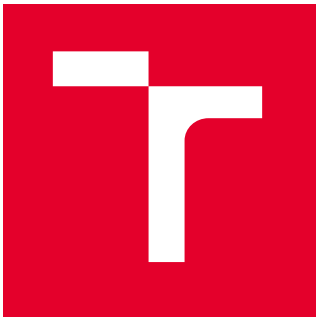
BRNO UNIVERSITY OF TECHNOLOGY

Faculty of Electrical Engineering
and Communication

DOCTORAL THESIS

Brno, 2025

Ing. Aneta Koláčková



BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF TELECOMMUNICATIONS

ÚSTAV TELEKOMUNIKACÍ

AI-AIDED AND DATA-DRIVEN ENERGY SAVINGS OPTIMIZATION FOR BASEBAND UNITS IN CELLULAR NETWORKS

OPTIMALIZACE ÚSPOR ENERGIE PRO ZÁKLADNOVÉ JEDNOTKY V MOBILNÍCH SÍTÍCH POMOCÍ UMĚLÉ INTELIGENCE A DATOVĚ ŘÍZENÉHO PŘÍSTUPU

DOCTORAL THESIS
DIZERTAČNÍ PRÁCE

AUTHOR — **Ing. Aneta Koláčková**
AUTOR PRÁCE

SUPERVISOR — **doc. Ing. Jan Jeřábek, Ph.D.**
ŠKOLITEL

BRNO 2025

ABSTRACT

This doctoral thesis addresses the high energy costs in 5G and beyond networks by focusing on Baseband Units (BBUs) - essential components that handle signal processing that often waste energy when traffic is low. Using real mobile network datasets from multiple sites, it develops accurate traffic-prediction models that capture both short-term spikes and long-term usage patterns. The research designs and tests a novel framework called PESBiU (Predictive Energy Saver for Baseband Units). It integrates advanced machine learning (CNN-LSTM) for traffic prediction with reinforcement learning algorithms (such as DQN and DDDQN) to determine when to switch BBUs to lower-power states without degrading service quality. The results show up to 40% reduction in BBU energy consumption, while keeping user throughput and latency within acceptable limits. By comparing multiple approaches-static rules, threshold methods, and different RL variants - the thesis underscores the importance of robust traffic forecasting and real-time adaptive control. Practical fallback and hybrid strategies further strengthen reliability under rapidly changing conditions. The findings emphasize the potential of AI-driven, data-focused methods for optimizing energy consumption in cellular networks. They also provide a stepping stone for operators and researchers to scale these techniques, addressing both environmental sustainability and the growing demands of 5G+ communications.

KEYWORDS

5G+ Networks, Baseband Units (BBUs), Predictive Energy Saver for Baseband Units (PESBiU), Energy Optimization, Traffic Prediction, Machine Learning (ML), Reinforcement Learning (RL), CNN-LSTM, Dueling Double Deep Q-Network (DDDQN), Real-world Dataset, BBU Sleep Mode, Quality of Service (QoS), Dynamic Resource Allocation

ABSTRAKT

Tato disertační práce se zabývá vysokými energetickými náklady v sítích 5G a budoucích generací, se zaměřením na základnové jednotky (BBU), které jsou klíčovými komponentami zajišťujícími zpracování signálu, avšak při nízkém provozu často zbytečně spotřebovávají energii. Na základě reálných dat z mobilních sítí z několika lokalit jsou navrženy přesné modely pro predikci provozu, které zachycují jak krátkodobé špičky, tak dlouhodobé vzorce. Výzkum představuje nový rámec s názvem PESBiU (Predictive Energy Saver for Baseband Units), který kombinuje pokročilé strojové učení (CNN-LSTM) pro predikci provozu s algoritmy zpětnovazebního učení (např. DQN a DDDQN) pro rozhodování, kdy přepínat BBU do úsporných režimů bez zhoršení kvality služeb. Výsledky ukazují úsporu energie až o 40% při zachování přijatelných hodnot propustnosti a latence. Srovnáním různých přístupů - statických pravidel, metod s prahovými hodnotami a různých variant RL - práce zdůrazňuje význam robustní predikce provozu a adaptivního řízení v reálném čase. Praktické záložní mechanismy a hybridní strategie dále zvyšují spolehlivost při rychle se měnících podmínkách. Závěr podtrhuje potenciál metod založených na umělé inteligenci a práci s daty pro optimalizaci spotřeby energie v mobilních sítích. Výsledky zároveň poskytují základ pro další škálování těchto technik ze strany operátorů a výzkumníků - směrem k udržitelnějším a výkonnějším sítím 5G+.

KLÍČOVÁ SLOVA

5G+ sítě, základnové jednotky (BBU), PESBiU, optimalizace spotřeby energie, predikce provozu, strojové učení (ML), zpětnovazební učení (RL), CNN-LSTM, DDDQN, reálná data, režim spánku BBU, kvalita služby (QoS), dynamické přidělování prostředků

KOLÁČKOVÁ, Aneta. *AI-Aided and Data-Driven Energy Savings Optimization for Baseband Units in Cellular Networks*. Doctoral thesis. Brno: Brno University of Technology, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2025. Advised by doc. Ing. Jan Jeřábek, Ph.D.

Author's Declaration

Author:	Ing. Aneta Koláčková
Author's ID:	173677
Paper type:	Doctoral thesis
Academic year:	2024/25
Topic:	AI-Aided and Data-Driven Energy Savings Optimization for Baseband Units in Cellular Networks

I declare that I have written this thesis independently, under the guidance of the Advisor and using exclusively the technical references and other sources of information cited in the thesis and listed in the comprehensive bibliography at the end of the thesis.

As the author, I furthermore declare that, with respect to the writing of this thesis, I have not infringed any copyright or violated anyone's personal and/or ownership rights. In this context, I am fully aware of the consequences of breaking Regulation § 11 of the Copyright Act No. 121/2000 Coll. of the Czech Republic, as amended, and of any breach of rights related to intellectual property or introduced within amendments to relevant Acts such as the Intellectual Property Act or the Criminal Code, Act No. 40/2009 Coll. of the Czech Republic, Section 2, Head VI, Part 4.

Brno ....

Author's signature*

*The author signs only in the printed version.

ACKNOWLEDGEMENT

The research work described in this thesis was performed at the Department of Telecommunications, Faculty of Electrical Engineering and Communication, Brno University of Technology over the years 2019-2025. This manuscript is to the best of my knowledge original, and neither this nor a substantially similar dissertation thesis has been submitted at any other university. I would like to express my sincere gratitude to my supervisors, Assoc. Prof. Jan Jeřábek Ph.D., and Assoc. Prof. Jiří Hošek Ph.D., for their invaluable support, professional guidance, and continuous encouragement throughout my doctoral research. Their insights and patience greatly contributed to the successful completion of this work. My appreciation also extends to Mgr. Markéta Brüstlova Ph.D., whose valuable input and constructive suggestions significantly improved the readability and clarity of this thesis. Finally, I would like to thank my family, in particular my parents, and my friends for their unwavering support and understanding throughout the entire Ph.D. journey.

Contents

List of Figures

List of Tables

1 Introduction

Artificial intelligence (AI) is transforming many aspects of communication engineering, particularly as networks evolve to accommodate rapidly increasing data demands. The complexities of 5G and beyond (5G+) networks and emerging 6G technologies call for solutions that can efficiently manage resources while preserving service quality. Against this backdrop, this thesis investigates how AI-aided approaches can be leveraged to optimize operational efficiency and sustainability in the Radio Access Network (RAN), focusing on practical implementations and robust evaluations with real-world data.

This thesis explores established conceptual and methodological frameworks [1], expanding them with additional experimental evidence and advanced algorithms. By integrating granular data-driven analysis, novel traffic prediction models, and reinforcement learning (RL) techniques, the aim of this research is to advance energy-saving measures for underexplored network components. Each chapter addresses a vital facet of such components, from data collection and analysis to the design and validation of AI-aided frameworks, culminating in contributions that push the boundaries of intelligent energy management in modern telecommunications.

The findings documented in this thesis not only underscore the promise of next-generation network architectures but also emphasize the feasibility of deploying AI-based optimizations at scale. Through iterative experimentation, simulation, and comparative analysis, this thesis illustrates how intelligent techniques can adapt to diverse traffic conditions, mitigate energy consumption, and ensure resilience in operational environments. Ultimately, this research seeks to provide both a theoretical foundation and a practical pathway for network operators and academic researchers striving to meet future connectivity challenges in a sustainable manner.

1.1 Context and Rationale

Energy consumption by AI is a major topic of discussion highlighting a need for sustainable solutions. Likewise, efficient management has become a strategic priority in modern mobile communications, particularly as the radio access segment continues to dominate the overall power consumption profiles in cellular networks [2]. Baseband Units (BBUs) handle computationally intensive signal processing tasks, such as wide-bandwidth operations and advanced technologies (e.g. massive Multiple-input multiple-output(MIMO)), thereby driving their energy consumption disproportionately high. Although they typically account for about 10–20% of a base station's total consumption, the air conditioning of these devices represents 40% of total power consumption, and yet BBUs are less explored in energy optimization research compared to other network components, creating an important yet under-addressed challenge [3].

Conventional energy-saving methods, often built on static heuristics, can falter when confronted with shifting usage patterns, adversely affecting Quality of Service (QoS). Consequently, there is a growing imperative to develop adaptive solutions that respond

dynamically to network changes. Recent advances in AI and machine learning (ML) underscore their potential to enable predictive management of BBUs [4], providing both transparency in decision-making [5, 6] and responsiveness to real-time traffic conditions. In light of the above, the present research examines how AI-aided techniques can substantially reduce BBU energy consumption while preserving critical performance benchmarks - thereby offering actionable insights for both academic research and real-world deployments in evolving 5G+ networks.

1.2 Research Objectives

Building on the challenges and rationale outlined above, this thesis aims to address energy efficiency and operational optimization of 5G+ networks using advanced intelligent techniques. The objectives are rooted in leveraging real-world datasets, exploring predictive models, and implementing innovative approaches to improve network performance and sustainability. The primary research objectives are as follows:

1. Data-Driven Traffic Prediction:
    - Utilize real-world datasets to predict traffic patterns, leveraging ML models such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU).
    - Analyze the effectiveness of hybrid datasets in improving the accuracy of traffic volume predictions across diverse network environments.
2. Comprehensive Dataset Analysis and Utilization:
    - Analyze two distinct datasets: a long-term dataset covering hourly metrics from multiple sites and a granular dataset capturing high-frequency data at 15-minute intervals.
    - Identify and address challenges in combining heterogeneous datasets, including interpolation and scaling issues, to enhance prediction accuracy.
3. Energy Consumption Optimization for BBUs:
    - Enhance and evaluate predictive algorithms to minimize energy consumption of BBUs, focusing on dynamic resource allocation and sleep mode transitions without compromising network performance.
    - Implement, create and compare reinforcement learning techniques, including Deep Q-Network (DQN), Dueling Double DQN (DDDQN), and Advantage Actor Critic (A2C), to enhance decision-making for energy management.
    - Develop hybrid models combining multiple AI-aided approaches, such as supervised learning, reinforcement learning, and transfer learning, to address complex optimization problems in 5G+ networks.
4. Integration, Validation and Practical Deployment:
    - Design framework and test energy-efficient algorithms for centralized architectures like Cloud Radio Access Networks (C-RAN).
    - Evaluate the impact of virtualization and dynamic resource management on energy consumption reduction using numerical simulations and performance metrics.

Through the above research objectives, this thesis aims to broaden the understanding

of integrating AI-aided techniques into modern telecommunication networks, that address both theoretical and practical aspects of energy efficiency and network optimization.

1.3 Structure of the Thesis

Each chapter of this thesis addresses important aspects of research on optimizing 5G+ network efficiency using intelligent techniques.

The present chapter 1 outlines the context and rationale of the research, emphasizing the growing demands on network infrastructure, in particular in terms of energy efficiency in 5G+ networks. It also sets forth the research objectives and provides the methodology used.

Chapter 2 provides a review of 5G network technologies, focusing on their architecture, energy efficiency strategies, and the role of AI-aided techniques in network optimization. It establishes the core concepts or fundamental principles that are necessary for the subsequent chapters.

Chapter 3 focuses on data collection and analysis. The chapter details the primary and secondary datasets used, highlights key metrics such as traffic volume and energy consumption, and describes the development of ML models (Stage 1) for traffic prediction and performance evaluation.

Chapter 4 presents AI-aided solutions for managing energy consumption in Baseband Units (BBUs). It introduces the Predictive Energy Saver for Baseband Units (PESBiU) framework and evaluates several algorithms (Stage 2), including reinforcement learning techniques, for optimizing energy usage.

Finally, chapter 5 provides a detailed analysis of the results, comparing various algorithms' performance in energy efficiency, latency reduction, and overall network optimization. It also discusses the broader implications of the findings and the potential for hybrid model integration.

Each chapter builds on the previous one, presenting a cohesive and logical progression through the research. Together, they aim to provide a comprehensive understanding of how AI-aided technology can transform the optimization of 5G network performance and energy efficiency.

The Conclusion summarizes the key findings of the research, evaluates the effectiveness of the proposed approaches, and offers recommendations for future research on 5G+ network optimization.

2 Technology Overview of 5G+ Networks

Innovations in 5G+ networks drive improvements in mobile communications by introducing advanced architectures, hardware, and AI-based optimization techniques. This chapter examines how technologies like C-RAN, energy-saving strategies, and dynamic resource management contribute to enhanced performance and sustainability. This chapter outlines the key technological advancements and deployment challenges of next-generation networks.

2.1 Overview of 5G Network Technologies

The initial vision for 5G networks promised significant advancements in wireless communication, including enhanced mobile broadband (eMBB), ultra-reliable low-latency communications (URLLC), and massive machine-type communications (mMTC). These were expected to enable next-generation applications such as augmented reality (AR), virtual reality (VR), autonomous vehicles, and large-scale Internet of Things (IoT) deployments [7]. However, in practice, the widespread rollout of 5G has faced several limitations, including the high cost of infrastructure, limited real-world improvements over 4G LTE in many areas, and ongoing challenges in spectrum efficiency and energy consumption.

While 5G does provide some improvements in network capacity and lower latency, its real-world impact has been more incremental than transformative. eMBB enables higher data rates, supporting applications like HD streaming and cloud gaming, but the actual user experience often depends on deployment density and available spectrum [8]. URLLC, which was expected to revolutionize mission-critical applications like remote surgery and autonomous driving, has struggled due to inconsistent latency performance outside controlled environments [9]. Similarly, mMTC has yet to deliver its full potential, with IoT adoption still largely relying on existing 4G LTE and LPWAN technologies such as NB-IoT and LoRa [10].

From a technical perspective, 5G integrates multiple advancements, but each comes with trade-offs. Millimeter-wave (mmWave) frequencies provide high bandwidth but suffer from short-range coverage and significant propagation loss, making widespread deployment expensive and impractical outside urban areas [11]. Massive MIMO improves spectral efficiency but significantly increases power consumption [12]. Network slicing enables flexible resource allocation, but its real-world implementation has been slower than anticipated due to operational complexities and interoperability issues [13]. As the industry moves forward, the focus is shifting beyond 5G and towards 6G, where AI-driven network optimization, improved energy efficiency, and more adaptive architectures will play a key role [5].

Nevertheless, 5G network architecture was designed to be highly flexible and efficient, comprising the Radio Access Network (RAN), the Core Network (5GC), and the Transport Network. The 5G RAN includes New Radio (NR) interfaces and technologies that enhance connectivity and coverage through advanced techniques like beamforming and carrier aggregation, improving signal quality and data rates [14]. The 5GC employs

a Service-Based Architecture (SBA) with a cloud-native design, providing greater flexibility, scalability, and efficient resource management [15].

Energy efficiency is a critical consideration in 5G+ network design and deployment. The increasing number of connected devices and higher data rates necessitate efficient energy management to ensure sustainable network operations. Advanced sleep modes for base stations and network components allow them to enter low-power states during periods of low traffic, significantly reducing energy consumption [16]. Additionally, intelligent resource management and dynamic adjustment of transmission power, guided by machine learning algorithms, optimize energy usage based on network conditions [17]. Edge computing and network slicing aim to further contribute to energy efficiency by reducing the need for long-distance data transmission and enabling localized processing, thus lowering latency and energy consumption [12].

Despite the numerous advantages, 5G+ technology faces challenges that need to be addressed. Spectrum availability remains a critical issue, requiring efficient management and sharing strategies to utilize available spectrum effectively [18]. The high infrastructure costs associated with deploying current 5G, particularly in urban areas, necessitate innovative financing models and cost-effective deployment strategies [19].

In conclusion, current 5G network technology offers advancements in speed, capacity, and reliability, enabling a wide range of new applications and services. However, to fully realize the potential of 5G+, continuous research and development efforts are essential to enhance network performance, expand coverage, and address the existing challenges.

2.2 Overcoming Key Limitations for 5G+ Network Efficiency

Although 5G introduced speed and capacity enhancements, real-world deployment has exposed critical inefficiencies. The shift towards 5G+ networks highlights the need for intelligent resource management, energy optimization, and seamless integration with legacy systems. This section examines the key challenges that hinder 5G's full potential and outlines strategies essential for making 5G+ truly effective.

Energy Consumption

One of the primary challenges in 5G networks is managing the energy consumption of network components, particularly the BBUs and RRUs [2]. As 5G networks support a massive number of devices and higher data rates, the energy demand increases significantly [3]. Traditional methods of energy management are inadequate for the dynamic and dense environments of 5G. This issue is expected to intensify further with the advent of 6G networks, which will introduce even more demanding use cases, such as holographic communications and pervasive AI-driven applications, requiring significantly higher energy efficiency strategies to ensure sustainable operations. Advanced strategies, such as employing ML for predictive energy saving, are necessary to reduce energy consumption without compromising performance [20, 21]. For instance, energy-efficient strategies for

BBUs and RRUs based on mixed statistical models have proven to be promising in reducing power consumption [22].

The shift to centralized architectures such as Cloud Radio Access Networks (C-RAN) introduces complexities but also enhances energy efficiency [23]. In C-RAN, BBUs are pooled together to serve multiple RRUs, which can lead to significant energy savings through centralized processing, see Fig. 2.1. In this way, BBUs allow dynamic resource allocation based on traffic demands [22]. However, the dynamic allocation of resources and the need for real-time energy management require sophisticated algorithms. Delay-aware energy-saving strategies, which consider both the processing delay and the energy consumption, have been envisaged to address these challenges effectively [22].

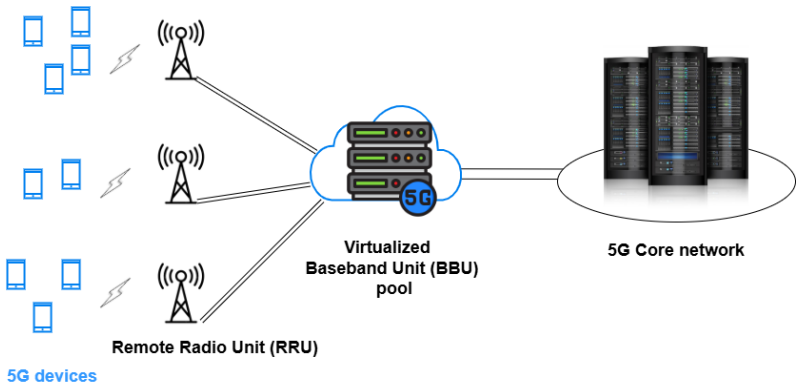


Fig. 2.1: Cloud Radio Access Network (C-RAN) architecture

Latency and Reliability

Achieving URLLC is another critical challenge. Applications such as autonomous driving, remote surgery, and industrial automation demand extremely low latency and extremely high reliability [24, 25]. The inherent variability in wireless communication environments, coupled with the need for seamless handovers and real-time data processing, makes it difficult to consistently meet these stringent requirements. Innovative approaches, such as edge computing and advanced network slicing, are being explored to bring computational resources closer to the user, thereby reducing latency [13].

Network Management

The increased complexity of 5G+ networks, characterized by heterogeneous network elements and dense deployments, requires advanced management techniques. Dynamic and flexible management strategies are required to handle varying traffic loads and ensure QoS. AI-aided technologies are playing an increasingly important role in optimizing these processes, providing predictive insights and automated adjustments based on real-time network conditions [26].

Heuristic-assisted deep reinforcement learning (DRL) has been shown to be effective in optimizing BBU aggregation in C-RAN, ensuring both energy efficiency and QoS. By leveraging AI techniques, these solutions can dynamically adjust resources in response

to changing network conditions, thereby maintaining optimal performance [26]. Moreover, monitoring tools and methodologies, like the netcalculator developed by the Czech Telecommunications Office, contribute to evaluating and optimizing network capacity and performance, ensuring that management decisions are based on accurate and comprehensive data [27].

Network Resource Utilization and Optimization

Efficient utilization of network resources, bandwidth, and infrastructure is essential for the successful deployment and operation of 5G networks. In a 5G environment, optimizing the allocation of resources involves a comprehensive approach that addresses capacity planning and infrastructure enhancements. Effective capacity planning ensures that the network can meet current demands while being scalable for future needs. By predicting user behavior and traffic patterns, network operators can allocate resources strategically to avoid congestion and ensure seamless service delivery [27].

Bandwidth optimization involves not only managing the available spectrum efficiently, but also implementing technologies such as carrier aggregation [28], massive MIMO, and beamforming to maximize data throughput. These technologies enable the network to support higher data rates and more users simultaneously by utilizing the spectrum more effectively. Additionally, network slicing allows for the creation of virtual networks tailored to specific applications, ensuring that critical services receive the necessary bandwidth and resources without interference from other traffic [13].

Infrastructure optimization focuses on enhancing the physical and logical components of the network to improve performance and reduce costs. This includes upgrading hardware components such as base stations and routers, as well as implementing software-defined networking (SDN) and network functions virtualization (NFV) to create a more flexible and programmable network environment [29]. By decoupling the network functions from the underlying hardware, SDN and NFV enable more efficient management and scaling of network resources [30].

Overall, the combination of strategic capacity planning, bandwidth optimization, and infrastructure enhancement creates a robust framework for optimizing network resources. These efforts ensure that the network can handle increased traffic demands, provide high-quality service, and adapt to future technological advancements and user needs.

Interoperability and Integration with Legacy Systems

The transition to 5G and even to 5G+ involves integrating new technologies with existing 4G and legacy systems, which presents significant challenges. Ensuring seamless interoperability and backward compatibility is crucial for a smooth transition and widespread adoption of 5G. This integration requires upgrades to existing infrastructure, which can be costly and complex. Additionally, maintaining consistent performance across different network generations and technologies necessitates robust management frameworks and protocols [23].

Virtualized BBU pools in C-RAN environments facilitate the integration of 5G with legacy systems by providing flexible and scalable processing resources. These pools can dynamically allocate resources to different network generations, ensuring consistent performance and efficient resource utilization [22, 23, 23].

In conclusion, while 5G networks offer substantial improvements over previous generations, achieving optimal network efficiency requires overcoming significant challenges. Energy management, latency reduction, effective network management, spectrum utilization and integration with legacy systems are critical areas that need continuous innovation and improvement. Addressing these challenges is essential for the successful deployment and operation of 5G networks, enabling them to meet the evolving demands of modern applications and services. However, due to the inherent complexity and broad scope of these challenges, this thesis focuses specifically on energy efficiency. By narrowing its scope, it aims to deliver in-depth insights and practical solutions for optimizing energy usage, recognizing that comprehensively addressing all identified issues exceeds the thesis's capacity.

2.3 Energy Efficiency in Current 5G Networks

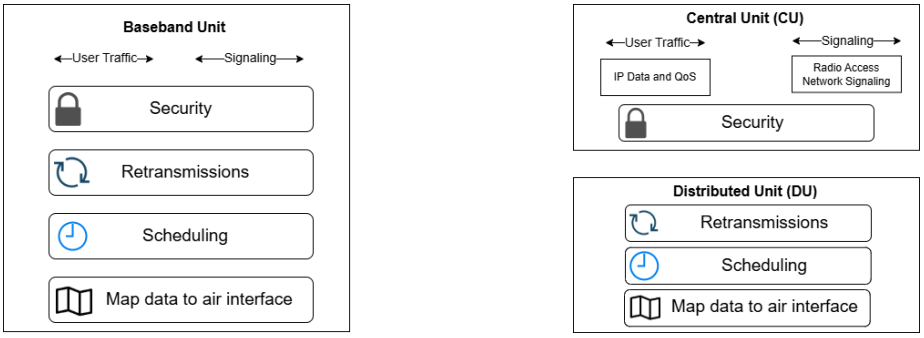
Energy efficiency is an essential concern in 5G networks due to the significant increase in data traffic and the need for sustainable operations. This section discusses the energy efficiency strategies in both BBUs and RRUs, which are indispensable components of 5G infrastructure and which are responsible for most of the energy consumption.

2.3.1 Baseband Units (BBUs)

BBUs are critical components in cellular networks, responsible for processing baseband signals and managing communication between the radio network and the core network. In 4G, BBU worked as one unified unit that was connected with RRU with fronthaul connection and when deployed, the BBU+RRU were optimized for throughtput and capacity in the network. Typically, it was deployed with purpose-build, pre-packaged hardware and sotware, which limited deployment options in many ways and created long upgrade cycles. With 5G, there is the option to break BBUs unit functions into more manageable components: Distributed Unit (DU) and Control Unit (CU) with open interfaces, which open the midhaul interface (between DU and CU) allowing different vendors to provide and integrate DU and CU components. This means operators can mix and match equipment from various manufacturers, such as Nokia, Ericsson, and Huawei, to optimize performance and cost-efficiency in their networks. This option for BBUs is in Fig. 2.2 [1].

This disaggregation and initiative from O-RAN alliance (Open-RAN) allow us to consider different possibilities in this field [31]. More intelligent, virtualized and interoperable RAN means that the software is independent from the hardware it runs on.

The CU and DU work together to manage user traffic and signaling in modern cellular networks, each with specific responsibilities that optimize performance and security.



(a) Old 4G version of BBU

(b) Separate 5G BBU functions

Fig. 2.2: Changes in BBUs with 5G technologies

The CU primarily handles IP data and QoS management, ensuring that user traffic is securely and efficiently processed. It also manages the signaling for the RAN, coordinating essential communication protocols between UE and the core network to maintain seamless connectivity. Security functions at the CU level protect data across various network layers, providing a safe experience for users [1].

On the other hand, the DU is responsible for real-time network operations that are crucial for maintaining performance and data integrity. It handles retransmissions to reduce data loss, and to ensure reliable communication, and oversees scheduling, which allocates network resources effectively to balance data flow. Additionally, the DU maps data to the air interface, enabling efficient radio transmission to the UE. However, even in some 5G deployment options, the CU and DU can still be aggregated together to do the same process in one unit.

In C-RAN, the centralization enables more efficient use of processing power and facilitates the implementation of advanced energy-saving techniques. But there are still some challenges: BBU usually must be located less than 15km from the RRU, so the delay is less than 100 µs by fiber run [1].

Energy Consumption Patterns

With the advent of 5G networks, the energy efficiency of BBUs has also become increasingly important due to higher data rates, increased number of connected devices, energy prices and the need for more complex signal processing. This section examines energy consumption patterns, and discusses current optimization techniques aimed at enhancing energy efficiency.

While BBUs typically account for about 10-20% of a base station's total energy consumption, their energy demands have increased with the advent of 5G networks [3]. According to Whitepaper [32], current BBUs are increasingly being virtualized to enable flexible and scalable cloud-based RAN deployments. However, such virtualization, despite its operational benefits, can lead to higher power consumption due to the inherent inefficiencies of general-purpose servers compared to dedicated hardware. As future network generations (5G+ and 6G) demand even more advanced digital processing capabilities, virtualized BBU energy consumption is likely to further increase, making it essential

to investigate strategies for achieving better BBU energy efficiency.

Moreover, recent literature supports this observed trend. The study "Experimental Evaluation of Power Consumption in Virtualized Base Stations" [33] presents detailed measurements of CPU power use, clearly showing that BBU power consumption increases significantly with the introduction of 5G virtualization technologies. The NGMN Alliance Whitepaper [34], "Metering in Virtualised RAN Infrastructure", further emphasizes this challenge by showing how power consumption scales notably with increased network loads in virtualized BBUs. The documented growth in energy demand highlights a practical concern: without targeted improvements in energy efficiency, the operational costs and environmental impacts of future 5G+ and 6G networks will rise significantly.

The energy consumption patterns of BBUs are influenced by various factors, including traffic load and operational states (active, idle, or sleep modes). During peak traffic periods, BBUs operate at full capacity, consuming the maximum amount of energy due to the high processing demands. Conversely, during off-peak periods, the energy consumption can be reduced by transitioning BBUs into lower power states. However, the ability to efficiently manage these transitions is critical to maintaining overall network performance and energy efficiency.

The introduction of 5G technologies has further complicated the energy consumption patterns of BBUs. The need to support higher data rates, increased user density, and more complex signal processing algorithms has led to a substantial increase in the power requirements of BBUs. Furthermore, the use of higher frequency bands in 5G, such as mmWave frequencies, requires more power-intensive signal processing techniques, further exacerbating the energy consumption challenges [35].

2.3.2 Remote Radio Units (RRUs)

RRUs are another critical component of 5G networks, responsible for transmitting and receiving radio signals to and from user equipment. The energy efficiency of RRUs is vital for the overall sustainability of the network. This section examines the architecture and functionality of RRUs, their energy consumption patterns, and current optimization techniques.

RRUs are deployed at cell sites and connected to BBUs, forming the radio part of the network. The architecture of RRUs includes power amplifiers, transceivers, antennas, and various signal processing components. RRUs handle the conversion of digital signals from BBUs into radio signals and vice versa, enabling wireless communication with user equipment. The functionality of RRUs involves transmitting and receiving data packets, managing signal modulation and demodulation, and ensuring efficient use of the radio spectrum. RRUs are designed to support multiple frequency bands and advanced technologies such as massive MIMO, which enhances the capacity and coverage of the network [4].

Possibility to split the RAN functions into the RU, DU, and CU (and possibly CU-User Plane (UP) and CU-Control Plane (CP)) gives the network operator a great deal of flexibility to deploy more intelligent functions to meet service and RAN performance

needs. Since the CU-CP does not have strict latency requirements, it can be centralized for resource sharing across many cells [1]. This benefit - the resource sharing across multiple cells - is sought in this thesis. The deployment examples are shown in Fig. 2.3.

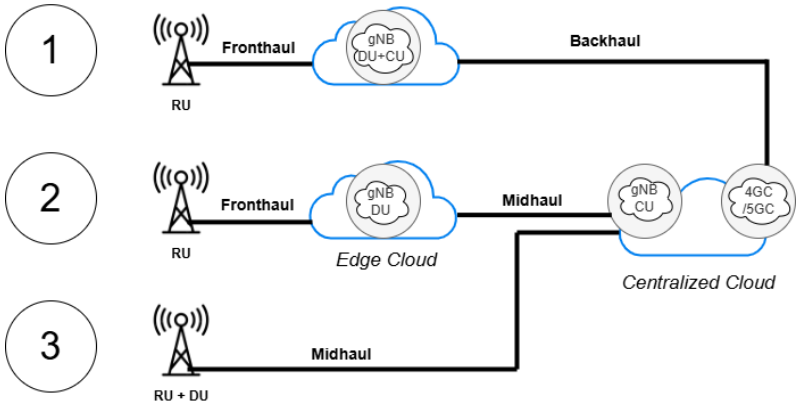


Fig. 2.3: Deployment examples for split RAN

Example 1, in Fig. 2.3, is unsplit 5G BBU leveraging existing 4G infrastructure. Example 2 optimizes the trade-off between round trip delay and efficient RAN resource sharing across multiple sites by strategically distributing processing functions. This approach ensures that latency remains within acceptable limits while allowing resource pooling, leading to improved spectral efficiency and cost savings, which is the case we consider for the real deployment of our solution. Example 3 achieves the best round-trip delay performance and requires the least transport bandwidth by keeping the DU colocated with the Radio Unit (RU) at the cell site while centralizing the CU and 5GC in the centralized cloud. This reduces fronthaul constraints, as midhaul transmission is more efficient, leading to lower overall latency. Also it minimizes the need for data transmission over fronthaul and midhaul links, reducing latency and dependency on high-capacity transport networks [1]. Our solution can be applied here as well.

Energy Consumption Patterns

The energy consumption of RRUs is influenced by several factors, including transmission power, signal processing requirements, and operational states. During peak usage times, RRUs consume more power due to the increased demand for data transmission and reception. Conversely, during off-peak times, there is potential to reduce energy consumption by transitioning RRUs into lower power states. In 5G networks, the need for higher data rates and increased user capacity has led to a significant rise in the energy consumption of RRUs. Moreover, advanced technologies such as massive MIMO, while improving network performance, also contribute to higher energy usage due to the increased number of antennas and associated signal processing tasks [4].

2.3.3 Current Optimization Techniques for BBUs and RRUs

In 5G networks, both BBUs and RRUs play an essential role in ensuring high data rates and reliable connectivity. However, these components also account for a significant portion

of the network's energy consumption. Over the years, various optimization strategies have been proposed to reduce their power usage without compromising performance. The following techniques highlight current approaches for improving the energy efficiency of BBUs and RRUs:

- **Dynamic Resource Allocation (primarily for BBUs):** Dynamic resource allocation involves adjusting the processing resources of BBUs in real-time based on the current traffic load and network conditions. By dynamically scaling the computational resources allocated to different tasks, BBUs can operate more efficiently during periods of low demand, thereby reducing energy consumption. This technique is particularly effective in C-RAN architectures, where centralized BBU pools can allocate resources to multiple RRUs as needed [4, 22, 35].
- **Advanced Sleep Modes (for BBUs and RRUs):** Implementing advanced sleep modes allows both BBUs and RRUs to transition into lower power states during periods of inactivity. For BBUs, these modes can range from partial sleep, where certain components are powered down, to deep sleep, where the BBU is almost completely shut down. RRUs similarly benefit from partial or full power-down states during low-demand intervals. The challenge lies in effectively managing these transitions to ensure quick resumption of full operation when needed [4, 22, 35, 36].
- **Machine Learning and Predictive Analytics (for BBUs and RRUs):** ML and predictive analytics are increasingly used to enhance the energy efficiency of both BBUs and RRUs. By analyzing historical traffic patterns and real-time network data, ML algorithms can predict future loads and adjust operational states accordingly. This predictive approach enables proactive management of resources, cutting unnecessary power usage during low traffic periods while ensuring adequate processing power during peak times. For RRUs specifically, AI-driven methods can dynamically adjust RRU configuration and deployment to further optimize energy consumption [6, 35, 36].
- **Virtualization and Network Slicing (mainly for BBUs):** Virtualization and network slicing allow for more flexible and efficient use of BBU resources. By creating virtual instances of BBUs, network operators can dynamically allocate processing power to different network slices based on current demand. This flexibility optimizes resource use, reduces energy consumption, and improves overall network efficiency. Network slicing also enables the isolation of different traffic types, allowing more tailored energy-saving strategies per slice [4, 22, 23].
- **Adaptive Power Control (mainly for RRUs):** Adaptive power control techniques modulate the transmission power of RRUs based on real-time network conditions and user demand. By reducing transmission power during periods of low demand or when users are physically close to the antenna, the system can minimize interference and significantly decrease energy consumption [36].
- **Energy-Efficient Hardware (for RRUs):** Developing and deploying energy-efficient hardware components for RRUs is a critical strategy. This includes using power-efficient transceivers, amplifiers, and signal processing units. Advances in semiconductor technology and circuit design are enabling the creation of RRUs that

consume less power while maintaining high performance [35].

In conclusion, reducing the energy consumption of both BBUs and RRUs is vital for sustainable 5G network operations. In this thesis, however, we focus specifically on the energy optimization of BBUs. While RRUs certainly require energy-efficient solutions, a substantial body of research has already addressed fundamental methods for managing RRU power, hardware design, and adaptive techniques. By contrast, BBUs are often less explored yet handle complex and computationally intensive tasks in 5G (e.g., wide-band signal processing and scheduling) and thus represent a particularly promising target for advanced optimization. Concentrating on BBUs allows us to explore more deeply sophisticated mechanisms such as dynamic resource allocation and machine learning–driven energy management, aiming to bridge an important gap in current research on sustainable and high-performance 5G+ network infrastructures.

2.4 AI-Aided Technologies

The evolution of 5G and 5G+ networks demands enhanced operational efficiency, scalability, and sustainability. This section explores the state-of-the-art of AI-aided technologies in 5G+ network optimization, with a particular focus on energy efficiency and resource management.

AI-aided techniques have fundamentally transformed the operational paradigms of mobile communication networks. By leveraging their ability to analyze vast datasets and derive actionable insights, these technologies have become integral to network management and optimization. In the RAN, for instance, ML algorithms are employed to optimize scheduling, mitigate interference, and allocate resources dynamically based on real-time conditions. Similarly, the core network benefits from AI-driven insights into user mobility patterns, security risks, and session flows, enabling predictive maintenance and proactive optimization [37].

Applications of AI-aided techniques extend beyond network management into areas such as QoS enhancements and user experience improvements. For example, video streaming services can utilize ML algorithms to predict network congestion and adapt streaming quality accordingly. These innovations not only improve user satisfaction, but also contribute to more efficient resource utilization, a critical factor in achieving sustainability in 5G+ networks [38].

AI-aided technologies combine two elements: learning approaches and algorithm types. Learning approaches describe how they improve by using data. AI/ML algorithms are based on these methods [39].

2.4.1 Learning Paradigms in AI/ML

AI/ML algorithms differ according to how they learn based on data and make decisions. These types of learning determine the effectiveness of different models in optimizing 5G+ networks. Understanding these paradigms is essential for selecting the appropriate models for tasks such as mobility prediction, anomaly detection, or resource allocation.

a) Supervised Learning

Supervised learning algorithms are trained on labeled datasets, where the desired output is known for each input. They are instrumental in solving classification and regression problems. In the context of 5G+ networks, supervised learning aids in tasks such as traffic prediction, resource allocation, and QoS estimation. Classification problems might involve predicting discrete categorical values, such as identifying types of network traffic or detecting anomalies, while regression problems could focus on forecasting continuous values such as future network load or user mobility patterns [39].

b) Unsupervised Learning

Unlike supervised learning, unsupervised learning operates on unlabeled data to discover hidden patterns. It is primarily used for clustering and reducing dimensionality. In 5G+ networks, clustering can help segment network users or identify common usage patterns, whereas dimensionality reduction is crucial for handling high-dimensional network data by reducing the number of features while retaining essential information. Applications of unsupervised learning include anomaly detection, user behavior analysis, and network topology optimization [39].

c) Semi-Supervised Learning

Semi-supervised learning combines labeled and unlabeled data, which is particularly beneficial when labeled data is scarce or expensive to obtain. This approach enhances model performance in tasks such as network intrusion detection and fault diagnosis by leveraging the available labeled data alongside the abundant unlabeled data [39, 40].

d) Self-Supervised Learning

Self-supervised learning leverages intrinsic structures within the data to generate supervisory signals without requiring manually labeled examples. In this paradigm, the model is tasked with predicting a missing or corrupted part of the input from its remaining context. For instance, generative language models are commonly pre-trained by learning to predict the next word in a sentence, while diffusion models for image generation learn to reverse a noise-injection process. Such self-supervised learning objectives have proven to be particularly effective in Generative AI (GenAI) technologies, as they enable the training of models on vast amounts of unlabeled data and help in learning rich, transferable representations [40, 41]. In the context of 5G+ networks, self-supervised learning can support the synthesis of realistic network scenarios and augment limited datasets for simulation, thereby enhancing the robustness of optimization algorithms [39].

e) Reinforcement Learning

RL involves an agent learning optimal actions through trial and error interactions with the environment, aiming to maximize cumulative rewards. This paradigm is highly relevant for dynamic resource management and handover decisions in 5G+ networks. Al-

gorithms like Q-Learning, Deep Q-Networks (DQN), and Advantage Actor-Critic (A2C) are prominent in optimizing network operations by continuously adapting to changing conditions [39].

f) Transfer Learning

Transfer Learning leverages knowledge from one problem domain to improve learning in a related domain. In 5G+ networks, models trained in one scenario (e.g., urban environment) can be adapted to another scenario (e.g., suburban areas), thereby reducing training time and data requirements. This capability is essential for deploying AI/ML models across diverse network environments with varying characteristics [39].

g) Ensemble Learning

Ensemble learning combines multiple models to enhance performance and robustness. Techniques such as bagging, boosting, and stacking are commonly used for tasks such as traffic prediction, anomaly detection, and resource allocation. In the context of 5G+ networks, ensemble learning improve the accuracy and reliability of predictions for network optimization tasks by aggregating the strengths of individual models [39].

h) Online Learning

Online learning algorithms update the model incrementally as new data arrives, making them suitable for environments where data is continuously generated, such as network traffic monitoring. This approach allows models to adapt to changes over time without the need for retraining from scratch, thereby maintaining relevance and accuracy in dynamic network conditions [39].

i) Federated Learning

Federated Learning (FL) enables decentralized model training across multiple devices while keeping data localized, which is essential for privacy-preserving network optimization. In 5G+ networks, FL allows base stations and user equipment to collaboratively train a global model without sharing sensitive user data, thereby enhancing privacy and security [39].

2.4.2 Algorithmic Implementations of AI/ML

While the previous section addressed the theoretical paradigm of learning, this section explores the specific AI/ML and their practical implementations that operationalize these concepts in real-world 5G+ network scenarios. These algorithms are the computational engines that execute tasks such as prediction, classification, and decision-making, each with distinct architectural features and performance profiles.

a) Shallow Algorithms

Shallow algorithms, such as Linear Regression, Support Vector Machines (SVM), and Decision Trees (DT), are foundational ML techniques used for various prediction and classification tasks. For instance, Linear Regression is employed to predict continuous network parameters like bandwidth demand, while SVMs are effective for classification tasks such as intrusion detection. Decision Trees aid in decision-making processes related to network configurations. Additionally, ensemble methods like Random Forest and AdaBoost enhance the performance of these shallow algorithms by combining multiple models to improve accuracy and robustness [39].

b) Deep Learning Algorithms

Deep learning algorithms extend the capabilities of traditional neural networks by incorporating multiple hidden layers, allowing for the extraction of high-level features from raw data. Artificial Neural Networks (ANNs), Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs) are fundamental DL architectures used in 5G+ network optimization. ANNs serve as basic models for various prediction tasks, while CNNs are particularly useful for spatial data analysis, such as network topology optimization. RNNs, including their advanced variants like LSTM networks, are adept at handling sequential data, making them ideal for time-series traffic prediction and user mobility forecasting [39].

c) Reinforcement Learning Algorithms

RL algorithms form a class of methods that adjust network parameters through feedback-driven updates. They learn from operational performance and modify decision rules as conditions change. RL algorithms are categorized into value-based, policy-based, and hybrid methods. Value-based techniques like Q-Learning and DQN compute expected returns for discrete actions. Policy-based methods, such as REINFORCE, update decision policies without intermediate value estimation. Hybrid approaches, including A2C and Deep Deterministic Policy Gradient (DDPG), merge both techniques to handle continuous control tasks. This paradigm fits dynamic network scenarios that demand real-time adjustments. RL algorithms simulate network conditions and adapt to fluctuating traffic patterns. Their application covers resource allocation, interference control, and energy optimization. Learning from limited initial data makes these methods practical for evolving network settings [39].

d) Generative AI

An emerging paradigm is GenAI. Based primarily on unsupervised and self-supervised learning principles, generative models, such as Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs) - aim to synthesize novel data that emulates real-world distributions. This capability might be particularly valuable in network optimization, where synthetic data can augment limited or imbalanced datasets, support robust

simulation of network conditions, and aid in stress-testing various operational scenarios. Moreover, advanced generative models often integrate RL fine-tuning via RL from human feedback (RLHF) to align generated outputs with specific operational objectives. In addition to these architectures, Large Language Models (LLMs) have emerged as a powerful generative paradigm. Although originally designed for natural language processing, LLMs are trained via self-supervised objectives on vast corpora and can be adapted to generate and interpret complex data patterns - including network logs and operational scenarios [40–42].

2.4.3 Applications of AI/ML in 5G+ Network Optimization

The rollout of 5G and the evolution toward 5G+ networks have underscored the importance of intelligent and dynamic optimization methods. To address these complexities, recent research has shown a strong trend toward employing AI-aided techniques at various layers of the RAN [2,4,38,43]. In particular, AI/ML-based approaches offer the ability to predict short-term traffic loads accurately, learn from historical patterns, and dynamically allocate network resources, thereby improving energy efficiency and QoS.

Enhancing Energy Efficiency and Adaptability

Conventional heuristics and static threshold-based methods often fail to capture the intricate variations in traffic and user mobility patterns [21, 23, 44]. Hence, numerous studies have explored applying ML-driven algorithms to make RAN components more adaptive and resource-efficient. For example, hybrid LSTM and CNN models have demonstrated promise in predicting resource usage with enhanced accuracy [45, 46], while DRL agents have demonstrated the ability to autonomously learn optimal policies by interacting with their environment [47–49]. In particular, some works on DRL [50] and [51] extend this foundational principle to optimize network performance in 5G+ scenarios. While these studies focus on joint beam forming, power control, and energy savings at the network level, our research targets BBU energy optimization by integrating traffic prediction and more advanced DQN, thereby filling a gap in this area.

While DRL-based methods show promise, they must overcome several hurdles, including slow convergence, overestimation bias, and performance degradation if traffic spikes deviate significantly from predictions [37, 52]. Some frameworks have attempted to mitigate these issues by combining supervised learning traffic predictions with RL-based resource scheduling, achieving notable gains in both energy and cost efficiency [20, 53]. However, these studies often address broader network features like cell switching or carrier aggregation, whereas our focus is specifically on fine-grained BBU power states.

Overcoming Limitations of Heuristics via RL

Early BBU energy optimization methods relied on simple, rule-based heuristics - either putting entire sites into sleep mode during off-peak times or applying threshold-based triggers for partial deactivations [54–56]. While effective in specific scenarios, these static

approaches risk overshooting (or undershooting) actual operational demands. More recent works have employed DQN and its variants to refine energy-management decisions, mitigating the risk of suboptimal or unstable policies [57–59]. By continuously learning from real-time metrics, RL agents can effectively handle the multi-dimensional state space - encompassing traffic load, latency, and power consumption - and converge toward near-optimal strategies.

However, RL-based methods alone can suffer from slow training and sensitivity to rapidly changing network conditions. Hybrid approaches, such as combining CNN-LSTM-based traffic forecasting with DRL, help stabilize training by providing more accurate forecasts for the agent's state space [45, 46, 49]. Similar to these works, our approach integrates a hybrid CNN-LSTM model for traffic prediction with a DDDQN architecture that manages BBU power states dynamically. In contrast to existing solutions focusing on carrier aggregation or Open RAN slicing [43,57,58], we target the specific challenge of BBU energy optimization. By leveraging granular datasets and advanced RL algorithms, our method balances energy efficiency with QoS - an essential trade-off often overlooked in earlier heuristic-based approaches [39, 60].

Balancing Energy Efficiency and QoS in 5G+

Despite significant advancements, several challenges remain in striking an optimal balance between energy efficiency and QoS requirements. For instance, turning off too many BBUs during low demand can lead to service degradation when traffic surges unexpectedly [16, 39], while underutilization of sleep modes fails to reduce overall power consumption. RL-based frameworks address this by learning policies that adaptively respond to traffic fluctuations in near real-time, ensuring that resource availability aligns with demand [43]. Yet, the inherent complexity of DRL solutions, such as policy convergence and exploration-exploitation trade-offs, underscores the need for more robust approaches.

To bridge these gaps, recent studies have combined proactive traffic prediction with dynamic scaling of network resources. The work in [44] employed a mesh-based network partitioning approach to identify active cells based on predicted cell traffic, and [23] proposed bandwidth-aware virtualization to enhance energy savings. However, these methods often rely on threshold-based switching, making them susceptible to inaccuracies in prediction. In contrast, a properly designed DDDQN agent can autonomously learn a more nuanced, continuous mapping between predicted load and the optimal power state, refining its decisions over time.

Overall, the synergy between ML-driven prediction and RL-based decision-making represents a critical step forward in managing the escalating energy demands of 5G+ networks. In this work, we propose a methodology that fuses advanced forecasting (hybrid CNN-LSTM) with a novel DDDQN approach to overcome limitations of heuristic and classical RL techniques. This integrated framework aims to ensure that energy efficiency gains do not come at the expense of QoS, offering a more robust and adaptable solution for next-generation network deployments.

2.4.4 Challenges and Future Research

Despite the significant advantages AI/ML brings to 5G+ network optimization, several challenges persist that hinder their full potential. One of the primary issues is data availability and quality. Effective AI/ML models require large, high-quality datasets for training, yet in dynamic network environments, such data can be difficult to acquire or maintain. The variability and heterogeneity of 5G network data further exacerbate this challenge, often leading to incomplete or noisy datasets that degrade model performance [37]. Moreover, many current approaches rely on static or synthetic datasets, limiting their real-world applicability [39]. In contrast, our research leverages a real-world dataset to ensure that the proposed AI-aided solutions are firmly grounded in the operational demands of 5G+ networks.

A second notable challenge is the computational complexity associated with advanced AI/ML models. Deep learning architectures, while powerful, demand substantial computational resources, including high-performance hardware and energy-intensive operations [43]. This poses a limitation for deploying AI solutions in resource-constrained network elements, such as edge devices, small cells, or low-power base stations - where hardware capabilities are limited and power efficiency is paramount.

Model interpretability remains an additional barrier. Many deep learning–based techniques operate as "black boxes", making it difficult for network operators and stakeholders to understand or trust their decision-making processes [2]. Although the O-RAN Alliance has spearheaded efforts to standardize open interfaces that facilitate data exchange and enable xApps and rApps - AI-driven applications in Open RAN that optimize network operations, with xApps handling near real-time control tasks and rApps focusing on non-real-time strategic optimizations - to make more transparent decisions [49], achieving practical explainable AI (XAI) for network management is still an active area of research [6].

Privacy and security also demand careful attention. AI/ML models often process sensitive user data, raising risks of data breaches and misuse - particularly in decentralized architectures like federated learning, where local data is retained at edge devices [38]. Ensuring secure data transmission, model integrity, and compliance with regional data regulations (such as the General Data Protection Regulation (GDPR), Lei Geral de Proteção de Dados (LGPD) or the California Consumer Privacy Act (CCPA)) is critical to building trust in AI-driven network optimizations.

Given these challenges, several promising research directions stand out for future 5G+ deployments:

- **XAI**: Developing AI models, that offer transparent and interpretable outputs, can build trust and facilitate human-in-the-loop decision making. XAI techniques, such as attention mechanisms or layer-wise relevance propagation, could demystify model outputs without sacrificing performance [6].
- **Edge AI and Distributed Intelligence**: By deploying AI/ML models closer to the data source - on edge servers or local controllers - researchers can reduce latency and backhaul load, thereby achieving real-time network optimization. This

approach also mitigates privacy concerns by minimizing data transfers to central cloud servers [6, 23].

- **Robust Hybrid Models**: Combining multiple AI paradigms, such as supervised learning for traffic prediction and DRL for resource allocation, has shown promise for more adaptive and comprehensive solutions [6, 45, 46]. Future work may extend these hybrid approaches to include federated or transfer learning, enabling more robust and transferable models across heterogeneous network environments.
- **Lightweight and Energy-Efficient Architectures**: Exploring pruning, quantization, or knowledge distillation techniques can substantially reduce model size and computational overhead, making advanced AI more feasible for low-power base stations and edge nodes [50].
- **Security-Enhanced Frameworks**: Enhanced security protocols and privacy preserving techniques like homomorphic encryption or secure multi-party computation can protect user data while maintaining robust model performance [39]. These frameworks can be integrated within RL-based solutions to ensure both energy efficiency and data integrity.

In sum, AI/ML holds transformative potential for 5G+ network optimization but requires continued innovation to overcome limitations in data quality, computational complexity, interpretability, and security. By pursuing advanced AI-aided architectures and by embedding explainability, edge intelligence, and robust security measures, future systems will be better poised to meet the evolving demands of next-generation networks. This thesis aims to contribute to these research directions, particularly by investigating hybrid ML and DRL frameworks tailored for BBU energy optimization and QoS preservation.

3 Real Mobile Data and ML Predictions

Datasets used in mobile network research vary widely in scope, granularity, and availability. Public datasets, such as those from OpenCelliD [61] or network regulators [62,63], provide broad coverage but lack the specific needs or are outdated for our usecase. Operator-collected data, while detailed, is often restricted due to privacy and security concerns, limiting open-access research. Most studies rely on either simulated datasets [64–66], which allow for controlled experimentation but lack real-world variability, or real-world measurements collected through proprietary logging tools [67–70]. Despite their limitations, these datasets are essential for understanding network behavior, predicting traffic patterns, and optimizing resource allocation. Researchers working with mobile network data typically fall into two categories: industry teams within telecom companies, who have access to internal performance logs, and academic researchers, who often rely on partnerships or synthetic data. This work uses real-world data, which aligns with example 1 from the deployment scenarios described previously (see Fig. 2.3), focusing on an unsplit 5G BBU configuration that leverages existing 4G infrastructure. The dataset's specific origin remains undisclosed to maintain confidentiality.

To address the limitations mentioned above and achieve practical applicability and reliability in ML-driven predictions, this research leverages real-world network data comprising two distinct datasets. These datasets capture complementary aspects of network performance, including long-term trends and short-term granular variations, enabling detailed analysis of traffic patterns and network dynamics. The primary dataset, collected over seven months, provides broad, hourly observations from multiple sites, while the secondary dataset offers higher-resolution data at 15-minute intervals from additional sites representing varied urban environments.

The use of two datasets, rather than a single one, provides a more comprehensive understanding of network performance by capturing both long-term trends and fine-grained variations. The primary dataset offers a broad temporal view, essential for observing seasonal and large-scale traffic patterns, while the secondary dataset delivers higher-resolution insights that allow for detailed short-term optimizations. This dual approach enables a more balanced and adaptable ML model, improving predictive accuracy and ensuring robust network performance across different environments and conditions. The following sections explore data analysis, pre-processing, and model evaluation, showcasing how ML techniques contribute to tackling challenges such as traffic prediction and energy efficiency.

3.1 Data Overview and Analysis

Building on the need for both grasping broad trends and conducting a fine-grained analysis, this research utilizes two distinct datasets to gain a deeper understanding of network performance and user behavior. The **primary dataset** encompasses data collected from four cell sites (A, B, C and D) containing 12 eNodeBs (BBU+RRH), capturing a seven-month period during the COVID-19 pandemic. The **secondary dataset** includes data

from three additional cell sites (Beta, Gama, and Delta) located in different cities of the similar size, containing also 12 eNodeBs (BBU+RRH). The additional sites were monitored for 2 months, thus enhancing the diversity and comprehensiveness of the analysis. There is a simple scheme on Fig. 3.1, that graphically explains the relationship.

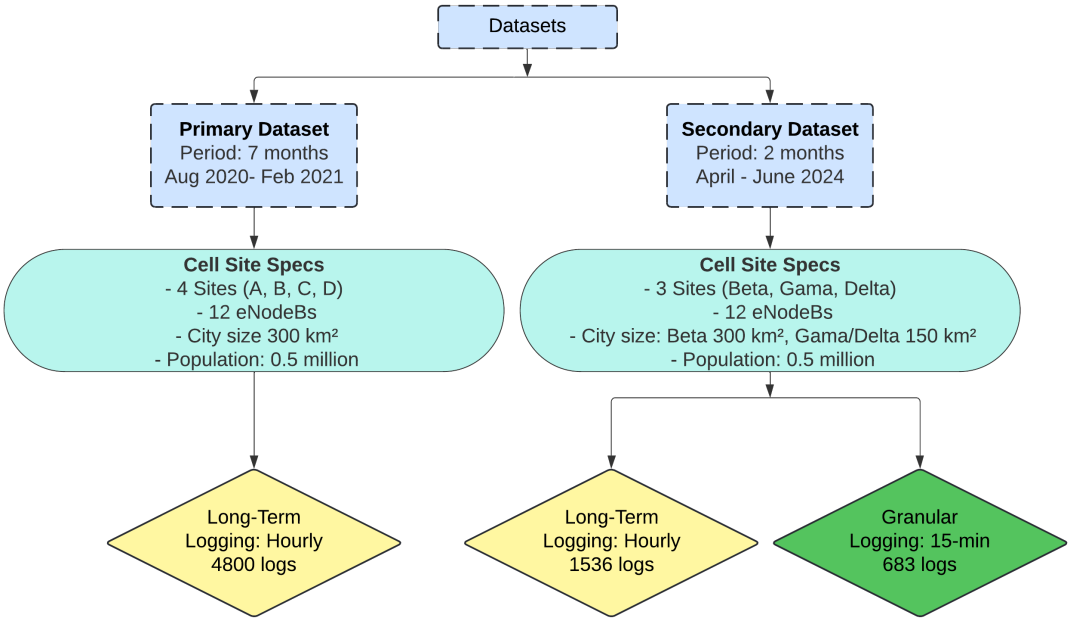


Fig. 3.1: Simple structure scheme of used datasets

Primary Dataset

The primary dataset spans from August 15, 2020, to February 28, 2021, covering a period of seven months. It includes detailed network quality information from 12 eNodeBs distributed across four cell sites (A, B, C, and D). Each cell site comprises multiple eNodeBs, with each eNodeB managing several RRHs. In total, these sites serve 212 000 users and handle approximately 1.5 terabytes of data traffic daily. Data was recorded on an hourly basis, resulting in 4 800 log records per eNodeB over the observation period, see Tab. 3.1.

Tab. 3.1: Physical communication parameters for primary dataset

Parameter	Value
City size	300 km²
Population density	0.5 million
Cell type	Macro
Number of cell sites	4
Number of eNodeBs	12
RRHs per BBU	4
Frequency bands	700 MHz, 850 MHz, 2100 MHz
Observation period	7 months, hourly

This dataset includes KPIs such as downlink and uplink data volumes, the number of connected UEs and packet loss.

Each cell site in the primary dataset is located in different areas of the same city, comprising a consistent number of BBUs and RRHs, see Tab. 3.2. These sites represent various environments, from business centers to residential areas, ensuring a diverse dataset for analysis. The layout of the cell site can be seen in Fig. 3.2. In following paragraphs, will be the sites explain in more details.

Site A (Office Area): This site is located in the city center, predominantly covering office buildings and business centers. It contains three eNodeBs (A1, A2, A3), each equipped with 4 RRHs, totaling 12 RRHs.

Site B (Office Area): Similar to Site A, Site B is located in an office area with specific antenna deployments on tall buildings. It also contains three eNodeBs (B1, B2, B3), each with 4 RRHs, totaling 12 RRHs. These eNodeBs are responsible for managing several sectors to support the heavy data load during peak business times.

Site C (Residential Area): This site is located in a densely populated residential area. It includes three eNodeBs (C1, C2, C3), each managing several sectors to provide extensive coverage. Each eNodeB is equipped with 4 RRHs, totaling 12 RRHs.

Site D (Residential Area): Situated at the edge of the city, Site D covers a mix of residential and light industrial areas. It comprises three eNodeBs (D1, D2, D3), each with 4 RRHs, totaling 12 RRHs. Each eNodeB manages multiple sectors to ensure comprehensive coverage and data collection.

Tab. 3.2: Number of BBUs and RRHs at Each Site

Site	Number of BBUs	Number of RRHs
A	3	12
B	3	12
C	3	12
D	3	12

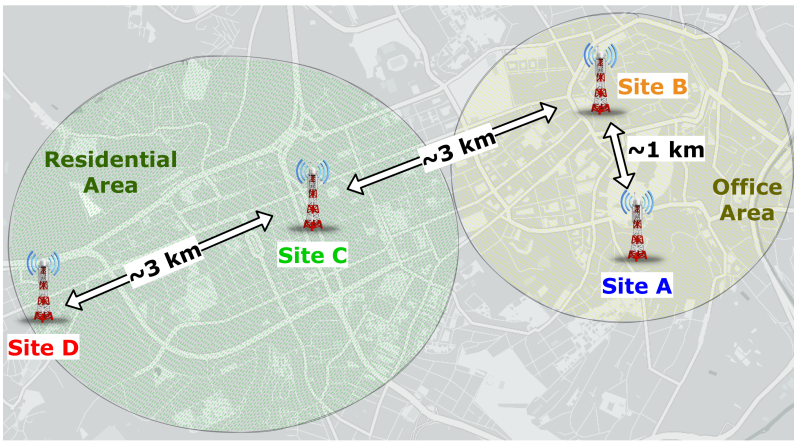


Fig. 3.2: Location of primary macro cell sites - Bird's-eye view

Data analysis shows that Sites A and B, located in office areas, have similar characteristics, with high traffic volumes during business hours and significant drops afterward. In contrast, Sites C and D, located in residential areas, maintain relatively high traffic volumes into the evening, reflecting residential usage patterns. These distinctions are crucial for understanding traffic behavior and optimizing network performance across different environments.

Secondary Dataset

The secondary dataset spans from April 2024, to June 2024 and is divided into two main categories: long-term data and granular data (short-term). These datasets provide insights into traffic volume, packet loss, latency, and energy consumption across three cell sites in different cities: Beta, Gama, and Delta. Each site is located in a different type of area: Beta in a remote area (outskirts of the city), Gama in a residential area and Delta in an office area. For the long-term dataset, we recorded 1 536 log records on an hourly basis per eNodeB over the observation period. For a granular dataset, we recorded 683 logs at 15 min intervals per eNodeB over the observation period, see Tab. 3.3.

Tab. 3.3: Physical communication parameters for secondary dataset

Parameter	**Value**
City size	B - 300 km²; C - 150 km²; D - 150 km²
Population density	B - 0.5 million; C - 0.5 million ; D - 0.5 million
Cell type	Macro
Number of cell sites	3
Number of eNodeBs	12
RRHs per BBU	2-4
Frequency bands	700 MHz, 850 MHz, 2100 MHz
Observation period	2 months hourly; 1 week 15 mins

Tab. 3.4 shows that each site has the same number of BBUs and a different number of RRUs, while Fig. 3.3 illustrates their geographic distribution in different cities.

Site Beta (Remote Area - City B): This site has three BBUs (B1, B2, B3), each managing multiple RRUs. B1 is associated with four RRUs; B2 operates also with four RRUs.; B3 controls again four RRUs. In total, the Beta site supports 12 RRUs.

Site Gama (Residential Area - City C): This site also has three BBUs (G1, G2, G3), but each BBU is linked just with two RRUs. The Gama site has a total of six RRUs.

Site Delta (Office Area - City D): The Delta site mirrors the Beta site in terms of RRU configuration, with three BBUs (D1, D2, D3) managing 12 RRUs in total.

Tab. 3.4: Number of BBUs and RRUs at each site

Site	**Number of BBUs**	**Number of RRUs**
Beta	3	12
Gama	3	6
Delta	3	12

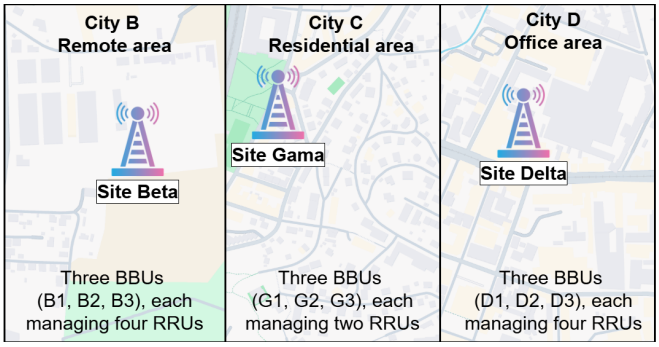


Fig. 3.3: Location of secondary macro cell sites - Bird's-eye view

3.1.1 Data Collection Process

Key Performance Indicators (KPIs) are critical metrics that are used to evaluate the performance and quality of mobile networks. These KPIs are collected and analyzed to monitor network health, optimize performance, and ensure that the network meets the service quality expectations of users.

The data collection process in mobile networks involves using Performance Management (PM) counters, which collect performance statistics from live traffic in the radio network. These measurements are taken without interfering with user connections and transmission handling, ensuring accurate and reliable data. The measurement of KPI in 5G + networks spans the UE, gNB, core elements, and the application server, as illustrated in Fig. 3.4. The PM counters capture data at each node, tracking metrics such as traffic volume, latency, energy consumption, packet loss, etc. In the radio domain, counters focus on transmissions and scheduling at the gNB. In the core, functions such as the Access and Mobility Management Function (AMF), Session Management Function (SMF), and the User Plane Function (UPF) log signaling events and user-plane performance. These measurements reveal performance from the UE application layer through the network and on to the server. They are gathered in the network management system, giving a practical view of how the network behaves [71–74].

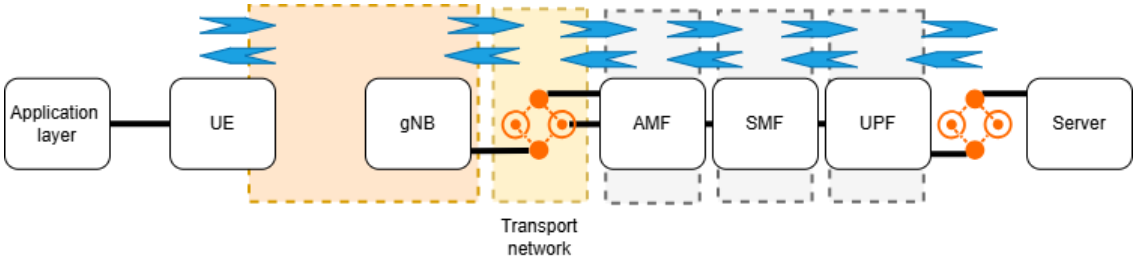


Fig. 3.4: Simplified KPI flow in 5G

KPIs can be categorized into several types based on their purpose and the aspect of the network they measure [71–75]:

- **Accessibility:** This category measures the ability of the network to grant service requests. An example of an accessibility KPI is the Initial E-RAB Establishment

Success Rate, which indicates the success rate of establishing an E-RAB during the initial setup and it is expressed as a percentage.

- **Retainability:** Retainability KPIs assess the network's ability to maintain a service once it has been established. An example is the E-RAB Retainability, shown as a percentage or ratio indicating the portion of sessions that remain active.
- **Integrity:** Integrity KPIs evaluate the quality of the service provided. For instance, Packet Loss Rate measured as a percentage, and Latency recorded in milliseconds are common Integrity KPIs that indicate the performance quality experienced by users.
- **Mobility:** These KPIs measure the network's ability to support user mobility, such as the Handover Success Rate, which tracks the success of handovers between cells, typically expressed as a percentage.
- **Availability:** Availability KPIs determine the network's ability to provide service at any given time. Cell Availability is a typical example, indicating the percentage of time the cell is available to users.
- **Utilization:** Utilization KPIs monitor how network resources are used, including traffic load and resource management metrics. These KPIs help to understand capacity and performance under different traffic conditions. Throughput is often measured in bits per second, traffic load may be given in Erlangs, and other resource metrics are expressed as percentages.

The collected KPI data is processed and aggregated to provide insights (see Sections below) at various levels, such as individual cells, clusters of cells, or entire geographic areas. Aggregation is also performed over time periods (hours, days, weeks) to identify trends and patterns. The measurement period of KPIs varies by gNodeB manufacturer. However, the 3GPP specification TS 28.550[1] [76] explicitly outlines acceptable KPI collection intervals, including 5, 15, 30, and 60 minutes. These varied intervals are designed to provide greater flexibility and availability in monitoring different KPIs, thereby enhancing the overall performance assurance. To further investigate the intervals options, a comparative analysis was created in Tab. 3.5, summarizing KPI collection intervals from major telecom vendors (Ericsson, Huawei, Nokia), clearly showing the variability and commonality among them.

Tab. 3.5: KPI Collection Intervals for 5G Network KPIs from Major Telecom Vendors

Vendor [A to Z]	**KPI Collection Intervals**	**References**
Ericsson	5 min (min), 15 min (default), 30 min, 1 hr, 2 hrs, 24 hrs	[77], [78], [79]
Huawei	Short: 5–15 min; Long: 30–60 min	[80]
Nokia	5 min (raw), 15 min (reporting), 1 hr, 1 day (aggregated)	[81], [82]

By systematically measuring and analyzing these KPIs, network operators can gain valuable insights into network performance, identify areas for improvement, and ensure that the network meets the expectations of users. The standardized KPIs also facili-

[1] https://itecspec.com/spec/3gpp-28-550-6-performance-assurance-specific-operations-and-notifications/

tate comparison and benchmarking against industry standards, such as those defined by the International Telecommunications Union-Telecommunications (ITU-T) [71–73, 83].

Long-Term KPIs Category

The long-term dataset category spans nine months in total (seven and two months), capturing hourly metrics from sites A, B, C, D and three secondary sites Beta, Gama and Delta. Due to differences in geographical locations and network conditions, these datasets cannot be directly merged. Instead, they are analyzed separately to account for variations in traffic patterns, population density, and environmental factors that influence network performance. This dataset includes bearer data such as traffic volume (downlink and uplink) in MBytes and packet loss percentages. The data is collected using advanced network monitoring tools that log these metrics continuously, providing insights into traffic patterns and network performance over an extended period.

a) Traffic Volume (Downlink and Uplink)

This KPI measures the total volume of bearer data transmitted over the network per time unit, in long-term data usually 1 hour. It provides information on the amount of data traffic handled by the network, which is crucial for understanding usage patterns and planning for capacity enhancements. Traffic volume is typically divided into two categories [72, 73]:

- **Downlink Traffic Volume (DL)**: This measures the amount of data transmitted from the network to users. High downlink traffic volume indicates heavy consumption of data services by users, such as streaming videos, downloading files, or browsing the internet.

$$\textit{Downlink Traffic Volume} = \sum \textit{Data transmitted from Nodes to UEs (MBytes per time)} \tag{3.1}$$

- **Uplink Traffic Volume (UL)**: This measures the amount of data transmitted from users to the network. High uplink traffic volume is often associated with activities such as uploading files, video calling, or sending emails with large attachments.

$$\textit{Uplink Traffic Volume} = \sum \textit{Data transmitted from UEs to Nodes (MBytes per time)} \tag{3.2}$$

b) Packet Loss Percentages

This KPI measures the proportion of data packets that are lost during transmission over the network. By analyzing packet loss percentages, network operators can identify and troubleshoot issues related to network congestion, hardware failures, or signal interference. Packet loss can significantly degrade the quality of service. There are three KPIs parameters [72, 73]:

- **Total Packet Loss Percentage**: This measures the overall packet loss rate across the network, providing a general indicator of network performance.

$$\textit{Total Packet Loss (\%)} = \frac{\textit{Number of Lost Packets}}{\textit{Total Number of Transmitted Packets}} \cdot 100\% \tag{3.3}$$

- **Downlink Packet Loss Percentage (DL)**: This measures the percentage of packets lost during transmission from the network to the users. High downlink packet loss can affect user experience, especially for applications requiring consistent and reliable data streams.

$$\textit{Downlink Packet Loss (\%)} = \frac{\textit{Number of Lost Packets in DL}}{\textit{Total Number of Transmitted Packets in DL}} \cdot 100\% \tag{3.4}$$

- **Uplink Packet Loss Percentage (UL)**: This measures the percentage of packets lost during transmission from users to the network. High uplink packet loss can impact the performance of user-upload activities and cloud-based applications.

$$\textit{Uplink Packet Loss (\%)} = \frac{\textit{Number of Lost Packets in UL}}{\textit{Total Number of Transmitted Packets in UL}} \cdot 100\% \tag{3.5}$$

Granular (Short-Term) KPIs Category

The granular dataset category covers a one-week period with a high frequency of data collection at 15-minute intervals. This dataset includes detailed metrics from Operations, Administration, and Maintenance (OAM) data and bearer data, providing a more granular view of the network's performance. The key KPIs for the granular dataset include total port throughput, UE throughput, UE packet loss, UE latency, and energy consumption for both BBUs and RRUs. Additionally, it includes hourly information on packet loss percentages for eUTRAN, eUTRAN latency, and temperature metrics (minimum, average, and maximum) over the week.

a) Total Port Throughput

This KPI measures total transmitted data across network ports, represented by counters such as PortMbpsIn and PortMbpsOut. It includes all traffic types (user and management) and reflects the load on network ports.

The data is collected using PM counters, which provide average throughput values over fixed intervals, typically 15 minutes (900 seconds). The calculation converts InterfaceInOctets to Mbps by multiplying by 8 to obtain bits, dividing by 1 000 000 to scale to Mbits, and dividing by 900 s to average over the interval:

$$\textit{PortMbpsIn} = \frac{8 \cdot \textit{InterfaceInOctets}}{1\,000\,000 \cdot 900} (\textit{Mbps}). \tag{3.6}$$

Averaging over a certain time interval simplifies the work with the dataset, providing a slightly simplified view of the network functioning. However, this approach results in the loss of granularity, such as peak and minimum throughput values on a per-second basis. Despite this limitation, PM's averaged throughput is effective for understanding long-term port utilization trends [72, 75].

$$\textit{Total Port Throughput} = \sum \textit{Data transmitted through all ports (Mbps).} \tag{3.7}$$

b) UE Throughput

This KPI quantifies the data rate experienced by UE, serving as an indirect indicator of perceived network performance. While throughput alone does not define QoS, higher values generally correspond to improved user experience, particularly for bandwidth-intensive applications. However, a low throughput value does not necessarily indicate poor network performance, as it may simply reflect lower user activity or background traffic. PM counters track data rates between Nodes and UEs, providing a measure of UE throughput that reflects network capacity and efficiency, particularly when the UE is actively transmitting or receiving data [72, 75].

$$UE\ Throughput = \sum Data\ rate\ experienced\ by\ UEs\ (kbps) \tag{3.8}$$

c) UE Packet Loss

This KPI measures the proportion of data packets that are lost during transmission from the UE to the network. Packet loss from UE can significantly impact service quality, especially for applications requiring real-time data transmission, such as video conferencing and online gaming. UE packet loss is tracked using network monitoring tools that count the number of packets sent by the UEs and the number of packets successfully received by the network, along with the packets that fail to reach their destination [72, 75].

$$UE\ Packet\ Loss\ (\%) = \frac{Number\ of\ Lost\ Packets\ from\ UE}{Total\ Number\ of\ Transmitted\ Packets\ from\ UE} \cdot 100\% \tag{3.9}$$

d) UE Downlink Latency

This KPI measures the time it takes for a data packet to travel from the network to the UE. Downlink latency is a critical metric for assessing the performance of applications that require real-time data transmission, such as video streaming, online gaming, and VoIP. High downlink latency can lead to delays, buffering, and a degraded user experience. UE downlink latency is monitored using network performance management tools that track the round-trip time of data packets sent from the network to the UE and back. Lower downlink latency values indicate better network performance and a higher QoS for end-users [72].

$$UE\ Downlink\ Latency = Time\ a\ packet\ travels\ from\ the\ network\ to\ the\ UE\ and\ back\ (ms) \tag{3.10}$$

e) Energy Consumption

This KPI measures the amount of energy consumed and the average power consumption by BBUs and RRUs. BBUs and RRUs send PM counters with values from their energy measurements. It is an important metric for assessing the energy efficiency of the network [84]. The energy consumption data is collected in 6-second intervals. Over a typical 15-minute period (900 s), there are 150 such intervals, alternatively, measurements may be aggregated over an hourly period. The “Consumed Energy” KPI sums the energy

values from each interval. To obtain the "Average Power Consumption", the total energy consumed (in Wh) is divided by the measurement period expressed in hours (for example, 15 minutes corresponds to 0.25 hours). This yields a value in Watts (W).

- **Consumed Energy (Wh)**: This KPI measures the total energy consumed by BBUs and RRUs over a specific period. It sums up energy consumption of all consumer units having an energy meter function. It is measured regardless of battery charge or discharge.

$$BBU\ Consumed\ Energy\ (Wh) = \sum Energy\ consumed\ by\ BBUs\ (Wh) \tag{3.11}$$

$$RRU\ Consumed\ Energy\ (Wh) = \sum Energy\ consumed\ by\ RRUs\ (Wh) \tag{3.12}$$

- **Average Power Consumption (W)**: This KPI measures the average power consumption by BBUs and RRUs, reflecting the rate of energy consumption over the measurement period. In these equations, time is expressed in hours:

$$BBU\ Average\ Power\ Consumption\ (W) = \frac{Sum\ up\ energy\ consumed\ by\ BBUs\ (Wh)}{Time\ (h)} \tag{3.13}$$

$$RRU\ Average\ Power\ Consumption\ (W) = \frac{Sum\ up\ energy\ consumed\ by\ RRUs\ (Wh)}{Time\ (h)} \tag{3.14}$$

Energy consumption data is collected using monitoring tools that track the power usage of network components. These metrics help in optimizing the energy efficiency of the network, reducing operational costs, and minimizing environmental impact.

f) eUTRAN Packet Loss

This KPI measures the percentage of packets lost within the eUTRAN (Evolved Universal Terrestrial Radio Access Network). It is critical for assessing the quality of service within the radio access network. These metrics are collected using PM tools that monitor packet transmission and latency within the eUTRAN, essential for identifying and resolving issues specific to the radio access network [73].

$$eUTRAN\ Packet\ Loss\ (\%) = \frac{Number\ of\ Lost\ Packets\ in\ eUTRAN}{Total\ Number\ of\ Transmitted\ Packets\ in\ eUTRAN} \cdot 100\% \tag{3.15}$$

g) eUTRAN Latency

This KPI measures the latency within the eUTRAN. Lower eUTRAN latency indicates better performance of the radio access network. These metrics are collected using PM tools that monitor packet transmission and latency within the eUTRAN, essential for identifying and resolving issues specific to the radio access network [73].

$$eUTRAN\ Latency = Time\ taken\ for\ a\ packet\ to\ travel\ within\ eUTRAN\ (ms) \tag{3.16}$$

h) Temperature Metrics

This KPI measures the temperature of network components, including minimum, average, and maximum values. It is collected using environmental monitoring tools that track the thermal conditions of network components. Monitoring and managing temperature is crucial for preventing overheating and ensuring optimal performance of devices [84, 85].

$$Temp\ [°C]\ MIN = Minimum\ temperature\ recorded \tag{3.17}$$

$$Temp\ [°C]\ AVG = Average\ temperature\ recorded \tag{3.18}$$

$$Temp\ [°C]\ MAX = Maximum\ temperature\ recorded \tag{3.19}$$

3.1.2 Collected Data Description

Collected Data Description focuses on the metrics that were gathered and their relevance for each dataset. Tab. 3.6 lists these metrics, along with the collection periods and dataset applicability. The primary dataset covers seven months, while the secondary dataset spans two months plus a one-week granular subset. This approach captures broad temporal patterns as well as short-interval variations.

Long-term (hourly) measurements (primary and secondary dataset) include traffic volume and packet loss details. These illustrate average load and performance over extended periods. The granular (15-minute intervals) measurements (secondary dataset) capture additional metrics such as total port throughput and UE throughput. They also include hourly eUTRAN statistics and temperature data. This combination offers a multifaceted perspective on network conditions and component behavior.

Table 3.6 shows certain metrics, such as total port throughput and energy consumption, that only appear in the granular subset. Others, including traffic volume and packet loss, span all three categories. The number of users is logged in full for the primary dataset and can be counted for both secondary segments, linking the long-term and granular measurements. Because the granular period overlaps with the secondary long-term period, these data can be counted and merged for a more comprehensive analysis. The primary dataset gives a longer historical perspective, while the secondary dataset integrates short-interval readings, covering both broad load trends and immediate performance details.

Tab. 3.6: Metrics, descriptions, and dataset applicability

Metric	Description	Primary Long-Term	Secondary Long-Term	Secondary Granular
Collection Period		7 months (Aug 2020–Feb 2021)	2 months (Apr 14, 2024–Jun 16, 2024)	1 week (within same period)
Date	The day data was collected.	✓	✓	✓
Time [hour]	Hourly time of data collection.	✓	✓	✓$_{\text{derived}}$
Traffic Volume (MBytes)	Total bearer data transmitted.	✓	✓	✓$_{\text{derived}}$
Traffic Volume DL (MBytes)	Downlink bearer data transmitted.	✓	✓	
Traffic Volume UL (MBytes)	Uplink bearer data transmitted.	✓	✓	
Packet Loss Total (%)	Percentage of total packets lost.	✓	✓	✓
Packet Loss DL (%)	Percentage of downlink packets lost.	✓	✓	
Packet Loss UL (%)	Percentage of uplink packets lost.	✓	✓	
Number of users (-)	Number of connected users.	✓	✓$_{\text{derived}}$	✓$_{\text{derived}}$
Time interval [hh]	15-minute data collection intervals.		✓$_{\text{derived}}$	✓
Total port Tput (Mbps)	Port throughput (OAM + bearer).			✓
UE Tput (kbps)	Throughput for UE.			✓
UE Packet loss (%)	Packet loss for user traffic.			✓
UE DL Latency (ms)	Downlink latency for UE.			✓
BBU Consumed Energy (Wh)	Energy used by the BBU.			✓
BBU AVG Power consumption (W)	Average BBU power usage.			✓
RRU Consumed Energy (Wh)	Energy used by the RRU.			✓
RRU AVG Power consumption (W)	Average RRU power usage.			✓
eUTRAN Latency (Hourly) (ms)	Hourly eUTRAN latency.			✓
BBU Temperature (Min, Avg, Max)	Hourly BBU temperature readings.			✓

3.2 Machine Learning Model Development and Integration

The primary and secondary datasets, while similar in scope, differ slightly in their structure and focus. These differences extended to the pre-processing methods applied to each dataset, reflecting their unique characteristics.

The primary dataset served as the foundation for developing the initial version of the framework: Predictive Energy Saver for Baseband Units (PESBIU 1.0). Building on the knowledge gained from the primary dataset's predictions, we were later able to refine and enhance the ML algorithms for predictions for the secondary dataset, leading to significant improvements in their performance and applicability. The process and results were published in [86, 87].

3.2.1 Data Prepossessing and Cleaning

For the purpose of this research, a series of common data cleaning and transformation steps were applied to both the primary and secondary datasets to ensure data integrity and to facilitate effective model training and evaluation. First, any records associated with anomalous events or maintenance periods were removed rather than imputed, reducing the risk of bias introduced by filling missing values with synthetic estimates. Although this approach can slightly reduce dataset size, it preserves the natural behavior and reliability of the underlying measurements.

Following the removal of unusable data, both datasets were split into training and testing subsets to evaluate the models' ability to generalize. While the exact proportions

(e.g., 75/25 - primary or 80/20 - secondary) varied between the datasets, each split reserved a substantial portion of data for training and kept a sufficient portion aside for unbiased performance assessment of unseen observations.

Because raw feature scales vary, a normalization or scaling process was then applied. In case of the primary dataset, both standard scaling and the Min-Max scaler were used. Standard scaling subtracts the mean and divides it by the standard deviation. The Min-Max scaler transforms values to the [0,1] range. In case of the secondary dataset, it was more appropriate to use the Min-Max scaler exclusively. These transformations ensured that no single feature dominates the model training process simply by having larger numeric values, leading to more robust and faster convergence during optimization.

a) Primary Dataset

The primary dataset comprised raw data collected from various eNodeBs in a mobile network (see Sec. 3.1 and Tab. 3.6). These data spanned multiple operational metrics, capturing variables such as traffic load, energy consumption, and performance indicators over a series of daily intervals. Because eNodeBs occasionally undergo maintenance, certain intervals exhibited gaps or anomalies; in total, about 0.8% of the records were excluded to ensure only normal operational behavior was analyzed. This removal step safeguarded model training against skewed trends and unreliable measurements [86].

After cleaning, the dataset provided a broad, real-world view of normal network conditions, reflecting fluctuations in traffic and performance across different times of the day. These characteristics made it possible to train ML models that closely captured daily operational patterns. By isolating routine eNodeB activity, the primary dataset formed a strong foundation for studying network energy consumption under typical loads, offering an effective baseline for model evaluation in this dissertation.

b) Secondary Dataset

This section discusses the methods applied to clean and pre-process the secondary dataset, which consists of both long-term and granular data, see Sec. 3.1 and Tab. 3.6. These diverse data types posed challenges that needed to be addressed through harmonization, cleaning, transformation, and effective data splitting for model training and testing [87].

The combination of long-term and granular measurements was explored with the hypothesis that their integration could enhance the predictive models by leveraging both the long-term trends captured in the hourly data and the detailed short-term fluctuations present in the granular data. The aim was to improve the model's ability to predict network behavior over a range of time scales, optimizing energy efficiency through a more comprehensive understanding of network dynamics. However, this integration required careful pre-processing to ensure compatibility between the datasets. We experimented with different ways of combining the datasets to determine the most effective approach for preserving their respective characteristics while maximizing their predictive power.

The first step in data pre-processing was harmonizing the mixed datasets. Since long-term measurements were collected at hourly intervals, while granular measurements were

collected at more frequent 15-minute intervals, the two measurement types needed to be aligned. To achieve this, downsampling was applied to the long-term measurements, converting it to a 15-minute interval format through linear interpolation. This interpolation generated four data points for each original hourly data point, ensuring that the model could capture more detailed fluctuations in network performance. However, careful attention was given to avoid introducing artificial trends during this interpolation process.

Conversely, the granular measurements were aggregated to match the hourly intervals of the long-term measurements. For metrics such as traffic volume and energy consumption, values from the four 15-minute intervals within each hour were summed to produce a single hourly data point. For other metrics such as latency and packet loss, weighted or simple averages were used depending on the nature of the data, ensuring consistency across the data types. The code itself is publicly available [2].

Feature engineering played a crucial role in enhancing the predictive power of the models by transforming raw data into meaningful features, especially with two different types of data - long-term and granular. One of the primary transformations involved creating temporal features from the 'Date' and 'Time Interval' columns. Features such as 'Hour of the Day,' 'Day of the Week,' and 'Month' were introduced to capture the cyclical patterns in network traffic and performance. These temporal features helped the models understand recurring patterns, such as daily peaks in traffic during business hours and weekly fluctuations in network activity. In addition to temporal features, lagged features were engineered to incorporate temporal dependencies within the dataset. For example, metrics such as traffic volume and latency from previous hours were used as predictors for current values, with a window size of ten hours (ten hours was selected as an empirically balanced window size-long enough to capture trend patterns, but short enough to avoid excessive noise or diminishing returns in feature relevance), allowing the model to recognize trends over time and enhance its forecasting abilities.

To further enhance model performance, interaction terms were introduced, as described in Sec. 3.2.2. These terms capture relationships between different network metrics by combining them into new features, thereby improving the model's predictive accuracy.

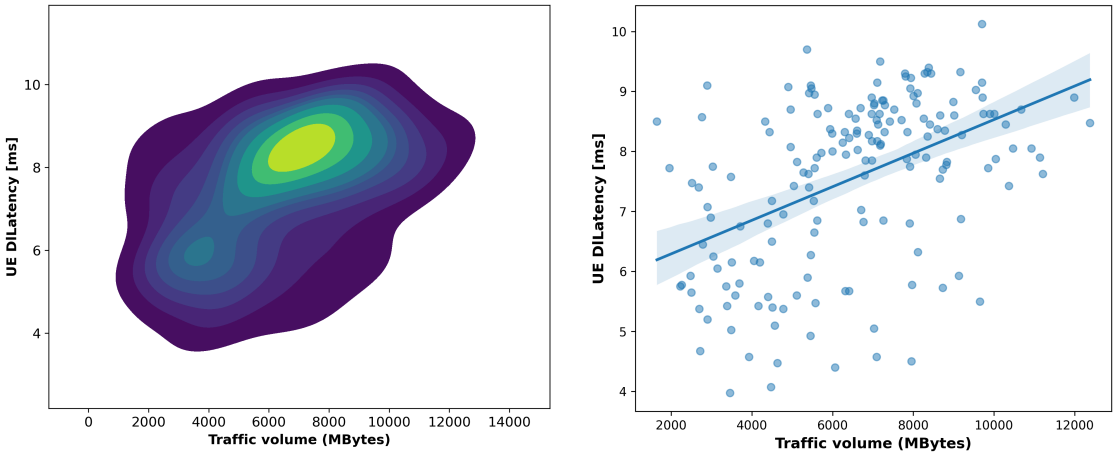
3.2.2 Interdependencies Between KPI Metrics

In addition to understanding and cleaning each dataset before modeling, analyzing the interplay among key KPIs is important to determine whether combining metrics (e.g., creating interaction terms) could enhance our predictive models and support energy-saving strategies without degrading user experience [87].

The focus was on interaction between traffic volume (DL+UL) and latency (UE DL Latency). A Kernel Density Estimation (KDE) (see Fig. 3.5a) and scatterplot with regression line (see Fig. 3.5b) shows a clear upward trend in latency (ms) as traffic volume (MBytes) moves from about 2 000 MBytes to 12 000 MBytes. When traffic remains below 4 000 MBytes, latency stays consistently low, but it begins to climb under moderate loads

[2] `https://github.com/vafekt/Stage1_PESBiU`

(6 000–8 000 MBytes) and spikes significantly beyond 10 000 MBytes, indicating mild to heavy network congestion.



(a) KDE figure

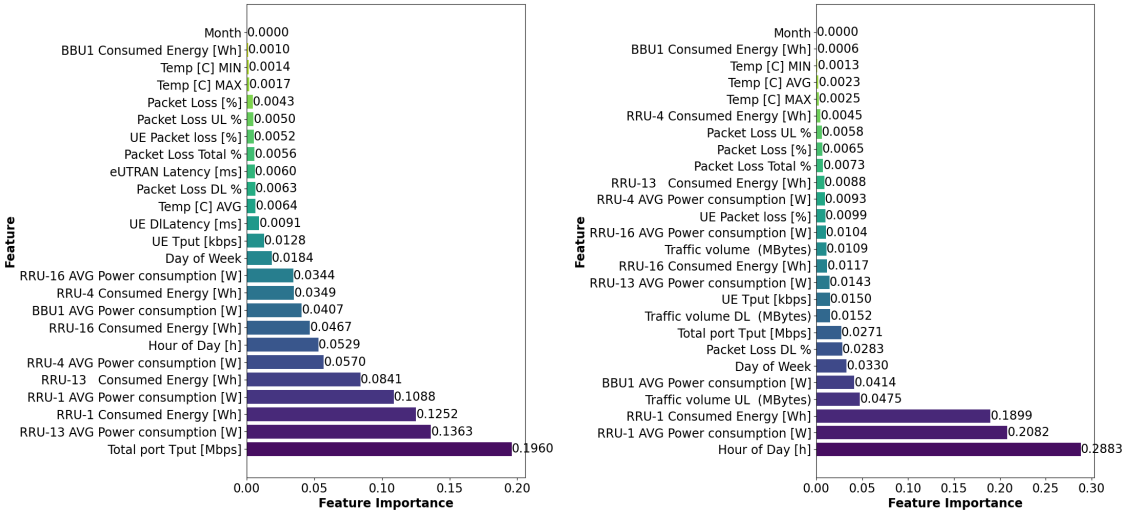
(b) Scatterplot with regression line

Fig. 3.5: Correlation between traffic volume and latency KPIs

To explore whether these metrics or others should be crossed (e.g., multiplied or otherwise combined) in feature engineering, a Random Forest model [88] was applied to assess feature importance, see Fig. 3.6, then Friedman's H-statistic [89] was used to quantify the strength of their interaction. These techniques were chosen to capture the genuine, often subtle, ways features co-vary. Random Forest feature importance offers a straightforward view of their relative contributions, but it does not fully expose how features collaborate to affect model outputs. That is why Friedman's H statistic was used - this statistic quantifies the strength of interaction between features, from 0 to 1. The relatively modest H-statistic of 0.1316 implies only mild synergy between traffic and latency. Although this value does not point to a very strong interaction, it does confirm that both variables influence each other to a meaningful extent.

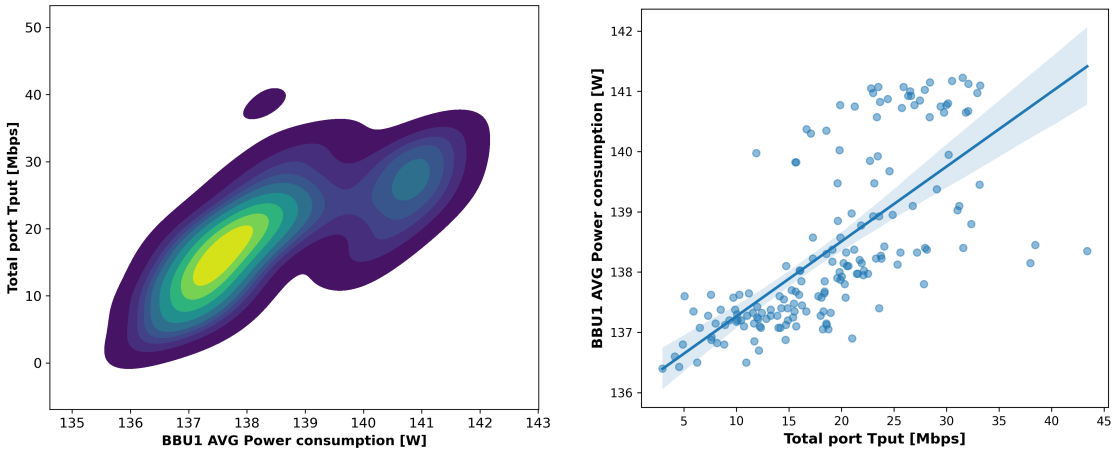
Another focus was on interaction emerged between Average Power Consumption (W) and Total Port Throughput (Mbps), for which one random BBU was used. The KDE in Fig. 3.7a highlights dense clusters of data at moderate throughput (5–20 Mbps) and power consumption (136–140 W), with a secondary region pushing throughput up to 30–40 Mbps at slightly higher power draws (140–141 W). Meanwhile, the scatterplot's upward-sloping regression line, Fig. 3.7b, confirms that power consumption rises as total port throughput increases. Although the confidence interval remains narrow at modest throughput, indicating consistent behavior, it broadens above 30 Mbps, reflecting more variability near higher load conditions.

Furthermore, the correlation between Average Power Consumption (W) of BBU and total port throughput (Mbps) had a significantly stronger Friedman H statistic of 0.5267 and the Random Forest Regression was used to show the features predictor's contribution as shown in Fig. 3.8. In predicting power consumption, environmental factors (like max, min, and average temperature) dominate, with total port throughput and time features (hour of day, day of week) also playing a strong role. In contrast, throughput predictions



(a) Predicting traffic volume (MBytes) (b) Predicting UD DL latency (ms)

Fig. 3.6: Feature importance evaluation for predicting metrics using Random Forest Regression

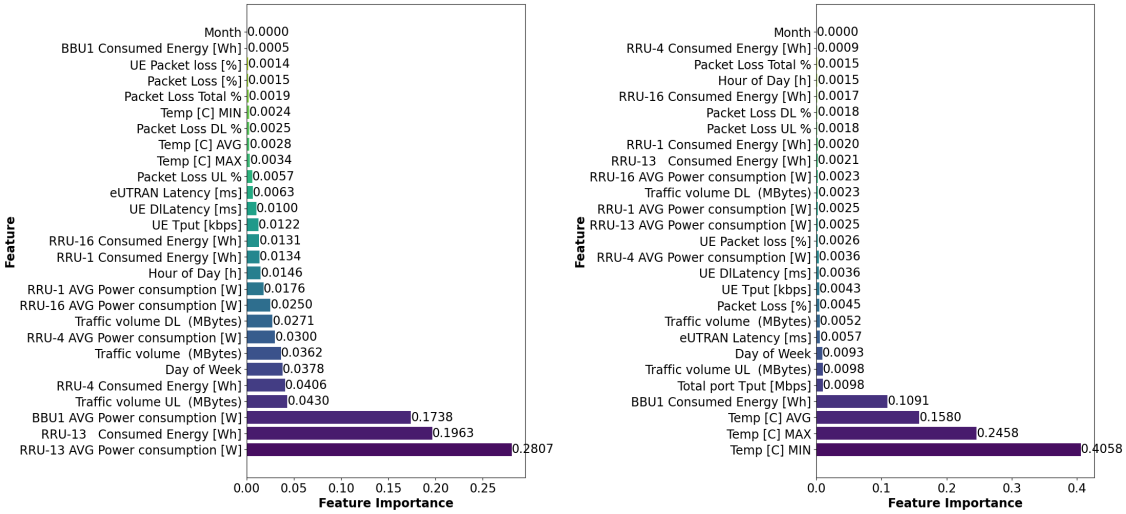


(a) KDE figure (b) Scatterplot with regression line

Fig. 3.7: Correlation between total port throughput and avg power consumption KPIs

more heavily rely on BBU and RRU energy measurements along with traffic metrics, reflecting that active power draws and packet flows are primary cues for throughput shifts.

In sum, these interdependencies reveal that traffic volume and latency have a mild but consistent mutual effect, while power consumption and throughput strongly shape each other's behavior. Targeted cross-features can boost prediction accuracy, making it easier to spot when and how to cut energy use without harming performance. This clarity is shaping our later modeling decisions by indicating which KPI pairs matter most for optimizing resources and reducing power costs.



(a) Predicting total port throughput (Mbps) (b) Predicting avg power consumption (W)

Fig. 3.8: Feature importance evaluation for predicting metrics using Random Forest Regression

3.2.3 Selected Models - Stage 1

To address the challenges of optimizing energy efficiency and network resource management in mobile networks, we have considered several advanced ML models, with distinct strengths in time-series forecasting and sequence learning, in order to find out which is the best suited for our research; see the general overview in Sec. 2.4. The models are summarized in Tab. 3.7. It is important to note that this research evolved continuously, and thus, not all models listed were applied to every dataset. Specifically, certain models were selectively implemented based on the previous results and findings.

This Stage 1 serves as a foundation by establishing predictive models that analyze network traffic, energy consumption, and QoS fluctuations. It provides the necessary forecasting capabilities required for optimization strategies in the subsequent stage. By first refining traffic predictions and validating model accuracy, we ensure that optimization methods in Stage 2 operate with high-confidence inputs, leading to superior energy efficiency without compromising QoS.

This section provides a detailed mathematical and theoretical overview of each model, highlighting its relevance and applicability to 5G+ network optimization.

Tab. 3.7: Overview of AI/ML models usage in this section

Stage	**Dataset**	**Model**	**Algorithm Type**
Stage 1	Primary	LSTM	Time series prediction
Stage 1	Primary	GRU	Time series prediction
Stage 1	Primary	FB's Prophet	Trend forecasting
Stage 1	Secondary	LSTM	Sequential modeling
Stage 1	Secondary	GRU	Sequential modeling
Stage 1	Secondary	CNN-LSTM, Hyper-CNN-LSTM	Spatio-temporal prediction

a) Long Short-Term Memory (LSTM)

LSTMs [90] are an extension of standard RNNs that address the vanishing gradient problem by introducing a more complex cell structure with three gates: forget, input, and output gates. Each gate decides which information to store, discard, or output at each step. The forget gate f_t, input gate i_t, and output gate o_t are defined as:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \tag{3.20}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \tag{3.21}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \tag{3.22}$$

where x_t denotes the current input at time t, h_{t-1} is the previous hidden state, σ represents the sigmoid activation function, W_f, W_i, W_o are the weight matrices for the respective gates, b_f, b_i, b_o are the bias terms.

The new candidate cell state $\tilde{C}_t$ is calculated as:

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C), \tag{3.23}$$

where W_C is the weight matrix for creating the candidate cell state, b_C is the corresponding bias term and tanh is the hyperbolic tangent activation function.

The cell state is then updated using the forget and input gates:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t, \tag{3.24}$$

where $\odot$ denotes the Hadamard product.

Finally, the new hidden state the new hidden state h_t is derived using the output gate:

$$h_t = o_t \odot \tanh(C_t). \tag{3.25}$$

The ability of LSTMs to retain long-term dependencies makes them highly effective for predicting complex, long-range temporal patterns in network data.

b) Gated Recurrent Unit (GRU)

The GRU [91] is a simplified version of the LSTM network, designed to overcome the vanishing gradient problem inherent in traditional RNNs. The GRU architecture consists of two gates: the update gate z_t and the reset gate r_t, which regulate the flow of information through the network:

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r), \tag{3.26}$$

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z), \tag{3.27}$$

where r_t determines how much of the previous hidden state h_{t-1} to forget, and z_t decides how much of the new information x_t to include. The candidate hidden state $\tilde{h}_t$ is then calculated as:

$$\tilde{h}_t = \tanh(W \cdot [r_t \odot h_{t-1}, x_t] + b), \tag{3.28}$$

where $\odot$ denotes the Hadamard product. Finally, the new hidden state h_t is a linear interpolation between the previous hidden state and the candidate hidden state:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t. \tag{3.29}$$

This structure reduces the complexity of LSTMs, resulting in faster training times, making GRU suitable for real-time 5G network traffic predictions.

c) Facebook's Prophet (FB's Prophet)

FB's Prophet [92] is a statistical model designed for time-series forecasting, particularly when data exhibits strong seasonal patterns and non-linear trends. FB's Prophet uses an additive regression model that combines three main components: trend ($g(t)$), seasonality ($s(t)$), and holidays or events ($h(t)$). The resulting forecasted value $y(t)$ is computed as:

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t, \tag{3.30}$$

where $g(t)$ captures the long-term trend, $s(t)$ models recurring seasonal variations, $h(t)$ includes special events that can disrupt regular patterns, and ϵ_t is the error term representing unexplained variability. This structure allows for flexible modeling of complex real-world time series, making it suitable for forecasting mobile network traffic with daily and weekly cycles.

d) Convolutional Neural Networks Long Short-Term Memory (CNN-LSTM)

CNN-LSTM [93] is a hybrid deep learning model that combines CNNs for spatial feature extraction and LSTM networks for temporal sequence learning. This architecture is particularly useful for time-series data with spatial correlations, such as 5G network traffic, where the spatial information (e.g., cell site configurations) can be leveraged along with temporal dynamics. The following list describes the layers and their dependencies in CNN-LSTM:

1. CNN Layer for Spatial Feature Extraction: The CNN layers are used to extract spatial features from the input data x_t. Each CNN layer consists of a series of convolutional filters applied to the input, producing a feature map F:

$$F = \sigma(W * x_t + b), \tag{3.31}$$

where $*$ denotes the convolution operation, W is the weight matrix of the convolutional filter, x_t is the input at time t, and b is the bias term.

The output feature map F is then passed through a non-linear activation function σ, such as ReLU, to capture non-linear relationships in the data.

2. LSTM Layer for Temporal Learning: The LSTM layer takes the feature maps from the CNN as input and learns temporal dependencies over time. Each LSTM cell in the sequence processes the spatial features from different time steps, using the memory cells and gating mechanisms described earlier to capture long-term dependencies. The LSTM

equations are defined as:

$$f_t = \sigma(W_f \cdot [h_{t-1}, F] + b_f), \tag{3.32}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, F] + b_i), \tag{3.33}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, F] + b_o), \tag{3.34}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, F] + b_C), \tag{3.35}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t, \tag{3.36}$$

$$h_t = o_t \odot \tanh(C_t). \tag{3.37}$$

3. Combining CNN and LSTM: The output from the CNN, F, is treated as the input to the LSTM at each time step, which allows the network to learn temporal patterns from spatially rich features. The combined architecture captures complex interactions in network traffic patterns.

4. Application in 5G Networks: The CNN-LSTM model is suitable for scenarios where both spatial and temporal patterns need to be understood. For example, in 5G networks, spatial features such as user mobility across different cell sites and temporal variations in data traffic can be efficiently captured by the CNN-LSTM architecture.

e) Convolutional Neural Networks Long Short-Term Memory with Hyperoptimizer (Hyper-CNN-LSTM)

Hyper-CNN-LSTM [94] is an advanced version of the CNN-LSTM that incorporates hyperparameter optimization techniques, such as Bayesian Optimization and Grid Search, to fine-tune the architecture and training parameters. The primary goal is to enhance the model's performance by systematically selecting the optimal set of hyperparameters, which include learning rate, batch size, number of layers, and filter sizes. The list below outlines the layers in Hyper-CNN-LSTM and their dependencies.

1. Hyperparameter Optimization Process: Hyperparameter optimization involves finding the set of hyperparameters θ that minimizes the loss function $\mathcal{L}$. Mathematically, the optimization problem is defined as:

$$\theta^* = \arg\min_{\theta} \mathcal{L}(y, f_\theta(x)), \tag{3.38}$$

where y is the true label, x is the input data, f_θ is the prediction function parameterized by θ, and $\mathcal{L}$ is the loss function, typically Mean Squared Error (MSE) for regression tasks.

2. Bayesian Optimization: Bayesian Optimization is used to model the loss function $\mathcal{L}$ as a Gaussian Process (GP) and iteratively search for the optimal hyperparameters. The acquisition function $\mathcal{A}(\theta)$ selects the next hyperparameters to evaluate based on a trade-off between exploration (searching new regions) and exploitation (refining known good regions):

$$\theta_{next} = \arg\max_{\theta} \mathcal{A}(\theta). \tag{3.39}$$

3. Grid Search: Grid Search exhaustively evaluates a predefined set of hyperparameters by computing $\mathcal{L}$ for each combination. While computationally intensive, it provides a comprehensive overview of model performance across a wide range of settings.

4. Integration of Hyperoptimization with CNN-LSTM: By integrating hyperoptimization with the CNN-LSTM model, the Hyper-CNN-LSTM achieves better generalization and lower prediction errors. This optimization process involves tuning both the CNN and LSTM layers' hyperparameters, such as kernel sizes in the CNN layers and the number of units in the LSTM cells.

5. Application in 5G Networks: The Hyper-CNN-LSTM is particularly useful in scenarios where the network traffic is highly dynamic and unpredictable, such as during peak usage hours or in environments with high user mobility. The model's ability to adaptively optimize its structure and parameters ensures high predictive accuracy and efficient resource utilization in 5G networks.

3.2.4 Model Evaluation Metrics

Model evaluation metrics are crucial for assessing the performance of predictive models, particularly in contexts where both accuracy and error magnitude are important considerations. For evaluating the models developed using both the primary and secondary datasets, Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) were employed as primary evaluation metrics. These metrics provided insights into the accuracy and reliability of predictions for network traffic and energy consumption.

a) Mean Absolute Error (MAE)

MAE is a widely used metric in regression analysis, measuring the average magnitude of prediction errors. It provides an intuitive understanding of prediction accuracy by quantifying the absolute difference between predicted values and the actual observed values. Mathematically, MAE is defined as:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|, \tag{3.40}$$

where:

- n is the number of observations,
- y_i is the actual value for the i-th observation,
- $\hat{y}_i$ is the predicted value for the i-th observation,
- $|y_i - \hat{y}_i|$ is the absolute error for each prediction.

MAE provides a direct measure of the average prediction error in the same units as the data, which makes it particularly useful for evaluating models that predict network traffic and energy consumption. Since it considers absolute values of errors, it avoids the issue of offsetting positive and negative errors. MAE is widely favored in practice due to its simplicity and interpretability [95, 96].

b) Mean Absolute Percentage Error (MAPE)

MAPE offers a scale-independent measure of prediction accuracy by expressing the error as a percentage of the actual values. This is particularly helpful when comparing models

across datasets where the scale of the target variable may vary significantly. MAPE is computed as:

$$\text{MAPE} = \frac{100\%}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|, \tag{3.41}$$

where:

- n is the total number of observations,
- y_i is the actual value for the i-th observation,
- $\hat{y}_i$ is the predicted value for the i-th observation,
- $\left| \frac{y_i - \hat{y}_i}{y_i} \right|$ represents the absolute percentage error for each prediction.

MAPE expresses the error in percentage terms, making it easier to interpret relative errors across different models and datasets. This metric is particularly useful for network management applications, where operators need to assess model performance across sites with varying traffic volumes and energy consumption. However, one limitation of MAPE is that it can produce large errors when the actual values are close to zero [97, 98].

Both MAE and MAPE were applied to evaluate the performance of the models developed on the primary and secondary datasets. The primary dataset, which spanned seven months of data from multiple NodeBs, provided an extensive basis for evaluating model performance across different cell sites. By using these metrics, we were able to quantify how closely the model's predictions aligned with actual observations, particularly in predicting network traffic volume, latency, and energy consumption.

3.2.5 ML Model Deployment Results

This subsection evaluates the performance of ML models in predicting key network metrics across datasets: the primary dataset and the secondary dataset. In addition, the secondary data set was examined from a hybrid perspective (long-term KPIs + granular KPIs), and from a purely granular KPIs.

The initial analysis focuses on the performance of models like LSTM, GRU, and FB's Prophet when applied to the primary dataset, which consists of long-term data with hourly intervals. Following this, the study investigates the hybrid approach, where both secondary long-term and granular data are integrated to improve prediction accuracy. Finally, we shift our attention to the granular dataset, sampled at 15-minute intervals, which demonstrates significant improvements in capturing short-term fluctuations. The goal of this subsection is to highlight the strengths and limitations of each dataset and model combination.

a) Primary Dataset

The primary dataset employed in this study was pivotal for assessing the performance of various ML models in predicting network traffic volumes within 5G networks. This dataset encompassed KPIs such as traffic volume, the number of active users, and packet loss across four distinct sites, see Sec. 3.1. These specific KPIs were selected due to their impact on network efficiency and QoS. Traffic volume reflects network demand, the number of active users provides insights into network load distribution, and packet loss serves as

a key reliability metric, collectively offering a comprehensive view of network performance. The dataset was also processed and segregated into training and testing subsets and the evaluation primarily focused on two metrics MAE and MAPE, as explained in detail in Sec. 3.2.1 and Sec. 3.2.4.

Tab. 3.8 and 3.9 present a comparative analysis of the performance of of FB's Prophet, LSTM networks, and GRU networks in forecasting downlink and uplink traffic volumes, respectively. The code and results are publicly available [3].

Tab. 3.8: Gained results: Comparison of network traffic volume predictions - downlink

Model	**Metric**	**Site A**	**Site B**	**Site C**	**Site D**
FB's Prophet	MAPE (%)	29.32	22.85	**13.46**	21.76
	MAE (MBs)	3687.24	2162.10	2447.59	**1485.08**
	Accuracy (%)	70.68	77.15	**86.54**	78.24
LSTM	MAPE (%)	7.01	10.04	**6.11**	6.57
	MAE (MBs)	844.31	869.26	1107.10	**449.28**
	Accuracy (%)	92.99	89.96	**93.89**	93.43
GRU	MAPE (%)	10.03	11.07	7.98	**7.79**
	MAE (MBs)	1197.33	1044.07	1328.82	**546.60**
	Accuracy (%)	89.97	88.93	92.02	**92.21**

Tab. 3.9: Gained results: Comparison of network traffic volume predictions - uplink

Model	**Metric**	**Site A**	**Site B**	**Site C**	**Site D**
FB's Prophet	MAPE (%)	62.49	31.78	**12.11**	32.54
	MAE (MBs)	844.49	448.39	**293.09**	374.35
	Accuracy (%)	37.51	68.22	**87.89**	67.46
LSTM	MAPE (%)	9.90	11.81	**7.10**	9.66
	MAE (MBs)	164.42	163.59	178.99	**107.28**
	Accuracy (%)	90.10	88.19	**92.90**	90.34
GRU	MAPE (%)	12.25	21.83	8.40	**16.23**
	MAE (MBs)	194.50	248.77	216.14	**167.68**
	Accuracy (%)	87.75	78.17	**91.60**	83.77

The analysis of the Tab. 3.8 and 3.9 reveals that the LSTM model consistently achieves the lowest MAE and MAPE values across most sites, indicating its superior ability to model and predict complex traffic patterns. For instance, in downlink traffic predictions at Site A, LSTM recorded a MAE of 844.31 MB and a MAPE of 7.01%, markedly outperforming FB's Prophet, which had a MAE of 3687.24 MB and a MAPE of 29.32%. Similarly, in uplink traffic at Site C, LSTM achieved a MAE of 178.99 MB and a MAPE of 7.10%, compared to FB's Prophet's 293.09 MB MAE and 12.11% MAPE.

The GRU model, while not as accurate as LSTM, still outperforms FB's Prophet in most scenarios, particularly in environments with stable traffic patterns. For example, in downlink traffic at Site D, GRU achieved a MAE of 546.60 MB and a MAPE of 7.79%,

[3] `https://github.com/lizzy262/PESBiU.git`

compared to FB's Prophet's 1485.08 MB MAE and 21.76% MAPE. However, GRU's performance is slightly less consistent than LSTM's, especially in more volatile environments such as Site B, where GRU exhibited a MAPE of 21.83% in uplink traffic compared to LSTM's 11.81%.

Following figures (Fig.3.9-3.13) highlight the inherent variability and trends within the dataset that the ML models must capture to make accurate predictions. However, these are not predictions itself, but rather a way to capture a trend in the dataset. Fig. 3.9 and 3.10 illustrate the average hourly downlink and uplink traffic volumes for each site, respectively. For instance, Site A and Site B, located in business districts, exhibit significant fluctuations in traffic volume throughout the day, which contribute to higher prediction errors for FB's Prophet due to its reliance on statistical trends. In contrast, Sites C and D, situated in residential areas, show more stable and predictable traffic patterns, resulting in lower error rates for all models, particularly LSTM and GRU.

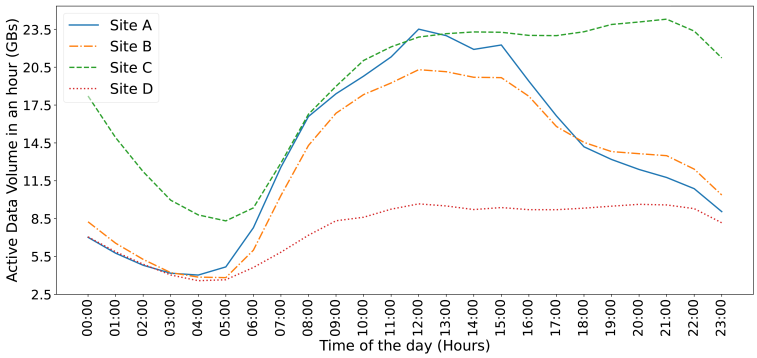


Fig. 3.9: Average hourly downlink traffic volume across sites during one day

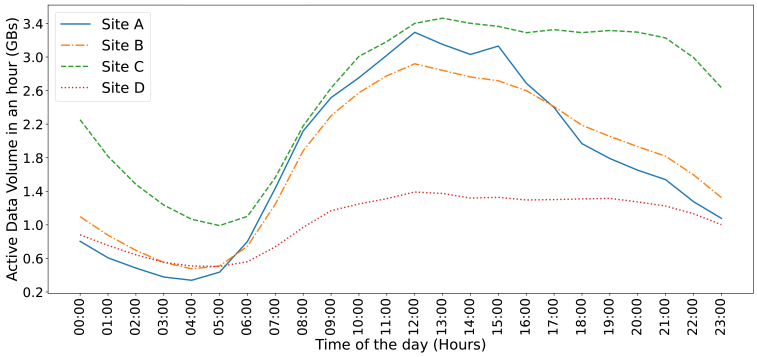


Fig. 3.10: Average hourly uplink traffic volume across sites during one day

Fig. 3.11 displays the average hourly number of active users across all sites, correlating with traffic volume trends depicted in Fig. 3.9 and 3.10. Sites with higher and more consistent number of users, such as Site C, tend to have lower prediction errors, reinforcing the effectiveness of LSTM and GRU in stable environments. Conversely, fluctuating user counts at Sites A and B contribute to higher variability in traffic, challenging the predictive capabilities of FB's Prophet more than those of LSTM and GRU.

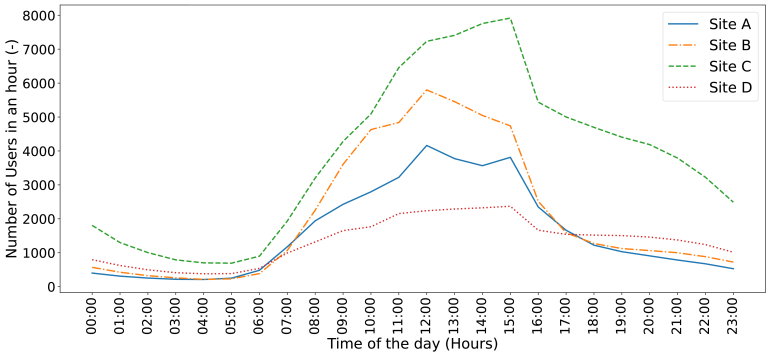


Fig. 3.11: Average hourly number of active users across sites during one day

Fig. 3.12 illustrates the average hourly packet loss across the network, which directly impacts the reliability and accuracy of traffic predictions. Higher packet loss rates, as observed in Site B during peak hours, introduce noise and inconsistencies in the data, thereby increasing the prediction errors for all models. This is particularly evident in the uplink traffic predictions, where packet loss significantly affects the reliability of the data being transmitted and subsequently predicted.

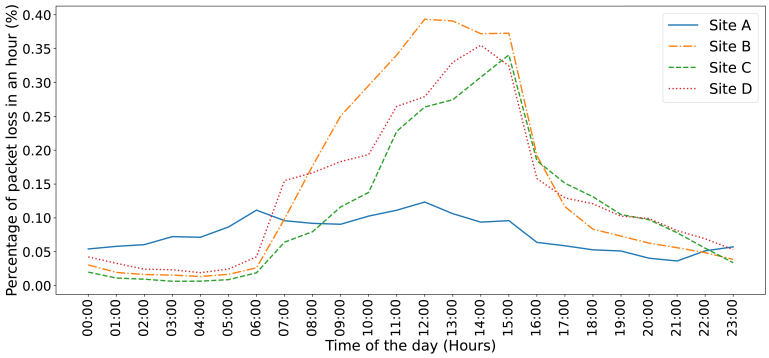


Fig. 3.12: Average hourly packet loss in the network across sites during one day

Lastly, Fig. 3.13 presents the average daily traffic volume across all sites for one week, revealing trends such as decreased traffic on weekends. Sites located in residential areas (C and D) exhibit more gradual declines compared to business districts (A and B), which experience sharper drops corresponding to typical office hours and workweek dynamics. This consistency in residential traffic patterns further explains the lower prediction errors achieved by LSTM and GRU in these areas.

To further illustrate the predictive capabilities of the models, we present visualizations of the traffic predictions over a sample week for Site A and Site B. These figures demonstrate how well each model captures the actual traffic trends.

Fig. 3.14 showcases the LSTM model's ability to accurately predict both downlink and uplink traffic volumes for Site A over a week. The close alignment between the predicted and actual values highlights the model's effectiveness in capturing the site's traffic dynamics.

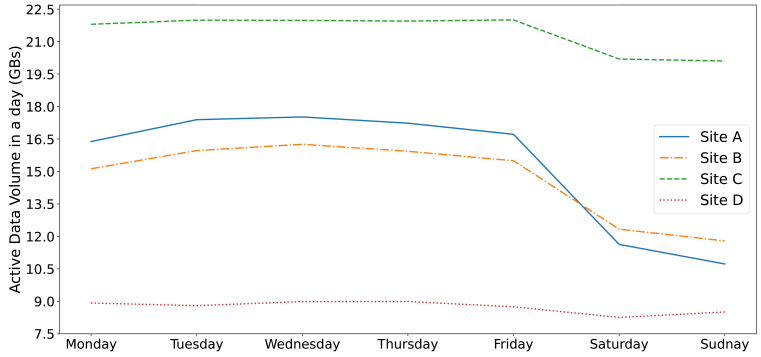


Fig. 3.13: Average daily traffic volume across sites during one week

Fig. 3.15 and 3.16 illustrates the average hourly traffic predictions for Site B using both LSTM and GRU models. Fig. 3.15 compares the individual performances of LSTM and GRU, while Fig. 3.16 demonstrates the combined implementation of both models. In this combined approach, we tried to leverage the strengths of both architectures by implementing an ensemble model where LSTM captures long-term dependencies in traffic trends, while GRU efficiently processes short-term variations with its simplified gating mechanism. The final predictions are obtained by blending the outputs of both models. The visualizations reveal that while both combined models perform well, LSTM maintains a slight edge in accuracy, particularly during peak traffic periods.

The comparative analysis of Tab. 3.8 and 3.9 alongside the visualizations in Fig. 3.9, 3.10, 3.11, 3.12, 3.13, 3.14, 3.15, and 3.16 underscores several key insights.

Firstly, the LSTM model demonstrates the highest accuracy across both downlink and uplink traffic predictions, particularly excelling in environments with complex and dynamic traffic patterns, such as Site A. Its ability to capture long-term dependencies and adapt to varying traffic conditions makes it the most reliable model among those tested. The visualizations in Fig. 3.14 corroborate this, showing how closely LSTM's predictions align with the actual traffic data over a sample week.

While the GRU model does not match LSTM's performance, it still outperforms FB's Prophet in most scenarios. GRU's simpler architecture offers computational efficiency, making it suitable for applications where resources are limited but reasonable prediction accuracy is still required. This is evident in the predictions for Site B, as shown in Fig. 3.15, where both LSTM and GRU models perform well, with LSTM maintaining a slight edge, especially during peak traffic periods.

FB's Prophet, on the other hand, significantly lags behind both LSTM and GRU in sites with high variability and packet loss, such as Site B. This highlights the limitations of traditional statistical models in handling noisy and unpredictable data, where deep learning models excel. The high MAPE and MAE values for FB's Prophet in such environments indicate its reduced effectiveness compared to more sophisticated ML approaches.

The impact of site characteristics on model performance is also evident. Sites C and D, located in residential areas, exhibit more consistent and predictable traffic patterns, resulting in lower prediction errors across all models. Fig. 3.9 and 3.10 illustrate the stable

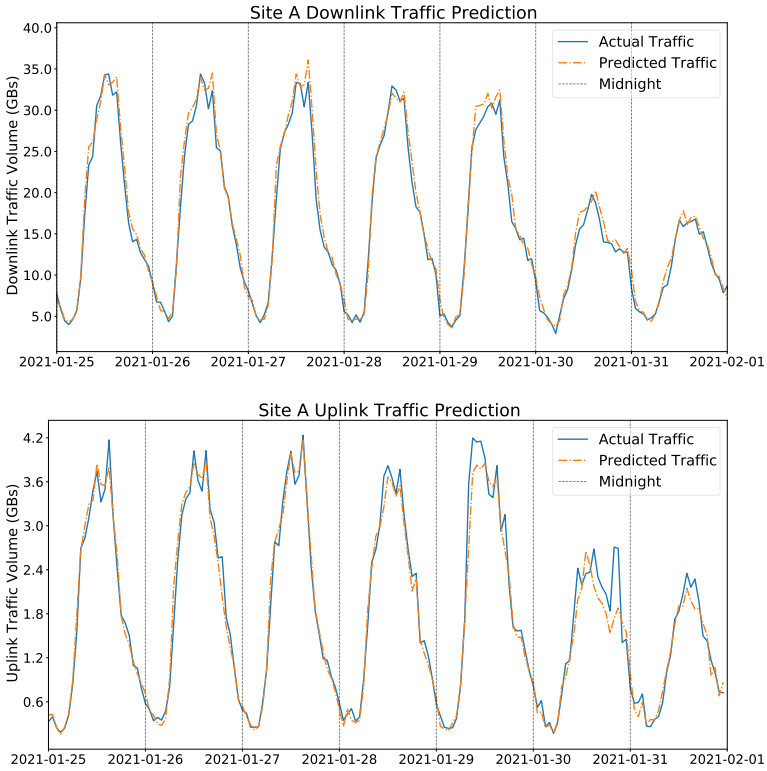


Fig. 3.14: Sample week of downlink and uplink traffic prediction for Site A using LSTM

daily traffic volumes for these sites, which contribute to the better performance of LSTM and GRU. In contrast, Sites A and B, situated in business districts, experience higher traffic variability and packet loss, as shown in Fig. 3.12, posing greater challenges for accurate predictions.

Furthermore, the distinction between downlink and uplink traffic predictions is notable. All models achieve better performance in predicting downlink traffic compared to uplink traffic. This is attributed to the more predictable nature of downlink traffic, which benefits from the temporal learning capabilities of LSTM and GRU, as well as the statistical trend analysis employed by FB's Prophet. Uplink traffic, being more susceptible to abrupt changes and packet loss, presents a more challenging prediction task, reflected in the higher MAPE and MAE values observed.

The visualizations in Fig. 3.14, 3.15 and 3.16 provide tangible evidence of the models' predictive capabilities. LSTM's close alignment with actual traffic trends in Site A and its superior performance in Site B demonstrate its robustness and adaptability. The combined implementation of LSTM and GRU in Site B, as shown in Figure 3.16, further enhances prediction accuracy, suggesting the potential benefits of ensemble approaches in complex network environments.

Overall, these findings demonstrate that while LSTM is the most effective model for complex and dynamic traffic patterns, GRU remains a viable alternative when computational efficiency is a concern. FB's Prophet, in contrast, is best suited for more stable environments, as seen in Site C. These insights are critical for optimizing network performance, especially in scenarios where traffic variability and data quality are paramount.

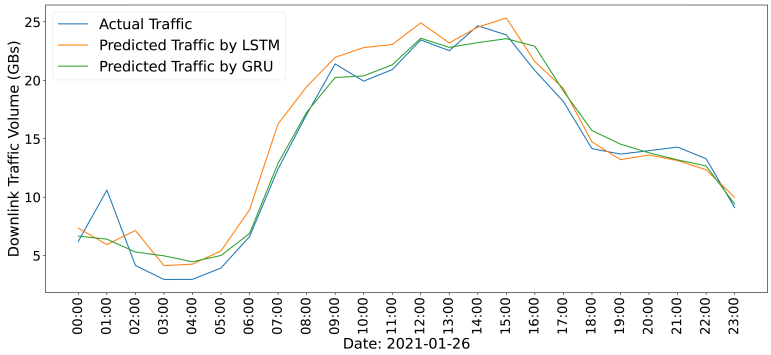


Fig. 3.15: Average hourly downlink traffic prediction using LSTM and GRU for Site B

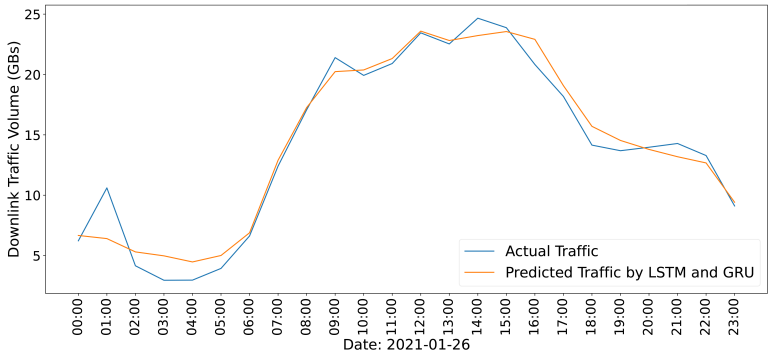


Fig. 3.16: Combined average hourly LSTM+GRU downlink traffic predictions for Site B

b) Secondary Dataset Hybrid (Long Term + Granular KPIs)

Building on the findings from the analysis of primary dataset, where the use of long-term data alone highlighted limitations in capturing short-term fluctuations, this section explores a hybrid approach by integrating granular data to address these gaps. The primary goal is to combine the strengths of long-term data, which captures overall trends, with granular data, which provides a finer temporal resolution, to improve predictive accuracy for key network metrics such as traffic volume and energy consumption in BBUs.

To evaluate this hybrid approach, four ML models (see Sec. 3.2.3 and previous Sec. a)), LSTM, CNN-LSTM, GRU and Hyper-CNN-LSTM, were applied across four configurations: (1) long-term data alone at 1-hour intervals, (2) granular data alone at 15-minute intervals, (3) a combination of long-term and granular data at 1-hour intervals, and (4) a combination of long-term and granular data at 15-minute intervals. Each configuration was analyzed to identify the optimal model and dataset pairing for improving prediction performance using this hybrid dataset structure. The following paragraphs evaluate each model's performance across these configurations.

1. **LSTM Model**: The LSTM model, known for its ability to handle temporal dependencies in sequential data, was initially tested across all configurations. While the LSTM demonstrated strong performance with long-term datasets, it struggled

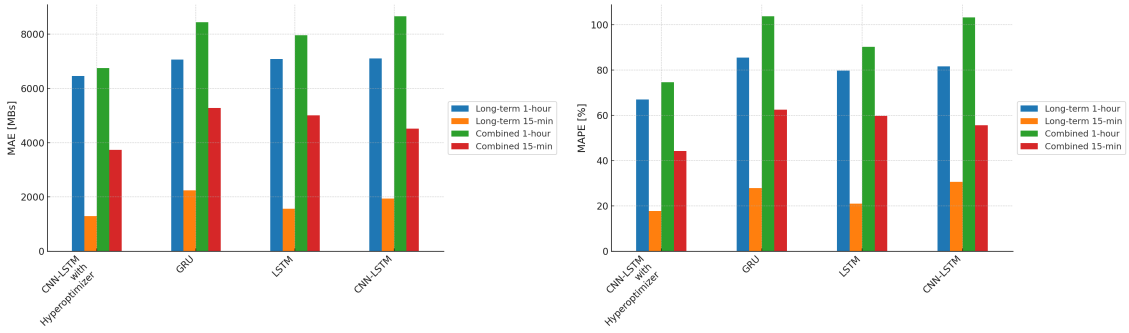
to capture rapid fluctuations in the hybrid datasets due to its sensitivity to noise and short-term variations inherent in granular data. The performance metrics, such as MAE and MAPE, indicated that the LSTM achieved its best results with long-term data alone, but did not benefit significantly from the hybrid dataset at 15-minute intervals.

2. **CNN-LSTM Model**: The CNN-LSTM architecture, which integrated convolutional layers for spatial feature extraction with LSTM layers for temporal learning, showed improved performance over the standalone LSTM. The convolutional layers effectively filtered noise and captured spatial dependencies within the hybrid dataset. This model performed particularly well with the combined long-term and granular datasets at 15-minute intervals, achieving a lower MAE and MAPE compared to the LSTM. The CNN-LSTM's ability to balance the spatial and temporal dynamics made it a strong candidate for hybrid datasets, especially for predicting BBU power consumption under varying traffic loads.
3. **GRU Model**: The GRU model, a variant of LSTM with a simplified gating mechanism, was evaluated to determine if its streamlined architecture would enhance performance with hybrid datasets. The GRU model displayed robust results with granular data alone, but its performance declined when the hybrid datasets were used. This suggested that GRU's simpler structure, while advantageous for short-term granular predictions, lacked the capacity to handle the complexity introduced by hybrid datasets that integrated multiple temporal granularities. The GRU, therefore, appears more suited for pure granular data scenarios rather than hybrid configurations.
4. **CNN-LSTM with Hyperparameter Optimization (Hyper-CNN-LSTM)**: The Hyper-CNN-LSTM, incorporating Bayesian optimization to fine-tune hyperparameters such as filter sizes, LSTM units, dropout rates, and learning rates, outperformed all other models across the four configurations. The optimization process allowed the model to adapt effectively to the varying characteristics of the hybrid datasets, balancing both short-term and long-term predictive elements. The Hyper-CNN-LSTM achieved the lowest MAE and MAPE for traffic volume and energy consumption prediction with the combined long-term and granular dataset at 15-minute intervals. The results highlighted the model's ability to integrate detailed short-term fluctuations with broader long-term trends, making it the optimal choice for hybrid dataset applications.

The comparative analysis of these models revealed that hybrid datasets, while offering potential improvements in predictive accuracy, also introduced additional complexity that not all models handled effectively. The CNN-LSTM and Hyper-CNN-LSTM architectures, with their capacity for simultaneous spatial and temporal learning, showed clear advantages in harnessing the strengths of both long-term and granular datasets. On the other hand, simpler models like LSTM and GRU were less effective, as they either overfitted to short-term fluctuations or failed to capture the detailed temporal patterns in the hybrid configurations.

The best overall performance was achieved by the Hyper-CNN-LSTM with the hy-

brid version of secondary dataset at 15-minute intervals, indicating that a finely tuned architecture could successfully integrate granular and long-term data for high-fidelity predictions. The results for all datasets and cell sites together can be seen in Fig. 3.17. It was counted by summing the absolute errors and absolute percentage errors, respectively, across all prediction steps and across all BBUs and sites, and then averaging these totals over the full evaluation period for each model.



(a) Cumulative MAE

(b) Cumulative MAPE

Fig. 3.17: Cumulative results for each method - all dataset and cell sites together

Each site had multiple BBUs: Beta (B1, B2, B3), Gama (G1, G2, G3), and Delta (D1, D2, D3). Each site represents a unique network setup with distinct traffic volume patterns, requiring specialized predictive models to optimize performance. The results for Hyper-CNN-LSTM are in Tab. 3.10.

Tab. 3.10: Traffic volume prediction results: Hyper-CNN-LSTM across secondary datasets

Site	**Long-term (1 h)**		**Combined (1 h)**		**Long-term (15 min)**		**Combined (15 min)**	
	MAE (MBs)	**MAPE (%)**	**MAE (MBs)**	**MAPE (%)**	**MAE (MBs)**	**MAPE (%)**	**MAE (MBs)**	**MAPE (%)**
Beta (B1)	112.30	2.99	347.05	6.30	**71.01**	**1.76**	152.68	2.97
Beta (B2)	146.24	1.55	399.15	3.57	**41.74**	**0.45**	134.61	1.30
Beta (B3)	113.44	1.82	250.05	4.97	**53.41**	**1.19**	137.99	2.91
Gama (G1)	179.93	3.94	320.63	4.34	**94.46**	**1.81**	177.04	2.77
Gama (G2)	305.73	2.10	842.34	4.91	**302.53**	**1.77**	463.80	2.43
Gama (G3)	229.59	6.61	695.00	11.22	**215.92**	**4.15**	346.35	6.78
Delta (D1)	254.36	2.74	2311.18	19.91	**117.40**	**1.09**	1279.34	11.67
Delta (D2)	244.34	4.17	1018.79	14.28	**197.43**	**3.27**	605.33	8.57
Delta (D3)	351.20	4.69	565.55	5.13	**202.17**	**2.33**	438.35	4.86

During the integration of long-term and granular datasets, linear interpolation was used to generate 15-minute data points from the original hourly long-term dataset. While this technique allowed the creation of a uniform temporal granularity across datasets, it introduced artificial smoothing effects, which masked short-term traffic spikes and led to suboptimal predictions in scenarios with highly variable traffic patterns. Additionally,

downsizing granular data to match the hourly intervals of the long-term dataset resulted in a loss of critical short-term dynamics, further diminishing predictive accuracy. This alignment issue highlighted the limitations of hybrid datasets, as the interpolated data could not fully replicate the detailed variations present in actual high-frequency granular data.

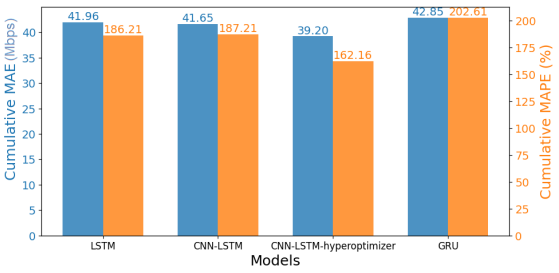
Based on our results, it was determined that the exclusive use of 15-minute granular data might be more effective for scenarios requiring high-fidelity short-term predictions, which definitely correspond to 5G+ technologies. The interpolation and aggregation techniques used to align long-term and granular data introduced inaccuracies, smoothing out essential short-term variations and impacting the model's ability to capture dynamic network behaviors. Consequently, we decided to prioritize the use of granular data at their native resolution to preserve critical traffic fluctuations, ensuring more reliable and precise predictions for network optimization.

c) Secondary Dataset (Granular KPIs)

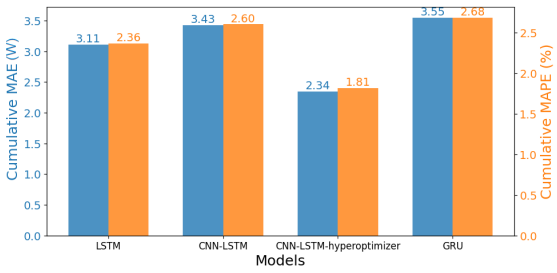
The secondary dataset, sampled at 15-minute intervals, enabled more accurate short-term predictions than in the case of primary dataset, due to its high temporal resolution. As in the case of hybrid approach, same ML models (LSTM, GRU, CNN-LSTM and Hyper-CNN-LSTM) were used and evaluated based on their performance for predicting total port throughput, energy consumption, and UE downlink latency. Each model architecture was configured to capture temporal dependencies, with LSTM and GRU utilizing two stacked layers of 50 units each and a dropout rate of 0.2 to prevent overfitting. The CNN-LSTM integrated a 1D convolutional layer with 64 filters before passing data through LSTM layers, enabling it to learn both spatial and temporal patterns.

It was observed that the CNN-LSTM model consistently outperformed both the LSTM and GRU models across all three metrics. Furthermore, the Hyper-CNN-LSTM model, achieved again the lowest MAE and MAPE. The optimized configuration included 100 LSTM units, 64 convolutional filters, a dropout rate of 0.35, and a learning rate of 0.001, enhancing the model's robustness in capturing high-frequency variations. The total cumulative MAE and MAPE for all cell sites for each model together can be seen in Fig. 3.18a for total port throughput, in Fig. 3.18b energy consumption and for UE downlink latency see Fig. 3.18c. The cumulative MAE and MAPE were calculated as the sum of absolute errors and absolute percentage errors, respectively, across all prediction steps and all BBUs per site, then aggregated over the entire test duration for each KPI.

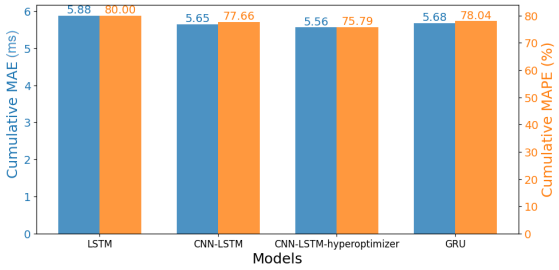
For total port throughput prediction, the Hyper-CNN-LSTM model yielded an MAE of 3.03 Mbps for BBU Beta (B2) and a MAPE of 9.18%, see Fig. 3.19. For energy consumption prediction, the best results were achieved with an MAE of 0.18 W and a MAPE of 0.16% for BBU Delta (D3), see Fig. 3.20a, but even BBU Gama (G3) achieved satisfactory results, see Fig. 3.20b. The Hyper-CNN-LSTM model also provided the highest accuracy for UE downlink latency, with an MAE of 0.49 ms and a MAPE of 6.08% for BBU Beta (B2), see Fig. 3.21a. Fig. 3.21b then shows also promising results for the UE downlink latency, with an MAE of 0.48 ms and a MAPE of 6.94% for BBU Gama (G2).



(a) Total Port Throughput Prediction



(b) Energy Consumption Prediction



(c) UE Downlink Latency Prediction

Fig. 3.18: Cumulative MAE and MAPE for each measured KPI and model for granular dataset across all cell sites

The GRU model, while less computationally intensive, showed lower predictive accuracy compared to LSTM-based models. For all the results of Hyper-CNN-LSTM, see Tab. 3.11.

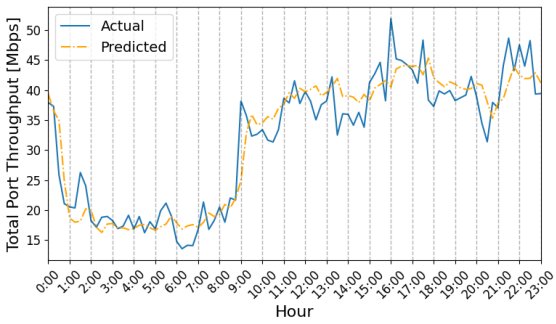
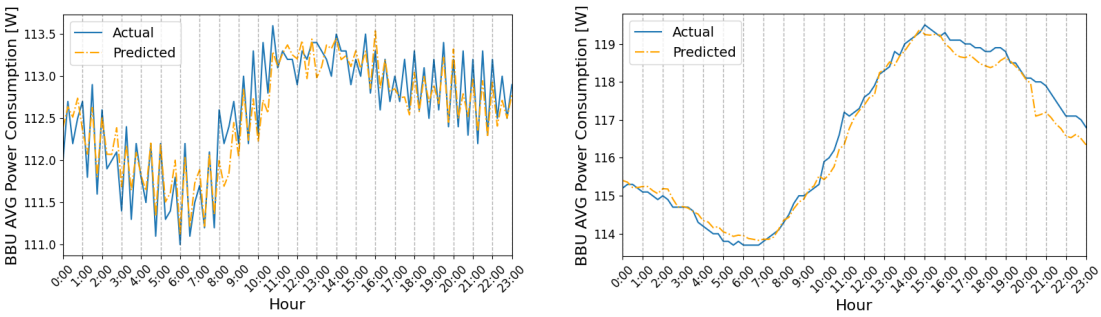


Fig. 3.19: Total port throughput prediction using Hyper-CNN-LSTM for BBU B2

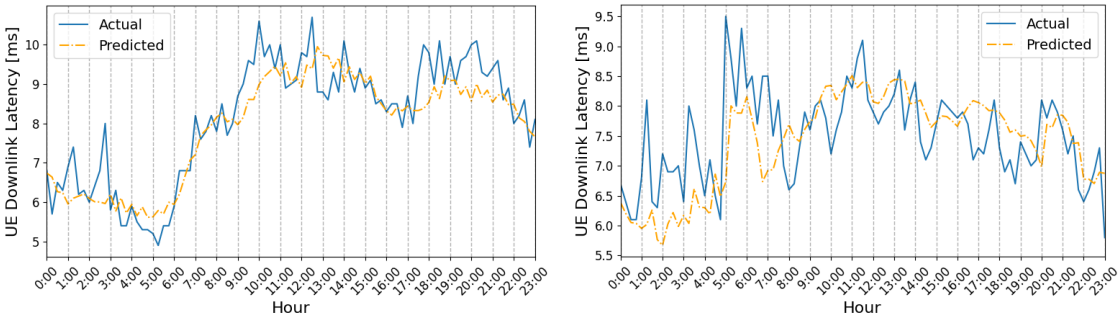
The results confirmed that Hyper-CNN-LSTM, with its ability to capture both temporal and spatial dependencies, is the most suitable model for granular datasets. The LSTM, despite being effective for some metrics, lacked the capacity to handle the short-term fluctuations as effectively as the Hyper-CNN-LSTM. Thus, we concluded that the granular dataset, in combination with a hyperparameter-optimized CNN-LSTM, is optimal for our further use in BBU energy management scenario. To support this claim that the total port throughput remains within acceptable limits, we show the worst case result for Beta (B1) reaching aprox. MAPE 22.10%, yet still closely aligned with the actual traffic, as shown in Fig. 3.22.



(a) Site Delta BBU D3

(b) Site Gama BBU G3

Fig. 3.20: Avg power consumption prediction using Hyper-CNN-LSTM



(a) Site Beta BBU B2

(b) Site Gama BBU G2

Fig. 3.21: UE downlink latency prediction using Hyper-CNN-LSTM

Tab. 3.11: Total port throughput, UE downlink latency and BBU average power consumption prediction results: Hyper-CNN-LSTM across granular dataset

BBU	**Throughput**		**Latency**		**Power Consumption**	
	MAE (Mbps)	**MAPE (%)**	**MAE (ms)**	**MAPE (%)**	**MAE (W)**	**MAPE (%)**
Beta (B1)	3.05	22.10	0.62	8.82	0.21	0.15
Beta (B2)	3.03	9.18	0.49	6.08	0.22	0.19
Beta (B3)	3.08	15.83	0.71	9.90	0.26	0.21
Gama (G1)	3.31	18.13	0.75	9.98	0.35	0.24
Gama (G2)	7.94	13.66	0.48	6.94	0.47	0.34
Gama (G3)	3.83	24.26	0.84	10.90	0.19	0.16
Delta (D1)	4.99	14.20	0.49	7.47	0.96	0.69
Delta (D2)	3.66	15.02	0.67	9.09	0.23	0.17
Delta (D3)	4.89	16.35	0.65	9.14	0.18	0.16

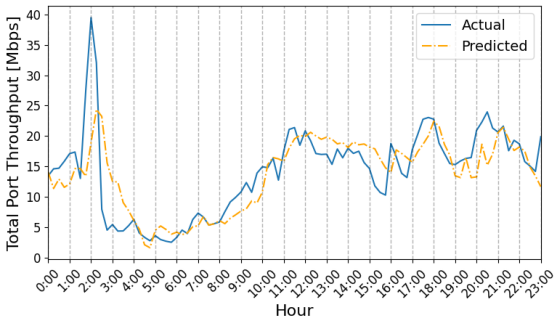


Fig. 3.22: The worst case result: Total port throughput prediction using Hyper-CNN-LSTM for BBU B1

3.3 Conclusion on Real Mobile Data and ML Predictions

This chapter on ML predictions demonstrated how diverse mobile data can improve model accuracy. The two datasets described here were collected to capture both broad temporal trends and fine-grained fluctuations. This dual focus was selected to expose short-range spikes and longer cycles, giving a richer view of traffic and energy behavior.

The use of primary dataset, logged over months, provided an extended window for understanding seasonal shifts and covered office and residential environments. Adding the secondary dataset brought high-frequency observations from multiple sites in different cities, helping to see abrupt variations and revealing sudden fluctuations that may be hidden in hourly logs. The aim was to reflect typical network conditions and gather realistic data distributions.

Model selection focused on balancing interpretability, and capacity to learn complex patterns. Recurrent networks, such as LSTM or GRU, proved to be effective for capturing multi-hour or daily sequences [86]. The LSTM model consistently outperformed others in predicting traffic volumes within the primary dataset. The GRU model also showed strong performance, though slightly less consistent than LSTM, while FB's Prophet struggled with high variability scenarios. This exploration with primary dataset motivated the adoption of more advanced CNN-LSTM and Hyper-CNN-LSTM architectures for secondary dataset, building on earlier lessons about model strengths and shortcomings. Although CNN-LSTM-based approaches performed well in granular data, Hyper-CNN-LSTM achieved even stronger performance by merging convolutional layers with recurrent layers and using tuned hyperparameters. Fig. 3.23 summarizes the datasets and models evolution.

A hybrid version of the secondary dataset with long-term and granular data was also examined. The results are publicly available [4] [87]. The combined approach exhibited promise, but interpolating hourly values into finer intervals introduced smoothing effects. This limitation affected accuracy for abrupt changes. Hyper-CNN-LSTM coped best with that challenge, yet direct use of granular data provided superior predictive power for short-term demands.

[4] `https://github.com/vafekt/Stage1_PESBiU.git`

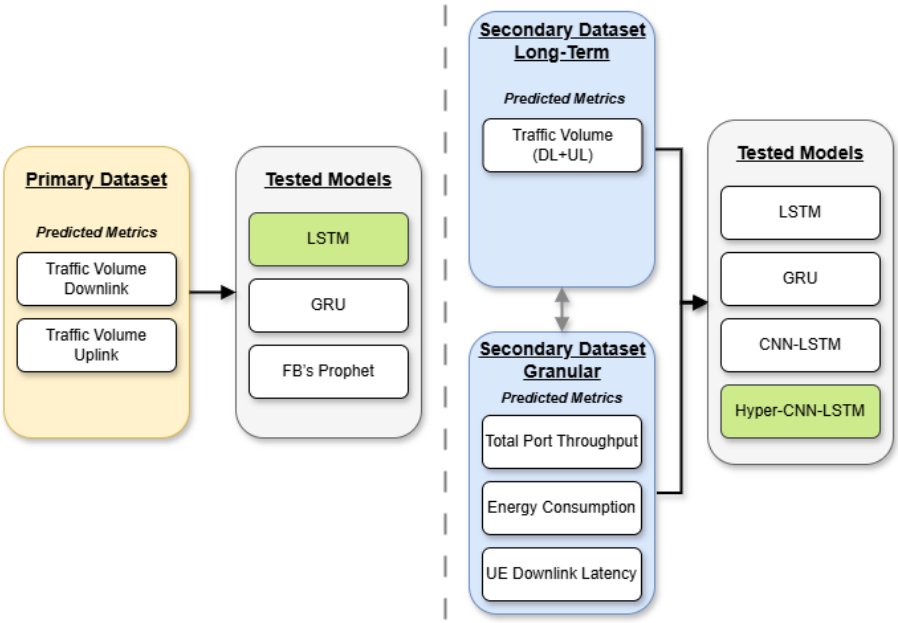


Fig. 3.23: Datasets and tested model evolution with the best performing ones highlighted by green color

This study's approach - leveraging actual operator data and adopting multiple time resolutions - was chosen so that the derived ML predictions remain relevant under changing traffic loads. The models covered varied network setups, from office blocks to urban edges. Looking ahead, the insights and results obtained from this analysis lay a solid foundation for the subsequent chapter (see Chap. 4), where the study will leverage these predictive capabilities to address energy efficiency in BBUs. Specifically, the next chapter will introduce the Predictive Energy Saving for Baseband Units (PESBiU) framework, which builds upon the predictive models developed here to proactively manage and optimize the BBU energy consumption. This progression from understanding and predicting network behavior to actively managing energy resources exemplifies the practical application of intelligent data-driven strategies to improve the sustainability and efficiency of 5G+ networks.

4 AI-Aided BBU Energy Consumption Management

As it was mentioned in Sec.2.3, energy efficiency is a critical concern in 5G networks due to the significant increase in data traffic and the need for sustainable operations. In comparison to the existing strategies detailed in Sec. 2.3, this chapter introduces the Predictive Energy Saving for Baseband Units (PESBiU), it is a method that we created, which uses predicted metrics for BBUs and which aims at saving energy. Initially, PESBiU 1.0 was created as the first version of the framework. Building on its foundation, we significantly improved the design and capabilities, resulting in the development of PESBiU 2.0. Both versions are using the predicted data[1] from the Stage 1 as described in previous Sec. 3.2. PESBiU 1.0 was created on top of the primary dataset, with long-term data. The only metric used was traffic volume (MBytes). By forecasting when traffic volume will reach specific thresholds, PESBiU 1.0 proactively transitions BBUs into sleep mode before they would do so autonomously (if at all). In PESBiU 2.0 we enhanced this framework in many ways. We used the secondary dataset that enabled us to use more granular data and on top of that we used three types of predicted metrics: total port throughput (Mbps), UE DL latency (ms) and BBU AVG power consumption (W). Moreover, we implemented an AI-aided decision engine that makes a choice when to put BBU to sleep instead of a simple threshold based on predicted metrics.

This whole BBU energy consumption management is based on real-world data, unlike other studies that rely on synthetic datasets. This fact allows us to validate the correlation between traffic, energy usage, and UE experience.

Through analyzing network traffic patterns in Sec. 3.2.5, we observe that data transfer volumes tend to decrease at predictable times, such as between 3:00 am and 5:00 am and in business districts during weekends. During these periods, we can turn off some BBUs to save energy. As detailed in Section 2.3, this strategy effectively reduces power consumption. The process involves redistributing the load from BBUs operating under low load conditions to others with available capacity, allowing some units to be shut down.

The description of the created framework is contained in the following two sections, 4.2 and 4.3. Firstly, we will describe its essentials, to understand the key notations and definitions used and we will explain the parameters and conditions upon which the algorithms operates. Then we demonstrate the efficiency of our PESBiU framework in simulated environment, using real world dataset. The results and the code itself are publicly available [2].

4.1 Selected Models - Stage 2

In Stage 2, we will move beyond predictive modeling (addressed in Stage 1, see Sec. 3.2.3) to the application of advanced RL models. Stage 1 provided accurate forecasts of network

[1] PESBiU 2.0 works also with non-predicted dataset for training the RL agents, to avoid influence by not only predicted data, but also by actual data.

[2] `https://github.com/lizzy262/PESBiU.git`

traffic, energy usage, and QoS metrics—essential inputs required by Stage 2's decision-making algorithms. Building upon these predictive capabilities, Stage 2 specifically focuses on optimizing real-time decisions for energy efficiency and resource management within 5G+ networks. This sequential structure, moving from prediction (Stage 1) to decision optimization (Stage 2), ensures precise and adaptive control over BBUs, enhancing overall network efficiency without sacrificing quality of service.

The models were chosen for their ability to optimize dynamic decisions, handle complex policy-based scenarios, and improve safety in resource allocation [99]. In this chapter, we test which model best aligns with our needs for optimizing real-time network operations, as summarized in Tab. 4.1. This section details the structure and theoretical background of each model for our AI-aided decision engine in a 5G+ network context.

Tab. 4.1: Overview of AI/ML models usage in this section

Stage	**Dataset**	**Model**	**Algorithm Type**
Stage 2	Stage 1 Outputs	DQN	Decision optimization
Stage 2	Stage 1 Outputs	DDDQN	Decision optimization
Stage 2	Stage 1 Outputs	A2C	Policy-based decision making
Stage 2	Stage 1 Outputs	Hybrid DDDQN	Safer decision optimization

a) Deep Q-Network (DQN)

DQN [47] is a RL algorithm that combines Q-learning with deep neural networks. Q-learning is a value-based method used to learn the optimal policy by estimating the Q-values, which represent the expected cumulative future reward for taking an action a in a given state s and following a policy π thereafter. The key idea behind DQN is to use a deep neural network as a function approximator to estimate the Q-values. The Q-value for a given state-action pair (s_t, a_t) is updated using the Bellman equation as follows:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right], \tag{4.1}$$

where: $Q(s_t, a_t)$ is the current Q-value for state s_t and action a_t, α is the learning rate, r_t is the reward received at time t, γ is the discount factor, which determines the importance of future rewards, s_{t+1} is the next state, a' is the action taken in the next state, and $\max_{a'} Q(s_{t+1}, a')$ is the maximum Q-value for the next state.

DQN uses two key innovations: experience replay and a target network. Experience replay involves storing past experiences (s_t, a_t, r_t, s_{t+1}) in a replay buffer and sampling mini-batches for training, which reduces correlations between consecutive updates. The target network, a separate copy of the Q-network, is used to compute the target Q-values $r_t + \gamma \max_{a'} Q(s_{t+1}, a')$, which stabilizes learning by holding the target Q-values constant for several iterations [47].

b) Dueling Double Deep Q-Network (DDDQN)

DDDQN[3] [100] is an enhancement of the traditional DQN that addresses the problem of overestimation in Q-values and improves learning stability. DDDQN employs two separately created improvements: Double Q-Learning and Dueling Network Architecture.

1. Double Q-Learning [101]: In DQN, the max operator in $\max_{a'} Q(s_{t+1}, a')$ can lead to overestimation of Q-values. To counter this, Double Q-learning uses two separate networks: the online network and the target network. The action selection is done using the online network, but the Q-value for that action is evaluated using the target network. Thus, the updated rule becomes:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_t + \gamma Q_{\text{target}}(s_{t+1}, \arg\max_{a'} Q_{\text{online}}(s_{t+1}, a')) - Q(s_t, a_t) \right]. \quad (4.2)$$

2. Dueling Network Architecture [48]: The dueling architecture decomposes the Q-value into two separate estimators: the state-value function $V(s)$ and the advantage function $A(s, a)$. This separation allows the network to learn which states are valuable regardless of the action taken. The final Q-value is computed as:

$$Q(s, a) = V(s) + A(s, a) - \frac{1}{|\mathcal{A}|} \sum_{a'} A(s, a'), \quad (4.3)$$

where $\frac{1}{|\mathcal{A}|} \sum_{a'} A(s, a')$ is the mean advantage across all actions. This modification leads to more stable and accurate value estimates, making DDDQN particularly useful in scenarios with complex state-action spaces, such as 5G network resource allocation.

c) Advantage Actor-Critic (A2C)

A2C [102] is a policy gradient method that combines the benefits of both policy-based and value-based RL. In A2C, two neural networks are trained simultaneously: the actor and the critic. The actor network $\pi(a|s; \theta)$ outputs the probability distribution over actions in a given state, and the critic network $V(s; \theta_v)$ estimates the state value function.

1. Advantage Function: A2C improves the stability of policy gradients by using the advantage function $A(s, a)$ instead of the raw return:

$$A(s_t, a_t) = r_t + \gamma V(s_{t+1}; \theta_v) - V(s_t; \theta_v). \quad (4.4)$$

2. Policy Update: The actor network is updated using the advantage function:

$$\mathcal{L}_{\text{policy}} = -\sum_t \log \pi(a_t|s_t; \theta) A(s_t, a_t), \quad (4.5)$$

where $\log \pi(a_t|s_t; \theta)$ is the log-probability of taking action a_t in state s_t.

3. Value Update: The critic network is updated by minimizing the Temporal Difference (TD) error:

[3] In some other research papers this combination of Double DQN (DDQN) and Dueling DQN is also called D3QN.

$$\mathcal{L}_{\text{value}} = \frac{1}{2} \sum_{t} \left[r_t + \gamma V(s_{t+1}; \theta_v) - V(s_t; \theta_v) \right]^2 . \tag{4.6}$$

4. Entropy Regularization: To encourage exploration, an entropy term is added to the loss function:

$$\mathcal{L}_{\text{entropy}} = -\beta \sum_{t} H(\pi(s_t; \theta)), \tag{4.7}$$

where $H(\pi)$ is the entropy of the policy and β is a scaling factor. The total loss for A2C is:

$$\mathcal{L} = \mathcal{L}_{\text{policy}} + \mathcal{L}_{\text{value}} - \beta \cdot \mathcal{L}_{\text{entropy}}. \tag{4.8}$$

A2C's simultaneous optimization of actor and critic networks leads to better convergence and stability, making it also well-suited for resource management in dynamic and complex environments such as 5G networks.

4.1.1 Evaluation of RL Models

Evaluating RL models differs fundamentally from evaluating time-series prediction models due to the dynamic nature of the environment and the agent's interaction with it. In RL, the primary objective is not only to make accurate predictions but to optimize a sequence of actions that maximize a cumulative reward. Therefore, traditional metrics like MAE and MAPE are insufficient for evaluating RL models, as they fail to capture the sequential dependencies and the interaction between actions and states.

Instead, RL models are typically evaluated using cumulative reward, policy stability, and constraint satisfaction. However, there is also an effective method of validating RL models through heuristic-based approaches. These approaches serve as a comparative baseline, where a predefined set of "correct" actions or rules is used to ensure the RL agent's behavior adheres to expected outcomes. Such heuristic methods allow for verifying the correctness of the learned policy without the need for explicit train-test-validation splits, which is a standard practice in supervised learning models [60].

The evaluation of RL models often centers on the expected cumulative reward, $\mathbb{E}[R]$, defined as:

$$\mathbb{E}[R] = \sum_{t=0}^{T} \gamma^t r_t, \tag{4.9}$$

where r_t is the reward at time step t, $\gamma \in [0, 1]$ is the discount factor, which determines the weight of future rewards and T is the time horizon over which rewards are accumulated.

The goal is to learn a policy π that maximizes $\mathbb{E}[R]$. Heuristic-based approaches use static or rule-based policies π_h to verify whether the learned policy π achieves comparable or superior cumulative rewards under similar conditions. This form of evaluation can be expressed mathematically as:

$$\Delta\mathbb{E}[R] = \mathbb{E}[R_\pi] - \mathbb{E}[R_{\pi_h}], \tag{4.10}$$

where $\mathbb{E}[R_\pi]$ is the expected reward obtained by the RL policy and $\mathbb{E}[R_{\pi_h}]$ is the expected reward obtained by the heuristic policy. If $\Delta\mathbb{E}[R] \geq 0$, then the RL policy is considered valid against the heuristic benchmark. This approach ensures that the RL model adheres to the predefined "correct" set of actions and does not deviate into suboptimal or unsafe behaviors [60].

Several studies have explored validating RL agents through heuristic methods. For instance, persistent rule-based reinforcement learning (IntRL) systems have been used to verify RL agents by providing rule-based advice during training, ensuring the agent's policy aligns with safety and performance expectations [103]. Similarly, the use of safety shields in RL incorporates static rules to act as fallback policies to prevent the agent from taking actions that violate critical constraints [104]. There are also other works, such as the study [105], which argue for systematic evaluation procedures to ensure fairness and reliability in comparing RL algorithms against heuristic counterparts.

Furthermore, hybrid approaches, such as case-based reasoning (CBR) combined with RL, demonstrate how heuristic rules can be used iteratively to refine the agent's learning process. In such systems, the RL agent is guided by a case base of predefined rules, and the policy is evaluated against these rules at each decision point [106].

One significant advantage of using heuristic-based validation is that it eliminates the need for train-test-validation splits, as is typical in supervised learning. Since the RL agent's performance is compared against a static policy, the evaluation can use the entire dataset without reserving separate subsets for testing [106, 107].

The validation process can thus be expressed as a comparison of the RL policy π against the heuristic policy π_h:

$$\text{Validation Score} = \sum_{t=0}^{T} \mathbb{I}(\pi(s_t) = \pi_h(s_t)), \tag{4.11}$$

where $\mathbb{I}$ is an indicator function that returns 1 if the RL agent's action $\pi(s_t)$ matches the heuristic action $\pi_h(s_t)$ for state s_t, and 0 otherwise. A high validation score indicates that the RL agent's policy aligns closely with the heuristic baseline.

4.2 PESBiU 1.0

PESBiU 1.0 is a first version of the Predictive Energy Saving for Baseband Units framework, that is using predicted traffic volume (MBytes) in hourly intervals from primary dataset and puts BBU to sleep via thresholds [86]. To approximate the logic used, it is important to note that while all BBUs utilize the same technology, variations such as the number of connected RRHs, microchip differences, or connection types can affect power consumption. This variability is evident at Site B in the Table 4.2, where despite BBU_1 having the highest power consumption, it handles the lowest traffic volume. Nevertheless, traffic flow and power consumption generally remain correlated, as shown in Fig. 4.1 and 4.2. The summary of observed values can be seen in the Table 4.2, which summarizes total traffic volume, average traffic volume, average power consumption, and consumed energy for the BBUs at Site B.

The specific power consumption characteristics of each BBU, determined by setup and hardware, ensure that despite lower traffic volume, BBU_1's higher power usage does not impact the PESBiU algorithm's decision-making process. The algorithm prioritizes traffic volume predictions using the LSTM model in case of PESBiU 1.0, which inherently aligns with power consumption. However, these specific power consumption characteristics are utilized in PESBiU 2.0 algorithm for even more precise energy optimization.

Tab. 4.2: Power behavior of BBUs for Site B (primary dataset) over a selected week

	Total traffic volume [GB]	**AVG traffic volume [GB]**	**AVG Power Consumption [W]**	**Consumed energy [Wh]**
BBU_1	1690	10.06	**136.77**	**5747.75**
BBU_2	**2530**	**15.05**	114.51	4812.00
BBU_3	1701	10.12	117.70	4951.50

Fig. 4.1 and 4.2 illustrate the average power consumption and consumed energy, respectively, in relation to traffic volume over a week, from primary dataset for Site B. The graphs display data from Monday to Sunday, showing the correlation between traffic volume and power usage for each BBU. While a general dependency on traffic can be observed, the strength of this correlation appears weaker for some BBUs. This suggests that although traffic load influences power usage, other factors such as static hardware characteristics or baseline operational requirements play a significant role, particularly in non-virtualized BBUs with less dynamic workload profiles.

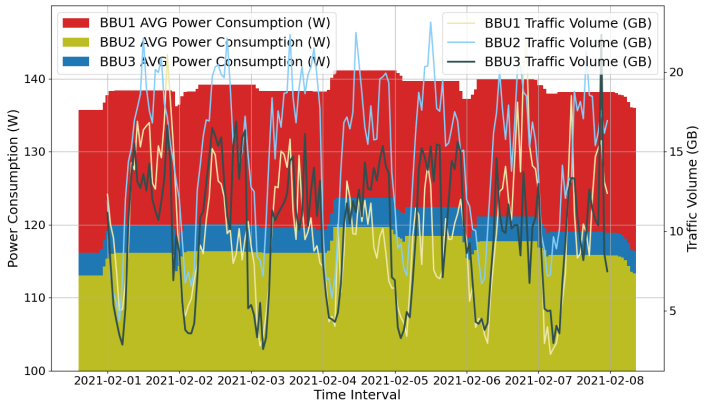


Fig. 4.1: Hourly AVG power consumption with traffic volume during one week for BBUs on Site B

For clarity, Fig. 4.3 presents a detailed view of the data during one workday (Tuesday), highlighting the specific power consumption and traffic volume correlation for that day. This finer resolution highlights minor variations that might be masked in the weekly view and is referenced later in Sec. 4.2.2.

Above we illustrated traffic patterns regards to energy consumption throughout the one week and one specific day from primary dataset. The following Sec. 4.2.1, will deal with mathematical notations explanation of the parameters and conditions upon which the PESBiU 1.0 framework operates.

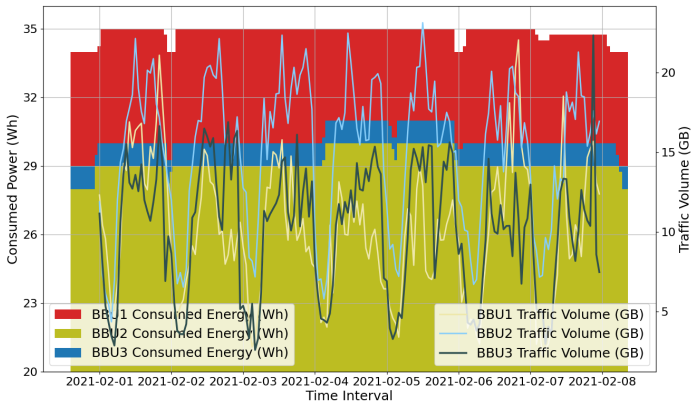


Fig. 4.2: Hourly consumed energy with traffic volume during one week for BBUs on Site B

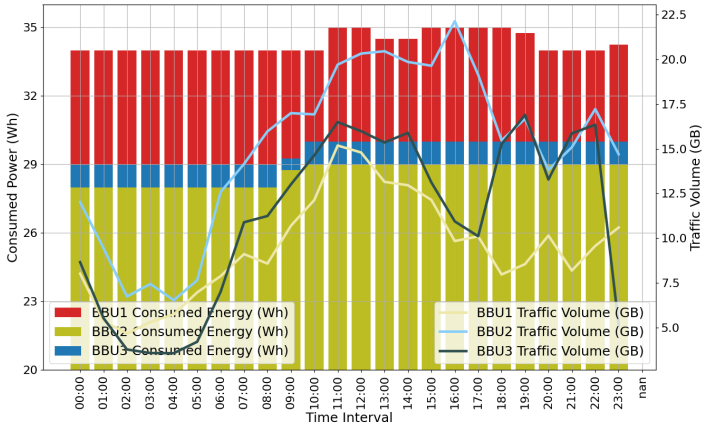


Fig. 4.3: Hourly consumed energy with traffic volume during one workday for BBUs on Site B

4.2.1 Notations, Definitions and Algorithm Description

Used variables include:

- $BBUs$: Set of BBUs $\{BBU_1, BBU_2, BBU_3\}$ [-]
- C_i : Capacity of BBU_i based on analysis (C_j is capacity of covering BBUs) [MBytes]
- T_{total} : Total traffic capacity, $T_{\text{total}} = \sum_{i=1}^{3} C_i$ [MBytes]
- T_{pred} : Predicted total traffic [MBytes]
- T_{pred_i} : Predicted traffic for BBU_i [MBytes]
- T_{actual} : Actual total traffic [MBytes]
- T_i : Current traffic of BBU_i [MBytes]
- Status_i : Status of BBU_i, can be “active” or “sleep”

The algorithm uses rules for traffic levels for the predicted total traffic volume and individual BBU traffic predictions to classify the traffic levels, which influences the decision by which BBUs can be put to sleep.

- Traffic Level for Cell Site - the overall predicted traffic level, which is derived from [108] (can also be changed, if necessary, on the basis of provider’s request) and is categorized as follows:

$$\begin{cases} \text{Low Traffic} & \text{if } T_{\text{pred}} < 0.3\,T_{\text{total}} \\ \text{Medium Traffic} & \text{if } 0.3\,T_{\text{total}} \leq T_{\text{pred}} \leq 0.7\,T_{\text{total}} \\ \text{High Traffic} & \text{if } T_{\text{pred}} > 0.7\,T_{\text{total}}. \end{cases}$$

- Traffic Level for Each BBU - each BBU's predicted traffic level is classified as:

$$\begin{cases} \text{BBU.Low Traffic} & \text{if } T_{pred_i} < 0.3\,C_i \\ \text{BBU.Medium Traffic} & \text{if } 0.3\,C_i \leq T_{pred_i} \leq 0.7\,C_i \\ \text{BBU.High Traffic} & \text{if } T_{pred_i} > 0.7\,C_i. \end{cases}$$

There is a Safe Switch Mechanism applied to prevent undesirable behavior in cases where traffic predictions may significantly deviate from real-time traffic conditions.

- Traffic Difference Monitoring - to ensure stability and avoid unpredicted changes, the algorithm monitors the difference between actual and predicted traffic:

$$\Delta T = \left| \frac{T_{\text{actual}} - T_{\text{pred}}}{T_{\text{pred}}} \right|.$$

- Threshold for Safe Switch - if the relative traffic difference ΔT exceeds 10%, all BBUs remain operational. This threshold was chosen as a conservative baseline to effectively balance energy savings and network stability, based on an analysis of real-world network traffic fluctuations observed in the primary dataset. This initial value can be changed based on the operator requirements:

$$\Delta T > 0.1 \implies \text{All BBUs remain operational.}$$

Implementation of PESBiU 1.0 Framework

The PESBiU Framework, see Alg. 1, effectively utilizes the defined notations and traffic level classifications to manage the power states of BBUs for optimized energy usage. It starts by predicting the total and individual BBU traffic (T_{pred} and T_{pred_i}), and classifies the overall traffic into Low, Medium, or High levels based on defined thresholds.

Depending on the traffic level, the framework selects BBUs with the lowest predicted traffic for potential sleep mode, setting individual wake-up thresholds at 30% of their capacities. Before deactivating selected BBUs, the framework verifies through a redistribute function that other active BBUs can comfortably absorb the redistributed traffic. This ensures that the added traffic does not cause these active BBUs to exceed their own thresholds or compromise network performance.

To maintain network stability, if the actual traffic differs from predicted values by more than 10% (which can be modified, depending on the operator's preference), no BBUs enter sleep mode, keeping the network fully operational to handle unpredicted traffic fluctuations.

Finally, the framework primarily uses predicted traffic data to reactivate any sleeping BBU once forecasts show that its load exceeds the predefined 30% wake-up threshold. Active BBUs or associated RRUs monitor actual traffic during sleep mode and triggers

immediate activation if real-time loads approach or exceed predictions. This ensures that the system stays responsive, prevents performance degradation, and reduces unnecessary energy consumption.

Algorithm 1 PESBiU 1.0 Framework

Input: T_{total}, T_{pred}, C_i for $i = 1, 2, 3$
Output: Updated statuses of BBUs ('sleep' or 'active')
Detecting Opportunity:
$T_{\text{pred}} = \text{PredictTotalTraffic}()$
$T_{\text{pred}_i} = \text{PredictTrafficForBBU}(i)$
$$\text{TrafficLevel} = \begin{cases} \text{L} & \text{if } T_{\text{pred}} < 0.3\,T_{\text{total}} \\ \text{M} & \text{if } 0.3\,T_{\text{total}} \leq T_{\text{pred}} \leq 0.7\,T_{\text{total}} \\ \text{H} & \text{if } T_{\text{pred}} > 0.7\,T_{\text{total}} \end{cases}$$
if TrafficLevel = L **then**
    Identify BBUs i_1, i_2, i_3 such that $T_{\text{pred}_{i_1}} \leq T_{\text{pred}_{i_2}} \leq T_{\text{pred}_{i_3}}$
    BBUsForSleep $\leftarrow \{i_1, i_2\}$
else if TrafficLevel = M **then**
    Identify BBUs i_1, i_2, i_3 such that $T_{\text{pred}_{i_1}} \leq T_{\text{pred}_{i_2}} \leq T_{\text{pred}_{i_3}}$
    BBUsForSleep $\leftarrow \{i_1\}$
else
    BBUsForSleep $\leftarrow \emptyset$
end if
for i in BBUsForSleep **do**
    WakeUpThreshold$_i \leftarrow 0.3C_i$
end for
Validation Check of Covering BBUs:
for i in BBUsForSleep **do**
    for j in BBUs $\setminus \{i\}$ **do**
        if Status$_j \neq$ sleep **and** $T_{pred_j} + T_{pred_i} <$ WakeUpThreshold$_j$ **and** WakeUpThreshold$_j \geq 0.7\,C_j$ **then**
            CoveringBBUs $\leftarrow$ CoveringBBUs $\cup \{j\}$
        end if
    end for
end for
Entering Sleep Mode:
for i in BBUsForSleep **do**
    Transfer T_{pred_i} to one of CoveringBBUs
    Status$_i \leftarrow$ sleep
end for
Safe Switch Mechanism:
if $\Delta T > 0.1$ **then**
    All BBUs remain operational
end if
Wake-Up BBUs:
for j in BBUs $\setminus$ BBUsForSleep **do**
    if $T_{pred_j} + T_{pred_i} >$ WakeUpThreshold$_j$ for any i in BBUsForSleep **then**
        Status$_i \leftarrow$ active
    end if
end for
for i in BBUsForSleep **do**
    if $T_{pred_i} >$ WakeUpThreshold$_i$ **then**
        Status$_i \leftarrow$ active
    end if
end for=0

4.2.2 Evaluation of PESBiU 1.0 Framework

To validate the PESBiU 1.0 framework, we tested it on a specific workday (Tuesday) within the selected week to evaluate potential energy savings. As illustrated in Fig. 4.3,

which utilizes values obtained through a standard function. The predicted data for the simulation were generated using LSTM recurrent neural networks. By forecasting future network traffic load and applying the PESBiU 1.0 framework, we were able to estimate the necessary number of BBUs required to operate the network efficiently without under-utilizing available BBUs.

Our observations included instances where one BBU was offline, and the remaining two BBUs had to compensate by covering its data traffic. This scenario enabled us to determine a coefficient representing the increase in energy consumption due to traffic re-distribution, set at 0.05 for every 1 percent increase in covered traffic. This coefficient was incorporated into our simulation to adjust the calculated energy consumption accurately.

The graphical results in Fig. 4.4 demonstrate that between 2:00 am and 5:00 am, it was possible to put two BBUs into sleep mode, reducing energy consumption from 273 Wh to 90 Wh during that 3-hour period. For the total of 5 hours, the network operated with only two BBUs. Overall, the application of the PESBiU 1.0 algorithm resulted in an energy saving of 332.35 Wh for 24 hours, which corresponds to a 15.12 % reduction for the entire day. This estimation is based on an assumed BBU capacity of 25 GB per hour, derived from weekly observations.

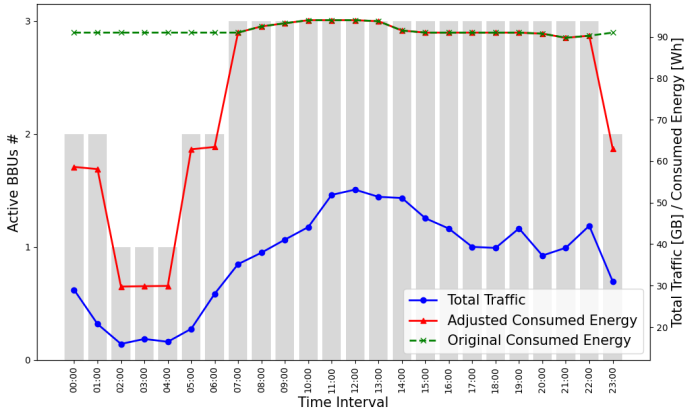


Fig. 4.4: Hourly predicted number of active BBUs, traffic volume, and energy consumption over one workday using PESBiU 1.0 with LSTM

4.3 PESBiU 2.0

PESBiU 2.0 builds on its predecessor PESBiU 1.0 and enhances the framework from all perspectives. First of all, it works with secondary dataset granular 15 mins data using Hyper-CNN-LSTM prediction compared to hourly data from primary dataset using LSTM prediction in PESBiU 1.0, which enables more accurate short-term predictions due to its higher temporal resolution.

Secondly, the change of dataset brings also another improvement. We use total port throughput (Mbps) instead of total traffic volume (MBytes) (their difference is explained in detail in Sec. 3.1.1), which gives us a more complex metric. Another difference is the use of BBU AVG power consumption (W) instead of consumed energy (Wh), due to the higher fluctuation since we lower the time interval. The last predicted metric that is used in PESBiU 2.0 is UE DL latency (ms), which gives us the QoS control check.

The biggest enhancement, however, is the use of AI-aided engine for making the decision when to put BBU to sleep instead of simple threshold based on predicted metrics. PESBiU 2.0 is an agentic system[4] that is trained on secondary granular dataset using actual and predicted data to efficiently manage network resources, particularly BBUs, to optimize throughput and latency while minimizing energy consumption. Training on actual data allowed agents to learn genuine relationships and patterns in network behavior. Moreover, with combination with predicted data, it enables agents to improve agents' robustness to data inaccuracies and ensure they can handle predicted data effectively.

PESBiU 2.0 builds on these improvements with a modular design that ties together data collection, prediction, and decision-making, as shown in Fig. 4.5. To explain the general structure, the primary modules include:

- **Data Acquisition Module:** This module gathers granular data at 15-minute intervals from operational KPIs of BBUs and RRUs. The collected metrics include traffic volume, UE latency, and energy consumption.
- **Prediction Module:** Advanced ML models, such as a hybrid CNN-LSTM architecture, are used to forecast traffic patterns and energy consumption under varying network loads.
- **Decision Engine:** An AI-aided decision-making mechanism optimizes the power states (active or sleep) of BBUs based on the input of predicted traffic and real-time KPI data. Techniques such as DDDQN and hybrid variants with threshold-based or fallback strategies ensure robustness against traffic spikes. The decision engine can be considered as an agent.
- **Control Module:** This module manages the implementation of energy-saving actions based on agent decision, dynamically adjusting the operational state of BBUs to improve energy savings while meeting QoS requirements.

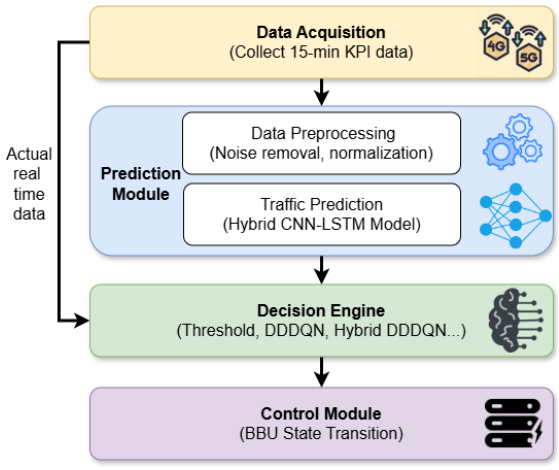


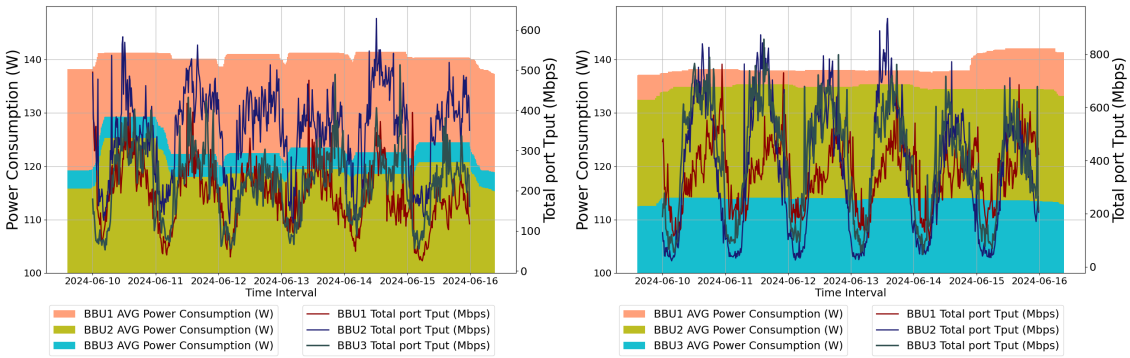
Fig. 4.5: General structure of PESBiU 2.0: The illustration of system workflow, detailing data acquisition, preprocessing and traffic prediction, decision-making and BBU state transitions through the control module

[4]Agentic system refers to the system's ability to act autonomously in optimizing energy consumption decisions for BBUs using RL.

4.3.1 Traffic Patterns and Correlations Across Cell Sites

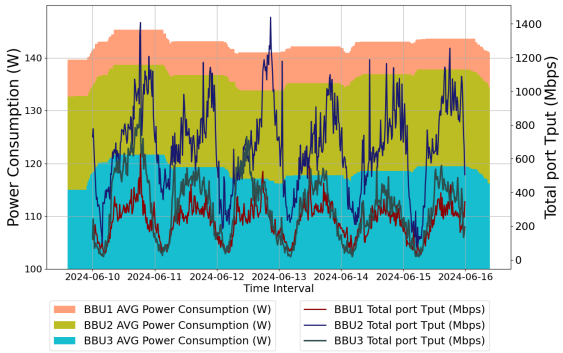
In PESBiU 2.0, we pretrained three agents to manage BBU energy-saving decisions by learning optimal sleep and active state transitions based on predicted/actual traffic patterns. Each agent was tailored for a specific location type: Remote area (Site Beta), Office area (Site Delta), and Residential area (Site Gama). This enables mobile operators to leverage transfer learning, eliminating the need to train agents from scratch in similar environments.

Fig. 4.6 illustrates the average power consumption in relation to Total Traffic Throughput over a week, from secondary granular dataset for Site Beta, Delta and Gama, respectively. The graphs display data from Monday to Sunday, showing the correlation between Traffic Throughput and power usage for each BBU.



(a) Site Beta

(b) Site Delta

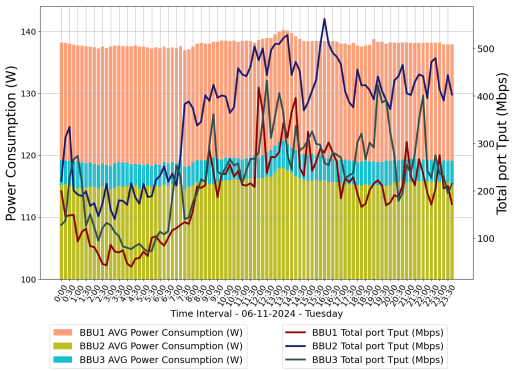


(c) Site Gama

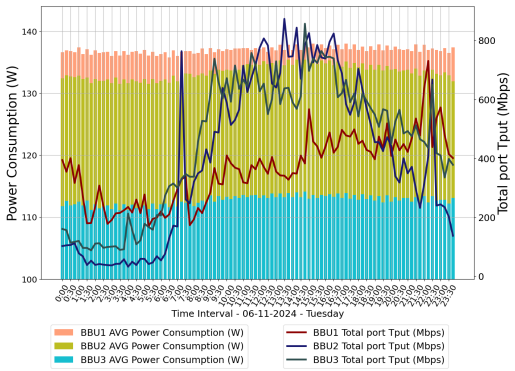
Fig. 4.6: Hourly AVG power consumption with total traffic throughput during one week for BBUs on site: (a) Beta (b) Delta (c) Gama

On the other hand, Fig. 4.7 shows the average power consumption, respectively, in relation to Total Traffic Throughput over one workday (Tuesday), from secondary granular dataset for all sites: Beta, Delta and Gama. The graphs display data for all 24 hours, every 15 min, showing the correlation between traffic throughput and power usage for each BBU.

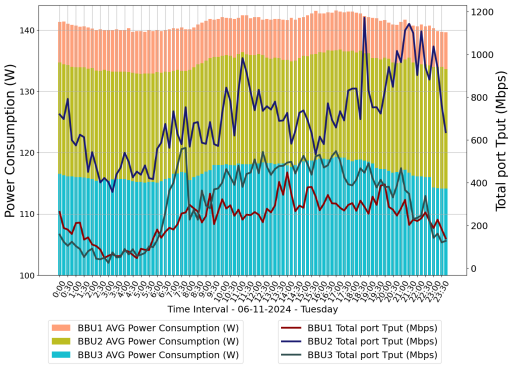
The last set of figures in Fig. 4.8 shows the UE downlink latency, in relation to Total Traffic Throughput over Tuesday, from secondary granular dataset for all sites: Beta, Delta and Gama. The graphs display data for all 24 hours, every 15 min.



(a) Site Beta

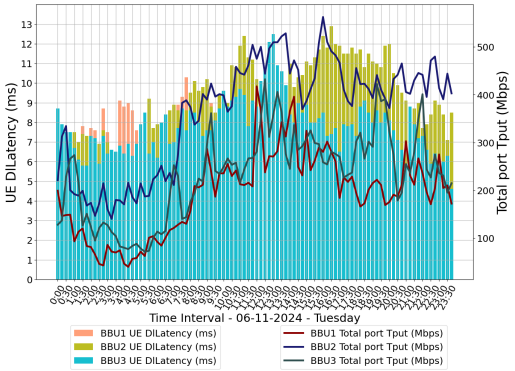


(b) Site Delta

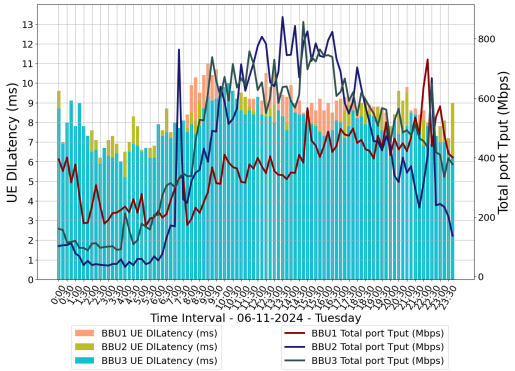


(c) Site Gama

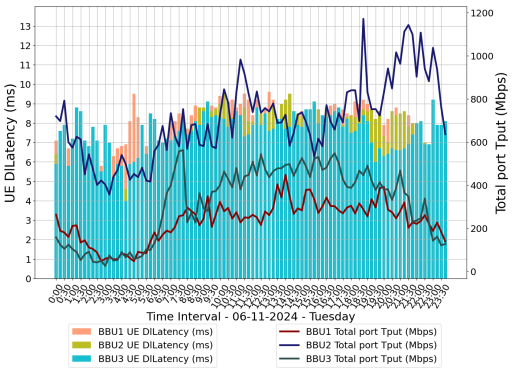
Fig. 4.7: Quarter-hourly AVG power consumption with total traffic throughput during one workday for BBUs on site: (a) Beta (b) Delta (c) Gama



(a) Site Beta



(b) Site Delta



(c) Site Gama

Fig. 4.8: Quarter-hourly UE downlink latency and total traffic throughput during one workday for BBUs on site: (a) Beta (b) Delta (c) Gama

4.3.2 Notations, Definitions and Algorithm Description

This section introduces the mathematical notations and clearly defines the parameters integral to the PESBiU 2.0 framework, which was explicitly designed, developed, and implemented as part of this thesis. PESBiU 2.0 comprises multiple algorithmic components, each precisely defined and explained both mathematically and in narrative form to showcase its innovative approach to network energy efficiency optimization.

Notations and definitions relevant to PESBiU 2.0 are defined as follows:

- $\mathcal{S}$: Set of cell sites, $\mathcal{S} = \{B, D, G\}$, where B is Beta, D is Delta and G is Gama.
- For each site $s \in \mathcal{S}$:
  - $\mathcal{B}_s$: Set of BBUs at site s, $\mathcal{B}_s = \{\text{BBU}_{s,1}, \text{BBU}_{s,2}, \text{BBU}_{s,3}\}$ [-].
  - $C_{s,i}$: Maximum capacity of $\text{BBU}_{s,i}$ after applying utilization factors:

$$C_{s,i} = \text{MaxThroughput}_{s,i} \cdot \text{max\_utilization} \cdot \text{coefficient} \ [Mbps] \tag{4.12}$$

  - $\text{Status}_{s,i}(t)$: Status of $\text{BBU}_{s,i}$ at time t, where $\text{Status}_{s,i}(t) \in \{0, 1\}$ (1 for active, 0 for sleep).
  - $T_{s,i}^{\text{pred}}(t)$: **Predicted** throughput [Mbps] of $\text{BBU}_{s,i}$ at time t.
  - $T_{s,i}^{\text{act}}(t)$: **Actual** throughput [Mbps] of $\text{BBU}_{s,i}$ at time t.
  - $E_{s,i}(t)$: Energy consumption [W] of $\text{BBU}_{s,i}$ at time t.
  - $L_{s,i}(t)$: Latency [ms] of $\text{BBU}_{s,i}$ at time t.
  - $k_{s,i}$, $E_{\text{base},s,i}$: Energy coefficients determined via linear regression for $\text{BBU}_{s,i}$.
  - $m_{s,i}$, $L_{\text{base},s,i}$: Latency coefficients determined via linear regression for $\text{BBU}_{s,i}$.
- Coefficients and Parameters:
  - α: Weight for energy savings [-].
  - β: Weight for latency penalty [-].
  - γ: Weight for overload penalty [-].
  - δ: Base overload prevention bonus [-].
  - θ: Threshold for safe switch mechanism [%].
  - $\theta_{s,i}(t)$: Adaptive fallback threshold, a time-varying threshold for site s and BBU i at time t (optional, see below).
  - MAX_LATENCY_INCREASE: Maximum allowable latency increase per step [ms].
  - $T_{\text{MAX\_OVERLOAD}}$: Maximum allowable overload traffic [Mbps].

The first crucial step within the PESBiU 2.0 framework is the state representation for each agent, see Algorithm 2.1. The state observation algorithm, explicitly designed and implemented as part of this thesis, collects crucial temporal and operational metrics for real-time decision-making. This procedure begins by taking the current time step t and relevant input data, including predicted throughput $T_{s,i}^{\text{pred}}(t)$, latency $L_{s,i}(t)$, energy consumption $E_{s,i}(t)$, and the status $\text{Status}_{s,i}(t)$ of each BBU. The decision to use these specific metrics was driven by their proven correlation with network performance and energy consumption, as demonstrated by preliminary analyses in Sec. 3.2.2. For each site s, the algorithm first observes the temporal features, specifically the decimal representation of the hour, the current day of the week, and the current month. Then it proceeds to record the predicted throughput, latency, and energy consumption for each of the three

BBUs at the site. Additionally, it observes the operational status (active or sleep) of each BBU. The algorithm compiles the gathered information into a state vector for the site. This state vector contains the time features, the predicted throughput, latency, energy and status of each BBU, forming the observed state $S_s(t)$, which is used for further decision-making processes.

Algorithm 2.1. State Observation for PESBiU 2.0

Input: Time t, Predicted throughput $T_{s,i}^{\text{pred}}(t)$, Latency $L_{s,i}(t)$, Energy $E_{s,i}(t)$, Status $\text{Status}_{s,i}(t)$
Output: Observed state $S_s(t)$ for each site s

for each site s **do**
  Observe current time features:
    $\text{Hour\_decimal}(t)$, $\text{Day}(t)$, $\text{Month}(t)$
  Observe BBUs' predicted throughput, latency, energy:
    $\left\{T_{s,i}^{\text{pred}}(t),\ L_{s,i}(t),\ E_{s,i}(t)\right\}_{i=1}^{3}$
  Observe BBUs' status:
    $\{\text{Status}_{s,i}(t)\}_{i=1}^{3}$
  Formulate state vector:
    $S_s(t) = \left[\text{Hour\_decimal}(t),\ \text{Day}(t),\ \text{Month}(t),\ \left\{T_{s,i}^{\text{pred}}(t),\ L_{s,i}(t),\ E_{s,i}(t)\right\}_{i=1}^{3},\ \{\text{Status}_{s,i}(t)\}_{i=1}^{3}\right]$
end for=0

The Safe Switch Mechanism (Fallback) described in Alg. 2.2 - which was also designed specifically in the framework of this thesis - maintains network stability by addressing significant discrepancies that occasionally arise between predicted and actual throughput values. This algorithm was created to enhance reliability, especially in situations of inaccurate predictions, by introducing a safeguard that prevents energy-saving mechanisms from negatively impacting network performance. Two options were created: A) static threshold (simpler) and B) adaptive threshold (more sophisticated), described below.

- Option A (Static): A constant threshold θ (e.g., 10%) is used. Whenever the relative deviation exceeds θ, the fallback immediately activates all BBUs at that site.
- Option B (Adaptive): Instead of a fixed θ, we define a time-varying threshold $\theta_{s,i}(t)$ that adapts to historical forecast accuracy and time-of-day patterns. This reduces unnecessary triggers when forecasts are stable, but lowers the threshold during periods of high error.

The algorithm receives three inputs: the predicted throughput $T_{s,i}^{\text{pred}}(t-1)$, actual throughput $T_{s,i}^{\text{act}}(t-1)$, and either a static or adaptive threshold θ (or $\theta_{s,i}(t)$). For each site s and each BBU i, the algorithm calculates the relative difference $\Delta_{s,i}(t-1)$ between predicted and actual throughput at the previous time step $(t-1)$ as follows:

$$\Delta_{s,i}(t-1) \;=\; \left| \frac{T_{s,i}^{\text{act}}(t-1) \;-\; T_{s,i}^{\text{pred}}(t-1)}{T_{s,i}^{\text{pred}}(t-1)} \right|. \tag{4.13}$$

The rationale behind calculating this relative difference is to provide a clear, normalized measure of prediction accuracy, directly guiding the activation of the fallback mechanism. If $\Delta_{s,i}(t-1)$ surpasses the threshold (or adaptive threshold), the fallback mechanism is triggered. This sets all BBUs at the affected site to active:

$$\text{Status}_{s,i}(t) \;=\; 1, \quad \forall\, i \in \{1,2,3\}. \tag{4.14}$$

To refine fallback activation for option B (adaptive threshold), one may define:

$$\theta_{s,i}(t) \;=\; \theta_0 \;-\; \alpha_{\text{acc}} \cdot \big(\text{AvgErr}(t,w)\big) \;-\; \alpha_{\text{tod}} \cdot \text{TodF}(t), \tag{4.15}$$

where θ_0 is a baseline threshold, $\text{AvgErr}(t,w)$ is the mean forecast error over the past w intervals, $\text{TodF}(t)$ accounts for time-of-day variations (e.g., lower values at night). The coefficients α_{acc} and α_{tod} determine the sensitivity to forecast errors and time-of-day adjustments, respectively. This allows the fallback threshold to increase in stable low-error scenarios (reducing false triggers) and decrease when forecast errors escalate.

Overall, the Safe Switch Mechanism ensures that no BBU is inadvertently put to sleep under conditions of considerable predictive error, thus proactively preserving system stability. Upon triggering this mechanism, the algorithm immediately terminates further evaluation within the current loop. This is a deliberate implementation choice to efficiently avoid redundant checks and rapidly respond to critical discrepancies.

Algorithm 2.2. Safe Switch Mechanism (Fallback) for PESBiU 2.0

Input: Predicted throughput $T_{s,i}^{\text{pred}}(t-1)$, Actual throughput $T_{s,i}^{\text{act}}(t-1)$, Static threshold θ or Adaptive threshold $\theta_{s,i}(t)$
Output: Updated status $\text{S}_{s,i}(t)$
for each site s **do**
  for each BBU $i \in \{1,2,3\}$ **do**
    Compute the relative difference:

$$\Delta_{s,i}(t-1) \;=\; \left| \frac{T_{s,i}^{\text{act}}(t-1) \;-\; T_{s,i}^{\text{pred}}(t-1)}{T_{s,i}^{\text{pred}}(t-1)} \right|.$$

    Option A (Static Threshold):
    Use a constant threshold θ.
    if $\Delta_{s,i}(t-1) > \theta$ **then**
      Fallback Triggered: Activate all BBUs at site s
      $\text{S}_{s,i}(t) \leftarrow 1 \quad \forall\, i \in \{1,2,3\}$
      Break from loop
    end if

    Option B (Adaptive Threshold):
    Use a time-varying threshold $\theta_{s,i}(t)$, computed as:

$$\theta_{s,i}(t) \;=\; \theta_0 \;-\; \alpha_{\text{acc}} \cdot \Big(\text{AvgErr}(t,w)\Big) \;-\; \alpha_{\text{tod}} \cdot \text{TodF}(t),$$

    Compare:

$$\Delta_{s,i}(t-1) \;>\; \theta_{s,i}(t).$$

    if $\Delta_{s,i}(t-1) \;>\; \theta_{s,i}(t)$ **then**
      Adaptive Fallback Triggered: Activate all BBUs at site s
      $\text{S}_{s,i}(t) \leftarrow 1 \quad \forall\, i \in \{1,2,3\}$
      Break from loop
    end if
  end for
end for=0

The Action Selection, see Alg. 2.3, was created to determine the optimal configuration of active and sleep states for the BBUs at each site. The algorithm utilizes inputs including

the observed state $S_s(t)$, the explicitly defined action space $\mathcal{A}$, and the learned policy π, which maps states to actions. For each site s, if the Safe Switch Mechanism (Fallback) has not been activated, the agent autonomously selects the optimal action $A_s(t)$ from the defined action set $\mathcal{A}$, based on the current observed state $S_s(t)$ and policy $\pi(S_s(t))$. This action specifies a particular combination of active and sleep states for the three BBUs at the site. The algorithm then iterates through each BBU i and updates its status $\text{Status}_{s,i}(t)$ based on the selected action's mapping. Specifically, the BBU statuses are set according to the predefined mappings in the ActionMapping function. This function assigns binary statuses (1 for active, 0 for sleep) to the BBUs, precisely reflecting the chosen action $A_s(t)$.

The action set was designed and implemented as follows:

$$\mathcal{A} = \{a_0, a_1, a_2, a_3, a_4, a_5, a_6\} \tag{4.16}$$

with explicitly defined BBU status mappings:

$$\begin{aligned}
a_0 &: [1,0,0] \quad \text{(Only BBU1 active)} \\
a_1 &: [0,1,0] \quad \text{(Only BBU2 active)} \\
a_2 &: [0,0,1] \quad \text{(Only BBU3 active)} \\
a_3 &: [1,1,0] \quad \text{(BBU1 and BBU2 active)} \\
a_4 &: [1,0,1] \quad \text{(BBU1 and BBU3 active)} \\
a_5 &: [0,1,1] \quad \text{(BBU2 and BBU3 active)} \\
a_6 &: [1,1,1] \quad \text{(All BBUs active)}
\end{aligned} \tag{4.17}$$

This particular design ensures each BBU precisely transitions to the appropriate state, in line with the agent's real-time predictive-driven policy decisions, marking a step forward from conventional static or heuristic-driven approaches.

Algorithm 2.3. Action Selection for PESBiU 2.0

Input: Observed state $S_s(t)$, Action space $\mathcal{A}$, Policy π
Output: Selected action $A_s(t)$, Updated status $\text{Status}_{s,i}(t)$

for each site s **do**
  if Fallback is **not** triggered **then**
    Agent selects action $A_s(t) \in \mathcal{A}$ based on policy $\pi(S_s(t))$
    for each BBU $i \in \{1,2,3\}$ **do**
      Update BBU status according to action mapping:

$$\text{Status}_{s,i}(t) = \text{ActionMapping}(A_s(t),\ i)$$

    end for
  end if
end for=0

The Environment Update and Traffic Redistribution, see Alg. 2.4 presents an approach developed explicitly to dynamically manage traffic redistribution when BBUs at a site enter a sleep state. The algorithm begins by using as input the current BBU statuses $\text{Status}_{s,i}(t)$, capacities $C_{s,i}$, and actual throughput $T^{\text{act}}_{s,i}(t)$ for each BBU. For each site s, the algorithm first calculates the total traffic that needs to be redistributed, defined as

the sum of the respective actual throughput of BBUs that are currently in a sleep state:

$$T_{\text{redistribute},s}(t) = \sum_{\substack{i \\ \text{Status}_{s,i}(t)=0}} T_{s,i}^{\text{act}}(t). \tag{4.18}$$

Next, it computes the available capacity in the active BBUs, which is determined by subtracting the current throughput from the maximum capacity for each active BBU:

$$C_{\text{available},s}(t) = \sum_{\substack{i \\ \text{Status}_{s,i}(t)=1}} \left(C_{s,i} - T_{s,i}^{\text{act}}(t) \right). \tag{4.19}$$

The algorithm then determines if there is any overload traffic that cannot be accommodated by the active BBUs, calculated as:

$$T_{\text{overload},s}(t) = \max\left(T_{\text{redistribute},s}(t) - C_{\text{available},s}(t),\ 0 \right). \tag{4.20}$$

If the available capacity is greater than zero and there is traffic to be redistributed, the algorithm redistributes the traffic proportionally among the active BBUs. For each active BBU i, the additional throughput $\Delta T_{s,i}(t)$ to be assigned is computed based on its available capacity relative to the total available capacity:

$$\Delta T_{s,i}(t) = \frac{C_{s,i} - T_{s,i}^{\text{act}}(t)}{C_{\text{available},s}(t)} \cdot T_{\text{redistribute},s}(t). \tag{4.21}$$

This additional throughput is then capped to ensure it does not exceed the BBU's remaining capacity, noting that the full capacity is typically not utilized in practice and may be further limited based on operator-defined thresholds:

$$\Delta T_{s,i}(t) \leftarrow \min\left(\Delta T_{s,i}(t),\ C_{s,i} - T_{s,i}^{\text{act}}(t) \right). \tag{4.22}$$

After updating the throughput for each active BBU, the algorithm computes the corresponding latency increase $\Delta L_{s,i}(t)$. This latency management component is computed based on redistributed throughput, using a predefined linear latency coefficient $m_{s,i}$ that directly links additional traffic to potential latency impact:

$$\Delta L_{s,i}(t) = m_{s,i} \cdot \Delta T_{s,i}(t). \tag{4.23}$$

Latency increases are carefully limited to avoid sudden spikes, while maintaining consistent user experience:

$$\Delta L_{s,i}(t) \leftarrow \min\left(\Delta L_{s,i}(t),\ \text{MAX\_LATENCY\_INCREASE} \right). \tag{4.24}$$

The new latency value is then updated accordingly:

$$L_{s,i}^{\text{after}}(t) \leftarrow L_{s,i}(t) + \Delta L_{s,i}(t). \tag{4.25}$$

Finally, the energy consumption is updated based on a linear relation between throughput and energy, using the base energy $E_{\text{base},s,i}$ and a coefficient $k_{s,i}$:

$$E_{s,i}^{\text{after}}(t) \leftarrow E_{\text{base},s,i} + k_{s,i} \cdot T_{s,i}^{\text{after}}(t). \tag{4.26}$$

If the available capacity cannot accommodate the redistributed traffic, the algorithm flags that the overload traffic cannot be handled. For BBUs that are put to sleep (status set to zero), the throughput and energy consumption are reset to zero:

$$T_{s,i}^{\text{after}}(t) \leftarrow 0, \quad E_{s,i}^{\text{after}}(t) \leftarrow 0. \tag{4.27}$$

This ensures that inactive BBUs do not consume any energy or handle traffic during their sleep state.

Algorithm 2.4. Environment Update and Traffic Redistribution for PESBiU 2.0

Input: BBU statuses $\text{Status}_{s,i}(t)$, Capacities $C_{s,i}$, Actual throughput $T^{\text{act}}_{s,i}(t)$
Output: Updated throughput $T^{\text{after}}_{s,i}(t)$, Latency $L^{\text{after}}_{s,i}(t)$, Energy $E^{\text{after}}_{s,i}(t)$

for each site s **do**

Compute total traffic to redistribute:

$$T_{\text{redistribute},s}(t) = \sum_{\substack{i \\ \text{Status}_{s,i}(t)=0}} T^{\text{act}}_{s,i}(t)$$

Compute available capacity in active BBUs:

$$C_{\text{available},s}(t) = \sum_{\substack{i \\ \text{Status}_{s,i}(t)=1}} \left(C_{s,i} - T^{\text{act}}_{s,i}(t)\right)$$

Compute overload traffic:

$$T_{\text{overload},s}(t) = \max\left(T_{\text{redistribute},s}(t) - C_{\text{available},s}(t),\ 0\right)$$

if $C_{\text{available},s}(t) > 0$ **and** $T_{\text{redistribute},s}(t) > 0$ **then**
  for each active BBU i **do**
    Compute additional throughput:

$$\Delta T_{s,i}(t) = \frac{C_{s,i} - T^{\text{act}}_{s,i}(t)}{C_{\text{available},s}(t)} \cdot T_{\text{redistribute},s}(t)$$

    Cap to available capacity:

$$\Delta T_{s,i}(t) \leftarrow \min\left(\Delta T_{s,i}(t),\ C_{s,i} - T^{\text{act}}_{s,i}(t)\right)$$

    Update throughput:

$$T^{\text{after}}_{s,i}(t) \leftarrow T^{\text{act}}_{s,i}(t) + \Delta T_{s,i}(t)$$

    Compute latency increase:

$$\Delta L_{s,i}(t) = m_{s,i} \cdot \Delta T_{s,i}(t)$$

    Cap latency increase:

$$\Delta L_{s,i}(t) \leftarrow \min\left(\Delta L_{s,i}(t),\ \text{MAX\_LATENCY\_INCREASE}\right)$$

    Update latency:

$$L^{\text{after}}_{s,i}(t) \leftarrow L_{s,i}(t) + \Delta L_{s,i}(t)$$

    Update energy consumption:

$$E^{\text{after}}_{s,i}(t) \leftarrow E_{\text{base},s,i} + k_{s,i} \cdot T^{\text{after}}_{s,i}(t)$$

  end for
else
  Overload traffic cannot be handled
end if
for each BBU i with $\text{Status}_{s,i}(t) = 0$ **do**
  Set:

$$T^{\text{after}}_{s,i}(t) \leftarrow 0, \quad E^{\text{after}}_{s,i}(t) \leftarrow 0$$

end for
end for=0

Our unique Reward Calculation, presented in Alg. 2.5, which was created and implemented, is responsible for evaluating the performance of the agents at each site based on energy savings, latency penalties, and overload management. The algorithm was designed to use as input the energy consumption values before and after ($E_{s,i}(t)$ and $E_{s,i}^{\text{after}}(t)$), throughput values before and after ($T_{s,i}^{\text{act}}(t)$ and $T_{s,i}^{\text{after}}(t)$), latency values before and after ($L_{s,i}(t)$ and $L_{s,i}^{\text{after}}(t)$), and the overload traffic $T_{\text{overload},s}(t)$.

First, to measure how much energy is saved, we sum the total energy consumption of all BBUs at site s before the action, and subtract from it the total energy consumption after the action. This idea is captured by:

$$E_{\text{savings},s}(t) = \sum_i E_{s,i}(t) - \sum_i E_{s,i}^{\text{after}}(t). \tag{4.28}$$

This difference directly shows the total power reduction achieved by our decision at the site in one control interval.

Next, we seek a value that is not just about raw energy savings but is also scaled by the original (baseline) total energy use. Therefore, the normalized energy savings (NES) is:

$$\text{NES}_s(t) = \frac{E_{\text{savings},s}(t)}{\sum_i E_{s,i}(t)}. \tag{4.29}$$

Dividing the energy saved at the site by the total baseline energy consumption helps to place each site's energy reduction on a comparable scale, allowing us to fairly measure improvements regardless of the site's overall power level.

We also need to gauge whether these energy savings come at the cost of poorer latency. To do that, we first compute how much the total latency has changed after taking our action. We weight the latency by the respective throughput, reflecting how many users or data streams might be affected by the delay. Concretely:

$$\Delta L_{\text{total},s}(t) = \sum_i T_{s,i}^{\text{after}}(t) L_{s,i}^{\text{after}}(t) - \sum_i T_{s,i}^{\text{act}}(t) L_{s,i}(t). \tag{4.30}$$

This term highlights whether our action triggered any increase in overall latency that would degrade the QoS.

To compare this latency change to the original situation at the site, we construct a normalized latency penalty (NLP):

$$\text{NLP}_s(t) = \frac{\Delta L_{\text{total},s}(t)}{\sum_i T_{s,i}^{\text{act}}(t) L_{s,i}(t)}. \tag{4.31}$$

Here, we divide the increase in total latency by the initial total latency, weighted by each BBU's previous throughput. This yields a relative measure, allowing us to assess the severity of the latency increase compared to the site's original baseline.

In addition to managing energy and latency, we also need a way to reflect any traffic overload that the network cannot serve. Therefore, we compute a normalized overload penalty (NOP):

$$\text{NOP}_s(t) = \min\left(\frac{T_{\text{overload},s}(t)}{T_{\text{MAX\_OVERLOAD}}},\ 1.0\right). \tag{4.32}$$

This expression caps the penalty at 1.0 to keep it within practical bounds. Here, $T_{\text{MAX\_OVERLOAD}}$ is the maximum allowable or "reference" overload traffic for one step, ensuring that extreme values do not overwhelm other reward terms.

We also reward decisions that completely avoid overload. When $T_{\text{overload},s}(t)$ is zero, the agent has successfully steered the site away from overload, which we incentivize through an overload prevention bonus (OPB). This bonus grows further if more BBUs are in a sleep state while still meeting the traffic demands. Formally:

$$\text{OPB}_s(t) = \begin{cases} \delta \left(1 + \dfrac{N_{\text{inactiveBBUs},s}(t)}{N_{\text{totalBBUs}}}\right), & \text{if } T_{\text{overload},s}(t) = 0 \\ 0, & \text{otherwise.} \end{cases} \tag{4.33}$$

This approach rewards the agent for achieving both: no congestion (thus optimal QoS) and energy savings (by deactivating some BBUs). The factor δ scales the weight of this bonus in the overall reward.

Finally, we combine all these terms into the total reward $R_s(t)$:

$$R_s(t) = \alpha \cdot \text{NES}_s(t) - \beta \cdot \text{NLP}_s(t) - \gamma \cdot \text{NOP}_s(t) + \text{OPB}_s(t). \tag{4.34}$$

We choose α, β, γ to reflect the relative importance of energy savings, latency penalty, and overload penalty. The reward thus balances multiple objectives: it increases with energy savings, decreases if latency or overload traffic gets worse, and expands if we avoid overload altogether. These coefficients give network operators a way to tune the agent's emphasis, for instance, to prioritize QoS or to meet stringent energy-saving targets.

The values of α, β, γ, and δ are constants that set how important each component is within the reward function. These coefficients were already introduced in the notations and definitions in the beginning, see Sec. 4.3.2. The following paragraphs will explain in greater detail why we designed them in this way to clarify how they reflect industry practices, regulatory requirements, and operational experience.

These predefined constants are set as follows:

- $\alpha = 0.4$: Scales how much energy savings contribute to the reward.
- $\beta = 0.2$: Sets how heavily latency penalties reduce the reward.
- $\gamma = 1.0$: Assigns high weight to overload penalties.
- $\delta = 0.12$: Determines the baseline bonus for overload prevention.
- $\theta = 0.5$: Threshold to decide a "safe-switch" mechanism.
- MAX_LATENCY_INCREASE = 10: Upper bound on tolerable per-step latency increase.
- $T_{\text{MAX\_OVERLOAD}} = 10$: Maximum allowable overload traffic.

In the subsequent discussion, we provide a more specific rationale for each setting:

1. **Alignment with Industry Standards:**
   Meeting recommendations from organizations such as 3GPP Release 15+ [109] and ITU IMT-2030 [110] ensures that energy efficiency, latency, and throughput remain balanced. Our choices for α, β, γ, and δ conform to guidelines that encourage dynamic, adaptive power control while safeguarding service integrity.

2. **Operational Priorities and Economic constraints:**
   Energy consumption is a major expense for operators [111], intensifying the need for cost-effective solutions. We place stronger emphasis on energy savings in α, consistent with heightened economic pressure and broader environmental goals.
3. **QoS Safeguards:**
   Although energy saving is prioritized, the reward function also keeps a focus on throughput and latency (with β), ensuring that user experience remains acceptable [112]. QoS thresholds from the ITU [110] and internal service-level agreements (SLAs) serve as guardrails to prevent under-provisioning.
4. **Empirical Validation:**
   Extensive tests on real network data helped refine the exact values. Trials [5] showed that $\alpha = 0.4$ yields substantial reductions in power usage without triggering unacceptable latency levels. Similarly, values for β and γ were iteratively calibrated to consistently avoid overload.
5. **Flexibility and Adaptability:**
   The final parameter set (θ, MAX_LATENCY_INCREASE, $T_{\text{MAX\_OVERLOAD}}$, etc.) can be readily updated to address future demands, such as stricter latency needs in URLLC scenarios or new regulatory limits on energy consumption [43].

In this way, our weighting scheme is neither arbitrary nor static. It is the outcome of adhering to recognized standards, matching economic realities, validating against real-world data, and ensuring the framework can evolve with shifting network priorities.

[5] `https://github.com/lizzy262/PESBiU.git`

Algorithm 2.5. Reward Calculation for PESBiU 2.0

Input: Energy $E_{s,i}(t)$, $E_{s,i}^{\text{after}}(t)$, Throughput $T_{s,i}^{\text{act}}(t)$, $T_{s,i}^{\text{after}}(t)$, Latency $L_{s,i}(t)$, $L_{s,i}^{\text{after}}(t)$, Overload traffic $T_{\text{overload},s}(t)$

Output: Reward $R_s(t)$

1: **for** each site s **do**

2: Compute energy savings:

$$E_{\text{savings},s}(t) = \sum_i E_{s,i}(t) - \sum_i E_{s,i}^{\text{after}}(t)$$

3: Compute normalized energy savings:

$$\text{NES}_s(t) = \frac{E_{\text{savings},s}(t)}{\sum_i E_{s,i}(t)}$$

4: Compute total latency increase:

$$\Delta L_{\text{total},s}(t) = \sum_i T_{s,i}^{\text{after}}(t) L_{s,i}^{\text{after}}(t) - \sum_i T_{s,i}^{\text{act}}(t) L_{s,i}(t)$$

5: Compute normalized latency penalty:

$$\text{NLP}_s(t) = \frac{\Delta L_{\text{total},s}(t)}{\sum_i T_{s,i}^{\text{act}}(t) L_{s,i}(t)}$$

6: Compute normalized overload penalty:

$$\text{NOP}_s(t) = \min\left(\frac{T_{\text{overload},s}(t)}{T_{\text{MAX\_OVERLOAD}}},\ 1.0\right)$$

7: Compute overload prevention bonus:

$$\text{OPB}_s(t) = \begin{cases} \delta\left(1 + \dfrac{N_{\text{inactiveBBUs},s}(t)}{N_{\text{totalBBUs}}}\right), & \text{if } T_{\text{overload},s}(t) = 0 \\ 0, & \text{otherwise} \end{cases}$$

8: Compute total reward:

$$R_s(t) = \alpha \cdot \text{NES}_s(t) - \beta \cdot \text{NLP}_s(t) - \gamma \cdot \text{NOP}_s(t) + \text{OPB}_s(t)$$

9: **end for**=0

The Agent Learning, see Alg. 2.6 is responsible for updating the policy of each agent based on observed experiences to optimize decision-making over time. The algorithm uses as input the observed states $S_s(t)$, the selected actions $A_s(t)$, the received rewards $R_s(t)$, and the resulting next states $S_s(t+1)$ for each site s. For each site, the agent stores the experience tuple $(S_s(t), A_s(t), R_s(t), S_s(t+1))$ into a replay buffer. This experience tuple captures the agent's interaction with the environment and is used to train the policy in a more stable manner by breaking the temporal correlations between consecutive experiences.

Several RL models have been explored to optimize the learning process, including a simple DQN, DDDQN, and A2C models. These models were not used simultaneously, but individually evaluated to determine the most effective. The selection was based on cumulative reward performance and the ability to generalize across different traffic profiles and site-specific constraints. Additionally, a hybrid approach was used during the initial learning phase, incorporating a Threshold algorithm (similar to PESBiU 1.0) to guide the

agent's behavior before switching to the learned policy. This hybrid approach ensured a more robust starting policy and improved the learning convergence by reducing the exploration space early on.

Once sufficient experiences are collected, the algorithm samples a mini-batch of experiences from the replay buffer. For each sampled experience in the mini-batch, the Q-values are updated according to the selected RL model. For the DQN and DDDQN models, target Q-values are computed using a separate target network to provide more stable learning targets. Specifically, the target Q-value for the selected action $A_s(t)$ is given by:

$$Q_{\text{target}} = R_s(t) + \gamma \cdot \max_{A'} Q_{\text{target}}(S_s(t+1), A'; \theta^-), \tag{4.35}$$

where γ is the discount factor, Q_{target} is the target Q-value function, and θ^- represents the weights of the target network. The online network weights θ are then updated to minimize the loss between the current Q-values and the target Q-values, defined as:

$$\text{Loss} = \left(Q(S_s(t), A_s(t); \theta) - Q_{\text{target}}\right)^2. \tag{4.36}$$

For the Advantage Actor-Critic (A2C) model, the policy π and the value function V are updated simultaneously. The policy is adjusted using the advantage function to improve action selection, while the value function minimizes the temporal difference (TD) error:

$$\text{Advantage} = R_s(t) + \gamma V(S_s(t+1)) - V(S_s(t)). \tag{4.37}$$

This advantage term helps the policy network focus on actions that yield higher-than-expected returns.

The agent's policy π is then updated based on the improved Q-values (for DQN models) or the advantage function (for A2C). This step ensures that the agent progressively learns to select actions that maximize the expected cumulative reward over time. By utilizing experience replay, a hybrid initial learning phase, and a target network, the algorithm stabilizes the learning process and mitigates issues such as divergence or overestimation, which are common in RL models.

Algorithm 2.6. Agent Learning for PESBiU 2.0

Input: Observed states $S_s(t)$, Actions $A_s(t)$, Rewards $R_s(t)$, Next states $S_s(t+1)$, Selected learning model
Output: Updated policy for each agent

for each site s **do**
  Store experience tuple $(S_s(t), A_s(t), R_s(t), S_s(t+1))$ in replay buffer
  Sample mini-batch from replay buffer
  if DQN or DDDQN is selected **then**
    Compute target Q-values with the target network:

$$Q_{\text{target}} = R_s(t) + \gamma \cdot \max_{A'} Q_{\text{target}}(S_s(t+1), A'; \theta^-)$$

    Minimize the Q-value loss:

$$\text{Loss} = (Q(S_s(t), A_s(t); \theta) - Q_{\text{target}})^2$$

    Update online network weights θ using gradient descent.
    Update policy π based on updated Q-values.
  else if A2C (Advantage Actor-Critic) is selected **then**
    Compute advantage function:

$$\text{Advantage} = R_s(t) + \gamma V(S_s(t+1)) - V(S_s(t))$$

    Update policy π using advantage-based policy gradients.
    Update value function V by minimizing TD error:

$$\text{TD Error} = (R_s(t) + \gamma V(S_s(t+1)) - V(S_s(t)))^2$$

  end if
  Hybrid Approach (if applicable): Threshold algorithm (from PESBiU 1.0) is used for initial training
    Use predefined rules to generate actions for initial policy training.
    Store experience tuples for future learning stages.
end for=0

Final Remarks on the PESBIU 2.0 Framework

The PESBiU 2.0 framework significantly extends its predecessor by employing a multi-agent RL framework where each agent controls the BBUs at a cell site. The agents use predicted data and compare it with actual data to make safe decisions. A safe switch mechanism is in place to prevent performance degradation when predictions are inaccurate. If the actual throughput deviates significantly from the predicted throughput, the agents activate all BBUs to ensure service quality. Agents learn to make intelligent decisions about putting BBUs to sleep, balancing energy savings with latency and overload penalties, and ensuring efficient network operation.

4.4 Comparison Between PESBiU 1.0 and PESBiU 2.0

Building upon the algorithm developments outlined in previous sections, this section provides a concise, comparative analysis between PESBiU 1.0 and PESBiU 2.0. Fig. 4.9 visually illustrates the architectural evolution from PESBiU 1.0's simpler threshold-based energy management to PESBiU 2.0's advanced machine learning-driven approach utilizing hybrid CNN-LSTM predictions and reinforcement learning algorithms like DDDQN. Further, Table 4.3 offers a detailed, feature-by-feature comparison, clearly highlighting the enhanced granularity of data, improved prediction accuracy, more dynamic decision-making, and substantial improvements in energy savings and QoS adaptability achieved by PESBiU 2.0. This structured comparison emphasizes PESBiU 2.0's significant advancements in energy savings, predictive capabilities, and robustness under dynamic 5G+ network conditions.

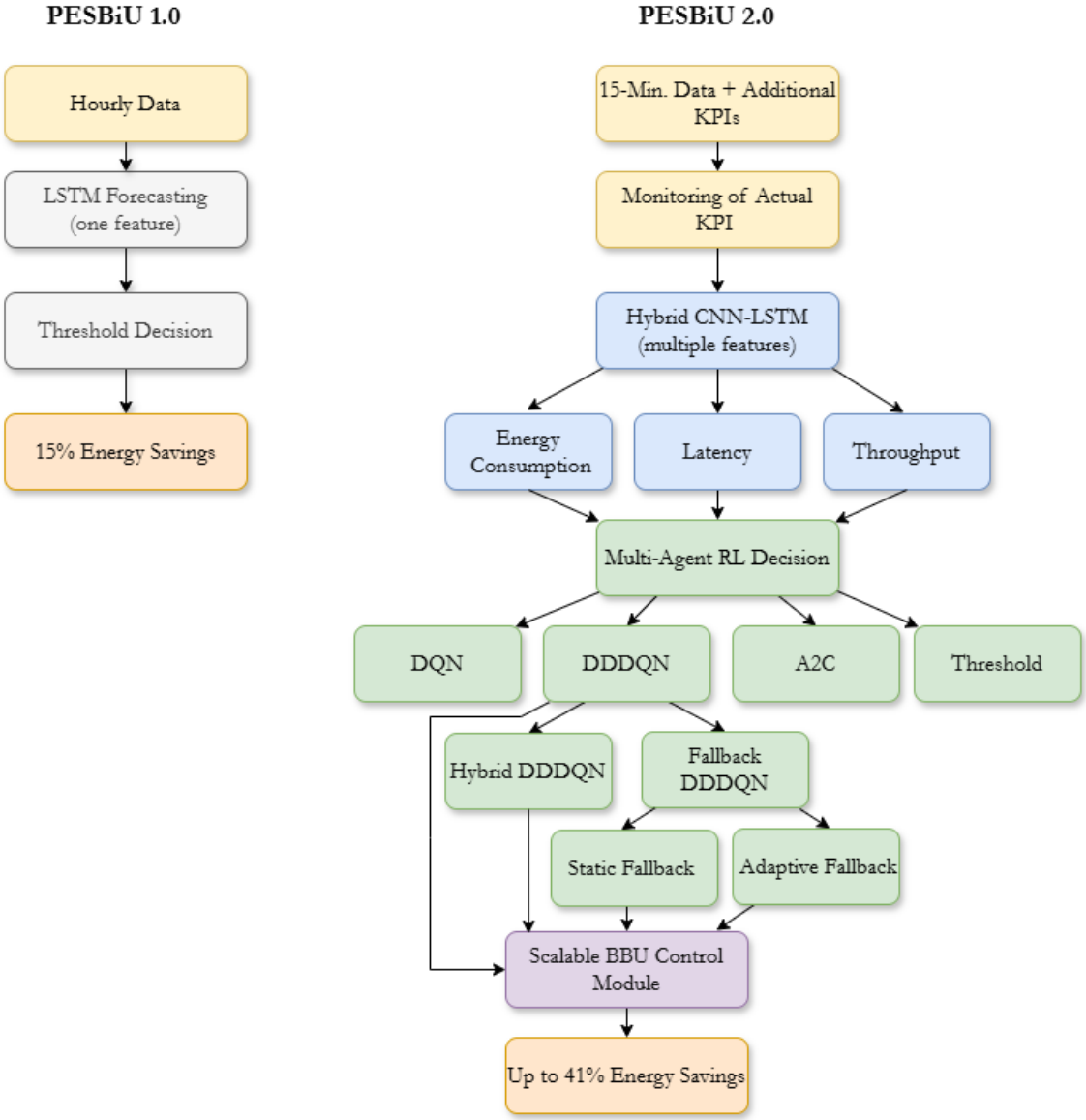


Fig. 4.9: Graphical Comparison of PESBiU 1.0 and PESBiU 2.0 architecture

Feature	PESBiU 1.0	PESBiU 2.0
Data Granularity	Hourly data from a primary dataset.	15-minute interval data from a secondary dataset, allowing finer predictions.
Prediction Model	LSTM-based forecasting.	Hybrid CNN-LSTM model for improved short-term forecasting.
Metrics Used	Traffic volume in MBytes.	Total port throughput (Mbps), UE DL latency (ms), and BBU AVG power consumption (W).
Decision Engine	Threshold rules for BBU sleep.	ML-driven mechanism using RL (DDDQN and variants) with fallback/adaptive and safe-switch mechanisms.
Energy Savings	Achieved 15.12% energy savings.	Achieved up to 41.21% energy savings (26.09% better than PESBiU 1.0).
Modularity	-	Modular design with separate modules for data acquisition, prediction, decision-making, and control.
QoS Handling	-	Integrated reward functions balancing energy and QoS; strong adaptability to spikes
Adaptability	-	Multi-agent RL system with transfer learning for different site types.
Energy Metric Expression	Consumed energy (Wh).	BBU average power consumption (W), suited to high-frequency data.
Overall Benefit	Entry-level solution that demonstrates feasibility of predictive BBU control	Robust multi-metric optimization with higher energy gains and sustained QoS

Tab. 4.3: Detailed comparison between PESBiU 1.0 and PESBiU 2.0.

4.5 Performance Assessment of PESBiU 2.0 Framework

In this section, we present the results for each tested algorithm, along with an explanation of the static rule-based algorithm used to validate the other tested algorithms for PESBiU 2.0. Figure 4.10 provides an overview diagram.

The PESBiU 2.0 framework was implemented and tested on a high-performance server equipped with three NVIDIA Tesla V100S-PCIE-32GB GPUs, each configured with 32 GB of VRAM. The system operated with the NVIDIA driver version 545.23.06 and utilized CUDA version 12.3. The server's total RAM was approximately 188 GB. Dedicated disk space was 262 GB. This setup provided sufficient resources for running the required computations and the training of the RL agents took approximately around 30-120 min.

It is important to highlight that the substantial hardware requirements mentioned

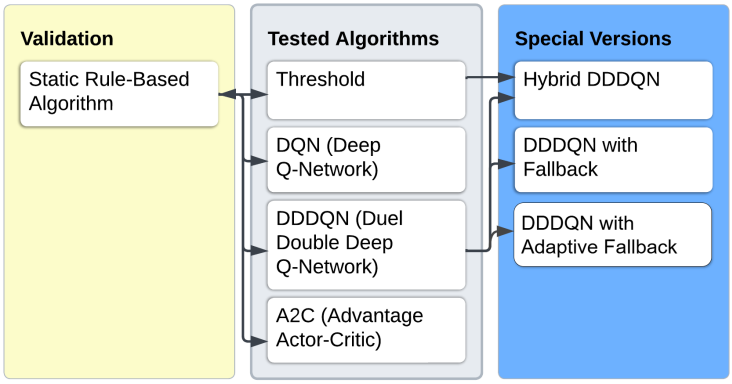


Fig. 4.10: Diagram of tested algorithms for PESBiU 2.0

above pertain only to the development and experimental evaluation phases. The operational deployment of PESBiU 2.0 in a real-world environment does not necessitate such high-performance hardware.

4.5.1 Static Rule-Based Algorithm

The static rule-based algorithm serves as a benchmark for evaluating the performance of the RL approach, providing a baseline against which the effectiveness of RL learned policies can be measured.

The static rule-based algorithm and the RL approach both aim to optimize energy consumption and network performance by deciding which BBUs to put to sleep at each time step. While they share common objectives and certain methodologies, there are fundamental differences in how they make decisions and adapt to changing network conditions. The static algorithm provides immediate heuristic decisions without learning. In contrast, the RL approach involves learning agents that adapt over time to optimize long-term rewards, demonstrating the potential for enhanced performance through RL. In the following paragraphs we will explain it more deeply.

Although the Static Rule-Based Algorithm performs well in controlled, simulated settings by maximizing energy savings with effective rules, it has significant limitations that make it unsuitable for real-world, dynamic 5G+ networks. It cannot adjust to changing traffic patterns, struggles to scale, and requires a lot of manual adjustments. These issues highlight the need for more flexible and intelligent methods, such as RL-based algorithms, which can automatically learn and improve energy management strategies in complex and changing network environments.

Similarities

Both the static rule-based algorithm and the RL approach utilize similar mathematical models, see Sec. 4.3.2, to estimate energy consumption and latency based on throughput. Specifically, they employ linear relationships:

$$E_{s,i}(t) = E_{\text{base},s,i} + k_{s,i} \cdot T_{s,i}(t), \tag{4.38}$$

$$L_{s,i}(t) = L_{\text{base},s,i} + m_{s,i} \cdot T_{s,i}(t), \tag{4.39}$$

where $E_{s,i}(t)$ and $L_{s,i}(t)$ represent the energy consumption and latency of BBU i at site s and time t, respectively; $E_{\text{base},s,i}$ and $L_{\text{base},s,i}$ are the base energy consumption and latency; $k_{s,i}$ and $m_{s,i}$ are coefficients (see Sec. 4.3.2); and $T_{s,i}(t)$ is the throughput.

Both approaches also consider energy savings, latency penalties, and overload penalties in their reward or objective functions to balance the trade-offs between performance and efficiency. When BBUs are put to sleep, both algorithms redistribute their traffic to the remaining active BBUs, ensuring that user demand continues to be met.

Differences in Decision-Making Process

Despite these similarities, the two approaches differ significantly in their decision-making processes. The static rule-based algorithm makes immediate decisions based on heuristic evaluations at each time step without learning from past experiences. In contrast, the RL approach involves agents that learn optimal policies over time through interactions with the environment, aiming to maximize long-term cumulative rewards.

In the static rule-based algorithm, the decision to put a BBU to sleep is based on a heuristic reward calculated for each BBU independently. For each BBU i at site s, the algorithm computes an energy efficiency metric and a latency penalty:

$$\text{EnergyEfficiency}_{s,i}(t) = \frac{E_{s,i}(t)}{T_{s,i}(t) + \epsilon}, \tag{4.40}$$

$$\text{LatencyPenalty}_{s,i}(t) = \left(L_{\text{base},s,i} + m_{s,i} \cdot \frac{T_{s,i}(t)}{2}\right) - L_{s,i}(t), \tag{4.41}$$

where ϵ is a small positive constant to prevent division by zero.

The heuristic reward for sleeping BBU i is then calculated as:

$$R_{\text{sleep},s,i}(t) = \text{EnergyEfficiency}_{s,i}(t) - \text{LatencyPenalty}_{s,i}(t). \tag{4.42}$$

BBUs are ranked based on $R_{\text{sleep},s,i}(t)$ in descending order. The algorithm attempts to put BBUs to sleep starting from the one with the highest reward, ensuring that capacity constraints are met and that at least one BBU remains active per site.

In the RL approach, agents observe the current state $S_s(t)$, which includes features such as time, predicted throughput, latency, energy consumption, and BBU statuses. Based on a learned policy $\pi(S_s(t))$, each agent selects an action $A_s(t)$ from the action space $\mathcal{A}$ to decide which BBUs to keep active or put to sleep.

The agents aim to maximize the expected cumulative reward:

$$R = \sum_t \gamma^t R_s(t), \tag{4.43}$$

where $R_s(t)$ is the reward at time t for site s, and γ is the discount factor.

The reward function, see Eq. 4.34, in the RL approach is similar to that of the static algorithm but is used differently. It incorporates normalized energy savings, latency penalties, overload penalties, and overload prevention bonuses. Agents use these rewards to update their policies over time, learning from the consequences of their actions to make better decisions in the future.

Adaptability and Learning

A key difference lies in adaptability and learning capability. The static rule-based algorithm does not adapt to changes in network conditions; it applies the same heuristic rules consistently without learning from past performance. In contrast, the RL approach allows agents to learn and adapt their policies based on feedback from the environment, improving decision making over time.

Results as a Benchmark

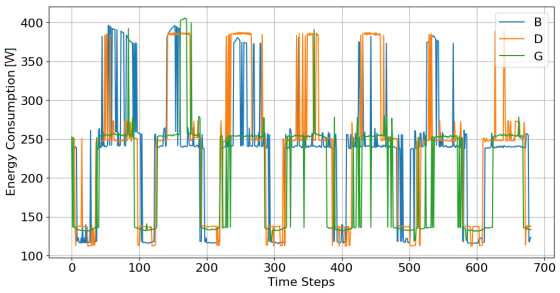
The results of the Static Rule-Based Algorithm for the period of one week, evaluated using three performance metrics - energy consumption, average latency, and total throughput, are shown in Fig. 4.11. The figure shows the algorithm's behavior in maintaining network performance under heuristic conditions. From it we can see the exchange processes and how energy consumption rises when total port throughput increases. Fig. 4.11 provides a benchmark, allowing us to compare with RL models to determine if they can manage higher peaks or balance metrics in the same way or even more effectively across the network. In the legend, B stands for Site Beta, D stands for Site Delta, and G stands for Site Gama.

In Fig. 4.12 we can see the BBU Sleep Counts charts for Sites Beta, Delta, and Gama, specifically the frequency with (how often) which each BBU is put to sleep under the static rule-based algorithm. It exhibits site-specific sleep patterns, indicating a reliance on certain BBUs for energy savings because of the highest reward that is calculated heuristically.

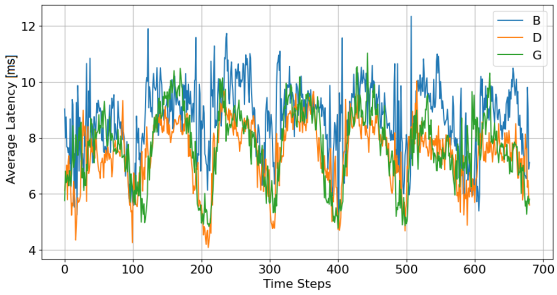
The following set of figures, Fig. 4.13, 4.14, 4.15, show results for Static Rule Based algorithm applied on the data from Tuesday. Tuesday was chosen as an graphical example similarly to PESBiU 1.0 and will be chosen and shown even for the other tested algorithms in following chapter for comparison. As it represents a normal workday. The results for the rest of the days are also compared in Chap. 5 and Appendix A.

Fig. 4.13 shows effectiveness in managing site Beta's BBU operations. The BBU sleep scheduling leads to significant energy savings, as shown by the substantial drop in energy usage post-action. This efficiency gain is achieved with minimal impact on network latency, which gains a slight increase, and throughput, which remains largely stable despite some fluctuations.

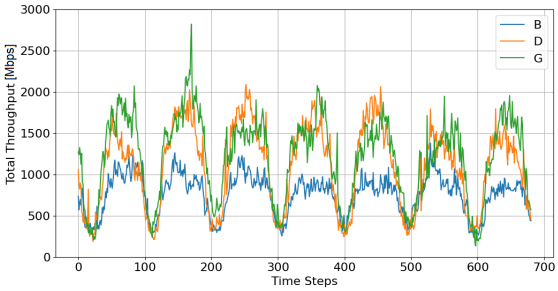
Fig. 4.14 illustrates that BBU sleep scheduling again reduces energy consumption from approximately 375 W to 125 W during sleep intervals, as seen in the green-shaded areas. However, the overall reduction is not as significant as in site Beta. While latency increases



(a) Energy consumption [W] over time

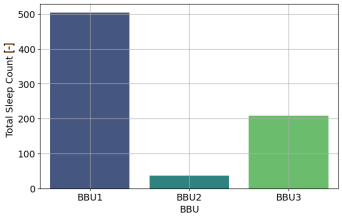


(b) Average latency [ms] over time

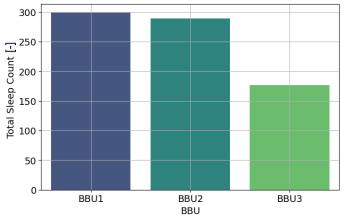


(c) Total throughput [Mbps] over time

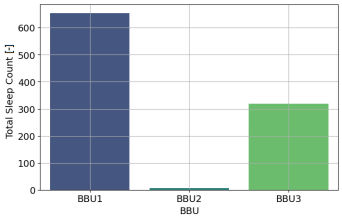
Fig. 4.11: Result of the metrics after applying Static Rule Based algorithm over a one week period measurement (one time step is one 15-min interval)



(a) BBU sleep count Site Beta



(b) BBU sleep count Site Delta

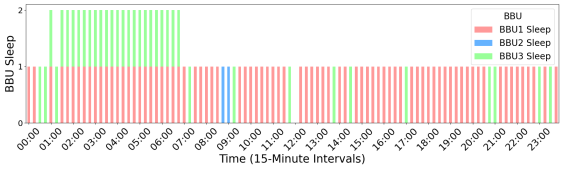


(c) BBU sleep count Site Gama

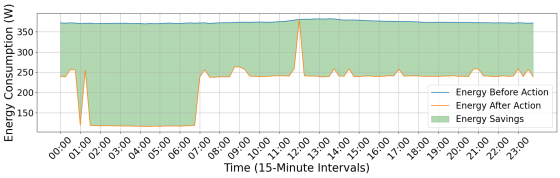
Fig. 4.12: Total sleep count of BBUs for each cell site over one week measurement with **Static algorithm**

slightly from around 6-7 ms to 7-8 ms, the throughput remains stable, ranging between 500 Mbps and 1750 Mbps with minor post-action variability.

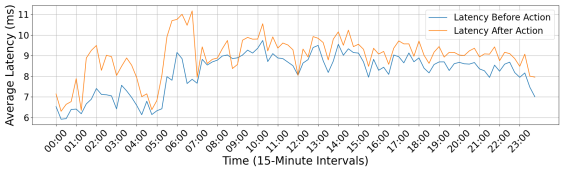
In Fig. 4.15, site Gama, the BBU sleep patterns show that BBU1 is frequently put to sleep for prolonged intervals, while BBU2 remains mostly active, with BBU3 showing sporadic activity, indicating that it is not set to sleep as frequently. This could be due to moderate traffic load or suboptimal energy efficiency gains compared to BBU1, making sleep transitions less favorable for BBU2 or even BBU3. This is caused by the calculated highest reward for energy efficiency for BBU3. Energy consumption decreases significantly after the sleep actions are applied. The latency chart shows a slight increase post-action, but remains within acceptable limits. Throughput remains steady, and no traffic is lost.



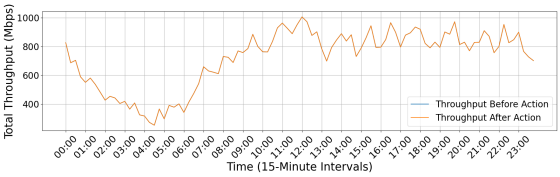
(a) BBUs put to sleep intervals



(b) Energy Consumption before and after

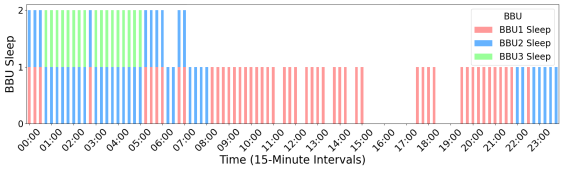


(c) Latency before and after PESBiU 2.0

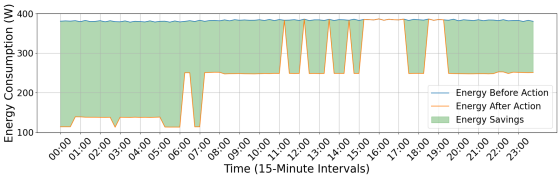


(d) Throughput before and after PESBiU 2.0

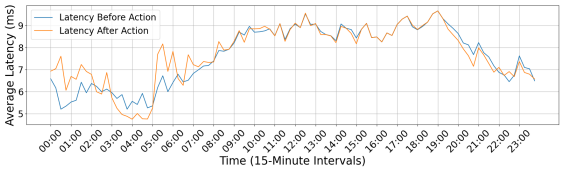
Fig. 4.13: Site Beta - workday - results before and after **Static algorithm's** decisions



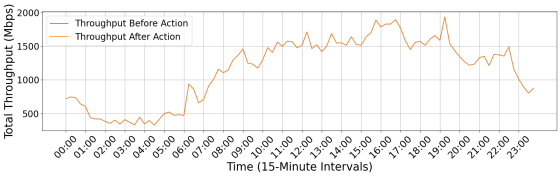
(a) BBUs put to sleep intervals



(b) Energy Consumption before and after

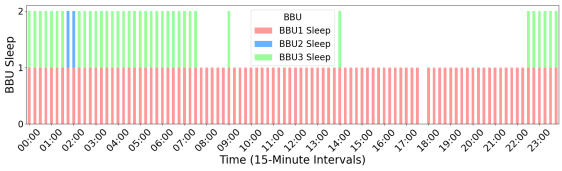


(c) Latency before and after PESBiU 2.0

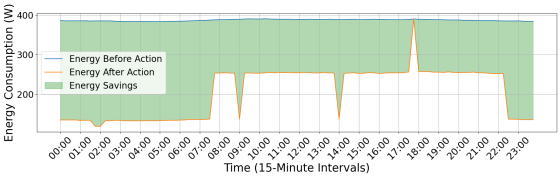


(d) Throughput before and after PESBiU 2.0

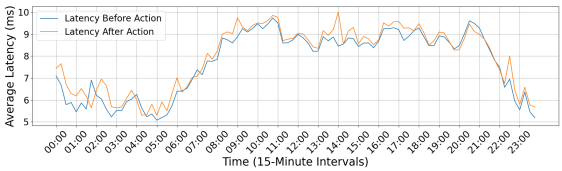
Fig. 4.14: Site Delta - workday - results before and after **Static algorithm's** decisions



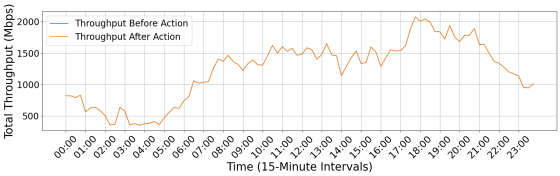
(a) BBUs put to sleep intervals



(b) Energy Consumption before and after



(c) Latency before and after PESBiU 2.0



(d) Throughput before and after PESBiU 2.0

Fig. 4.15: Site Gama - workday - results before and after **Static algorithm's** decisions

4.5.2 Threshold Algorithm

The Threshold algorithm provides an alternative heuristic approach for managing BBUs, and is built up on the same logic as PESBiU 1.0. The Threshold-based algorithm leverages predefined utilization and load thresholds to guide its decisions. It is included in PESBiU 2.0 development to manage the BBU sleep modes in a more conservative way. This option will also be used in a special version of PESBiU 2.0 algortihm, Sec. 4.5.6 Hybrid DDDQN.

Again, as with the Static Rule-based algorithm, the Threshold algorithm utilizes similar mathematical models to estimate energy consumption and latency based on throughput, as outlined in Sec. 4.3.2. Specifically, the linear relationships as follow in Eq. 4.38 and Eq. 4.39.

The Threshold-based algorithm makes decisions based on predefined utilization and load thresholds. For each BBU i at site s, the algorithm calculates its utilization percentage:

$$\text{Utilization}_{s,i}(t) = \frac{T_{s,i}(t)}{C_{s,i}} \cdot 100\%, \tag{4.44}$$

where $C_{s,i}$ is the maximum capacity of BBU i at site s after applying utilization factors.

The algorithm then assesses the overall load of the site:

$$\text{SiteLoad}_s(t) = \frac{\sum_{i=1}^{3} T_{s,i}(t)}{\sum_{i=1}^{3} C_{s,i}} \cdot 100\%. \tag{4.45}$$

Based on these calculations, the algorithm determines the number of BBUs that can be put to sleep:

- If $\text{SiteLoad}_s(t) < 30\%$, up to two BBUs can be put to sleep.
- If $30\% \leq \text{SiteLoad}_s(t) \leq 70\%$, one BBU can be put to sleep.
- If $\text{SiteLoad}_s(t) > 70\%$, no BBUs are put to sleep to ensure adequate capacity.

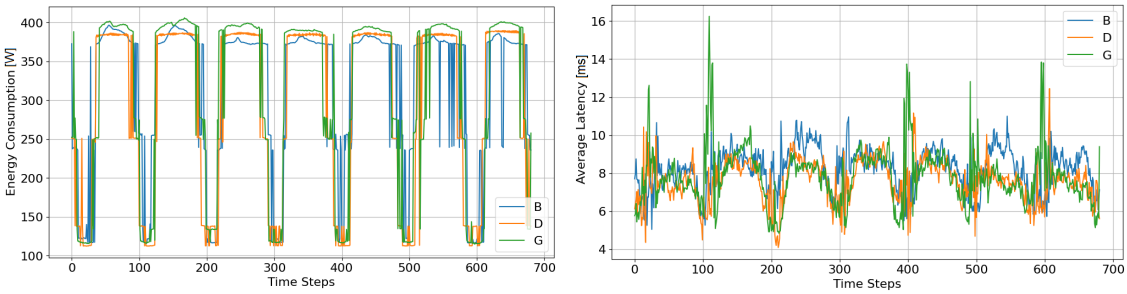
The algorithm prioritizes putting BBUs below a certain load threshold (e.g., 30%) to sleep, ensuring that the remaining active BBUs can handle the redistributed throughput without exceeding their capacities.

The benefits lie in simplicity of the applied logic, and in the fact that it is a more conservative solution which is less aggressive. However, the disadvantage is that it depends solely on threshold values, which may require precise tuning to match the specific operational environment.

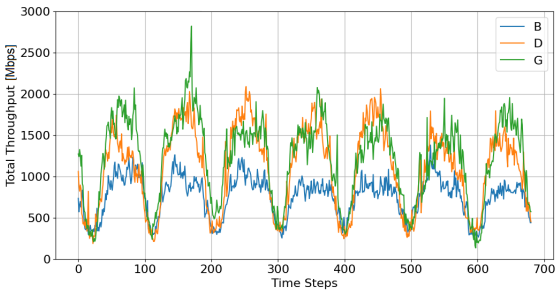
Results for Threshold Algorithm

The results of the Threshold algorithm for the period of one week, evaluated using three performance metrics - energy consumption, average latency, and total throughput, are shown in Fig. 4.16. From figures we can see that the energy saving is not as high as in the case of ideal case of Static Rule-Based algorithm in Fig. 4.11, but on the other hand, the latency is lower than in case of static version.

Fig. 4.17 shows BBU Sleep Counts charts for Sites Beta, Delta, and Gama over one week and the amount of time when BBUs went to sleep. The sleep pattern is more spread out and balanced.

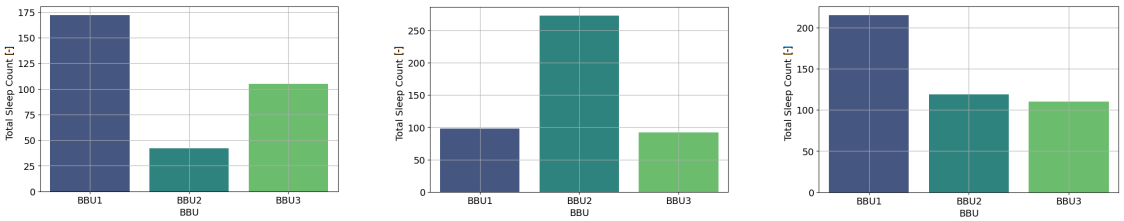


(a) Energy consumption [W] over time (b) Average latency [ms] over time



(c) Total throughput [Mbps] over time

Fig. 4.16: Result of the metrics after applying **Threshold Algorithm** over a one week period measurement



(a) BBU sleep count Site Beta (b) BBU sleep count Site Delta (c) BBU sleep count Site Gama

Fig. 4.17: Total sleep count of BBUs for each cell site over one week measurement with **Threshold Algorithm**

4.5.3 Deep Q-Network (DQN)

Following the Threshold Algorithm, which provides a heuristic approach for energy optimization, the Deep Q-Network (DQN) introduces a dynamic and learning-driven mechanism. Unlike the deterministic nature of threshold-based methods, DQN leverages RL to adapt to varying traffic conditions by learning optimal policies over time. The algorithm employs a neural network to approximate Q-values, representing the expected rewards of actions given specific states, thus enabling more nuanced decisions for BBU energy management [47].

The technical implementation of DQN for the PESBiU 2.0 algorithm was configured with a robust set of hyperparameters to balance exploration, learning stability, and computational efficiency. The Q-network architecture consisted of three fully connected hidden layers, each with 64 neurons and ReLU activations, paired with a dropout layer of 0.1

to prevent overfitting. The training process utilized an Adam optimizer with a learning rate of $1 \cdot 10^{-5}$ and a weight decay of $1 \cdot 10^{-4}$ for regularization. A discount factor (γ) of 0.99 was applied to prioritize long-term rewards. To ensure stability in learning, a target network was updated every 100 training steps, and experience replay was used to decorrelate training samples, leveraging a buffer size of 1000 transitions. The agent adopted an ϵ-greedy policy for action selection, with ϵ decaying from 1.0 to 0.01 over episodes to transition from exploration to exploitation.

Results for DQN

The Fig. 4.18 shows the learning progression of the DQN agents over 500 training episodes. Each training episode represents a single simulation run in which the agent interacts with the environment to learn and refine its decision-making policy based on the observed rewards. The total rewards, represented on the y-axis, increase steadily across episodes, demonstrating the agents' improved policies for energy management. Site Gama (G) slightly outperforms Sites Beta (B) and Delta (D) in reward accumulation, indicating better optimization. Early reward variability reflects exploration, while later stabilization highlights convergence to effective energy-saving strategies.

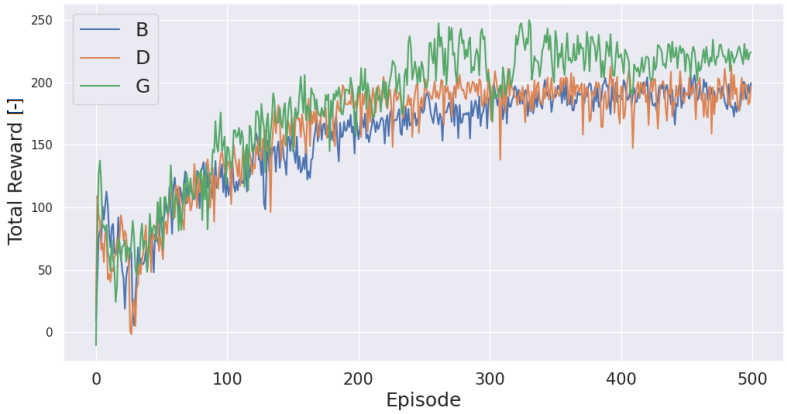
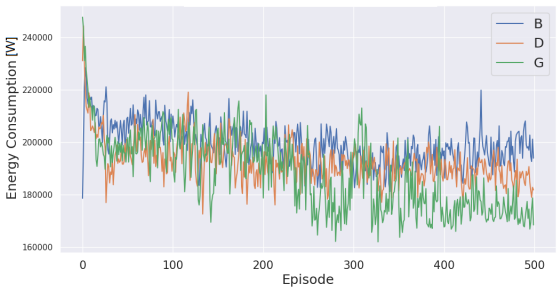
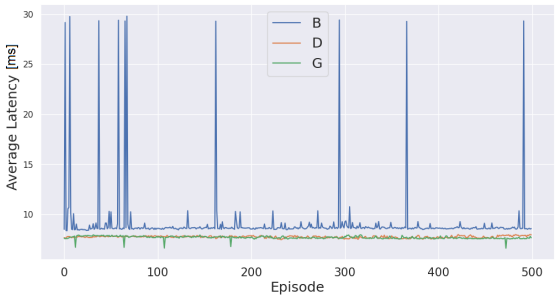


Fig. 4.18: DQN Total reward [-] over episodes

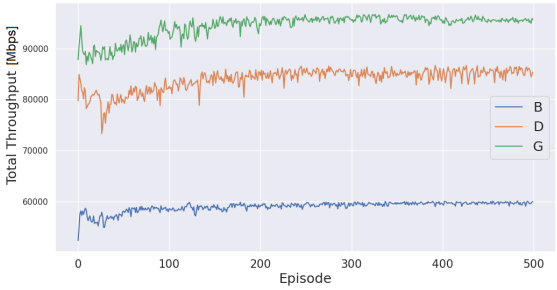
Fig. 4.19a shows the energy consumption per episode. The trends show an initial reduction during early training episodes, followed by stabilization as the agents converge to efficient energy management policies. Sites Delta and Gama display slightly lower energy usage compared to Site Beta, indicating more effective optimization for these sites. Fig. 4.19b shows the average latency per episode. While Sites Delta and Gama maintain low and stable latency, Site Beta exhibits occasional spikes, indicating higher variability in latency for this site. Despite these fluctuations, the overall latency remains within acceptable thresholds. Fig. 4.19c presents the total throughput per episode. Site Gama consistently achieves the highest throughput, followed by sites Delta and Beta respectively. These trends demonstrate the agents' ability to maintain high throughput levels while optimizing energy consumption.



(a) Energy consumption [W] over time



(b) Average latency [ms] over time



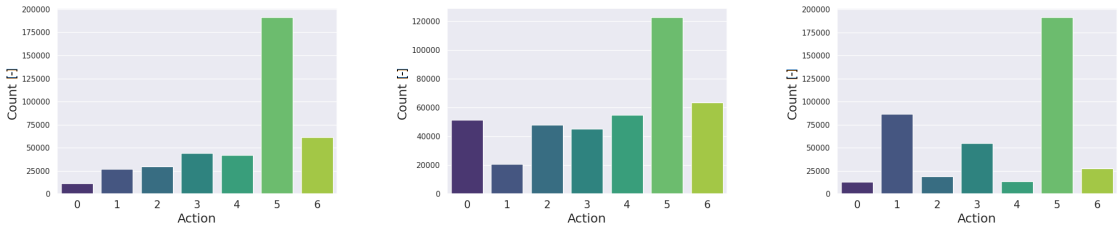
(c) Total throughput [Mbps] over time

Fig. 4.19: Result of the metrics after applying **DQN** over one week measurement

Fig. 4.20a depicts the distribution of actions (see Eq. 4.16) taken by the DQN agent for Site B during training. Action 5, corresponding to BBUs 2 and 3 being active while BBU 1 is sleeping, is the most frequently selected. This correspond to the results for Static algorithm. Fig. 4.12a indicates the agents' preferred configurations with energy savings while maintaining network performance. Fig. 4.20b shows the action distribution for Site Delta. Similar to Site Beta, the most frequent action is 5, indicating a preference for BBUs 2 and 3 being active while BBU 1 is in sleep mode. However, actions with two active BBUs (e.g., Actions 3 and 4) also appear with significant frequency, suggesting that the agents adapted their decisions to varying traffic conditions at this site. Again, it aligns with Fig. 4.12b. Fig 4.20c illustrates the action distribution for Site Gama. The agent overwhelmingly favored action 5, where BBUs 2 and 3 remain active while BBU 1 sleeps. This reflects the agent's optimization of energy consumption at Site Gama, which is consistent with the results for Static algorithm. The higher frequency of action 1, where only BBU 2 is active, further highlights site-specific traffic patterns influencing agent's decisions.

4.5.4 Dueling Double Deep Q-Network (DDDQN)

Building on the Deep Q-Network (DQN), the Dueling Double Deep Q-Network (DDDQN) incorporates two major advancements: a dueling network architecture and double Q-learning. The dueling architecture decomposes the Q-value estimation into separate streams for advantage and value functions, allowing the agent to differentiate the value of states from the relative importance of actions within those states. Meanwhile, double



(a) Action distribution Beta (b) Action distribution Delta (c) Action distribution Gama

Fig. 4.20: Total sleep count of BBUs for each cell site over one week measurement with **DQN**

Q-learning mitigates the overestimation bias of standard DQN by decoupling action selection from action evaluation. These enhancements enable the DDDQN to perform more robustly in complex scenarios by improving both learning stability and decision-making accuracy [100].

The DDDQN implementation features a neural network architecture with separate streams for value and advantage computation. The common feature extraction layer consists of one fully connected layer with 64 neurons and a ReLU activation function. Both the value and advantage streams use two hidden layers, each with 64 neurons and ReLU activations, followed by their respective output layers. A dropout layer with a probability of 0.1 is applied to enhance generalization.

The training process employs double Q-learning, where the target Q-value is computed by using the main Q-network for action selection and the target Q-network for evaluation. The Adam optimizer is used with a learning rate of $1 \cdot 10^{-5}$ and weight decay of $1 \cdot 10^{-4}$ for regularization. The replay buffer size is set to 1000, and mini-batches of size 64 are sampled for training. Similar to the standard DQN, the target network is updated every 100 steps, and the ϵ-greedy policy is used for exploration, with ϵ decaying from 1.0 to 0.01 over episodes.

Results for DDDQN

Fig. 4.21 describes the learning progress of the DDDQN agents across Sites Beta, Delta, and Gama over 500 training episodes. Compared to the standard DQN, the agents demonstrate faster convergence and achieve consistently higher rewards across all sites. Site Gama outperforms the others, achieving the highest steady-state reward, indicating the agent's effectiveness in optimizing energy consumption and maintaining quality of service. Sites Delta and Beta also show stable performance, with Site Delta achieving slightly higher rewards than Site Beta. The smooth convergence and reduced variability highlight the benefits of the DDDQN architecture.

Fig. 4.22a shows the energy consumption over 500 training episodes for the DDDQN agents. The energy consumption stabilizes more efficiently compared to the standard DQN, with Site Gama achieving the lowest and most stable energy usage. Sites Delta and Beta also demonstrate reduced energy consumption after initial episodes. Notably, the DDDQN outperforms the standard DQN by converging faster and achieving lower

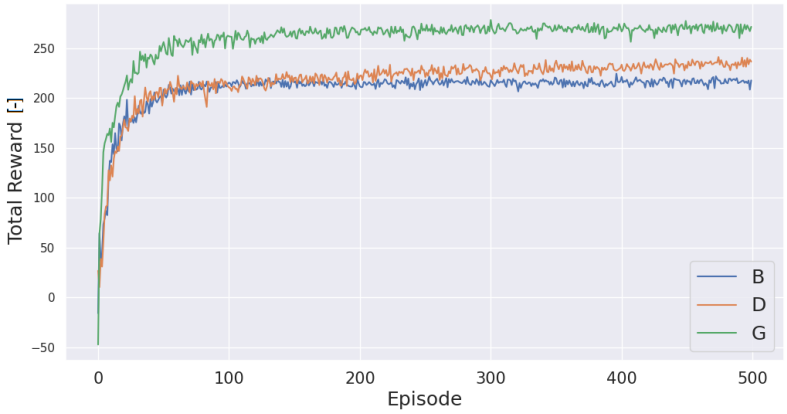
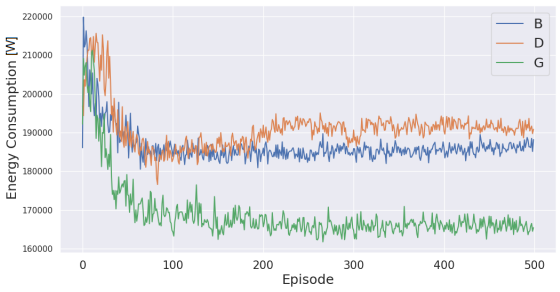


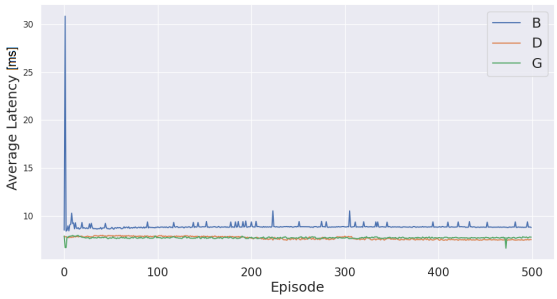
Fig. 4.21: DDDQN Total reward [-] over episodes

overall energy consumption across all sites. Fig. 4.22b illustrates the average latency per episode for the DDDQN agents. Similar to the standard DQN, Sites Delta and Gama maintain consistently low latencies, while Site Beta exhibits occasional spikes. However, these spikes are less frequent and less pronounced compared to the DQN, indicating improved stability in latency management with the DDDQN. Fig. 4.22c shows the total throughput. The throughput trends are consistent with the standard DQN, with Site Gama achieving the highest throughput, followed by Sites Delta and Beta respectively. However, the throughput for DDDQN stabilizes more quickly and shows less variability, particularly at Site Beta. This improved consistency highlights the enhanced optimization capabilities of the DDDQN over the standard DQN.

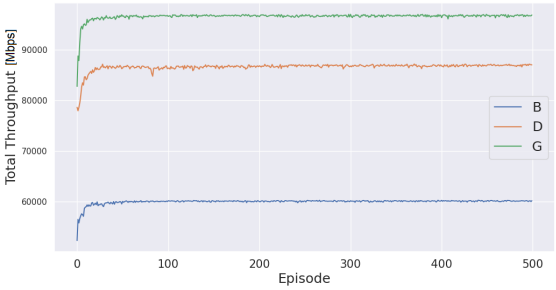
Fig. 4.23a presents the action distribution (see Eq. 4.16) for the Site Beta under the DDDQN agent. Action 5, where BBUs 2 and 3 are active and BBU 1 is in sleep mode, remains the dominant choice, similar to the standard DQN results. However, action 1, where only BBU 2 is active, occurs more frequently than in the DQN, indicating enhanced optimization for traffic patterns that allow for a greater reduction in active BBUs. The reduced frequency of less efficient actions like 2 and 6 highlights the agent's improved decision-making. Fig. 4.23b shows the action distribution for Site Delta. As for Site Beta, action 5 is the most frequent, reflecting a consistent strategy to prioritize energy savings. The relative frequency of actions 4 and 6 is higher compared to the DQN, suggesting that the DDDQN better adapts to site-specific traffic conditions, allowing for more effective combinations of active BBUs while maintaining performance. Fig. 4.23c displays the action distribution for Site Gama. Action 5 remains overwhelmingly dominant, similar to the results for Site Gama under the DQN. However, there is a noticeable increase in the frequency of action 1, indicating that the agent leverages the site-specific traffic dynamics to maximize energy savings by frequently deactivating BBUs 1 and 3. The improved distribution of actions compared to the standard DQN suggests that the DDDQN achieves greater adaptability and efficiency across different traffic scenarios.



(a) Energy consumption [W] over time

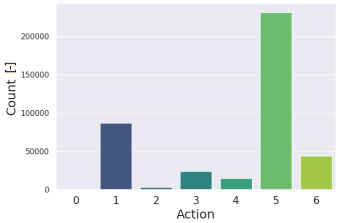


(b) Average latency [ms] over time

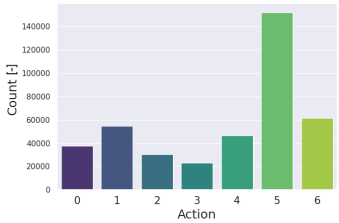


(c) Total throughput [Mbps] over time

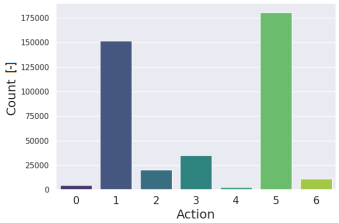
Fig. 4.22: Result of the metrics after applying **DDDQN** over one week measurement



(a) Action distribution Beta



(b) Action distribution Delta



(c) Action distribution Gama

Fig. 4.23: Total sleep count of BBUs for each cell site over one week measurement with **DDDQN**

4.5.5 Advantage Actor Critic (A2C)

The A2C algorithm represents an evolution in RL by incorporating separate actor and critic networks. The actor network optimizes the policy by learning to select actions that maximize long-term rewards, while the critic network evaluates the value of the states encountered. This dual-network approach allows A2C to efficiently balance exploration and exploitation. By leveraging the advantage function, which measures the relative quality of an action compared to others in the same state, A2C should ensure more stable and efficient policy updates compared to policy-gradient-only methods [102].

The A2C implementation consists of an actor network and a critic network, each with distinct architectures. The actor network includes three fully connected layers, each with 64 neurons, ReLU activation functions, and a final softmax layer to output a probability distribution over actions. The critic network employs a simpler structure with two hidden

layers of 32 neurons each and ReLU activations, followed by an output layer to estimate state values.

The training process involves optimizing the actor network by using the advantage function, computed as the difference between the estimated return and the critic's value prediction. The critic network minimizes the mean squared error between its predicted value and the actual return. An entropy regularization term is added to the actor loss to encourage exploration, weighted by an entropy coefficient of 0.01. Both networks are trained using the Adam optimizer with learning rates of $1 \cdot 10^{-4}$ for the actor and $1 \cdot 10^{-3}$ for the critic. A mini-batch size of 24 is used, and gradient clipping is applied to enhance stability during updates. To ensure reproducibility and diversity across agents, unique seeds are assigned for each site. This was the setup, where the best results were achieved. However, they did not surpass the results from DQN and DDDQN.

Results for A2C

Fig. 4.24 illustrates the total rewards for sites Beta, Delta, and Gama under the A2C algorithm. Site Gama demonstrates a strong performance, achieving consistently high rewards by episode 100 and maintaining stability thereafter. This mirrors the efficiency observed with DDDQN but shows slightly larger fluctuations, likely due to the exploration mechanism of entropy regularization in A2C.

Site Delta shows a delayed convergence compared to Site Gama, stabilizing only after approximately 200 episodes. Although the rewards remain positive, they are noticeably lower than the results achieved by DDDQN, indicating a slower adaptation to the site's specific traffic dynamics. For Site Beta, the rewards remain consistent but slightly lag behind both the DDDQN and DQN approaches, highlighting potential challenges in optimizing simultaneously energy efficiency and latency in this environment.

Compared to DQN, A2C demonstrates improved reward stability in Sites Delta and Gama, particularly after convergence. However, it falls short of the higher rewards achieved by DDDQN, suggesting that while A2C balances exploration and exploitation effectively, the absence of value-action decoupling may limit its performance in high-dimensional environments.

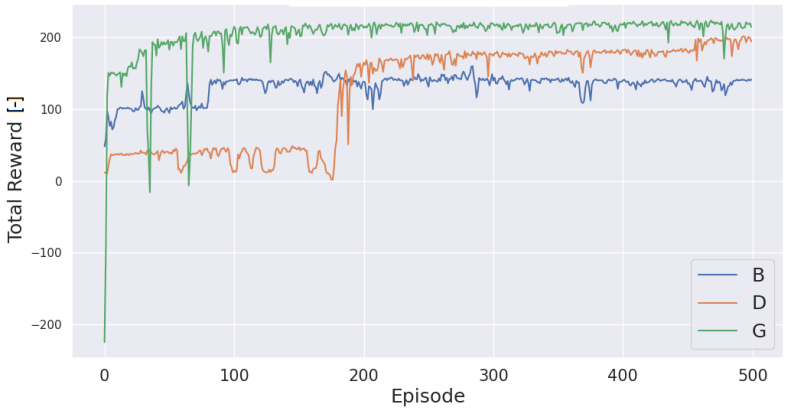
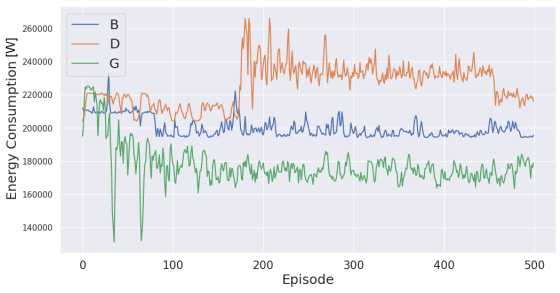


Fig. 4.24: A2C Total reward [-] over episodes

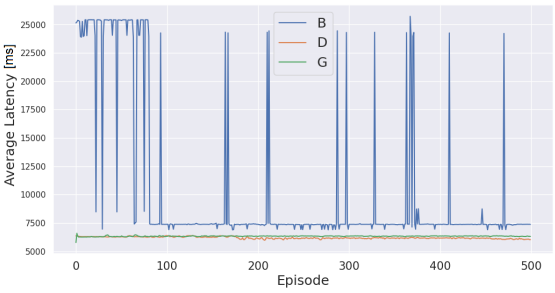
The energy consumption over episodes for the A2C algorithm is shown in Fig. 4.25a. Site Gama demonstrates the most efficient energy consumption, stabilizing at a relatively low level after initial fluctuations. Site Beta follows with slightly higher values, while Site Delta peaks significantly higher, indicating less efficient energy management compared to both DQN and DDDQN. Overall, A2C exhibits higher variability in energy consumption, and its performance is not superior to DQN or DDDQN.

Fig. 4.25b shows the average latency over episodes for A2C. Site Gama maintains stable and low latency, similar to the latency observed with DDDQN. In contrast, Site Beta experiences frequent large spikes, significantly higher than those seen with DQN or DDDQN, indicating A2C's challenges in managing latency consistently.

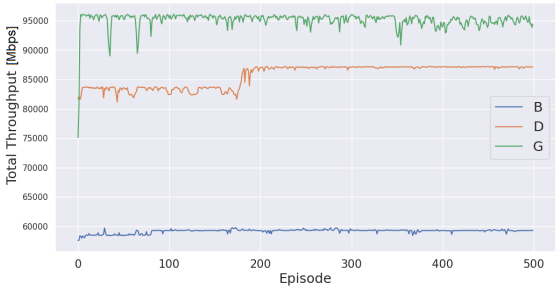
The total throughput for A2C, presented in Fig. 4.25c, shows moderate performance for all sites, since the result should be stable to avoid any traffic loss. The overall patterns are not as clear as in case of DQN or DDDQN, therefore A2C delivers comparable but not superior results.



(a) Energy consumption [W] over time



(b) Average latency [ms] over time



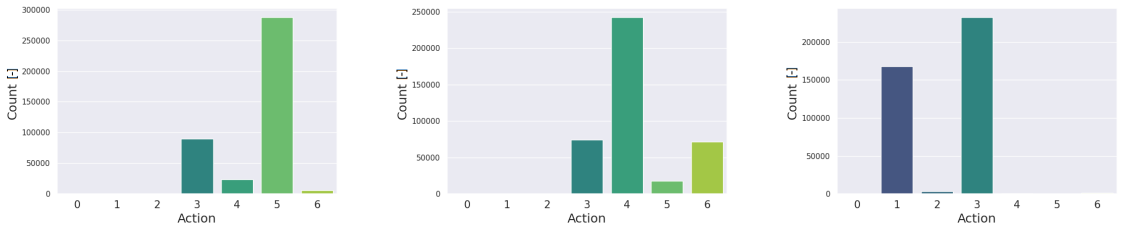
(c) Total throughput [Mbps] over time

Fig. 4.25: Result of the metrics after applying **A2C** over one week measurement

The action distributions for sites under the A2C algorithm are presented in Fig. 4.26. For Site Beta, action 5, which corresponds to keeping BBUs 2 and 3 active, dominates with the highest frequency, followed by action 3 (BBUs 1 and 2 active). Site Delta shows a similar pattern, with action 4 (BBUs 1 and 3 active) being the most frequent, followed by action 6 (all BBUs active), indicating that the model ensures sufficient capacity during higher demand. For Site Gama, action 3 (BBUs 1 and 2 active) is the most frequent, suggesting a more balanced approach between capacity and energy efficiency for this site.

Comparing these distributions with the DQN and DDDQN results reveals notable

differences. Both DQN and DDQN algorithms demonstrate a broader utilization of higher-capacity actions such as 6 (all BBUs active). In contrast, A2C favors energy-saving strategies while maintaining sufficient capacity through intermediate actions, such as 3, 4, and 5. However, A2C exhibits slightly less diversity in action selection, which could suggest over-prioritization of specific energy-saving configurations, which is not ideal in our case.



(a) Action distribution Beta (b) Action distribution Delta (c) Action distribution Gama

Fig. 4.26: Total sleep count of BBUs for each cell site over one week measurement with **A2C**

4.5.6 Hybrid Dueling Double Deep Q-Network (DDDQN) with Threshold

Building upon the capabilities of DDDQN and the simplicity of threshold-based heuristics, we created the Hybrid Dueling Double Deep Q-Network (DDDQN) with Threshold. This hybrid approach aims to synergize the strengths of both methodologies trying to achieve even better energy efficiency and network performance in BBU management. By integrating threshold-based decision-making within the DDDQN framework, the hybrid model ensures reliable baseline actions while leveraging the adaptive learning prowess of RL to optimize complex decision scenarios.

Explanation of the algorithm

The Hybrid DDDQN with Threshold employs a two-tier decision-making process. The initial tier utilizes predefined threshold rules to identify candidate BBUs for sleep, based on their individual utilization metrics and the overall site load. The subsequent tier applies the DDDQN to these candidates to fine-tune the decisions, ensuring optimal traffic redistribution and adherence to capacity constraints. The overall algorithm operates by selecting actions based on probability, derived from either the threshold-based heuristic or the DDDQN agent.

The threshold-based action is determined by using predefined utilization and load thresholds, similar to the standalone threshold algorithm described in Sec. 4.5.2. Specifically, for each site s at time t, the number of BBUs to be put to sleep, $N_{\text{sleep},s}(t)$, is determined based on the overall site load $\text{SiteLoad}_s(t)$:

- If $\text{SiteLoad}_s(t) < 30\%$, then $N_{\text{sleep},s}(t) = 2$.
- If $30\% \leq \text{SiteLoad}_s(t) \leq 70\%$, then $N_{\text{sleep},s}(t) = 1$.
- If $\text{SiteLoad}_s(t) > 70\%$, then $N_{\text{sleep},s}(t) = 0$.

Subsequently, BBUs with utilization below a certain threshold (e.g. 30%) are prioritized for deactivation to achieve the desired $N_{\text{sleep},s}(t)$.

Once the threshold-based selection identifies potential BBUs for sleep, the DDDQN takes over to make more informed and optimized decisions. The DDDQN uses its learned policies to evaluate the best possible actions for redistributing traffic and ensuring that network performance remains optimal. By considering the broader context of the network state, the DDDQN can adjust the initial suggestions from the threshold rules to better suit dynamic conditions and varying traffic patterns.

To decide which action to select from which algorithm, we used probability-based action selection. At each decision epoch t, for each BBU i at site s, the algorithm decides whether to activate or deactivate the BBU based on a confidence probability $\rho(t)$. This probability determines the likelihood of using the threshold-based action versus the DDDQN-derived action

$$a_{s,i}(t) = \begin{cases} a_{\text{threshold},s,i}(t) & \text{with probability } \rho(t), \\ a_{\text{DDDQN},s,i}(t) & \text{with probability } 1-\rho(t). \end{cases} \tag{4.46}$$

Here, $a_{\text{threshold},s,i}(t)$ represents the action suggested by the threshold algorithm, and $a_{\text{DDDQN},s,i}(t)$ denotes the action proposed by the DDDQN agent.

To ensure that the algorithm gradually relies more on the DDDQN agent as it gains experience, the confidence probability $\rho(t)$ decays over time:

$$\rho(t) = \max\left(\rho_0 \cdot \gamma^t, \rho_{\min}\right), \tag{4.47}$$

where ρ_0 is the initial confidence probability, $\gamma \in (0,1)$ is the decay rate, and $\rho_{\min}$ is the minimum confidence probability to guarantee a fallback to the threshold-based heuristic.

Results for Hybrid DDDQN+Threshold

Fig. 4.27 shows the total reward over 500 episodes for the Hybrid DDDQN+Threshold algorithm across the three sites. The Hybrid approach demonstrates a consistent learning curve with rewards steadily increasing in the early episodes before stabilizing. Site Gama achieves the highest final reward of approximately 250, while sites Delta and Beta plateau at around 220 and 200, respectively.

When compared to the DDDQN alone (Fig. 4.21), the Hybrid algorithm exhibits slightly lower overall rewards across all sites. This difference may stem from the influence of the threshold-based rules integrated into the Hybrid model. While the threshold heuristic ensures a more conservative and stable decision-making process during early episodes, it may limit the algorithm's ability to achieve the same level of exploration and reward optimization as the pure DDDQN.

It is important to note that the Threshold algorithm itself does not use reward-based performance metrics. Instead, it serves as a safe mechanism for the Hybrid approach, particularly during the initial training phases. This integration of threshold rules likely contributes to the Hybrid algorithm's steady, albeit slightly lower, reward trajectory,

as it prioritizes stability and operational safety over aggressive reward maximization. The Threshold algorithm will be compared in more detail in Chap. 5.

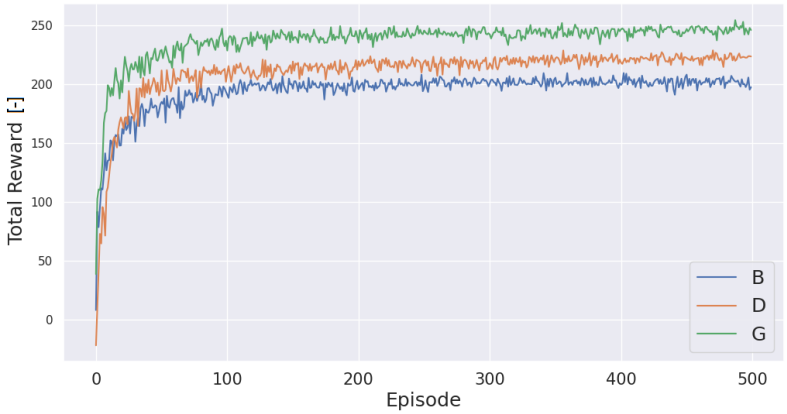


Fig. 4.27: Hybrid DDDQN+Treshold Total reward [-] over episodes

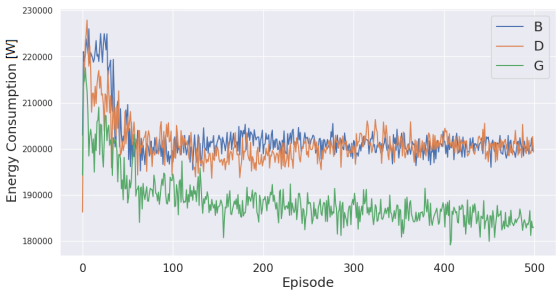
Fig. 4.28a presents the energy consumption patterns. At the beginning, the energy consumption is relatively high for all sites, reflecting the initial exploration phase of the algorithm. As the training progresses, the Hybrid model learns to optimize BBU activation patterns, leading to a notable reduction in energy consumption. Compared to DDDQN (Fig. 4.22a), the Hybrid algorithm achieves slightly higher energy consumption across all sites. This difference is likely due to the inclusion of the threshold heuristic in the Hybrid approach, which emphasizes operational safety and stability over aggressive energy minimization. The threshold algorithm ensures that BBUs are put to sleep only under safe conditions, which limits the energy savings potential but ensures consistent and reliable performance.

The performance of the Hybrid DDDQN+Threshold algorithm in terms of average latency over episodes is depicted in Fig. 4.28b. The results show relatively stable latency similarly to pure DDDQN algorithm (Fig. 4.22b), except for Site Beta, where the Hybrid algorithm occasionally experiences significant latency spikes.

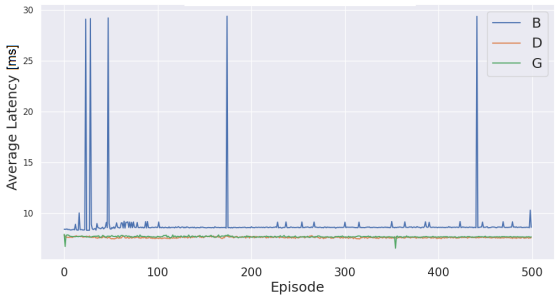
The total throughput over episodes, as shown in Fig. 4.28c, indicates a rapid convergence to a high throughput value for all scenarios, achieving stability after approximately 100 episodes. In comparison, for the DDDQN (Fig. 4.22c) the results are very similar, the only slight difference is at the beginning, where the Threshold algorithm secured higher throughput right at the start.

The action distribution for Site Beta, see Fig. 4.29a, highlights the preference for activating two BBUs (action 5: BBU2 and BBU3 are active) in both the Hybrid and DDDQN approaches. In the Hybrid model, action 5 is selected most frequently, followed by action 1 (only BBU2 is active). The DDDQN results also prioritize action 5, but with slightly less variability across other actions. Notably, action 0 (only BBU1 is active) and 2 (only BBU3 is active) are almost never selected in both models, showcasing consistent optimization patterns.

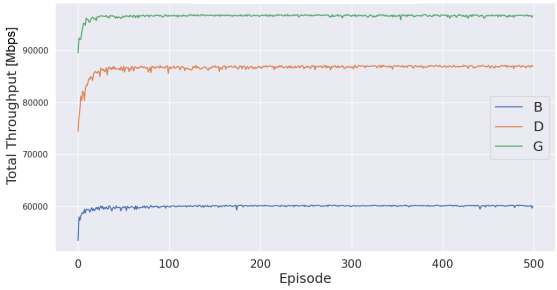
For Site Delta, Fig. 4.29b and Fig.4.23, both the Hybrid model and DDDQN algo-



(a) Energy consumption [W] over time



(b) Average latency [ms] over time

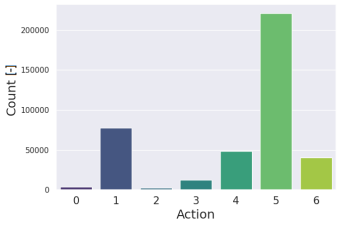


(c) Total throughput [Mbps] over time

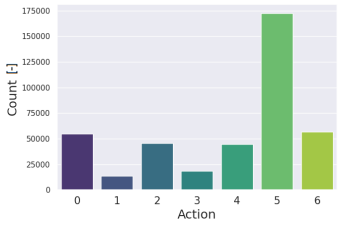
Fig. 4.28: Result of the metrics after applying **Hybrid DDDQN+Treshold** over one week measurement

rithms show a strong preference for action 5 (BBU2 and BBU3 are active). The biggest difference is in choosing action 1 (only BBU2 is active), where in case of DDDQN is chosen more.

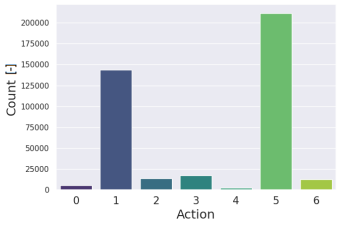
For Site Gama, Fig. 4.29c, both Hybrid and DDDQN approaches favor action 5 (activation of BBUs 2 and 3), followed by action 1 (only BBU2 is active). DDDQN demonstrates a slightly more diverse action distribution compared to the Hybrid model, particularly for actions 2 and 3, while both methods rarely select action 4 (BBU1 and BBU3 are active).



(a) Action distribution Beta



(b) Action distribution Delta



(c) Action distribution Gama

Fig. 4.29: Total sleep count of BBUs for each cell site over one week measurement with **Hybrid DDDQN+Treshold**

The introduction of the Hybrid DDDQN with Threshold marks an enhancement in the management of BBUs within 5G+ networks. By fusing Threshold-based rules with the adaptive learning capabilities of DDDQN, the Hybrid model achieves a harmonious balance between reliability and adaptability. This integration not only ensures consistent

energy savings and network performance but also empowers the system to evolve and optimize its policies in response to dynamic network conditions. Future work may explore further refinements of the Hybrid model, such as dynamic threshold adjustments and more sophisticated learning algorithms, to push the boundaries of energy-efficient network management.

4.5.7 Dueling Double Deep Q-Network (DDDQN) with Fallback

In this section, we introduce the DDDQN with Fallback, a safety enhancement designed to mitigate the impact of inaccurate predictions and unforeseen traffic bursts. This method extends the capabilities of DDDQN and provides a static fallback trigger, ensuring that whenever large discrepancies between predicted and actual throughput are observed, the algorithm promptly reactivates all BBUs. Consequently, this mechanism preempts potential overloads or quality degradation by maintaining sufficient capacity during unexpected traffic surges.

Even with advanced DDDQN decision making, disparities between $T_{s,i}^{\text{pred}}(t-1)$ and $T_{s,i}^{\text{act}}(t-1)$ can arise due to sudden changes in user demand or prediction inaccuracies. Here, the fallback acts as a safeguard: if the following condition

$$\Delta_{s,i}(t-1) \;=\; \left| \frac{T_{s,i}^{\text{act}}(t-1) \;-\; T_{s,i}^{\text{pred}}(t-1)}{T_{s,i}^{\text{pred}}(t-1)} \right| \;>\; \theta \tag{4.48}$$

is met for any BBU i at site s, then all BBUs at site s revert to active status, i.e., $\text{Status}_{s,i}(t) = 1$ for all $i \in \{1, 2, 3\}$. It is described in more detail mathematically in Alg. 2.2. This conservative static threshold θ (e.g., 10%) has proven effective in balancing energy efficiency with stability for many use cases. However, overly frequent triggers may occur under moderate but recurring forecast errors.

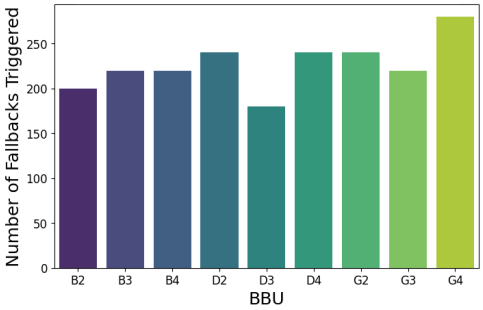
By adding this fallback mechanism to the DDDQN agent, PESBiU 2.0 ensures that the decision engine never puts the network at risk of overload when significant deviations are detected. Although energy savings may be modestly reduced during these triggered intervals, the fallback approach provides good performance in real-world deployments where some degree of predictive error is unavoidable.

Performance Evaluation and Results for DDDQN with Fallback

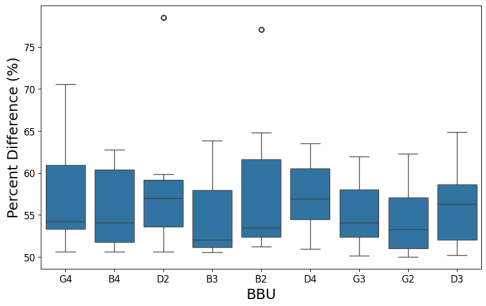
The fallback mechanism in the RL framework was tested by introducing controlled discrepancies between predicted and actual traffic through the `preprocess_states` function, which applied random normal variations to the predicted throughput. The standard deviation of these variations was set based on the worst prediction outputs from Stage 1 traffic predictions, around 0.2, simulating real-world unpredictability. During each environment step, the system calculated the relative difference between predicted and actual traffic for each BBU, and if this difference exceeded a predefined threshold, the fallback was triggered, forcing all BBUs at the affected site to remain active to prevent overloads.

These fallback events were logged, capturing details such as the frequency of fallbacks triggered by each BBU, which can be seen in Fig. 4.30a, and percentage difference as

shown in Fig. 4.30b. This testing ensured that the fallback mechanism reliably safeguarded against unexpected traffic surges, enhancing the overall robustness of the PESBiU 2.0. The overall results are further explained in Chap. 5.



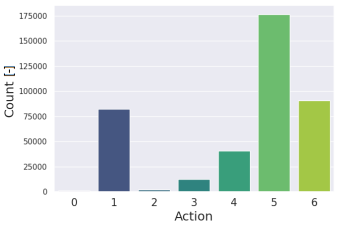
(a) Number of Fallbacks Triggered



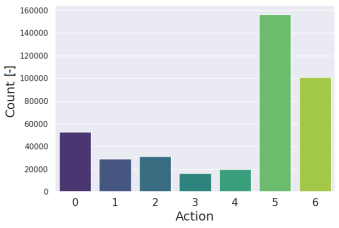
(b) Percent Differences of Fallbacks

Fig. 4.30: Triggered Fallback and related percent differences over a week

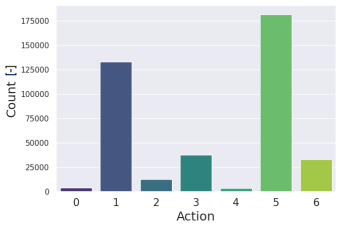
The effectiveness of DDDQN with Fallback can be seen in Fig. 4.31, where the number of times action 6 was chosen increased rapidly compared to Fig. 4.23.



(a) Action distribution Beta



(b) Action distribution Delta



(c) Action distribution Gama

Fig. 4.31: Total sleep count of BBUs for each cell site over one week measurement with **DDDQN+fallback**

The DDDQN with Fallback represents an enhancement in PESBiU 2.0 framework, balancing energy efficiency with network performance. By integrating advanced learning architectures with safety mechanisms, this approach optimizes energy consumption while ensuring service quality. Although energy savings may be slightly lower compared to more aggressive strategies, this trade-off results in reduced latency and improved network stability, particularly during unexpected traffic spikes that might be harder to predict. It should be noted that the simulations intentionally introduced prediction errors through the `preprocess_states` function, so that the real-world performance may exhibit even better outcomes. The fallback mechanism ensures that critical services remain uninterrupted, making it a robust and practical solution for real-world deployments.

4.5.8 Dueling Double Deep Q-Network (DDDQN) with Adaptive Fallback

While the Static Fallback presented above is straightforward and reliable, it may seem overly conservative or may be easily triggered under fluctuating forecast conditions. Hence,

we proposed an Adaptive Fallback threshold, which adjusts dynamically, based on historical accuracy and time-of-day variations.

Instead of using a constant threshold θ, we define a time-varying threshold $\theta_{s,i}(t)$:

$$\theta_{s,i}(t) \;=\; \theta_0 \;-\; \alpha_{\text{acc}}\,\big(\text{AvgErr}(t,w)\big) \;-\; \alpha_{\text{tod}}\,\text{TodF}(t). \tag{4.49}$$

Here, θ_0 is a baseline threshold (e.g., 10%), $\text{AvgErr}(t,w)$ is the rolling average of relative prediction errors over the last w intervals, and $\text{TodF}(t)$ captures time-of-day effects (e.g. higher forecast volatility during busy hours). Coefficients α_{acc} and α_{tod} control sensitivity to recent accuracy and diurnal cycles, respectively.

The Adaptive Fallback triggers reactivation of all BBUs at site s if

$$\Delta_{s,i}(t-1) \;>\; \theta_{s,i}(t), \tag{4.50}$$

for any BBU i. By tightening or loosening the threshold according to real-time conditions, the system avoids unnecessary fallback events when forecasts are stably accurate, yet swiftly responds under higher error conditions. This refined approach improves efficiency in scenarios with moderate variability, as fewer false alarms reduce the number of forced wake-ups.

Empirical evaluations indicate that the Adaptive Fallback balances responsiveness with reliability, generally outperforming the static fallback in environments characterized by predictable baseline traffic but intermittent high spikes. It also maintains service continuity by reverting to a fully active configuration whenever prediction errors become large.

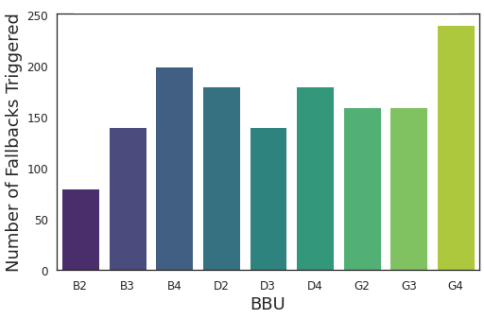
Performance Evaluation and Results for DDDQN with Adaptive Fallback

The Adaptive Fallback mechanism was evaluated under the same conditions as the Static Fallback baseline. However, instead of relying on a fixed threshold, the system continuously adjusted the trigger level, based on recent prediction accuracy and time-of-day effects. This dynamic adaptation aimed to reduce overly frequent fallback events, thereby balancing energy savings with QoS more effectively.

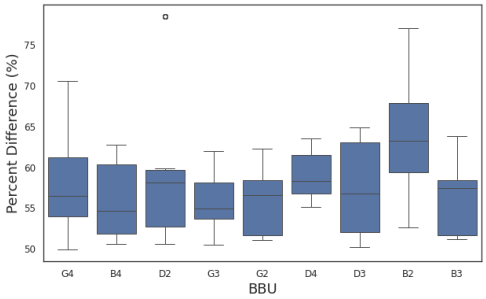
Similar to the Static Fallback setup, random normal variations were introduced to the predicted throughput in `preprocess_states`, simulating moderate forecasting errors seen in Stage 1. Whenever the relative error exceeded the current adaptive threshold, the fallback was activated for all BBUs at the affected site. Fewer site-wide reactivations were observed due to tightening or loosening the adaptive threshold based on historical accuracy. The logs recorded both the frequency and the magnitude of these triggers for each BBU, as shown in Fig. 4.32.

As seen in Fig. 4.32a, the total number of fallback events decreased compared to the Static Threshold setup, see Fig. 4.30a. The distribution of percent differences in Fig. 4.32b showed more large deviations that triggered a system-wide reactivation. These outcomes confirm that the mechanism successfully curbed unnecessary fallback operations while still responding quickly to substantial forecast inaccuracies.

The resulting action flow under Adaptive Fallback was similar to the Static Fallback approach in terms of general policies. However, fewer abrupt transitions appeared,



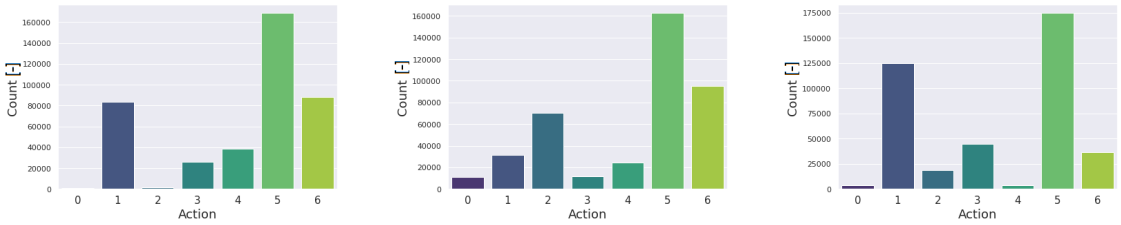
(a) Number of Adaptive Fallbacks Triggered



(b) Percent Differences of Adaptive Fallbacks

Fig. 4.32: Triggered Adaptive Fallback and related percent differences over a week

because the refined threshold avoided reactivations under moderate error conditions. Fig. 4.33 illustrates how the action distributions remained consistent with the original DDDQN strategy, see Fig. 4.23, with only moderate increases in fully active states during high error intervals.



(a) Action distribution Beta (b) Action distribution Delta (c) Action distribution Gama

Fig. 4.33: Total sleep count of BBUs for each cell site over one week with **DDDQN+Adaptive fallback**

These findings demonstrate that adopting a dynamic threshold reduces fallback frequency. At the same time, the BBU power savings remain on par with the static approach, as the system only tightens the criterion under stable forecasting conditions. This adaptability translates into higher overall efficiency, less service interruption during modest prediction errors, and better responsiveness to sudden spikes. Additional analysis and broader results are discussed in Chap. 5.

Overall, the DDDQN with Adaptive Fallback feature enhances PESBiU 2.0 and the fallback itself by refining the trade-off between safety-driven reactivations and unnecessary fallback triggers. In practical deployments, this approach further stabilizes the network's operational flow and energy footprint, reflecting a balanced strategy for maintaining QoS under fluctuating traffic conditions.

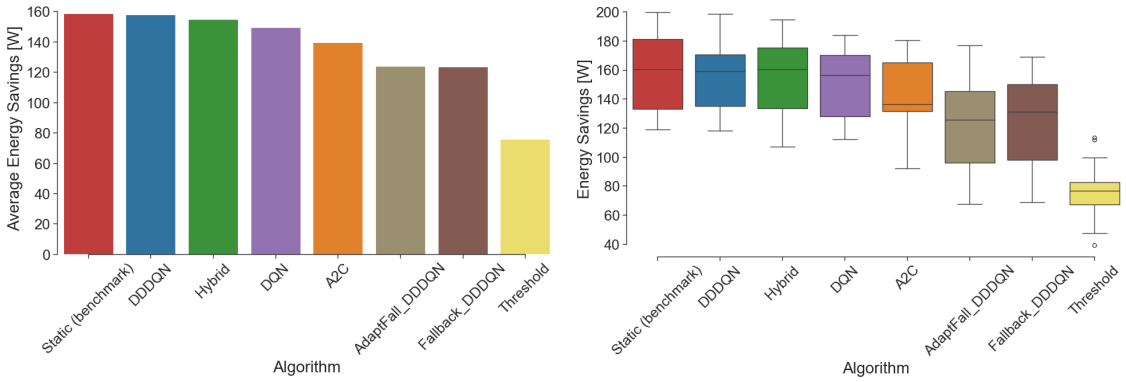
5 Analysis and Discussion

This chapter analyzes the performance of the various algorithms implemented in the PESBiU 2.0 framework for energy consumption management in BBUs. The evaluated algorithms include the Static rule-based algorithm, Threshold algorithm, Deep Q-Network (DQN), Dueling Double Deep Q-Network (DDDQN), Advantage Actor-Critic (A2C), and three special versions: Hybrid DDDQN with Threshold, DDDQN with Fallback and DDDQN with Adaptive Fallback. Among these, the Hybrid DDDQN, DDDQN with Fallback and Adaptive Fallback, Static rule-based, and Threshold algorithms represent original contributions specifically designed and developed for this research. The remaining algorithms - DQN, DDDQN and A2C are well established, however, substantial effort was dedicated to finetuning, optimizing hyperparameters, and adapting them to ensure their effectiveness within our unique dataset and operational context. Moreover, leveraging DDDQN in PESBiU 2.0 decision engine makes it the first known implementation of DDDQN specifically tailored for BBU energy consumption optimization in 5G and beyond networks.

Assessment presented in this chapter focuses on comparing the overall performance of the implemented algorithms across multiple metrics simultaneously - average energy savings, latency, throughput, and overall efficiency. Unlike the previous chapter, which individually assessed the algorithm behavior and specific design choices, this chapter provides a comparative analysis. It highlights relative strengths and trade-offs between the algorithms, using data from simulations conducted over a one-week period at 15-minute intervals from three distinct cell sites (Beta, Delta, Gama). The aim here is to offer a broader perspective, identifying the most effective solutions for balancing energy consumption and network performance under realistic operating conditions.

5.1 Energy Savings Analysis

The primary objective of the PESBiU 2.0 framework is to reduce the energy consumption of BBUs by intelligently managing their active and sleep states. Fig. 5.1a illustrates the average energy savings per day achieved by each algorithm over the one-week simulation period for all cell sites together. The Static algorithm serves as the benchmark, achieving an average saving of approximately 160 W, due to its aggressive approach of putting BBUs to sleep based on predefined heuristics without considering dynamic network conditions. It is followed closely by DDDQN and the Hybrid approach. Moreover, the Hybrid algorithm, which combines the strengths of DDDQN and the Threshold algorithm, outperformed the DQN and A2C algorithms. However, DQN and A2C also provided competitive savings. The DDDQN with Fallback/Adaptive Fallback and Threshold algorithms achieve lower energy savings. In case of Threshold algorithm it was due to conservative strategy designed to prioritize network stability over aggressive energy reduction. In case of DDDQN with Fallback/Adaptive Fallback it was due to simulating moderate forecasting errors in Stage 1.



(a) Average Energy Savings (b) Distribution of Energy Savings

Fig. 5.1: Comparison of average daily energy savings for all algorithms across all cell sites in one week

Fig. 5.1b provides a deeper look into the variability of energy savings for each algorithm. Notably, the Static algorithm exhibits consistent performance with a narrow inter-quartile range. In comparison, the RL methods (DQN, A2C, and DDDQN) show wider variability, suggesting dynamic adaptation to environment but occasional underperformance. The Hybrid DDDQN merges stability with dynamic learning, yielding a strong median. Compared to plain DDDQN, DDDQN with Fallback and with Adaptive Fallback maintain near-similar medians but show broader spreads. Fallback mechanisms are activated when traffic forecasts deviate significantly from reality, pulling BBUs back online to avoid overloads. In the Adaptive Fallback approach, thresholds are continuously recalibrated using historical prediction accuracy and time-of-day data, preventing unnecessary fallback triggers during predictable traffic and lowering them under higher volatility, for better energy savings than in case of Static Fallback. This being said, overall, Threshold-based algorithm underperforms due to its rather conservative threshold settings, which restrict larger energy gains.

To be able to compare it with PESBiU 1.0, energy savings needs to be expressed as a percentage. Below is Tab. 5.1, which presents average energy savings in Wh and percentage per site per day using PESBiU 2.0. We achieved a **41.21%** energy savings with DDDQN, compared to 15.12% with PESBiU 1.0, representing an improvement of 26.09%.

Fig. 5.2 presents the average energy savings per day over one week achieved by each algorithm for each cell site separately, highlighting the variations in performance across different network environments. Site Gama demonstrates the highest energy savings across all algorithms, with the Static, DDDQN, and Hybrid algorithms leading. This indicates that Site Gama has favorable conditions for optimization due to its specific network configurations (such as carrier aggregation across multiple frequency bands), hardware (such as newer-version of BBUs software/updates or different configurations of connected antennas supporting additional frequency bands), and traffic patterns in that area. Site Delta shows slightly lower savings compared to the Site Gama, but the performance gap between algorithms is negligible. The Hybrid algorithm maintains competitive savings, even outperforming the Static and DDDQN algorithms. Site Beta exhibits energy savings

Algorithm	Avg. Default Energy Consumption (W)	Avg. Energy Saving (W)	Energy Saved (Wh/day)	Percentage Saved (%)
Static (benchmark)	383.19	158.62	3806.94	41.40
DDDQN	383.19	157.94	3790.56	41.21
Hybrid	383.19	154.80	3715.20	40.41
DQN	383.19	149.32	3583.68	38.97
A2C	383.19	139.66	3351.84	36.45
AdaptFall DDDQN	383.19	123.98	2975.52	32.37
Fallback DDDQN	383.19	123.30	2959.20	32.18
Threshold	383.19	75.68	1816.42	19.75

Tab. 5.1: Analysis of energy savings with PESBiU 2.0 framework: Comparison of average daily energy usage and savings for all algorithms, highlighting energy-saving performance and percentage reductions across average cell sites

similar to Site Delta, with all algorithms performing well. However, the DDDQN with Fallback/Adaptive Fallback and Threshold-based methods fall behind, due to simulating forecast errors and conservative strategies.

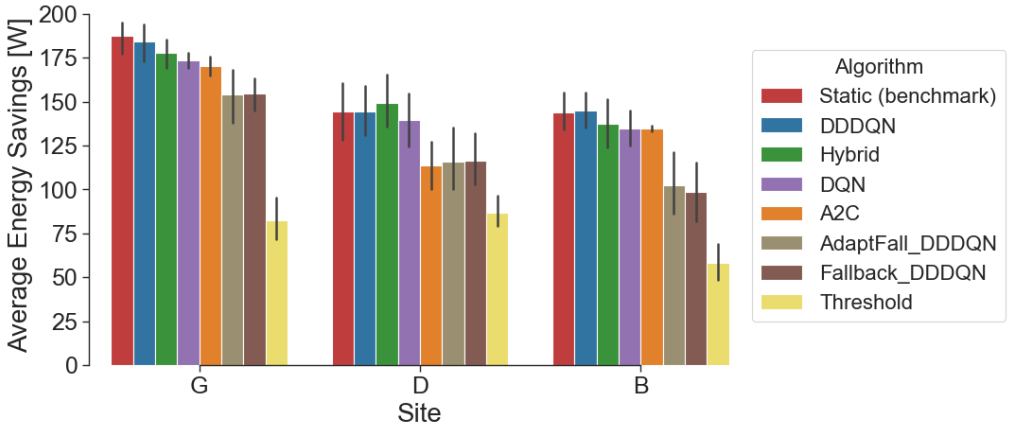


Fig. 5.2: Site-specific energy savings achieved by different algorithms: Comparison of average daily energy savings for Gama, Delta, and Beta sites across all algorithms, highlighting algorithm performance variations by site

The visual representation in Fig. 5.3 summarizes the per-site energy savings and highlights how specific environments and configurations, such as those at Sites Gama, Delta, and Beta, influence optimization results. Site Gama stands out with the highest energy savings due to its favorable network design and traffic characteristics, as noted earlier.

Additionally, the figure specifies the best-performing algorithm for each site, reinforcing the adaptability and effectiveness of AI-aided energy management approaches. This visual complement supports the tabular and graph-based analysis, offering a clearer understanding of how energy efficiency varies geographically and algorithmically.

5.2 Latency and Throughput Analysis

While energy savings are important, maintaining acceptable levels of network performance is equally important. We evaluated the impact of each algorithm on KPIs such as average

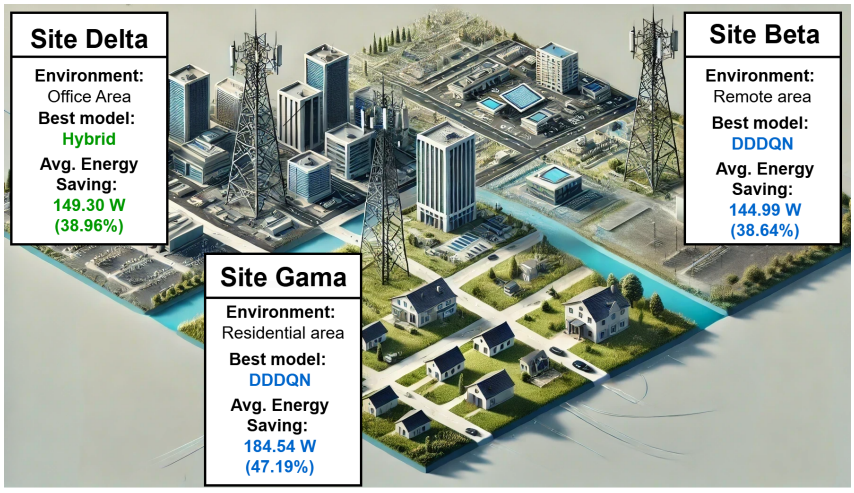
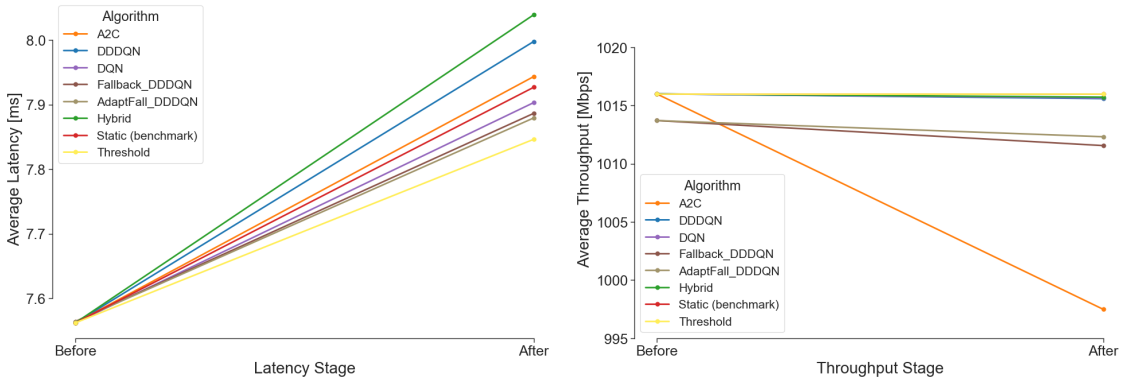


Fig. 5.3: Energy optimization across network sites: Highlights include the best-performing models (DDDQN and Hybrid) and corresponding average energy savings

latency and throughput. Fig. 5.4a compares the average UE latency before and after applying each algorithm.



(a) Latency impact

(b) Throughput impact

Fig. 5.4: Comparison before and after applying each algorithm: (a) Average latency trends across algorithms: Average downlink latency, highlighting variations in QoS impact (b) Average throughput across algorithms: Overlapping lines for Threshold, Static, Hybrid, DQN and DDDQN algorithms indicate minimal impact on network throughput, while noticeable variations are observed with A2C and Fallback/Adaptive DDDQN algorithms, highlighting their effect on network performance

The results indicate a minor, consistent increase in average latency from approximately 7.6 ms to slightly below 8.0 ms across all algorithms after their application. This increase is negligible and well within acceptable QoS requirements. The Hybrid algorithm exhibited the highest latency increase, while the Threshold algorithm showed the smallest. This negligible rise underscores the minimal trade-off between energy savings and network latency, aligning with industry practice where such minor increments are tolerated for significant efficiency gains.

Fig. 5.4b compares total port throughput before and after applying PESBiU 2.0 with different algorithms. The results show minimal changes in throughput for most algorithms, maintaining an average close to 1015 Mbps before and after application, which means that the traffic drop did not occur. However, A2C demonstrates a decrease in throughput, dropping to 997.5 Mbps post-application. This suggests that while A2C maintains reasonable performance, it introduces a minor trade-off in network throughput.

5.3 Overall Efficiency Comparison

The overall efficiency metric is computed by combining normalized values of energy savings, latency, and throughput, with each metric weighted according to its relative importance. In this evaluation, energy savings are given the highest priority (weight of 2.0), followed by throughput (weight of 1.5), and finally latency (weight of 0.4). This weighting reflects the emphasis on energy savings and the understanding that slight increases in latency and modest reductions in throughput have a minor impact on user experience [113].

Fig. 5.5 presents a radar chart comparing the overall efficiency of each algorithm. A larger polygonal area indicates better performance across the weighted metrics. For a clearer comparison, Fig. 5.6 provides a bar chart showing algorithms sorted by weighted energy savings with relevant throughput and latency.

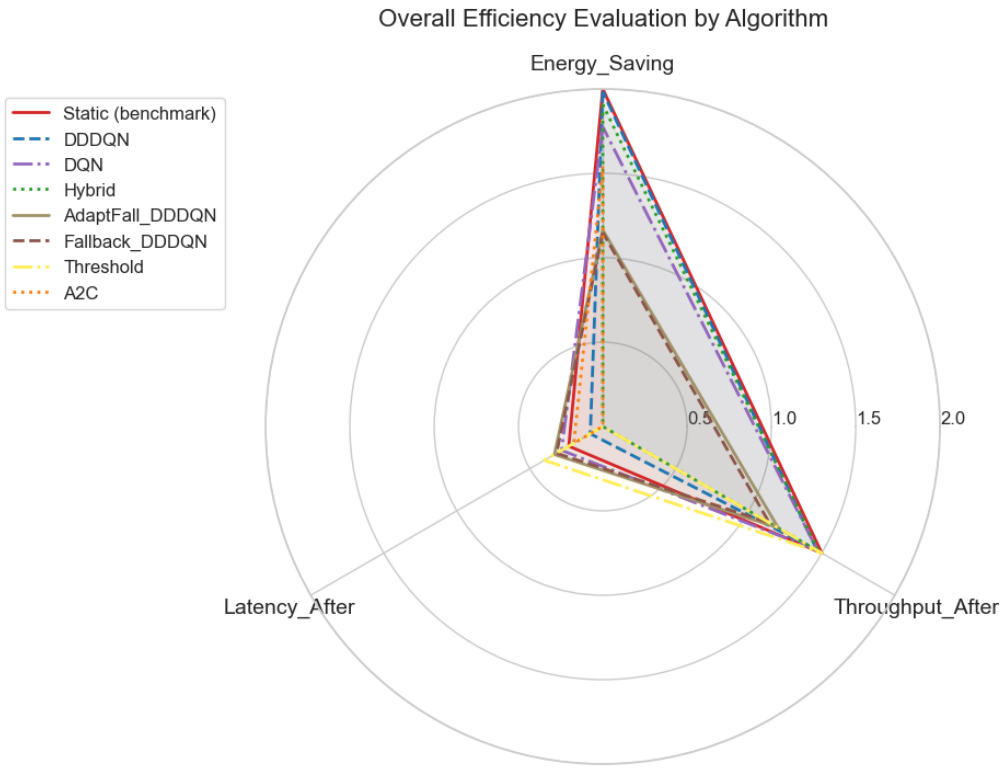


Fig. 5.5: Overall efficiency comparison of algorithms: Radar plot evaluating all algorithms based on energy savings, post-application latency, and throughput performance, highlighting trade-offs across key metrics

Based on the radar and bar chart, the results across the algorithms are close to Static

algorithm that serves as the benchmark, demonstrating that all approaches achieve a reasonable trade-off between energy savings, latency. However, DDDQN excels due to its strong energy savings and competitive throughput. Despite a slight increase in latency, its high energy and throughput scores outweigh this minor drawback, allowing it to achieve substantial overall efficiency compared to other algorithms.

DQN maintains a balanced profile with decent energy savings and throughput, but does not exceed the top performers in terms of total weighted score. The slightly lower energy gains place DQN just behind DDDQN in overall efficiency.

Hybrid DDDQN+Threshold achieves notable energy savings. Although not quite on par with DDDQN, its efficient energy profile and acceptable throughput make it a strong contender even with the highest latency increase, placing it near the top in overall efficiency.

Fallback and Adaptive Fallback DDDQN were affected by simulated forecast errors and this caused their lower overall efficiency.

Threshold does not pursue aggressive energy savings with such strong force. Consequently, it does not gain as high a score under the chosen weights, resulting in comparatively lower overall efficiency.

A2C's latency does not compensate for its lower energy savings and reduced throughput. Given the heavier emphasis on energy and throughput, A2C is low in overall efficiency.

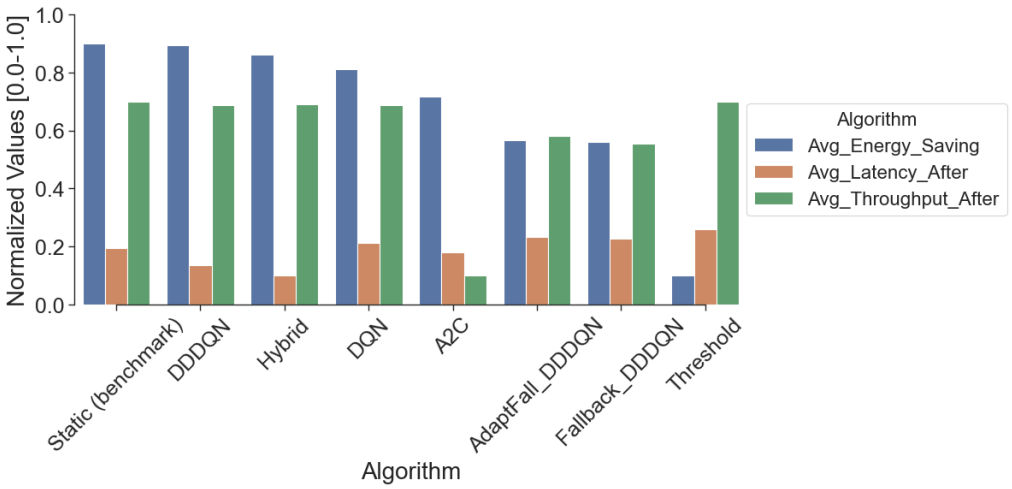


Fig. 5.6: Overall efficiency comparison of algorithms: Weighted and normalized (0.0-1.0) metrics by algorithm sorted by energy savings

5.4 Benefits of Special Versions

The following subsections describe the advantages offered by specialized algorithms designed to address the challenges of real 5G+ environments.

5.4.1 Hybrid DDDQN with Threshold

This algorithm integrates the DDDQN RL approach with a conservative Threshold algorithm to balance energy efficiency and network stability. This hybrid mechanism ensures that the BBUs are not deactivated under conditions that could degrade performance in the beginning, while the DDDQN agent learns optimal policies over time.

The Hybrid approach has a range of benefits that make it both practical and effective in managing network operations. By incorporating a Threshold component, it ensures enhanced stability during the early training phases of the DDDQN agent, preventing erratic actions and providing a reliable operational baseline. This stability allows the algorithm to focus on learning more efficiently, as the heuristic rules provide a solid foundation, enabling the DDDQN agent to refine its decisions in more complex scenarios where simple rules are insufficient. Furthermore, the combination of learning-based adaptability and rule-based reliability ensures that the system performs well under a variety of network conditions, striking a balance between flexibility and dependability. Although the energy savings are slightly lower than standalone DDDQN, the Hybrid algorithm still delivers good performance, making it an effective choice for network operators prioritizing energy efficiency without compromising service quality.

5.4.2 DDDQN with Fallback

The DDDQN with Fallback integrates a safety mechanism that is triggered whenever there is a large gap between predicted and actual traffic values. This mechanism ensures that the network remains stable under abrupt traffic surges or unexpected behavioral shifts. By reactivating all BBUs when such deviations occur, it prevents any single cell site from being overloaded due to incorrect forecasts or sudden demand.

The fallback mechanism offers several important advantages for maintaining a stable and reliable network. It acts as a safety net, ensuring that the network continues to perform even when prediction errors or unexpected traffic patterns occur. This reliability helps prevent disruptions and keeps performance consistent. It also brings stability by taking a conservative approach during unpredictable periods, preventing sudden changes in BBU statuses and promoting smoother, more predictable operations. Additionally, the fallback allows the DDDQN agent to focus on typical scenarios by managing outlier situations, which helps improve learning efficiency and refine decision-making. In settings where traffic can be highly erratic, the fallback feature presents a practical safeguard, preserving QoS and preventing excessive latency or throughput drops. As a result, DDDQN with Fallback proves well-suited for environments that must remain robust despite occasional forecasting errors.

5.4.3 DDDQN with Adaptive Fallback

The DDDDQN with Adaptive Fallback enhances the conventional fallback approach by dynamically adjusting its thresholds based on historical traffic prediction accuracy and current time-of-day conditions. Unlike the static fallback, which uses a fixed threshold to

trigger fallback actions, the adaptive variant continuously recalibrates thresholds, ensuring that fallback actions are only activated when truly necessary.

The primary advantage of the adaptive fallback mechanism lies in its dynamic responsiveness. By calculating the threshold based on past prediction errors and time-of-day trends, it significantly reduces false-positive fallback triggers during periods of predictable traffic, thus enhancing overall system efficiency. During periods of higher traffic volatility or less predictable conditions, the adaptive mechanism proactively lowers the threshold, ensuring network stability by swiftly activating fallback actions if needed.

Consequently, this approach achieves a more refined balance between energy savings and network reliability, providing tailored responses to varying operational conditions. This flexibility not only preserves service quality by preventing unnecessary activations of BBUs but also improves the learning efficiency of the DDDQN by minimizing disruptions from unpredictable network states.

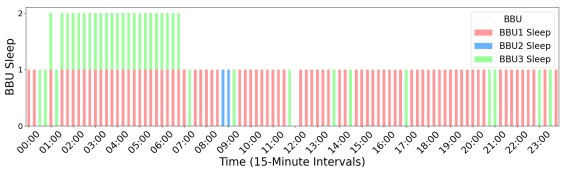
5.5 BBU Sleep Patterns on a Workday

To illustrate the practical implications of the algorithms, we analyze the BBU sleep patterns for the Site Beta on a workday - Tuesday, as this site represents a mix of office and residential network environment on the outskirt of the city, and Tuesday represents typical weekday traffic patterns. Fig. 5.7 shows the BBUs being set to sleep (0 for no BBU sleeping, 1 for one BBU sleeping, 2 for two BBUs sleeping) and their corresponding energy consumption savings by the Static, DDDQN and Hybrid algorithms throughout the day, divided into 15-minute intervals. Appendix A shows how each algorithm puts BBUs to sleep and the energy consumption saving for each site on a Tuesday.

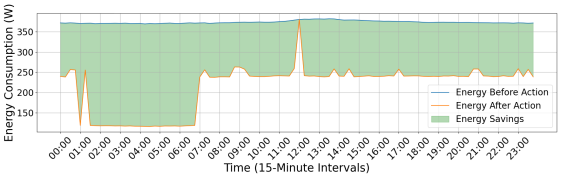
The Static algorithm serves as the benchmark, representing a consistent, rule-based approach to energy savings, while the DDDQN and Hybrid algorithms attempt to replicate or improve upon these results using adaptive, learning-based methods.

In terms of BBU sleep decisions, all the above - mentioned three algorithms aim to achieve similar outcomes by deactivating BBUs during periods of low traffic. However, the increasing dynamics and variability of traffic in 5G+ networks present significant challenges for purely static or rule-based methods. The Static algorithm operates strictly within predefined parameters, resulting in predictable and rigid scheduling that does not accommodate unexpected traffic variations effectively. In contrast, both the DDDQN and Hybrid algorithms leverage advanced RL techniques to dynamically adapt to these fluctuating conditions. They introduce nuanced variations in BBU sleep scheduling by actively responding to real-time traffic changes, thus providing greater flexibility and responsiveness compared to static methods.

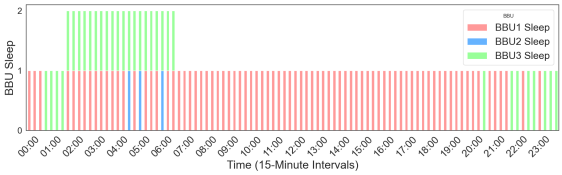
For example, in some 15-minute intervals, the DDDQN or Hybrid algorithms may choose to deactivate different BBUs than the Static algorithm would do, showcasing their ability to proactively manage the network's resources according to immediate demand. This adaptability becomes particularly beneficial during peak fluctuations or unanticipated shifts in user behavior, where static heuristics fall short. Consequently, these dynamic algorithms are better equipped to maintain network stability, optimize energy



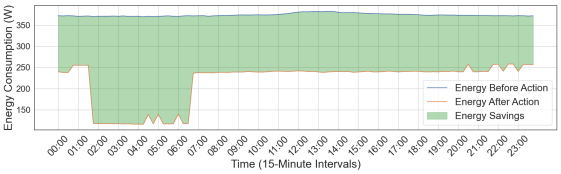
(a) Static algorithm – Distribution of BBUs entering sleep mode at specific intervals based on a static heuristic schedule



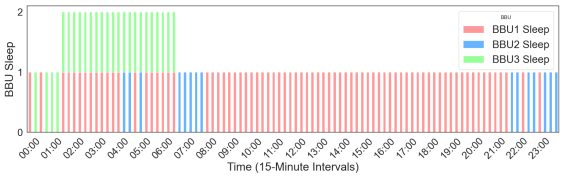
(b) Static algorithm – Energy consumption and corresponding savings achieved using the static algorithm across 15-min intervals



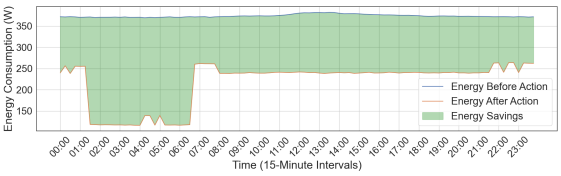
(c) DDDQN – Adaptive management of BBUs entering sleep mode based on traffic dynamics by the DDDQN agent



(d) DDDQN – Energy usage reduction through DDDQN, highlighting adaptability to traffic conditions



(e) Hybrid DDDQN+Threshold – Combined heuristic and DDDQN-based approach for putting BBUs to sleep, balancing learning-based adaptability with stability



(f) Hybrid DDDQN+Threshold – Energy savings achieved by the hybrid approach, maintaining a trade-off between dynamic optimization and stability

Fig. 5.7: Site Beta: Top-performing algorithms compared in terms of BBUs in sleep mode and energy consumption (Workday)

savings, and ensure QoS across a diverse range of operational scenarios typical of 5G+ environments. Their performance highlights the importance of deploying adaptive and intelligent systems capable of continuous learning and adjusting to complex and variable network conditions.

The results of energy consumption further emphasize these similarities and yet differences. Both aim to achieve energy savings comparable to Static benchmark.

In conclusion, the DDDQN and Hybrid algorithms strive to achieve the same outcomes as the Static algorithm benchmark. While overall their results align closely, minor differences in specific decisions highlight their adaptability and the trade-offs involved in learning-based versus rule-based approaches.

5.6 Key Findings and Reflections

In terms of average energy savings, the algorithms are ranked as follows: DDDQN, Hybrid, DQN, A2C, DDDQN with Fallback/Adaptive Fallback, and Threshold.

The analysis demonstrates that reinforcement learning-based algorithms, particularly the DDDQN and Hybrid versions, effectively balance energy efficiency and network performance in managing BBU energy consumption. Notably, to our best knowledge, this study represents the first application of the DDDQN algorithm in view of optimizing energy efficiency within 5G networks. The DDDQN algorithm outperforms other RL models such as DQN and A2C by incorporating a dueling network architecture and double Q-learning, which enhances learning stability and decision-making accuracy.

The Hybrid DDDQN with Threshold further refines this approach by integrating heuristic safety measures, offering a practical solution for real-world deployment.

The DDDQN with Fallback/Adaptive Fallback provides an added layer of reliability by safeguarding against prediction errors, making it suitable for networks with highly variable traffic patterns.

The proposed PESBiU 2.0 framework can be practically deployed at the layer managing BBU operations in a modern RAN architecture. Concretely, it can be installed within the centralized BBU pool or integrated as a software module in the baseband processing node (DU/CU) for scenarios where splitting is employed. By placing the framework at this level, it has direct access to real-time metrics (e.g. traffic throughput, latency, and energy consumption) and can immediately execute power-state changes or fallback decisions.

Only minor network modifications are required. Operators already gather performance logs (via SNMP or vendor-specific counters) for routine KPI measurements. PESBiU requires these logs at a higher frequency (e.g. every 15 minutes), so any changes focus only on adjusting data collection intervals or enabling more granular monitoring. Additional hardware replacements are unnecessary; existing servers or cloud-based nodes that host the BBU software can handle the agent.

Regarding computing demands, RL-based modules may increase CPU/GPU load during model training, but standard off-the-shelf hardware or modestly scaled edge servers suffice. Once trained, the model's inference step is fast, consuming minimal resources and causing negligible overhead on real-time operations. Operators can also shift heavy training jobs to central data centers or the cloud, and then periodically update local models.

In summary, PESBiU 2.0 can be introduced with a modest adjustment in data monitoring intervals and an existing compute node to host its agent. No radical hardware overhaul is required, and real-world integration can leverage the same control interfaces already used for baseband management in 5G RAN deployments.

Conclusion

This thesis explored the potential of integrating ML and AI into 5G+ networks to address some critical challenges, particularly energy efficiency, traffic prediction, and resource management. By leveraging real-world datasets and advanced algorithms, the presented research proposed innovative solutions to optimize the performance and sustainability of modern mobile networks.

Chapter 2 outlined the key technological components of 5G+ networks, emphasizing advancements such as network slicing, cloud-radio access networks (C-RAN), and millimeter-wave technologies. Although these innovations enable transformative capabilities, they also introduce complexity in energy management and scalability. The chapter identified energy efficiency and dynamic resource allocation as fundamental areas requiring optimization through AI/ML.

Chapters 3 and 4 highlighted the significant advancements achieved through data-driven optimization strategies. By leveraging long-term and granular datasets from diverse network environments, ML models were developed and rigorously evaluated for traffic prediction and energy consumption management. In Chapter 3, the primary dataset revealed that LSTM provided the most accurate traffic predictions. However, further refinement using the secondary dataset led to the development of the Hyper-CNN-LSTM model, which outperformed LSTM in prediction accuracy.

Chapter 4 introduced the Predictive Energy Saver for Baseband Units (PESBiU) framework, showcasing a range of reinforcement learning techniques, including Deep Q-Networks (DQN), Advantage Actor-Critic (A2C), and the Dueling Double Deep Q-Network (DDDQN). Notably, this work is groundbreaking as to our best knowledge it marks the first-ever application of the DDDQN algorithm for optimizing energy efficiency in 5G networks. Among these, DDDQN demonstrated the highest efficiency in energy savings, reaching a benchmark set by heuristic Static algorithm and achieving an impressive energy saving of 41.21%. Additionally, the hybrid version that we created by combining DDDQN with a Threshold-based mechanism also demonstrates strong performance, achieving a 40.41% energy savings.

Finally, the analysis presented in Chapter 5 validated the practical applicability of the proposed solutions through comprehensive comparative evaluations across key performance metrics—energy savings, latency, throughput, and overall efficiency. The Hybrid approach, despite slightly lower energy savings compared to standalone DDDQN, emerged as a compelling choice due to its enhanced reliability and suitability for real-world operational scenarios. Additionally, special versions such as DDDQN with Static and Adaptive Fallback introduced practical mechanisms for managing forecasting errors and traffic volatility, thus further improving network resilience and reliability.

This work has highlighted the essential importance of dynamic resource management, predictive optimization, and the adoption of hybrid AI/ML approaches. By addressing gaps in the literature and proposing innovative methodologies, this research has provided invaluable insights to both academic and practical domains of network optimization.

Future research should focus on the challenges identified in this thesis, including data

quality issues, computational complexity, and the need for explainable AI models. Addressing the above mentioned limitations will pave the way for more adaptive, efficient, and sustainable networks, ensuring that 5G+ systems meet the growing demands of modern applications and services. Moreover, expanding the scope of hybrid models to encompass additional performance metrics and incorporating federated learning for privacy-preserving data analysis could further advance this field of research.

Ultimately, it is our sincere hope that the findings of this thesis serve as a foundation for the continued evolution of telecommunication networks, positioning AI/ML as indispensable tools in the pursuit of next-generation network efficiency and innovation.

Bibliography

[1] Alliance, ORAN, "O-RAN AI/ML Workflow Description and Requirements 1.03," *ML Workflow Description and Requirements*, vol. 1, 2021. [Online]. Available: https://www.o-ran.org/specifications

[2] NGMN Alliance, "Green Future Networks – Sustainability Challenges and Initiatives in Mobile Networks," NGMN, Tech. Rep., 2021, accessed: 2024-12-29. [Online]. Available: https://www.ngmn.org/publications/sustainability-challenges-and-initiatives-in-mobile-networks.html

[3] Mavenir and Intel, "A Holistic Study of Power Consumption and Energy Savings Strategies for Open vRAN Systems," Mavenir, Tech. Rep., 2023, https://builders.intel.com/docs/networkbuilders/a-holistic-study-of-power-consumption-and-energy-savings-strategies-for-open-vran-systems-1676628842.pdf.

[4] D. López-Pérez, A. De Domenico, N. Piovesan, G. Xinli, H. Bao, S. Qitao, and M. Debbah, "A survey on 5g radio access network energy efficiency: Massive mimo, lean carrier design, sleep modes, and machine learning," *IEEE Communications Surveys Tutorials*, vol. 24, no. 1, pp. 653–697, 2022.

[5] S. Wang, M. A. Qureshi, L. Miralles-Pechuán, T. Huynh-The, T. R. Gadekallu, and M. Liyanage, "Explainable AI for 6G Use Cases: Technical Aspects and Research Challenges," *IEEE Open Journal of the Communications Society*, vol. 5, pp. 2490–2540, 2024.

[6] B. Brik, H. Chergui, L. Zanzi, F. Devoti, A. Ksentini, M. S. Siddiqui, X. Costa-Pérez, and C. Verikoukis, "Explainable AI in 6G O-RAN: A Tutorial and Survey on Architecture, Use Cases, Challenges, and Future Research," 2024. [Online]. Available: https://arxiv.org/abs/2307.00319

[7] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. K. Soong, and J. C. Zhang, "What will 5g be?" *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1065–1082, 2014.

[8] P. Popovski, K. F. Trillingsgaard, O. Simeone, and G. Durisi, "5G Wireless Network Slicing for eMBB, URLLC, and mMTC: A Communication-Theoretic View," *IEEE Access*, vol. 6, pp. 55 765–55 779, 2018.

[9] E. Calvanese Strinati and S. Barbarossa, "6G networks: Beyond Shannon towards semantic and goal-oriented communications," *Computer Networks*, vol. 190, p. 107930, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128621000773

[10] T. S. Rappaport, S. Sun, R. Mayzus, H. Zhao, Y. Azar, K. Wang, G. N. Wong, J. K. Schulz, M. Samimi, and F. Gutierrez, "Millimeter wave mobile communications for 5g cellular: It will work!" *IEEE Access*, vol. 1, pp. 335–349, 2013.

[11] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta, "Massive MIMO for next generation wireless systems," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 186–195, 2014.

[12] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[13] X. Foukas, G. Patounas, A. M. Elmokashfi, and M. K. Marina, "Network Slicing in 5G: Survey and Challenges," *IEEE Communications Magazine*, vol. 55, pp. 94–100, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:206456479

[14] E. Dahlman, S. Parkvall, and J. Skold, *5G NR: The Next Generation Wireless Access Technology*. Academic Press, 2020.

[15] X. Li, M. Samaka, H. A. Chan, D. Bhamare, L. Gupta, C. Guo, and R. Jain, "Network slicing for 5G: Challenges and opportunities," *IEEE Internet Computing*, vol. 21, no. 5, pp. 20–27, 2017.

[16] H. Zhang, S. Huang, C. Jiang, K. Long, V. C. M. Leung, and H. V. Poor, "Energy Efficient User Association and Power Allocation in Millimeter-Wave-Based Ultra Dense Networks With Energy Harvesting Base Stations," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 9, pp. 1936–1947, 2017.

[17] N. Sharma and K. Kumar, "Energy Efficient Clustering and Resource Allocation Strategy for Ultra-Dense Networks: A Machine Learning Framework," *IEEE Transactions on Network and Service Management*, vol. 20, no. 2, pp. 1884–1897, 2023.

[18] J. G. Andrews, S. Buzzi, C. Wan, S. V. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, "Modeling and analyzing millimeter wave cellular systems," *IEEE Transactions on Communications*, vol. 65, no. 1, pp. 403–430, 2017.

[19] M. Shafi, A. F. Molisch, P. J. Smith, T. Haustein, P. Zhu, P. J. De Silva, F. Tufvesson, A. Benjebbour, and G. Wunder, "5G: A tutorial overview of standards, trials, challenges, deployment, and practice," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 6, pp. 1201–1221, 2017.

[20] M. Zhu, J. Gu, T. Shen, C. Shi, and X. Ren, "Energy-Efficient and QoS Guaranteed BBU Aggregation in CRAN Based on Heuristic- Assisted Deep Reinforcement Learning," *Journal of Lightwave Technology*, vol. 40, no. 3, pp. 575–587, 2022.

[21] T. Saraiva, D. Duarte, I. Pinto, and P. Vieira, "An Improved BBU/RRU Energy Consumption Predictor for 4G and Legacy Mobile Networks using Mixed Statistical Models," in *2020 International Conference on Computing, Networking and Communications (ICNC)*, 2020, pp. 320–325.

[22] A. Yusuf, H. Wen, M. Zhang, and Q. Tang, "Delay-Aware Energy-Saving Strategies for BBU Pool in C-RAN: Modeling and Optimization," *IEEE Access*, vol. 8, pp. 123 456–123 470, 2024.

[23] M. R. Aktar, M. S. Anower, M. Z. I. Sakar, H. K. Bappy, and A. A. R. Janee, "Energy Efficient Bandwidth Aware Virtualized BBU Pool for 5G C-RAN," *2024 6th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT)*, pp. 1–6, 2024.

[24] S. Saafi, J. Hosek, and A. Kolackova, "Cellular-enabled Wearables in Public Safety Networks: State of the Art and Performance Evaluation," in *2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 2020, pp. 201–207.

[25] ——, "Enabling Next-Generation Public Safety Operations with Mission-Critical Networks and Wearable Applications," *Sensors*, vol. 21, no. 17, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/17/5790

[26] W. Chen, L. Zhang, and J. Zhao, "Energy-Efficient and QoS-Guaranteed BBU Aggregation in C-RAN Based on Heuristic-Assisted Deep Reinforcement Learning," *IEEE Transactions on Network and Service Management*, vol. 17, no. 3, pp. 1008–1020, 2024.

[27] A. Kolackova and J. Jerabek, "Strategic Capacity Planning and Optimization in Communication Networks: A Case Study," *Proceedings II of the 30st Conference STUDENT EEICT 2024: Selected papers.*, vol. 237, pp. 166–170, 2024. [Online]. Available: https://dspace.vut.cz/items/4d039d6a-093b-45c0-a922-fa3c8c47d9c5

[28] A. Kolackova, S. Saafi, P. Masek, J. Hosek, and J. Jerabek, "Performance Evaluation of Carrier Aggregation in LTE-A Pro Mobile Systems," in *2020 43rd International Conference on Telecommunications and Signal Processing (TSP)*, 2020, pp. 627–632.

[29] A. Kolackova, J. Hosek, J. Jerabek, and P. Masek, "Experimental Verification of Segment Routing Methods in 5G+ Mobile Transport Networks," in *2021 13th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 2021, pp. 38–43.

[30] P. Hao and F. Yang, "SRv6 Strategy Based on Network State and Behavior Prediction in 5G Network," *2023 9th International Conference on Computer and Communications (ICCC)*, pp. 1044–1055, 2023.

[31] M. Wani, M. Kretschmer, B. Schröder, A. Grebe, and M. Rademacher, "Open RAN: A Concise Overview," *IEEE Open Journal of the Communications Society*, vol. 6, pp. 13–28, 2025.

[32] Huawei Technologies Co., Ltd., "5G Power Whitepaper," https://carrier.huawei.com/~/media/CNBG/Downloads/Spotlight/5g/5G-Power-White-Paper-en.pdf, 2019, published: 2019-02-27. Accessed: 2025-03-19.

[33] J. A. Ayala-Romero, I. Khalid, A. Garcia-Saavedra, X. Costa-Perez, and G. Iosifidis, "Experimental Evaluation of Power Consumption in Virtualized Base Stations," in

ICC 2021 - IEEE International Conference on Communications, Proceedings. IEEE, 2021, pp. 1–6. [Online]. Available: https://doi.org/10.1109/ICC42927.2021.9500323

[34] NGMN Alliance, "Green Future Networks: Metering in Virtualised RAN Infrastructure," https://www.ngmn.org/publications/metering-in-virtualised-ran-infrastructure.html, 2024, accessed: 2025-03-20.

[35] H. Zhang, J. Liu, M. Wen, and Y. Lin, "Toward Greener 5G and Beyond Radio Access Networks: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 2, pp. 1167–1190, 2023.

[36] D. Liu, X. Liu, R. Wang, L. Sun, P. Zhao, Y. Chang, H. Yao, and H. Zhang, "Research on Energy Consumption Optimization Method of 5G Base Station Based on Federation Learning," pp. 603–607, 12 2023.

[37] A. Ichimescu, N. Popescu, E. C. Popovici, and A. Toma, "Energy Efficiency for 5G and Beyond 5G: Potential, Limitations, and Future Directions," *Sensors*, vol. 24, no. 22, 2024. [Online]. Available: https://www.mdpi.com/1424-8220/24/22/7402

[38] Mughees, A., Tahir, M., Sheikh, M. A., Ahad, A., "Towards energy efficient 5G networks using machine learning: Taxonomy, research challenges, and future research directions." *IEEE Access*, vol. 8, p. 187498–187522, 2020.

[39] T. Taleb, C. Benzaïd, R. A. Addad, and K. Samdanis, "AI/ML for beyond 5G systems: Concepts, technology enablers solutions," *Computer Networks*, vol. 237, p. 110044, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128623004899

[40] D. Bergmann, "What is self-supervised learning?" 2023, accessed: 2025-02-08. [Online]. Available: https://www.ibm.com/think/topics/self-supervised-learning

[41] R. Balestriero, M. Ibrahim, V. Sobal, A. Morcos, S. Shekhar, T. Goldstein, F. Bordes, A. Bardes, G. Mialon, Y. Tian, A. Schwarzschild, A. G. Wilson, J. Geiping, Q. Garrido, P. Fernandez, A. Bar, H. Pirsiavash, Y. LeCun, and M. Goldblum, "A Cookbook of Self-Supervised Learning," 2023. [Online]. Available: https://arxiv.org/abs/2304.12210

[42] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[43] S. Yeh, S. Bhattacharya, R. Sharma, and H. Moustafa, "Deep Learning for Intelligent and Automated Network Slicing in 5G Open RAN (ORAN) Deployment," *IEEE Open Journal of the Communications Society*, vol. 5, pp. 64–70, 2024.

[44] Y. Ma, T. Li, Y. Zhou, L. Yu, and D. Jin, "Mitigating Energy Consumption in Heterogeneous Mobile Networks Through Data-Driven Optimization," *IEEE Transactions on Network and Service Management*, vol. 21, no. 4, pp. 4369–4382, 2024.

[45] L. Chen, D. Yang, D. Zhang, C. Wang, J. Li, and T.-M.-T. Nguyen, "Deep mobile traffic forecast and complementary base station clustering for C-RAN optimization," *Journal of Network and Computer Applications*, vol. 121, pp. 59–69, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804518302455

[46] Y. Zhu and S. Wang, "Traffic prediction enabled dynamic access points switching for energy saving in dense networks," *Digital Communications and Networks*, vol. 9, no. 4, pp. 1023–1031, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352864822001109

[47] V. e. a. Mnih, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.

[48] Z. Wang, T. Schaul, M. Hessel, H. van Hasselt, M. Lanctot, and N. de Freitas, "Dueling Network Architectures for Deep Reinforcement Learning," 2016. [Online]. Available: https://arxiv.org/abs/1511.06581

[49] T. Pamuklu, M. Erol Kantarci, and C. Ersoy, "Reinforcement Learning Based Dynamic Function Splitting in Disaggregated Green Open RANs," 02 2021.

[50] F. B. Mismar, B. L. Evans, and A. Alkhateeb, "Deep Reinforcement Learning for 5G Networks: Joint Beamforming, Power Control, and Interference Coordination," *IEEE Transactions on Communications*, vol. 68, no. 3, pp. 1581–1592, 2020.

[51] D.-H. Tran, N. V. Huynh, S. Kaada, V. N. Vo, E. Lagunas, and S. Chatzinotas, "Deep Reinforcement Learning for Network Energy Saving in 6G and Beyond Networks," 2024. [Online]. Available: https://arxiv.org/abs/2408.10974

[52] R. Tan, Y. Shi, Y. Fan, W. Zhu, and T. Wu, "Energy Saving Technologies and Best Practices for 5G Radio Access Network," *IEEE Access*, vol. 10, pp. 51 747–51 756, 2022.

[53] B. Salama Attia, A. Elgharably, M. Nabil Aboelwafa, G. Alsuhli, K. Banawan, and K. G. Seddik, "Self-Optimized Agent for Load Balancing and Energy Efficiency: A Reinforcement Learning Framework With Hybrid Action Space," *IEEE Open Journal of the Communications Society*, vol. 5, pp. 4902–4919, 2024.

[54] F. Salahdine, J. Opadere, Q. Liu, T. Han, N. Zhang, and S. Wu, "A survey on sleep mode techniques for ultra-dense networks in 5G and beyond," *Computer Networks*, vol. 201, p. 108567, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128621004801

[55] D. Shukla and S. D. Sawarkar, "Enhancing energy efficiency and QoS in 5G networks with dynamic resource optimisation green communication protocol," *Int.*

J. Wire. Mob. Comput., vol. 28, no. 1, p. 68–85, Dec. 2024. [Online]. Available: https://doi.org/10.1504/ijwmc.2025.143015

[56] B. J. R. Sahu, S. Dash, N. Saxena, and A. Roy, "Energy-Efficient BBU Allocation for Green C-RAN," *IEEE Communications Letters*, vol. 21, no. 7, pp. 1637–1640, 2017.

[57] F. Khoramnejad, R. Joda, A. B. Sediq, H. Abou-Zeid, R. Atawia, G. Boudreau, and M. Erol-Kantarci, "Delay-Aware and Energy-Efficient Carrier Aggregation in 5G Using Double Deep Q-Networks," *IEEE Transactions on Communications*, vol. 70, no. 10, pp. 6615–6629, 2022.

[58] M. Bordin, A. Lacava, M. Polese, S. Satish, M. A. Nittoor, R. Sivaraj, F. Cuomo, and T. Melodia, "Design and Evaluation of Deep Reinforcement Learning for Energy Saving in Open RAN," 2025. [Online]. Available: https://arxiv.org/abs/2410.14021

[59] M. Diamanti, G. Kapsalis, E. E. Tsiropoulou, and S. Papavassiliou, "Energy-Efficient Rate-Splitting Multiple Access: A Deep Reinforcement Learning-Based Framework," *IEEE Open Journal of the Communications Society*, vol. 4, pp. 2397–2409, 2023.

[60] S. Jordan, Y. Chandak, D. Cohen, M. Zhang, and P. Thomas, "Evaluating the Performance of Reinforcement Learning Algorithms," 06 2020.

[61] U. Labs, "OpenCelliD: The World's Largest Open Database of Cell Towers," 2025, accessed: 2025-02-14. [Online]. Available: https://opencellid.org/

[62] Ofcom, "UK Mobile Network Coverage and Performance Data," 2024, accessed: 2025-02-14. [Online]. Available: https://www.ofcom.org.uk/about-ofcom/our-research/opendata/

[63] Federal Communications Commission, "Mobile Broadband Coverage and Performance Data," 2024, accessed: 2025-02-14. Available at https://www.fcc.gov/reports-research. [Online]. Available: https://www.fcc.gov/reports-research

[64] B. Oancea, D. Salgado, S. Barragán, and M. Necula, "Use of Simulation Models for the Development of a Statistical Production Framework for Mobile Network Data with the simutils Package," 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:246063355

[65] B. Sliwa, M. Patchou, and C. Wietfeld, "The Best of Both Worlds: Hybrid Data-Driven and Model-Based Vehicular Network Simulation," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6.

[66] H. Bhute, G. T. Chavan, and A. N. Bhute, "Analysis of Location Management Schemes for MANET using Synthetic Mobility Models," *ArXiv*, vol. abs/1602.02493, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:14674095

[67] S. Chowdhury, S. Di Nardo, A. Hindle, and Z. M. Jiang, "An exploratory study on assessing the energy impact of logging on android applications," *Empirical Software Engineering*, vol. 23, pp. 1422–1456, 2018.

[68] S. Manas Kala, M. Mishra, A. Biju, V. Sathya, T. Amano, T. Higashino, H. Yamaguchi, and B. Reddy Tamma, "Architecture, Performance, and Usability of Mobile Cellular Network Monitoring Applications for Data-Driven Analysis," *IEEE Access*, vol. 12, pp. 88 426–88 444, 2024.

[69] A. Kolackova, J. Jerabek, and J. Hosek, "Integrating Real-Environment 4G and 5G Mobile Data Patterns into Network Simulations for Delay Analysis," *Proceedings II of the 29th Conference STUDENT EEICT 2023: Selected papers.*, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:260039628

[70] M. Conti, Q. Li, A. Maragno, and R. Spolaor, "The dark side(-channel) of mobile devices: A survey on network traffic analysis," 2018. [Online]. Available: https://arxiv.org/abs/1708.03766

[71] International Telecommunication Union, "Recommendation E.800: Definitions of Terms Related to Quality of Service," ITU-T, Tech. Rep., 2008, accessed: 2024-07-01. [Online]. Available: https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-E.800-200809-I!!PDF-E&type=items

[72] "Telecommunication management; Key Performance Indicators (KPI) for Evolved Universal Terrestrial Radio Access Network (E-UTRAN): Definitions," ETSI, Tech. Rep. TS 132450 V17.0.0, April 2022. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/132400_132499/132450/17.00.00_60/ts_132450v170000p.pdf

[73] "Telecommunication management; Key Performance Indicators (KPI) for Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Requirements," ETSI, Tech. Rep. TS 132451 V16.0.0, November 2020. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/132400_132499/132451/16.00.00_60/ts_132451v160000p.pdf

[74] "NGMN 5G White Paper: End-to-End 5G KPIs," NGMN Alliance, Tech. Rep., 2015. [Online]. Available: https://www.ngmn.org/wp-content/uploads/NGMN_5G_White_Paper_V1_0.pdf

[75] "Telecommunication management; Performance Management (PM); Performance measurements Evolved Universal Terrestrial Radio Access Network (E-UTRAN)," 3GPP, Tech. Rep. TS 32.425, 2015. [Online]. Available: https://www.3gpp.org/DynaReport/32425.htm

[76] 3rd Generation Partnership Project (3GPP), "Management and orchestration; Performance assurance," 3GPP, Technical Specification TS 28.550, 2018. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3411

[77] "rApps for More Sustainable Networks," https://www.ericsson.com/en/ran/intelligent-ran-automation/intelligent-automation-platform/rapps/rapps-for-more-sustainable-networks, accessed: 2025-03-09.

[78] Ericsson, "Transport Network Performance Indicators," Ericsson AB, Tech. Rep., 2020, version 5/0363-90/FCP 130 0532 Uen Rev C.

[79] ——, "Key Performance Indicators," Ericsson AB, Tech. Rep., 2024, version 37/1553-LZA 701 6014/1 Uen AC.

[80] Huawei Technologies, "5G RAN2.0 KPI Introduction," Huawei, Tech. Rep., 2023, accessed: 2025-01-09. [Online]. Available: https://slideshare.net/slideshow/4092827765gran20kpiintroductionpptx/266260726#1

[81] Nokia, "OAM-PM Latency (NSP) Report," https://documentation.nokia.com/nsp/24-8/Analytics/oam_pm_latency_nsp.html, accessed: 2025-03-09.

[82] A. Awada *et al.*, "Mobility Modeling and Performance Evaluation of Multi-Connectivity in 5G Intra-Frequency Networks," *Proceedings of the 2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, pp. 1–5, 2018.

[83] "ITU-T Recommendation Y.4900/L.1600: Overview of key performance indicators in smart sustainable cities," International Telecommunication Union, 2016. [Online]. Available: https://www.itu.int/rec/T-REC-L.1600-201606-I

[84] "Environmental Engineering (EE); Assessment of mobile network energy efficiency," ETSI, Tech. Rep. ES 203 228 V1.1.7, November 2016. [Online]. Available: https://www.etsi.org/deliver/etsi_es/203200_203299/203228/01.01.07_60/es_203228v01107p.pdf

[85] *Environmental Testing – Part 1: General and guidance*, International Electrotechnical Commission Std. IEC 60 068-1, 2017. [Online]. Available: https://webstore.iec.ch/publication/6344

[86] A. Kolackova, S. Sevgican, M. Fatih Ulu, J. Sadreddini, P. Masek, J. Hosek, J. Jerabek, and T. Tugcu, "Exploring Potential of ML-Aided Mobile Traffic Prediction for Energy-Efficient Optimization of Network Resources Using Real World Dataset," *IEEE Access*, vol. 12, pp. 93 606–93 622, 2024.

[87] A. Kolackova, P. Viet Anh, E. Younesian, J. Hosek, J. Jerabek, and N. Khatib, "Enhancing Energy Efficiency in BBUs: Predictive Analysis with Long-Term and Granular Data," in *Proceedings of the 9th International Conference on Advanced Engineering - Theory and Applications*, Nov. 2024.

[88] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[89] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.

[90] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[91] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," 2014. [Online]. Available: https://arxiv.org/abs/1406.1078

[92] S. J. Taylor and B. Letham, "Forecasting at Scale," *The American Statistician*, vol. 72, no. 1, pp. 37–45, January 2018. [Online]. Available: https://ideas.repec.org/a/taf/amstat/v72y2018i1p37-45.html

[93] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4580–4584.

[94] J. Bergstra, B. Komer, C. Eliasmith, D. Yamins, and D. D. Cox, "Hyperopt: a python library for model selection and hyperparameter optimization," *Computational Science & Discovery*, vol. 8, no. 1, p. 014008, 2015.

[95] C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance," *Climate Research*, vol. 30, no. 1, pp. 79–82, 2005.

[96] J. Varun, E. Tejas, and K. K. T G, *Performance Analysis of Machine Learning Algorithms Over a Network Traffic*, 03 2021, pp. 1–10.

[97] A. De Myttenaere, B. Golden, B. Le Grand, and F. Rossi, "Mean absolute percentage error for regression models," *Neurocomputing*, vol. 192, pp. 38–48, 2016.

[98] S. K. G. Peesapati, M. Olsson, M. Masoudi, S. Andersson, and C. Cavdar, "An analytical energy performance evaluation methodology for 5g base stations," in *2021 17th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2021, pp. 169–174.

[99] M. Sewak, "Deep q network (dqn), double dqn, and dueling dqn: A step towards general artificial intelligence," in *Deep reinforcement learning: frontiers of artificial intelligence*. Springer, 2019, pp. 95–108.

[100] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1995–2003.

[101] H. van Hasselt, A. Guez, and D. Silver, "Deep Reinforcement Learning with Double Q-learning," 2015. [Online]. Available: https://arxiv.org/abs/1509.06461

[102] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*. PmLR, 2016, pp. 1928–1937.

[103] N. Churamani, S. Cruz, S. Griffiths, and P. Barros, "Persistent Rule-based Interactive Reinforcement Learning," *Neural Computing and Applications*, vol. 13, pp. 1013–1028, 2023.

[104] T. Könighofer, F. Lorber, N. Jansen, and R. Bloem, "Shield Synthesis for Reinforcement Learning," in *Proceedings of the International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA 2020)*, ser. Lecture Notes in Computer Science (LNCS), vol. 12476. Springer, 2020, pp. 290–306.

[105] W. C. Dabney, "Adaptive step-sizes for reinforcement learning," 2014.

[106] R. Glatt, F. L. da Silva, R. A. C. Bianchi, and A. H. R. Costa, "DECAF: Deep Case-based Policy Inference for Knowledge Transfer in Reinforcement Learning," *Expert Systems with Applications*, vol. 156, p. 113420, 2020.

[107] H. Mao, Y. Chen, M. Jaeger, T. D. Nielsen, and B. Nielsen, "Learning Deterministic Probabilistic Automata from a Model Checking Perspective," *Machine Learning*, vol. 105, no. 2, pp. 255–299, 2020.

[108] E. ETSI, "202 706-1," *Environmental Engineering (EE) Metrics and Measurement Method for Energy Efficiency of Wireless Access Network Equipment, European Telecommunications Standards Institute, Part 1: Power consumption - static measurement method*, 2022.

[109] 3rd Generation Partnership Project (3GPP), "Study on Central Unit (CU) - Distributed Unit (DU) lower layer split for NR," 3GPP, Technical Report TR 38.816, 2018. [Online]. Available: https://www.3gpp.org/DynaReport/38816.htm

[110] NGMN Alliance, "ITU-R Framework for IMT-2030: Review and Future Direction," Tech. Rep., February 2024. [Online]. Available: https://www.ngmn.org/wp-content/uploads/ITU-R_FRAMEWORK_FOR_IMT-2030.pdf

[111] Ericsson, "Ericsson Mobility Report: Explore the latest trends, forecasts and analysis from across the telecom industry," 2024. [Online]. Available: https://www.ericsson.com/en/reports-and-papers/mobility-report

[112] 3rd Generation Partnership Project (3GPP), "Management and orchestration; 5G end to end Key Performance Indicators (KPI)," 3GPP, Technical Specification TS 28.554, 2023. [Online]. Available: https://www.3gpp.org/DynaReport/28554.htm

[113] Cecilia Andersson, Jonas Bengtsson, Greger Byström,Pål Frenger, Ylva Jading and My Nordenström, "Improving energy performance in 5G networks and beyond," Ericsson, Tech. Rep., 2022, accessed: 2024-12-17. [Online]. Available: https://www.ericsson.com/en/reports-and-papers/ericsson-technology-review/articles/improving-energy-performance-in-5g-networks-and-beyond

Selected Publications by Author

[114] A. Kolackova, P. Viet Anh, J. Jerabek, S. Andreev, and J. Hosek, "ML-Driven Energy Savings for Cellular Baseband Units via Traffic Prediction." In Submission process to *IEEE Open Journal of the Communications Society*, Mar. 2025.

[86] A. Kolackova, S. Sevgican, M. Fatih Ulu, J. Sadreddini, P. Masek, J. Hosek, J. Jerabek, and T. Tugcu, "Exploring Potential of ML-Aided Mobile Traffic Prediction for Energy-Efficient Optimization of Network Resources Using Real World Dataset," *IEEE Access*, vol. 12, pp. 93 606–93 622, 2024.

[25] S. Saafi, J. Hosek, and A. Kolackova, "Enabling Next-Generation Public Safety Operations with Mission-Critical Networks and Wearable Applications," *Sensors*, vol. 21, no. 17, 2021. [Online]. Available: `https://www.mdpi.com/1424-8220/21/17/5790`

[29] A. Kolackova, J. Hosek, J. Jerabek, and P. Masek, "Experimental Verification of Segment Routing Methods in 5G+ Mobile Transport Networks," in *2021 13th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 2021, pp. 38–43.

[87] A. Kolackova, P. Viet Anh, E. Younesian, J. Hosek, J. Jerabek, and N. Khatib, "Enhancing Energy Efficiency in BBUs: Predictive Analysis with Long-Term and Granular Data," in *Proceedings of the 9th International Conference on Advanced Engineering - Theory and Applications*, Nov. 2024.

[24] S. Saafi, J. Hosek, and A. Kolackova, "Cellular-enabled Wearables in Public Safety Networks: State of the Art and Performance Evaluation," in *2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 2020, pp. 201–207.

[28] A. Kolackova, S. Saafi, P. Masek, J. Hosek, and J. Jerabek, "Performance Evaluation of Carrier Aggregation in LTE-A Pro Mobile Systems," in *2020 43rd International Conference on Telecommunications and Signal Processing (TSP)*, 2020, pp. 627–632.

[69] A. Kolackova, J. Jerabek, and J. Hosek, "Integrating Real-Environment 4G and 5G Mobile Data Patterns into Network Simulations for Delay Analysis," *Proceedings II of the 29th Conference STUDENT EEICT 2023: Selected papers.*, 2023. [Online]. Available: `https://api.semanticscholar.org/CorpusID:260039628`

[27] A. Kolackova and J. Jerabek, "Strategic Capacity Planning and Optimization in Communication Networks: A Case Study," *Proceedings II of the 30st Conference STUDENT EEICT* 2024: Selected papers., vol. 237, pp. 166–170, 2024. [Online]. Available: `https://doi.org/10.13164/eeict.2024.166`

Symbols and Abbreviations

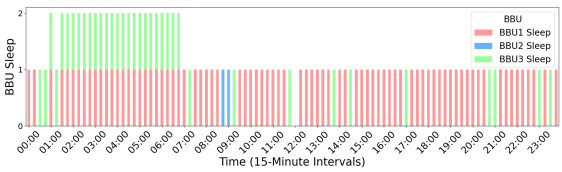
5G Fifth Generation of Wireless Networks
5GC 5G Core Network
6G Sixth Generation of Wireless Networks
A2C Advantage Actor-Critic
AI Artificial Intelligence
ANN Artificial Neural Network
AMF Access and Mobility Management Function
BBU Baseband Unit
CU Control Unit
DU Distributed Unit
CP Control Plane
DP Data Plane
C-RAN Cloud Radio Access Network
CNN Convolutional Neural Network
CNN-LSTM Convolutional Neural Network-Long Short-Term Memory
DDDQN Dueling Double Deep Q-Network
DDPG Deep Deterministic Policy Gradient
DQN Deep Q-Network
DRL Deep Reinforcement Learning
DT Decision Tree
eMBB Enhanced Mobile Broadband
GAN Generative Adversarial Network
GRU Gated Recurrent Unit
HA-DRL Heuristic-assisted Deep Reinforcement Learning
IoT Internet of Things
KDE Kernel Density Estimation
KPI Key Performance Indicator
LLM Large Language Model
LSTM Long Short-Term Memory
LPWAN Low-Power Wide-Area Network
LoRa Long Range
MAE Mean absolute error
MAPE Mean absolute percentage error
MIMO Multiple Input Multiple Output
ML Machine Learning
MSE Mean Squared Error
mMTC Massive Machine-Type Communications
NFV Network Functions Virtualization
NB-IoT Narrowband Internet of Things
NR New Radio
O-RAN Open Radio Access Network

PESBiU Predictive Energy Saver for Baseband Units
PM Performance Management
QoS Quality of Service
RAN Radio Access Network
RNN Recurrent Neural Network
RL Reinforcement Learning
RLHF Reinforcement Learning from Human Feedback
RRU Remote Radio Unit
SBA Service-Based Architecture
SDN Software Defined Networking
SMF Session Management Function
SVM Support Vector Machine
UPF User Plane Function
URLLC Ultra-Reliable Low Latency Communication
UE User Equipment
VAE Variational Autoencoder
XAI Explainable Artificial Intelligence

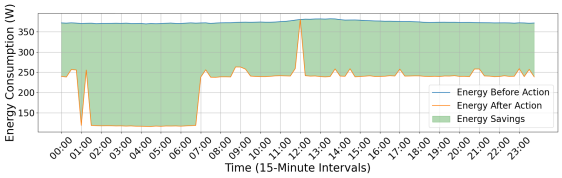
A PESBiU 2.0 Decisions for BBU Sleep and Energy Consumption for Tested Algorithms

List of figures are as follows:

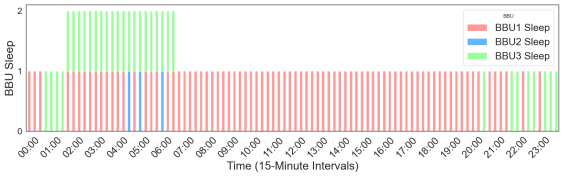
- Fig. A.1: Site **Beta** - Distribution of BBUs entering sleep mode at defined intervals for one workday, along with their energy consumption and the resulting savings achieved using all tested algorithms.
- Fig. A.2: Site **Delta** - Distribution of BBUs entering sleep mode at defined intervals for one workday, along with their energy consumption and the resulting savings achieved using all tested algorithms.
- Fig. A.3: Site **Gama** - Distribution of BBUs entering sleep mode at defined intervals for one workday, along with their energy consumption and the resulting savings achieved using all tested algorithms.



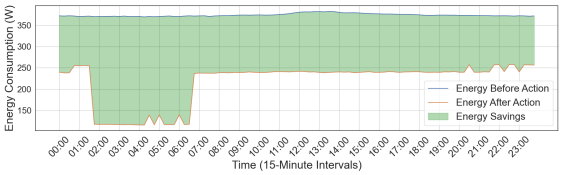
(a) Static algorithm - BBUs put to sleep



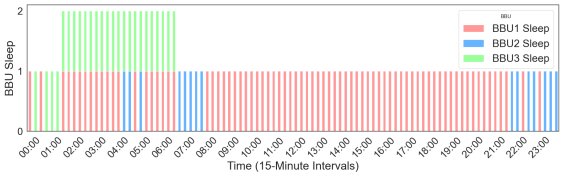
(b) Static algorithm - Energy Consumption



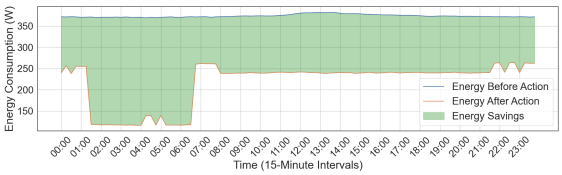
(c) DDDQN - BBUs put to sleep



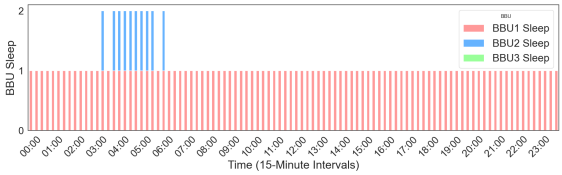
(d) DDDQN - Energy Consumption



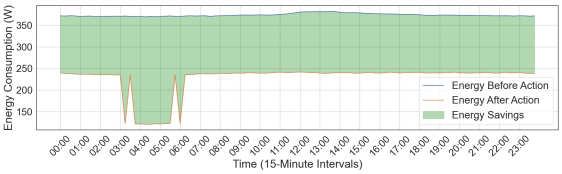
(e) Hybrid - BBUs put to sleep



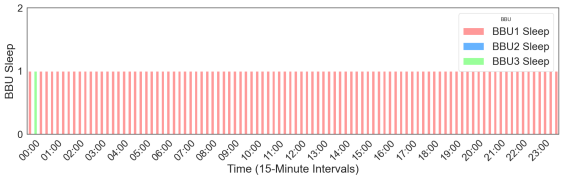
(f) Hybrid - Energy Consumption



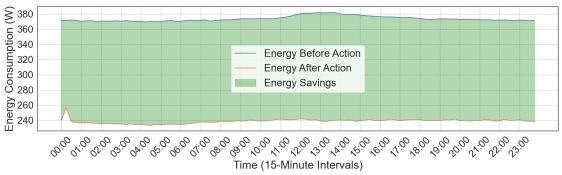
(g) DQN - BBUs put to sleep



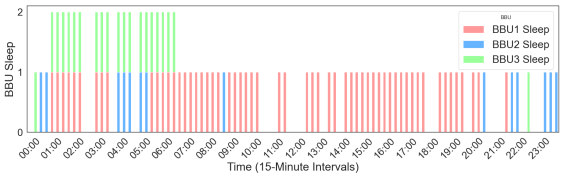
(h) DQN - Energy Consumption



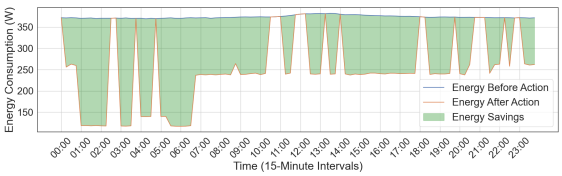
(i) A2C - BBUs put to sleep



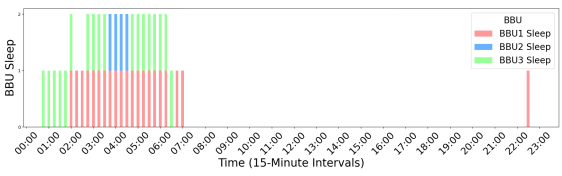
(j) A2C - Energy Consumption



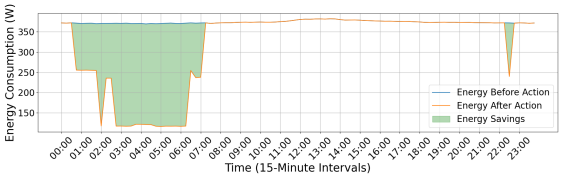
(k) DDDQN+fallback - BBUs put to sleep



(l) DDDQN+fallback - Energy Consu.

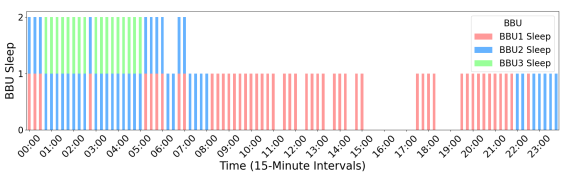


(m) Threshold - BBUs put to sleep

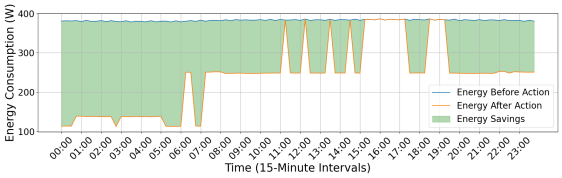


(k) Threshold - Energy Consumption

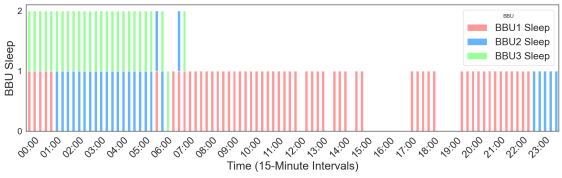
Fig. A.1: Site **Beta** - BBUs put to sleep by all algorithms along with resulting savings



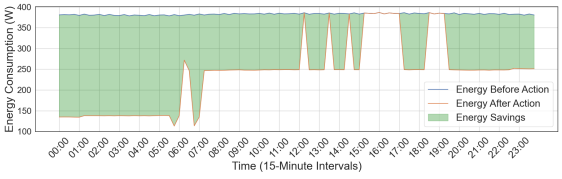
(a) Static algorithm - BBUs put to sleep



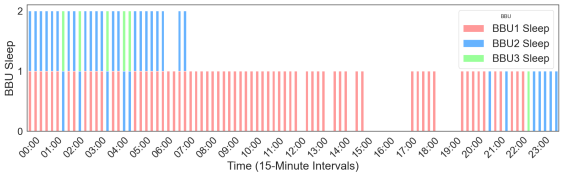
(b) Static algorithm - Energy Consumption



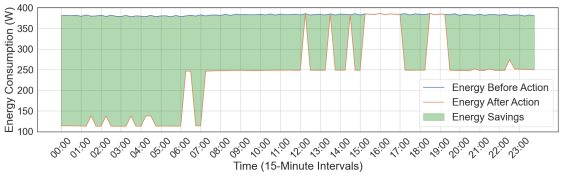
(c) DDDQN - BBUs put to sleep



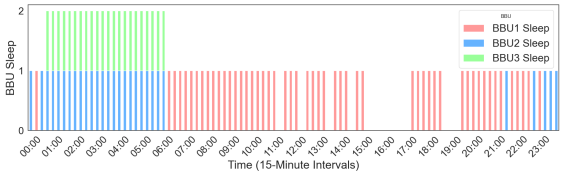
(d) DDDQN - Energy Consumption



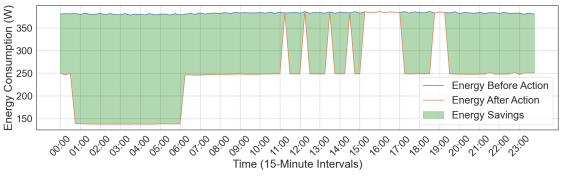
(e) Hybrid - BBUs put to sleep



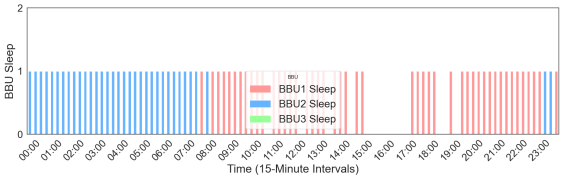
(f) Hybrid - Energy Consumption



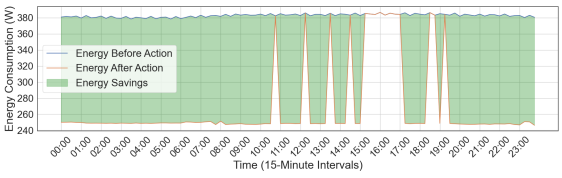
(g) DQN - BBUs put to sleep



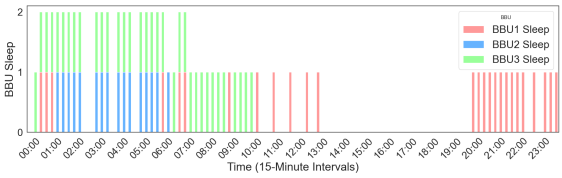
(h) DQN - Energy Consumption



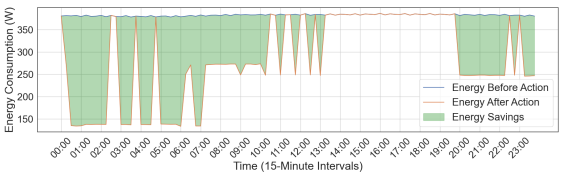
(i) A2C - BBUs put to sleep



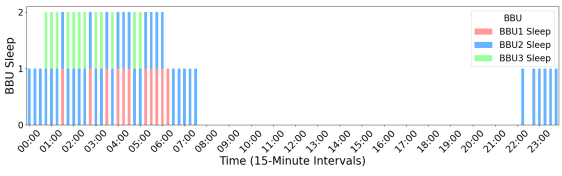
(j) A2C - Energy Consumption



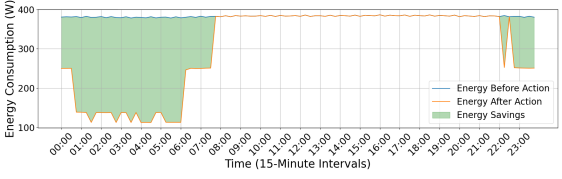
(k) DDDQN+fallback - BBUs put to sleep



(l) DDDQN+fallback - Energy Consu.

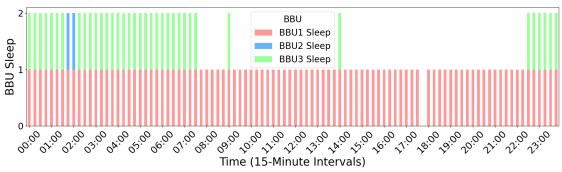


(m) Threshold - BBUs put to sleep

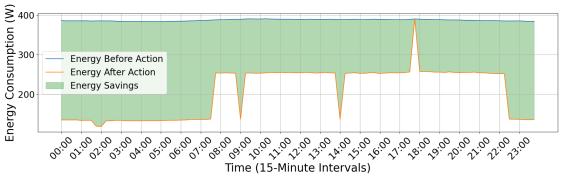


(k) Threshold - Energy Consumption

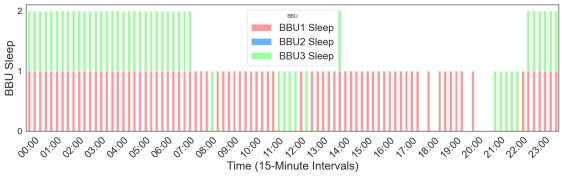
Fig. A.2: Site **Delta** - BBUs put to sleep by all algorithms along with resulting savings



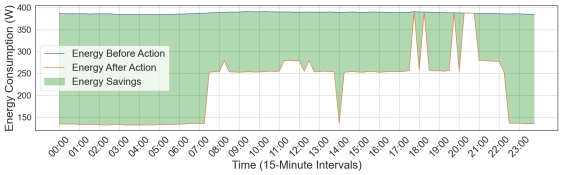
(a) Static algorithm - BBUs put to sleep



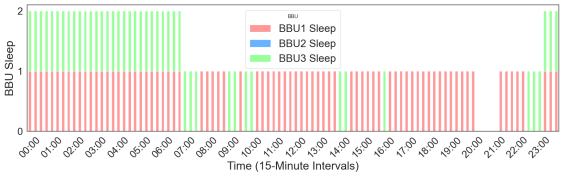
(b) Static algorithm - Energy Consumption



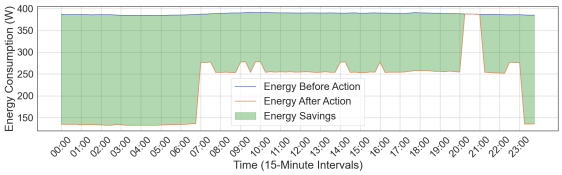
(c) DDDQN - BBUs put to sleep



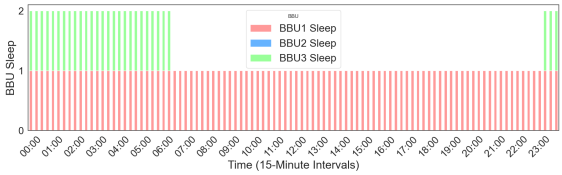
(d) DDDQN - Energy Consumption



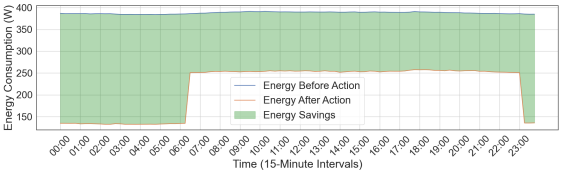
(e) Hybrid - BBUs put to sleep



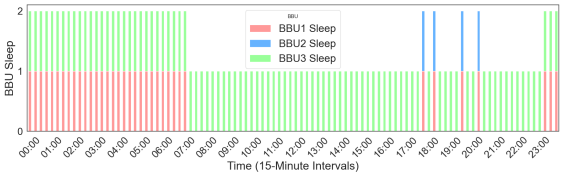
(f) Hybrid - Energy Consumption



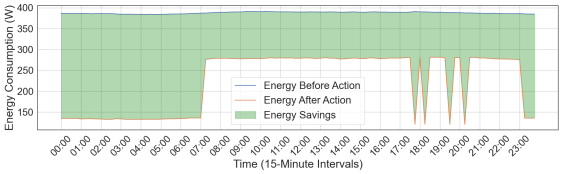
(g) DQN - BBUs put to sleep



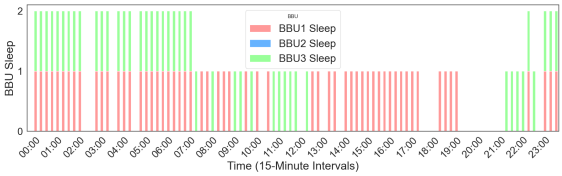
(h) DQN - Energy Consumption



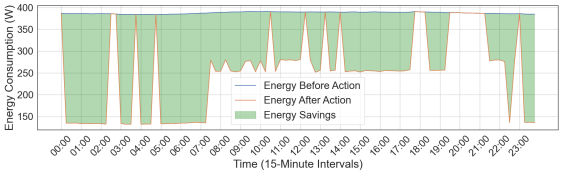
(i) A2C - BBUs put to sleep



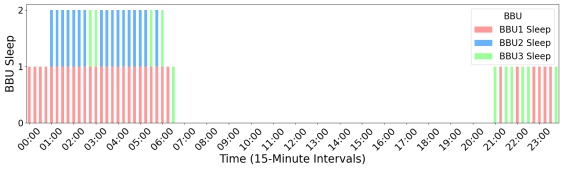
(j) A2C - Energy Consumption



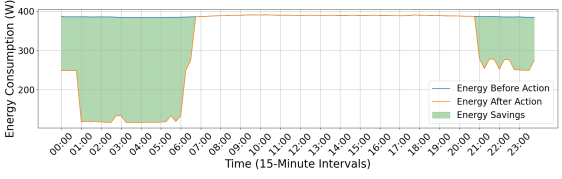
(k) DDDQN+fallback - BBUs put to sleep



(l) DDDQN+fallback - Energy Consu.



(m) Threshold - BBUs put to sleep



(k) Threshold - Energy Consumption

Fig. A.3: Site **Gama** - BBUs put to sleep by all algorithms along with resulting savings