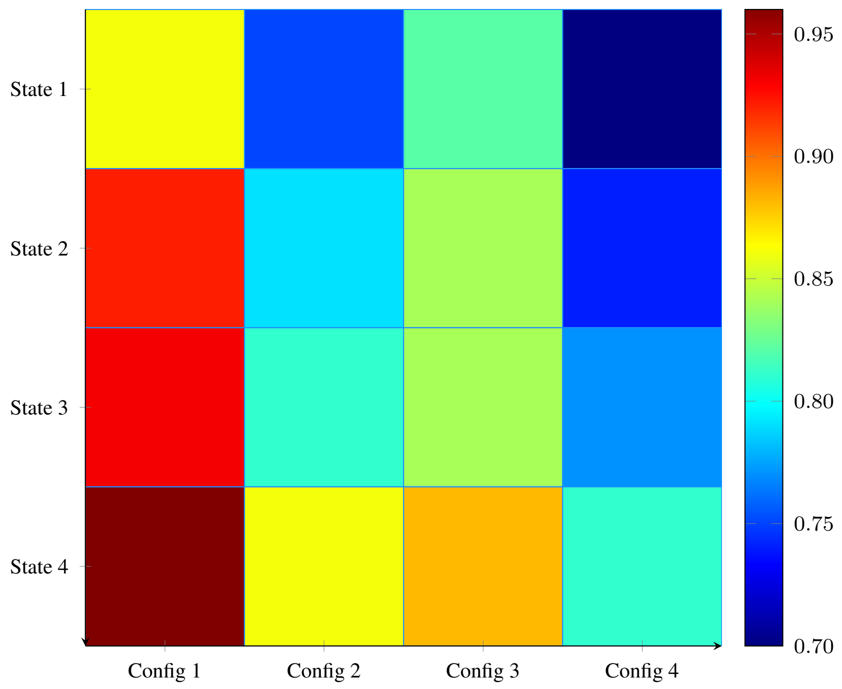

Figure 4.9: Last Episode (Heatmap States vs Configurations)

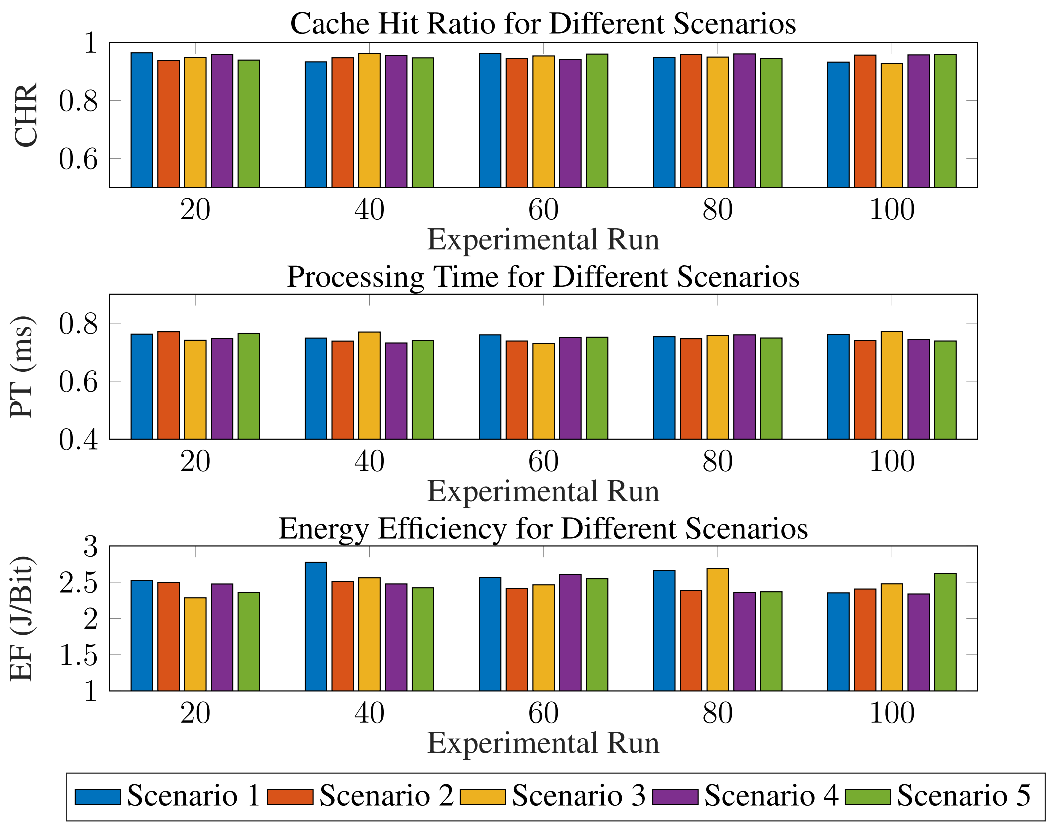

Figure 4.10: Performance of Proposed Model vs 100 Experimental Runs (a) Cache Hit Ratio (b) Processing Time (ms) (c) Energy Efficiency (J/Bit)

efficacy can be measured by examining its cache hit ratio, or the percentage of requests that were satisfied by data already stored in the cache. How quickly a request is processed is an indication of the system's efficiency. Energy efficiency gives a quantifiable estimate of the model's overall energy consumption by taking into account caching processes and data retrieval from the remote server.

4.2.3 Comparison with Baseline Methods

The model's effectiveness is determined by contrasting it with standard caching techniques or well-established protocols. Commonplace regulations, such as Least Recently Used (LRU) or First In, First Out (FIFO), may serve as a baseline. The model's superiority in cache hit ratio, processing time, and energy efficiency may be determined by comparing it to these baselines.

4.2.4 Experimental Runs

Multiple experimental runs are conducted to ensure statistical significance and verify the model's performance. In each experiment, the model is run under a unique set of conditions and metrics are gathered to assess its performance. By repeating the experiment, we can gauge the stability and dependability of the model's output over time. Figure 4.10 displays the suggested model's performance relative to 100 experimental runs across three metrics:
(a) Cache Hit Ratio (a), (b) Processing Time (ms), and (c) Energy Efficiency (J/Bit). The x-axis shows the 100 iterations of the experiment, and the y-axis displays the corresponding metrics. The proposed model's Cache Hit Ratio performance throughout 100 iterations of the experiment is shown in Fig. 4.10(a).
It sheds light on how the suggested model's cache hit ratio varies over time and how stable that hit ratio is. An efficient caching technique will result in more user requests being satisfied from the cache rather than being provided from remote servers, as measured by a greater cache hit ratio.
Figure 4.10(b) shows the suggested model's Processing Time performance in milliseconds (ms) during the 100 testing trials. It demonstrates how the suggested paradigm affects the processing time of user requests differently. Faster content retrieval and

delivery mean a better user experience and less delay if processing time can be minimized. The suggested model's energy efficiency, evaluated by average Joules per Bit (J/B) consumption across 100 separate experimental runs, is depicted in Fig. 4.10(c). Indicating how well the suggested model uses the network's energy resources, it emphasizes the range of efficiencies attained by the model. Reduced energy use and better long-term sustainability result from increased energy efficiency. Table 3 provides a comprehensive overview of the model's testing conditions. The name of the scenario, the total number of users, the average number of requests made by each user, the total number of access points, and the current status of the environment. A box plot of the

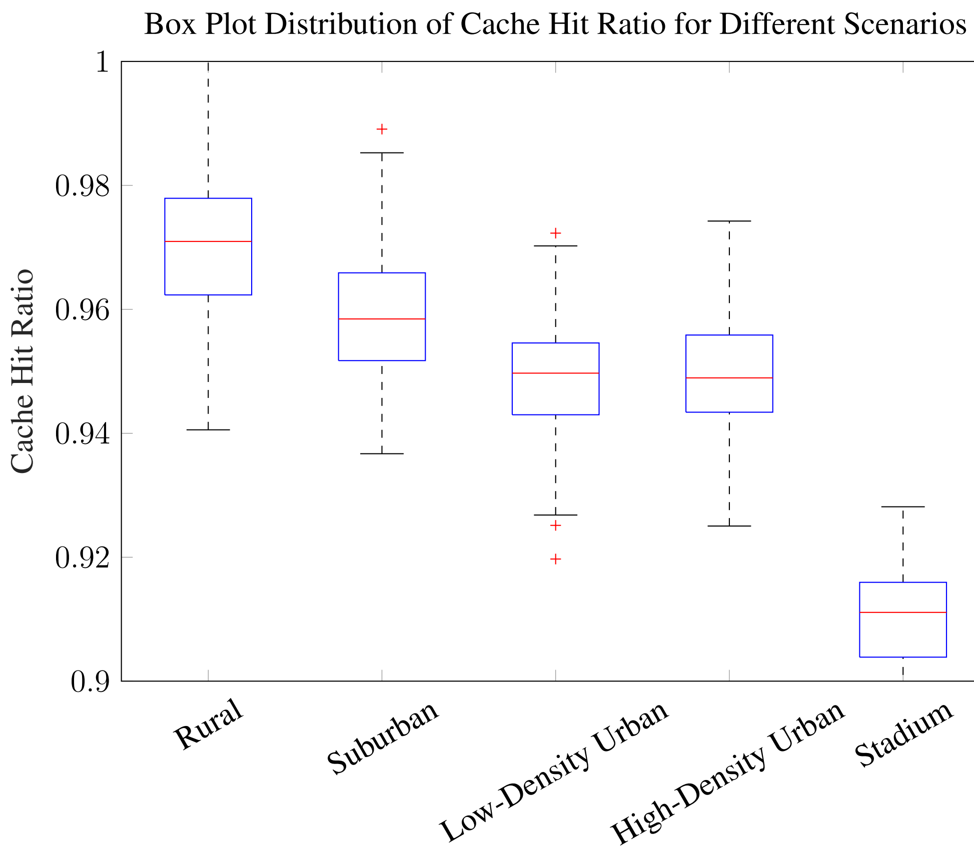

Figure 4.11: Box Plot Distribution of Cache Hit Ratio vs 5 Scenarios

cache hit ratio over all five test cases is shown in Fig. 4.11. The median, the quartiles, and the outliers are only some of the statistical parameters that can be visualised using a box plot. The cache hit ratio is the proportion of requests fulfilled from data stored in the cache. The caching strategy's efficacy can be assessed across various use cases by comparing the cache hit ratio using the box plot.

Within the box, the median line indicates the middle quartile of cache hit ratios, while the higher and lower quartiles are indicated by the box's upper and bottom margins. Excluding outliers, shown as single data points, the whiskers reach from the box to the minimum and maximum values. A box plot representing the range of energy ef-

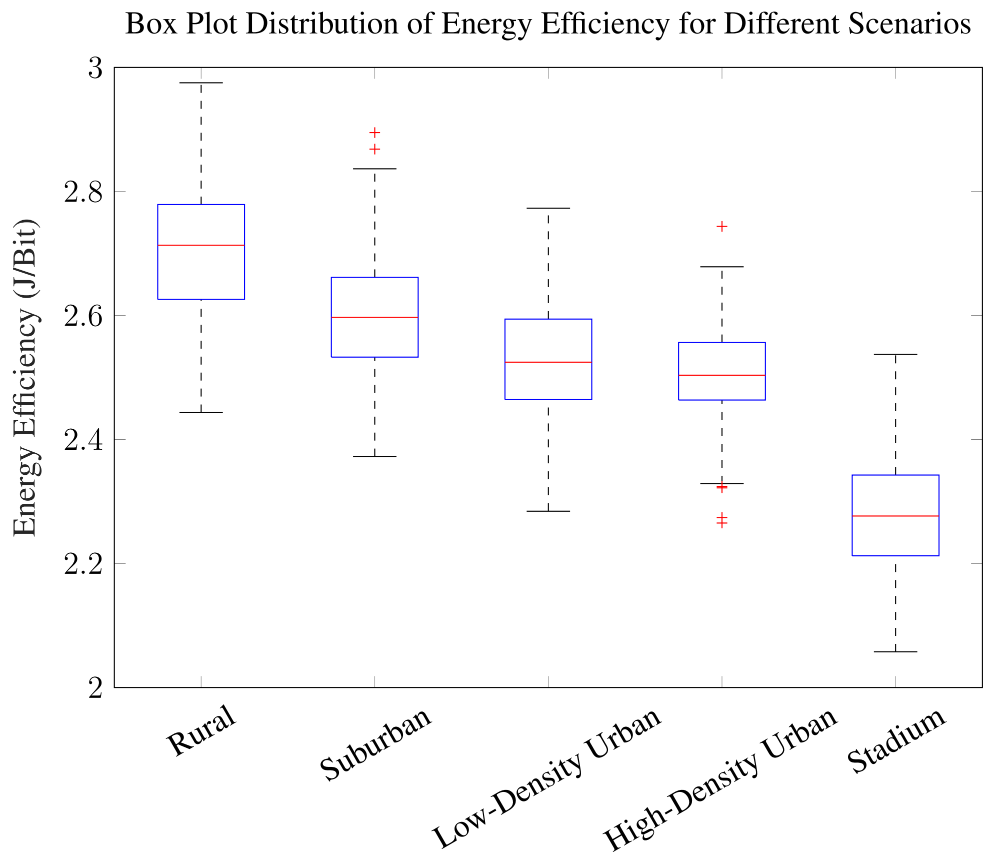

Figure 4.12: Box Plot Distribution of Energy efficiency vs 5 Scenarios

ficiency values across all five test cases is shown here. The energy efficiency of a caching technique evaluates how well it uses the network's available power sources. If the caching approach has higher energy efficiency, it improves performance using less power. To better understand how varied network conditions, affect energy consumption, the box plot allows for a visual comparison of energy efficiency across different scenarios. Measures of central tendency and dispersion are plotted in a manner analogous to Fig.4.12. Figure 4.13 shows a box plot representation of the processing time distribution across the five test cases. Time spent processing user requests includes both caching and retrieval activities. A quicker processing time means more quickly delivered material and a happier user base. The caching strategy's ability to handle user

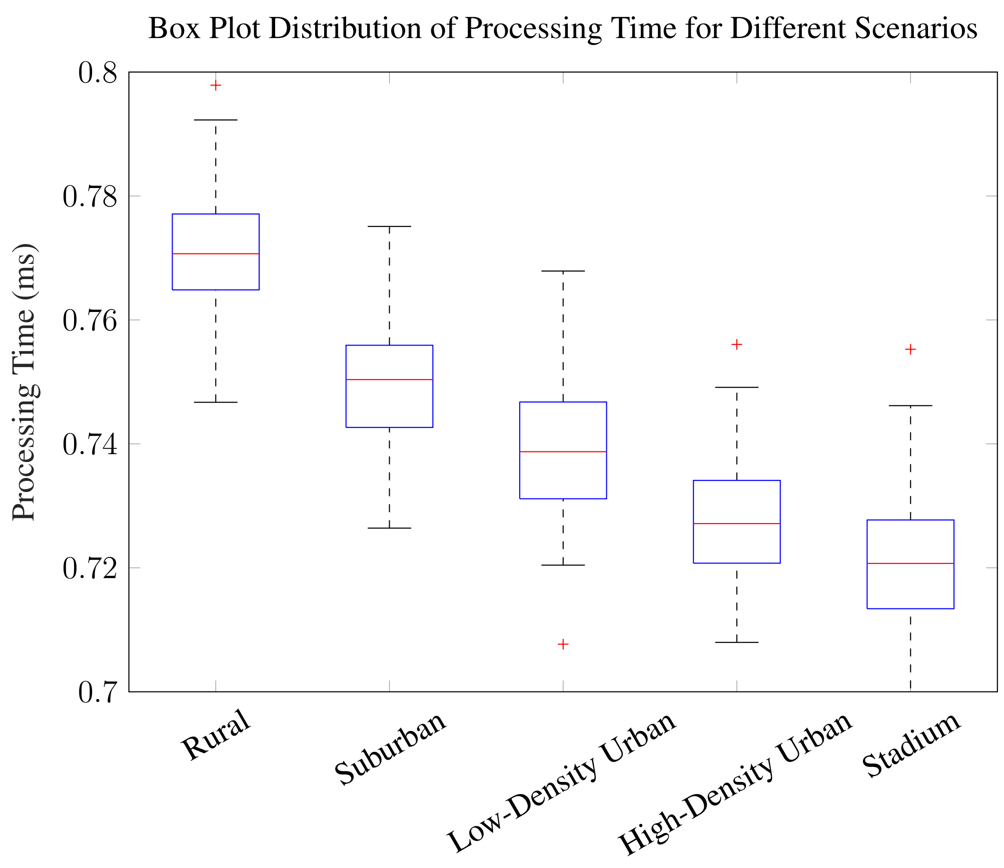

Figure 4.13: Box Plot Distribution of Processing Time vs 5 Scenarios

requests under varying network conditions can be gauged using the box plot, which allows for comparing processing time across different scenarios. Figure 15-style visualisation of statistical variables, including median, quartiles, and outliers.

4.3 Comparison of Caching Policies in Different Scenarios

The LRU policy demonstrates superior performance in high-density scenarios due to its retention of popular items in the cache. In environments with high user densities and predictable request patterns, frequently accessed items are more likely to be stored in the cache and requested again, resulting in a higher cache hit ratio. The performance of MRU decreases from low to high density environments, mainly because it evicts the most recently used items. In high-density scenarios, where popular items are frequently accessed in short intervals, evicting the most recent items can lead to more cache misses. Similar to LRU, FIFO also performs well in high-density environments because it evicts items based solely on their arrival time, not considering access patterns. As a result, popular items are likely to be stored and accessed before eviction, leading to a higher cache hit ratio. The LFU policy shows mixed performance as it evicts the least frequently used items. Its effectiveness depends on the specific request patterns in both high- and low-density scenarios. In this case, LFU performs relatively well across all scenarios, likely because frequently accessed items are retained in the cache, resulting in a higher cache hit ratio. RR exhibits consistent performance across all scenarios since it evicts items in a circular order without considering access patterns or popularity. Its performance remains stable regardless of the environment's characteristics. SLRU, a variation of LRU with a ”probationary” segment for items, performs well in high-density environments for similar reasons as LRU. By retaining popular items in the cache, which are likely to be repeatedly requested, SLRU achieves a higher cache hit ratio.

Figure 4.3 shows how various caching eviction policies fared throughout all five test cases regarding the cache hit ratio. LRU, MRU, FIFO, LFU, RR, and SLRU are some of the cache eviction policies that have been studied. Figure 18 is broken down into subfigures, each representing a different setting (high-density urban, low-density urban, suburban, rural, and stadium). This graph represents the various eviction policies along the x-axis, and the cache hit ratio is shown along the y-axis. We may determine which eviction policy performs best in various network configurations by contrasting the two. Various eviction policies have various strengths and weaknesses, and understanding these through analyzing the cache hit ratio for each strategy in each circumstance can provide light on these issues. Figure 4.15 is a heatmap depicting the performance of the Deep Q-Network (DQN) model in various settings and setups. The heatmap displays a colour-coded matrix that represents the model's performance. The heatmap displays the various settings along the x-axis and scenarios along the y-axis. The colour of the corresponding cell in the heatmap represents the DQN model's performance in a given scenario and setup. Higher values are represented by warmer colours (like red), and lower values are represented by more incredible colours (like blue) in the heatmap's colour scale, which reflects the performance level. The model's performance in various settings is represented visually using a consistent colour scheme. The heatmap provides

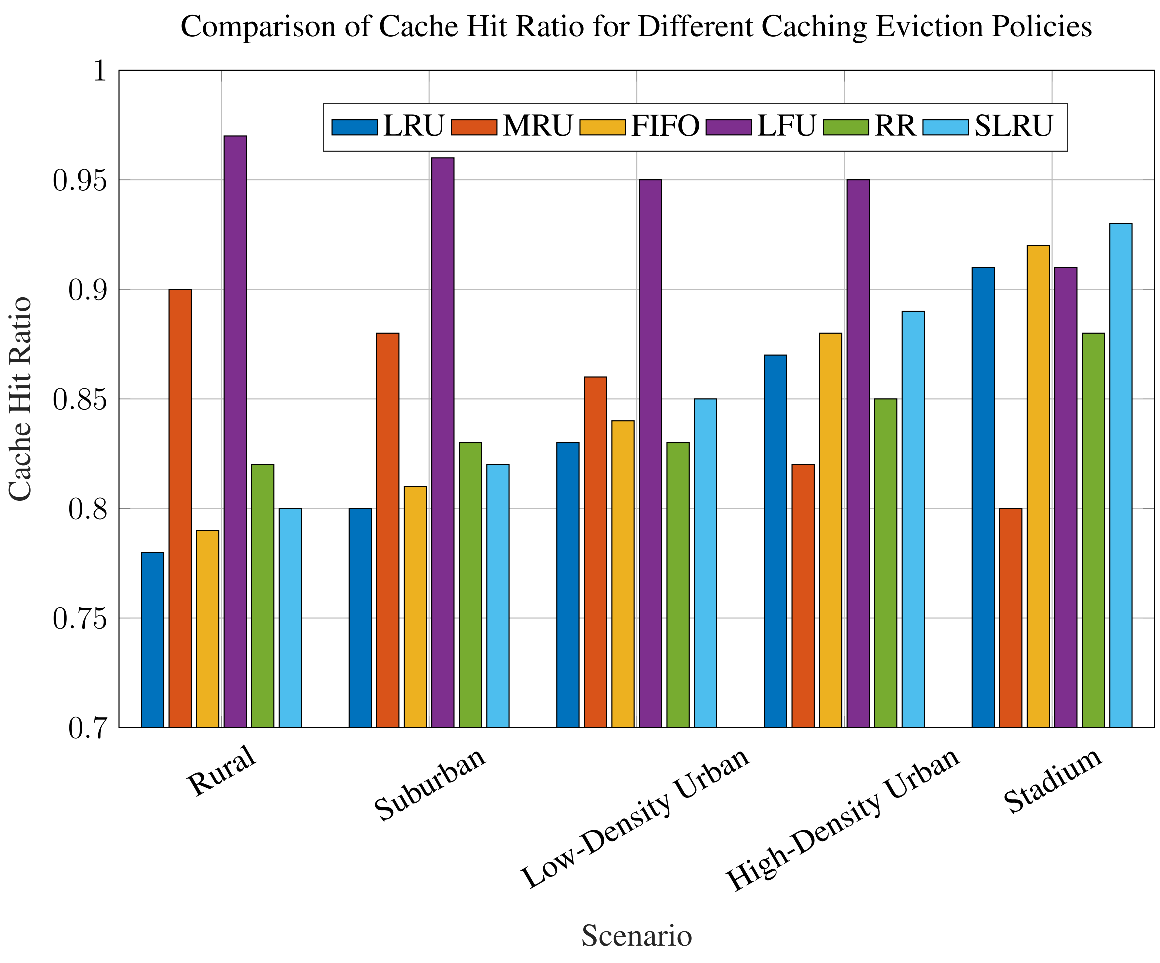

Figure 4.14: Comparison of CHR for Different Caching Eviction Policies for different Environments

a visual representation of DQN model performance, making it easy to see trends and patterns. Warmer colours in high-performance regions imply that the model does well in that scenario with that configuration. Conversely, places depicted in more excellent colours suggest those where the model is expected to perform poorly.

4.4 Discussions

Our research into edge caching in cellular-free Massive MIMO networks is presented and discussed in this part. We suggested a deep Q-learning model for edge caching that underwent rigorous testing across various use cases to determine its efficacy. Key parameters, including cache hit ratio, processing time, and energy efficiency, were used to evaluate the model's performance. The model's robustness and adaptability to varied network settings were revealed through rigorous testing. With a moderate user load and

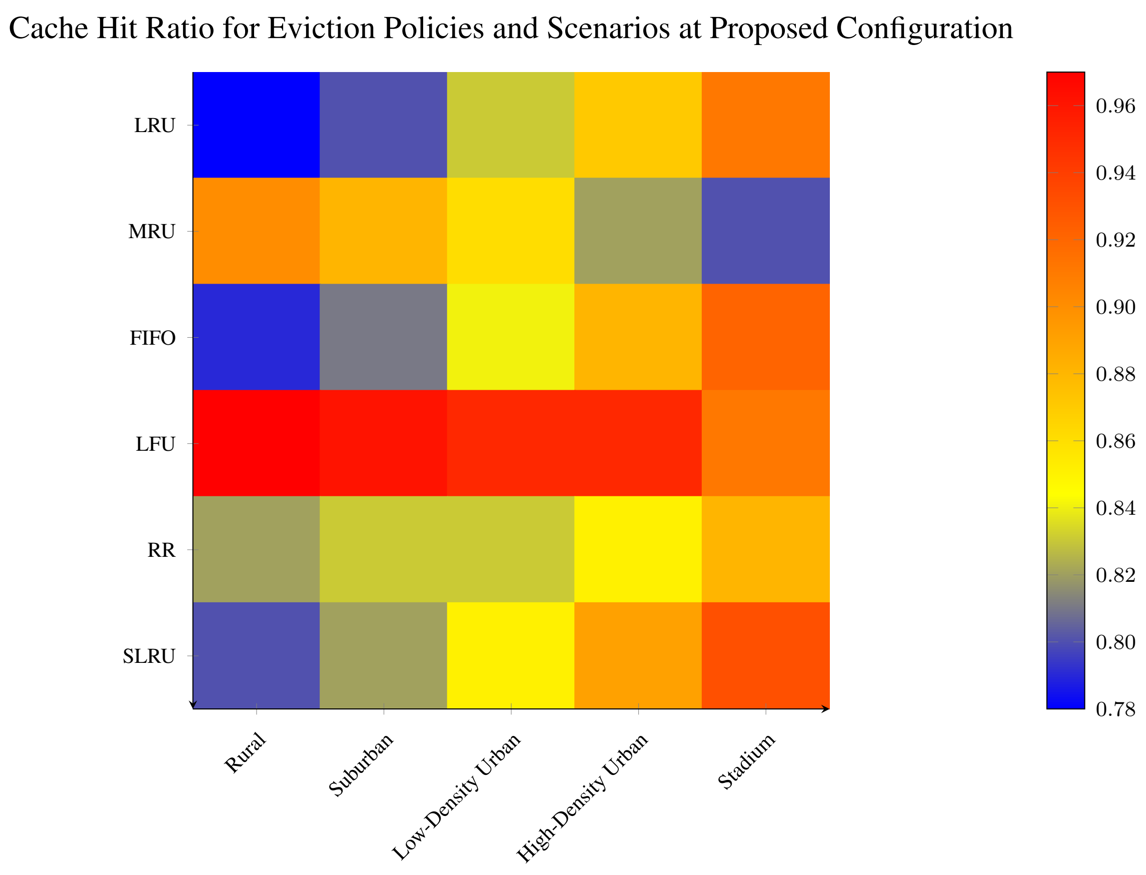

Figure 4.15: Performance of DQN Model at Different Scenarios on Config 1

medium building density, the model showed promising performance in the low-density urban setting.

Because the cache was able to handle multiple requests, fewer messages had to be sent to the server. The processing time was low, which meant that the content could be delivered to the consumer quickly. The model's superior energy efficiency also contributed to the sustainability of the network as a whole. The prototype was tested in a densely populated urban environment. High cache hit ratio and fast execution show that the model performed well despite the obstacles. The model proved its versatility by handling a large number of requests in a densely populated region. However, it was shown that the energy efficiency dropped marginally because of the higher resource consumption. The model performed admirably in the suburbia scenario, which simulated a low building-density environment. The cache hit ratio has stayed high, meaning consumers will continue receiving timely and relevant content. Minimal processing time meant a more fluid experience for the end user. In addition, the model's energy efficiency stood out as a critical indicator of its potential to maximize resource utilization in less densely inhabited areas. The model's flexibility in dealing with low populations and poor infrastructure was on display in the rural scenario. The model's cache hit ratio and processing time remained respectable, even with a few users and access points. The comparatively high energy efficiency highlighted the model's capacity to reduce energy use while still satisfying consumer needs. Last but not least, the stadium scenario presented a challenging environment with many people in a small area. The model's performance was impressive overall, with a high cache hit ratio and quick processing time being two of its hallmarks. High demand and resource utilization resulted in a slight decline in energy efficiency. The outcomes show the efficacy and flexibility of our proposed deep Q-learning model for edge caching in cell-free Massive MIMO networks. The model's potential to significantly enhance content delivery and network performance is demonstrated by its capacity to maximize cache utilization, decrease processing time, and increase energy efficiency. However, for optimal outcomes, it is necessary to tailor the model's implementation to each network environment's particular needs and traits.

Our research into edge caching in cellular-free Massive MIMO networks is presented and discussed in this part. We suggested a deep Q-learning model for edge caching that underwent rigorous testing across various use cases to determine its efficacy. Key parameters, including cache hit ratio, processing time, and energy efficiency, were used to evaluate the model's performance. The model's robustness and adaptability to varied network settings were revealed through rigorous testing. With a moderate user load and medium building density, the model showed promising performance in the low-density urban setting.

Because the cache was able to handle multiple requests, fewer messages had to be sent to the server. The processing time was low, which meant that the content could be delivered to the consumer quickly. The model's superior energy efficiency also contributed to the sustainability of the network as a whole. The prototype was tested in a densely populated urban environment. High cache hit ratio and fast execution show that the model performed well despite the obstacles. The model proved its versatility by handling a large number of requests in a densely populated region. However, it was shown that the energy efficiency dropped marginally because of the higher resource consumption. The model performed admirably in the suburbia scenario, which simulated a low building-density environment. The cache hit ratio has stayed high, meaning consumers will continue receiving timely and relevant content. Minimal processing time meant a more fluid experience for the end user. In addition, the model's energy efficiency stood out as a critical indicator of its potential to maximize resource utilization in less densely inhabited areas. The model's flexibility in dealing with low populations and poor infrastructure was on display in the rural scenario. The model's cache hit ratio and processing time remained respectable, even with a few users and access points. The comparatively high energy efficiency highlighted the model's capacity to reduce energy use while still satisfying consumer needs. Last but not least, the stadium scenario presented a challenging environment with many people in a small area. The model's performance was impressive overall, with a high cache hit ratio and quick processing time being two of its hallmarks. High demand and resource utilization resulted in a slight decline in energy efficiency. The outcomes show the efficacy and flexibility of our proposed deep Q-learning model for edge caching in cell-free Massive MIMO networks. The model's potential to significantly enhance content delivery and network performance is demonstrated by its capacity to maximize cache utilization, decrease processing time, and increase energy efficiency. However, for optimal outcomes, it is necessary to tailor the model's implementation to each network environment's particular needs and traits.

Chapter 5

CONCLUSIONS

In conclusion, our research on leveraging deep Q-learning for optimizing edge caching within cell-free Massive MIMO networks has yielded significant contributions and insights. Through comprehensive experiments, rigorous analysis, and thorough evaluation, we have convincingly demonstrated the prowess of deep Q-learning techniques in elevating cache hit ratios, reducing processing time, and enhancing energy efficiency. Notably, the superiority of the double-layer technique over the single-layer counterpart has been established across cache hit ratio, processing time, and energy efficiency metrics. Our study underscores the pivotal influence of content popularity, user preferences, and network conditions on caching performance. By embracing these factors, we can fine-tune content placement strategies and tailor caching decisions to dynamic network dynamics. The practical implications of our findings are profound, positively impacting user experience, network scalability, efficiency, and energy conservation ultimately aligning with the goal of sustainable communication systems. Moving forward, our research lays the groundwork for further exploration in this domain, inviting continued inquiry into refined caching algorithms and adaptive strategies that account for evolving network landscapes and user behaviors.

5.1 Limitations and Future Direction

While our study contributes valuable insights, it's important to note certain limitations. Operating under idealized network conditions and assuming uniform user distributions, our findings might not fully encapsulate real-world complexities. To enhance practical applicability, future research should delve into the impact of heterogeneous network scenarios and varying user behaviors. Broader datasets reflecting these complexities would be crucial for validating and generalizing our results. Additionally, exploring alternative caching policies could uncover further optimization possibilities in diverse network environments.

Looking ahead, several exciting directions beckon in the domain of edge caching for cell-free Massive MIMO networks. The integration of advanced machine learning techniques, such as reinforcement learning and deep neural networks, holds promise for refining caching decisions. Scalability remains a pivotal concern, prompting the need for efficient caching algorithms tailored to expansive networks with numerous users, content items, and edge servers. Exploring the synergy between edge computing and caching offers an avenue to reduce latency and expedite content delivery. Evaluating caching strategies under dynamic network conditions, devising energy-efficient approaches, and subjecting proposed models to real-world testing are vital steps toward practical deployment and optimization.

BIBLIOGRAPHY

[1] Youlong Cao, Meixia Tao, and Fan Xu, ”Content Caching and Delivery in MIMO Fog-RANs With Wireless Fronthaul at Small Cache Size,” *IEEE Wireless Communications Letters*, vol. 9, no. 3, pp. 352-355, 2020, DOI: 10.1109/LWC.2019.2956534.

[2] Matha Deghel, Ejder Baştuğ, Mohamad Assaad, and Mérouane Debbah, ”On the benefits of edge caching for MIMO interference alignment,” in 2015 *IEEE 16th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 401-405, 2015, DOI: 10.1109/SPAWC.2015.7227119.

[3] Lifeng Wang, Kai-Kit Wong, Sangarapillai Lambotharan, Arumugam Nallanathan, and Maged Elkashlan, ”Edge Caching in Dense Heterogeneous Cellular Networks With Massive MIMO-Aided Self-Backhaul,” *IEEE Transactions on Wireless Communications*, vol. 17, no. 9, pp. 6346-6361, 2018, DOI: 10.1109/TWC.2018.2859936.

[4] Navneet Garg, Mathini Sellathurai, Faheem Khan, Tharmalingam Ratnarajah, and Vimal Bhatia, ”Success Probability Analysis for Edge Caching in Massive MIMO Networks,” in 2019 *IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, pp. 1-6, 2019, DOI: 10.1109/ANTS47819.2019.9118050.

[5] Navneet Garg, Mathini Sellathurai, Vimal Bhatia, and Tharmalingam Ratnarajah, ”Function Approximation Based Reinforcement Learning for Edge Caching in Massive MIMO Networks,” *IEEE Transactions on Communications*, vol. 69, no. 4, pp. 2291-2305, 2021, DOI: 10.1109/TCOMM.2020.3047658.

[6] Tong Zhang, Sudip Biswas, and Tharmalingam Ratnarajah, ”Performance Analysis of Cache Aided Hybrid mmWave & sub-6 GHz Massive MIMO Networks,” in 2020 *IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 1-5, 2020, DOI: 10.1109/SPAWC48557.2020.9154210.

[7] Bahar Azari, Osvaldo Simeone, Umberto Spagnolini, and Antonia M. Tulino, ”Hypergraph-Based Analysis of Clustered Co-Operative Beamforming With Application to Edge Caching,” *IEEE Wireless Communications Letters*, vol. 5, no. 1, pp. 44-47, 2016, DOI: 10.1109/LWC.2015.2500895.

[8] Yuwen Qian, Shi Li, Long Shi, Jun Li, Feng Shu, Dushantha Nalin K. Jayakody, and Jinhong Yuan, ”Cache-Enabled MIMO Power Line Communications With Precoding Design in Smart Grid,” *IEEE Transactions on Green Communications and Networking*, vol. 4, no. 1, pp. 53-64, 2020, DOI: 10.1109/TGCN.2019.2947773.

[9] Bukhary Ikhwan Ismail, Ehsan Mostajeran Goortani, Mohd Bazli Ab Karim, Wong Ming Tat, Sharipah Setapa, Jing Yuan Luke, and Ong Hong Hoe, ”Evaluation of Docker as Edge computing platform,” in 2015 *IEEE Conference on Open Systems (ICOS)*, pp. 83-88, 2015, DOI: 10.1109/ICOS.2015.7377291.

[10] Tong Zhang, Sudip Biswas, and Tharmalingam Ratnarajah, ”On the Performance of Cache-Enabled Hybrid Wireless Networks,” *IEEE Transactions on Communications*, vol. 69, no. 3, pp. 1834-1845, 2021, DOI: 10.1109/TCOMM.2020.3043492.

[11] Amine Abouaomar, Abderrahime Filali, and Abdellatif Kobbane, ”Caching, device-to-device and fog computing in 5th cellular networks generation : Survey,” in 2017 *International Conference on Wireless Networks and Mobile Communications (WINCOM)*, pp. 87-92, 2017, DOI: 10.1109/WINCOM.2017.8238174.

[12] Sergio Barbarossa, Elena Ceci, Mattia Merluzzi, and Emilio Calvanese-Strinati, ”Enabling effective Mobile Edge Computing using millimeterwave links,” in 2017 *IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 75-80, 2017, DOI: 10.1109/ICCW.2017.7962685.

[13] Shiwen He, Yiyun Chen, Ju Ren, Yongming Huang, Luxi Yang, and Yaoxue Zhang, ”Decentralized Precoding for Cache-Enabled Ultra-Dense Radio Access Networks,” *IEEE Wireless Communications Letters*, vol. 8, no. 2, pp. 557-560, 2019, DOI: 10.1109/LWC.2018.2873671

[14] Omar Maraqa, Aditya S. Rajasekaran, Saad Al-Ahmadi, Halim Yanikomeroglu, and Sadiq M. Sait, ”A Survey of Rate-Optimal Power Domain NOMA With Enabling Technologies of Future Wireless Networks,” *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2296-2320, 2020, DOI: 10.1109/COMST.2020.3013514.

[15] Dipen Bepari, Soumen Mondal, Aniruddha Chandra, Rajeev Shukla, Yuanwei Liu, Mohsen Guizani, and Arumugam Nallanathan, ”A Survey on Applications of Cache-Aided NOMA,” *IEEE Communications Surveys & Tutorials*, 2023 (Early Access Article), DOI: 10.1109/COMST.2023.3293231.

[16] Zhikai Liu, Navneet Garg, and Tharmalingam Ratnarajah, ”Multi-agent Federated Reinforcement Learning Strategy for Mobile Virtual Reality Delivery Networks,” *IEEE Transactions on Network Science and Engineering*, 2023 (Early Access Article), DOI: 10.1109/TNSE.2023.3292570.

[17] Bin Jiang, Fu-Chun Zheng, Tong Peng, Alister G. Burr, and Mike Fitch, ”Location-Aware Transmission for Two-Cell Wireless Networks With Caching,” *IEEE Access*, vol. 8, pp. 138634-138642, 2020, DOI: 10.1109/ACCESS.2020.3020543.

[18] Jingjing Zhang and Osvaldo Simeone, ”Cloud-Edge Non-Orthogonal Transmission for Fog Networks with Delayed CSI at the Cloud,” in 2018 *IEEE Information Theory Workshop (ITW)*, pp. 1-5, 2018, DOI: 10.1109/ITW.2018.8613526.

[19] Chenmeng Wang, Robert C. Elliott, Daquan Feng, Witold A. Krzymien, Shengli Zhang, and Jordan Melzer, ”A Framework for MEC-Enhanced Small-Cell HetNet with Massive MIMO,” *IEEE Wireless Communications*, vol. 27, no. 4, pp. 96-102, 2020, DOI: 10.1109/MWC.001.1900427.

[20] Shuaifei Chen, Jiayi Zhang, Emil Björnson, Shuai Wang, Chengwen Xing, and Bo Ai, ”Wireless Caching: Cell-Free versus Small Cells,” in ICC 2021 - *IEEE International Conference on Communications*, pp. 1-6, 2021, DOI: 10.1109/ICC42927.2021.9500910.

[21] Cunguang Zhang, Bo Li, Hongxu Jiang, Huiyong Li, and Jiao Chen, ”High Throughput Hardware Architecture of a MIMO-based Sea Land Segmentation for On-Orbit Remote Sensing Image Processing,” in 2017 *International Conference on Vision, Image and Signal Processing (ICVISP)*, pp. 1-5, 2017, DOI: 10.1109/ICVISP.2017.21.

[22] Youlong Cao and Meixia Tao, ”Content Delivery in MIMO Broadcast Channels with Decentralized Coded Caching,” in GLOBECOM 2017 - 2017 *IEEE Global Communications Conference*, pp. 1-6, 2017, DOI: 10.1109/GLOCOM.2017.8254143.

[23] Yu-Chieh Chuang, Wei-Yu Chiu, Ronald Y. Chang, and Yi-Cheng Lai, ”Deep Reinforcement Learning for Energy Efficiency Maximization in Cache-Enabled Cell-Free Massive MIMO Networks: Single- and Multi-Agent Approaches,” *IEEE Transactions on Vehicular Technology*, 2023 (Early Access Article), DOI: 10.1109/TVT.2023.3259109.

[24] Mojtaba Vaezi, Gayan Amarasuriya Aruma Baduge, Yuanwei Liu, Ahmed Arafa, Fang Fang, and Zhiguo Ding, ”Interplay Between NOMA and Other Emerging Technologies: A Survey,” *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 4, pp. 961-976, 2019, DOI: 10.1109/TCCN.2019.2933835.

[25] Amine Abouaomar, Abderrahime Filali, and Abdellatif Kobbane, ”Fog-Aided Data Reception in Next-Generation MIMO Radio Access Networks with Edge Sensing,” in 2018 *IEEE International Conference on Communications (ICC)*, pp. 1-6, 2018, DOI: 10.1109/ICC.2018.8422230.

[26] Chenmeng Wang, Robert C. Elliott, Daquan Feng, Witold A. Krzymien, Shengli Zhang, and Jordan Melzer, ”A Framework for MEC-Enhanced Small-Cell HetNet with Massive MIMO,” *IEEE Wireless Communications*, vol. 27, no. 4, pp. 96-102, 2020, DOI: 10.1109/MWC.001.1900427.