Reinforcement Learning For Adaptive Distribution Network Reconfiguration

by

Nastaran Gholizadeh

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Software Engineering and Intelligent Systems

Department of Electrical and Computer Engineering
University of Alberta

© Nastaran Gholizadeh, 2023

Abstract

The increasing demand for electricity driven by the widespread adoption of electric vehicles necessitates effective distribution network reconfiguration methods. However, existing distribution network reconfiguration approaches often rely on precise network parameters, leading to scalability and optimality challenges. To overcome these issues, this thesis proposes a data-driven reinforcement learning-based algorithm for distribution network reconfiguration which is divided into three parts.

In the first part, five reinforcement learning algorithms, including deep Q-learning, dueling deep Q-learning, deep Q-learning with prioritized experience replay, soft actor-critic, and proximal policy optimization, are compared for the distribution network reconfiguration problem in 33- and 136-node test systems. Additionally, a new deep Q-learning-based action sampling method is introduced to reduce the action space size and optimize system loss reduction.

In the second stage of this research, an innovative action-space sampling method is developed which utilizes a graph theory-based algorithm named Yamada-Kataoka-Watanabe to find all minimum spanning trees in the network structure as an undirected graph. Subsequently, powerflow analysis is conducted for all these spanning tree structures to rank them from the most optimal to the least in terms of system loss. Notably, this new sampling method stands out from the previous deep Q-learning-based approach as it offers greater versatility and can be seamlessly applied to any test system. This method is applied to the 33-, 119-, and 136-node test systems. Comparative analysis against conventional methods demonstrated the effectiveness, scalability, and efficiency of the proposed method in reducing system losses and managing electricity demand effectively.

While reinforcement learning methods offer fast decision-making capabilities, the lack of transparency in their decision processes hinders their application in critical decision-making scenarios. In particular, distribution network reconfiguration involves altering switch states, which can significantly impact switch lifespan, requiring careful consideration. To address this transparency issue, in the third part of this study, a novel approach is introduced that employs an explainer neural network to analyze and interpret reinforcement learning-based decisions in distribution network reconfiguration. The explainer network is trained using the reinforcement learning agent's decisions, considering active and reactive power of the buses as inputs and generating line states as outputs. Later, the utilization of attribution methods enabled a deeper understanding of the relationship between inputs and outputs, offering valuable insights into the agent's decision-making process.

Overall, this thesis presents a comprehensive and innovative approach to distribution network reconfiguration leveraging data-driven reinforcement learning algorithms for decision making, graph theory-based action sampling for improving the optimality of decisions, and an explainer neural network for decision interpretation.

Preface

The research presented in this thesis was conducted at the ENergy digiTizAtIon Lab (ENTAIL) under the guidance of Dr. Petr Musilek. This thesis is predicated on three principal publications:

1. N. Gholizadeh, N. Kazemi and P. Musilek, "A Comparative Study of Reinforcement Learning Algorithms for Distribution Network Reconfiguration With Deep Q-Learning-Based Action Sampling," IEEE Access, vol. 11, pp. 13714-13723, 2023.

2. N. Gholizadeh, and P. Musilek, "A Generalised Deep Reinforcement Learning Model for Distribution Network Reconfiguration with Powerflow-based Action Space Sampling," Journal of Modern Power Systems and Clean Energy, 2023, under review.

3. N. Gholizadeh, and P. Musilek, "Explainable Reinforcement Learning for Distribution Network Reconfiguration," Applied Energy, 2023, under review.

During my enriching journey through graduate studies, I dedicated myself to exploring various projects that revolved around the fascinating realm of machine learning applications in power systems. In addition to the content presented in this thesis, there were noteworthy outcomes from these projects as follows that deserve recognition:

4. N. Gholizadeh and P. Musilek, "Distributed Learning Applications in Power Systems: A Review of Methods, Gaps, and Challenges," Energies, vol. 14, no.

12, p. 3654, Jun. 2021.

5. N. Gholizadeh, P. Musilek, “Federated Learning with Hyperparameter-Based Clustering for Electrical Load Forecasting,” Internet of Things, Volume 17, 2022.

Chapter 2 presents our research detailed in the IEEE Access paper [26]. In this chapter, I address the distribution network reconfiguration problem, formulating it as a Markov decision process. Furthermore, I apply a range of cutting-edge reinforcement learning algorithms to identify the most effective approach for solving this complex problem. Through rigorous experimentation and analysis, we gain valuable insights into the performance of these algorithms and their suitability for addressing the distribution network reconfiguration problem.

Chapter 3 is based on the paper [27] submitted to the Journal of Modern Power Systems and Clean Energy and is currently under review. Since the selection of the action space in [26] remained largely random, we developed an improved action space sampling method in this paper in order to make the proposed reinforcement learning-based approach for network reconfiguration applicable to any test system.

Finally, chapter 4 draws from our third paper [29] which is submitted to Applied Energy and addresses the transparency issue in reinforcement learning agent’s decision making for distribution network reconfiguration by designing an explanation approach. In all the mentioned three papers, I took charge of designing the research methodology, writing the code, analyzing the results, and writing the paper. Dr. Petr Musilek, as the supervisory author, offered significant help with concept formation and manuscript editing.

Acknowledgements

I am immensely grateful for the unwavering support and guidance provided by my esteemed supervisor Dr. Petr Musilek throughout this research. His invaluable expertise and encouragement has been instrumental in shaping the success of this project. I would also like to thank my family members for their emotional and mental support and their encouragement during my PhD studies and my stay in Canada. I am deeply appreciative of their unwavering faith in me, which has consistently fueled my motivation. I am genuinely thankful for their existence in my life. Finally, I would like to thank my friends and colleagues for their emotional and mental support throughout this journey and the members of the ENTAIL group for their suggestions to enhance this project.

Table of Contents

List of Tables

List of Figures

Abbreviations

AACO	Adaptive ant colony optimization.
BCE	Binary cross-entropy.
BCSAC	Batch-constrained soft actor-critic.
BDDQN	Branching double deep Q-learning.
CDBAS	Chaos disturbed beetle antennae search.
CNN	Deep convolutional neural network.
CSA	Cuckoo search algorithm.
DDPG	Deep deterministic policy gradient.
DDQN	Double deep Q-learning.
DG	Distributed generation.
DMADQN	Decomposed multi-agent deep Q-network.
DNR	Distribution network reconfiguration.
DQN	Deep Q-learning.
GA	Genetic algorithm.
HHO	Harris Hawks optimization.
LDBAS	Levy flight and chaos disturbed beetle antennae search.
LSTM	Long short-term memory.
MA-SAC	Multi-agent soft actor-critic.

MDP Markov decision process.

MG Microgrid.

MIQP Mixed-integer quadratic programming.

MP Mathematical programming.

MPSAC Multi-policy soft actor-critic.

PD Probability distribution.

PPO Proximal policy optimization.

PSO Particle swarm optimization.

RL Reinforcement learning.

SAC Soft actor-critic.

SOE Switch opening and exchange.

TOPSIS Technique for order preference by similarity to an ideal solution.

TRPO Trust region policy optimization.

WCA Water cycle algorithm.

Chapter 1

Introduction

The ever-increasing demand for electricity, coupled with the complexities of power systems and the cost of expanding existing infrastructure, has driven power system researchers to explore innovative approaches for optimizing power system operation. Among the diverse techniques employed, distribution network reconfiguration (DNR) has emerged as a critical method to minimize system losses and maintain controlled voltage levels. However, achieving an efficient and effective DNR process is a challenging task, demanding advanced tools that can handle the complexities of power systems.

Recently, the incorporation of machine learning methods has revolutionized the power systems domain, offering new avenues for enhancing operational efficiency and reliability. Various power system applications have effectively utilized machine learning techniques with notable success. These include but are not limited to power system security [114], anomaly detection [16], demand and generation forecasting [83], energy storage control [35], unit commitment [68], reliability evaluation [55], voltage stability prediction [87], and optimal power flow [19].

The focus of this research is on the utilization of reinforcement learning (RL) for DNR optimization. As a branch of machine learning built upon the principles of action and response, RL's ability to learn and adapt over time makes it a compelling candidate for automating the DNR process in power systems. By leveraging RL, the proposed method aims to minimize network losses via DNR while ensuring compliance with critical constraints such as voltage levels, radiality, and demand balance. One of the standout features of this approach is its flexibility, allowing it to be seamlessly applied to various distribution networks.

In the following sections, we delve into the details of our research focusing on the RL-based DNR optimization. Through this innovative application of machine learning, we aim to pave the way for more efficient and sustainable power grid management in the face of mounting electricity demand and evolving grid complexities.

1.1 Thesis Motivations and Objectives

In the field of DNR optimization, the use of RL remains relatively limited, with existing studies lacking comprehensive analysis, scalability, and flexibility. This research aims to address these limitations by developing a flexible RL framework for DNR, leveraging RL's learning capabilities to expedite reconfiguration execution and avoid the DNR's NP-hard optimization problem without using any simplifications for radiality and AC power flow constraints. The vast switching action space in power systems presents a challenge, as many actions may be impractical in real distribution systems. Therefore, a key objective is to devise an algorithm capable of identifying feasible actions within the network. Furthermore, given the inherent opacity of RL algorithms, this study explores the development of an explanation model to enhance understanding and trust in RL-based systems. By achieving these objectives, this research seeks to advance the application of RL in DNR and potentially contribute to improved efficiency and reliability in power system operations. The motivations, objectives, and findings derived from our research studies are summarized as follows:

Research Study 1: "A Comparative Study of Reinforcement Learning Algorithms for Distribution Network Reconfiguration with Deep Q-learning-based Action Sampling" [26] (Chapter 2)

Motivation: Current research employing RL for DNR optimization is limited and often lacks comprehensive analysis and flexibility. Moreover, practical challenges, such as the vastness of the switching action space and the impracticality of many actions in real-world distribution systems, remain inadequately addressed.

Objective: The main objective of this study is to devise an algorithm capable of identifying the feasible actions within the distribution network and perform a comparative study of various RL algorithms to find the most suitable ones for the DNR problem.

Findings: We have developed a method based on sequential opening of loops in the network. This technique successfully identified 14,401 feasible actions for the 33-node system and 56,996 feasible actions for the 136-node system. However, due to the large size of the feasible action space, we further introduced an advanced action space sampling method based on deep Q-learning. This involved training a separate agent using the deep Q-learning algorithm on a larger action space to effectively select the DNR agent's action space. However, since the action space selection for this agent was random, the selection of the action space for the DNR agent remained largely random.

Research Study 2: "A Generalised Deep Reinforcement Learning Model for Distribution Network Reconfiguration with Powerflow-based Action Space Sampling" [27] (Chapter 3)

Motivation: The feasible action space size in a DNR problem is still very large. Our research showed that the feasible action space consisted of 50,751 actions for the 33-node system and over 3,500,000 actions for the 119- and 136-node test systems. Therefore, once feasible actions are determined, the next step involves extracting the most optimal action space for RL application.

Objective: The primary objective of this research is to develop a new action-space sampling method for the RL algorithms and validate the effectiveness of this method by comparing the optimality of the solutions obtained by the RL agent trained on this action space against other math- and algorithm-based approaches in literature.

Findings: We adapted a graph analysis-based algorithm in this study to find more feasible actions for our test systems. This method was based on the Yamada-Kataoka-Watanabe algorithm and found 50,751 actions in the 33-node system and over 3,500,000 actions in the 119- and 136-node test systems. To choose the action space for the DNR agent, we proposed a new power flow-based method to rank the actions from the most optimal to the least. Applying this method on the 33-, 119-, and 136-bus test systems revealed that the DNR agent trained on this action space was able to find solutions within 0.3% optimality gap of the best solution presented in the literature for the 33-node system and within 10.58% and 1.95% optimality gap for the 119- and 136-node test systems, respectively.

Research Study 3: "Explainable Reinforcement Learning for Distribution Network Reconfiguration" [29] **(Chapter 4)**

Motivation: The motivation behind the third part of this research lies in the inherent opacity of RL algorithms which leads to skepticism and mistrust among users and poses challenges in ensuring the reliability, fairness, and acceptance of RL-based systems in crucial domains such as DNR.

Objective: The third objective of this study is to develop an explanation model for the trained RL agents to understand the logic behind their decision making in DNR.

Findings: We developed a novel approach which involved training an explainer network based on the DNR agent's actions as outputs and system state as input and used three attribution methods to explain the relationship between the inputs and the outputs. The results showed that the reactive power of bus 30 had a major impact on the reconfiguration of the 33-bus system. Additionally, the active power of bus 105 was crucial in determining the configuration of the 136-bus system.

1.2 Comparing RL Algorithms for DNR

DNR is a critical task in power system operation that involves altering the topology of the distribution network by changing the status of switches and reconfiguring the network layout. The objective of DNR is to minimize power losses, improve voltage stability, and ensure load balancing while adhering to various constraints such as radiality of the network. However, DNR optimization is known to be an NP-hard mixed-integer nonlinear problem due to the non-linearity of AC power flow and the radiality constraint [117]. Therefore, this research employs RL for DNR to circumvent the challenges associated with addressing an NP-hard mixed-integer nonlinear problem.

In our first study [26], we concentrated on formulating the DNR problem within the framework of a Markov decision process. Extensive experimentation was conducted to determine the optimal representation of the actions and the content of the system state. Initially, we tested a binary action space consisting of opening or closing single lines. However, this approach often led to the creation of loops or network disconnections, resulting in significantly negative rewards for the agent. Consequently, we had to revise our action representation to a vector format, which encompassed commands for the open/close state of all lines. This adjustment also prevented the agent's neural network from growing excessively as the system's size increased.

Furthermore, we conducted an array of experiments to determine the most suitable elements for inclusion in the system state. Initially, we considered solely incorporating the active power of the buses as the system state. However, our experiments demonstrated that including the information about reactive power and switch states led to improved performance of our RL algorithm. In addition, we carefully fine-tuned the coefficients in our reward function through a series of trial and errors.

After formulating the system as a Markov decision process, the primary challenge was dealing with the vast action space. In its current binary format, a system with 33 lines yielded 2^{33} possible actions. However, a significant portion of these actions led to system loops or disconnections, making them impractical for real-world application within a distribution system. Consequently, in this study, we also developed an algorithm to find only the feasible actions. This algorithm centered around the sequential opening of switches within the network until no loops remained within it.

It was found that the solutions produced by the RL agent did not attain the level of optimality observed in existing literature. The optimality gap was notably substantial. Consequently, we endeavored to enhance our approach by isolating the most optimal actions from the feasible action space. For this purpose, we initiated a randomized selection of n actions, and subsequently, employed deep Q-learning (DQN) and waited until convergence. We selected our final action space by only keeping the actions with the highest Q-values. This method effectively improved the optimality of solutions achieved by the RL agent. Nonetheless, it's worth noting that the agent's performance remained highly dependent on the initial random selection of the action space.

In addition to the previously mentioned research endeavors in this study, we have also provided a thorough comparison of different RL algorithms for DNR, a topic that has not been adequately addressed in prior literature.

1.3 Action Space Sampling for DNR

Our second study primarily centered on identifying the most optimal actions for RL in the context of DNR. This study diverged significantly from our previous research in that our main objective was to enhance the optimality of solutions generated by RL agents in DNR, making them competitive with solutions proposed by advanced methods in the existing literature. Furthermore, as our previous study largely relied on random selection for the final action space, this investigation aimed to rectify this deficiency.

To achieve this objective, our study employed a graph theory-based approach. We conceptualized our network as an undirected graph, where buses served as nodes, and power lines represented the edges. We adapted the Yamada-Kataoka-Watanabe graph theory technique to identify all the minimum spanning trees within the network. This method significantly expanded the number of feasible actions for our networks when compared to the previous approach of sequentially opening switches, as outlined in our prior study. Subsequently, we devised a power flow-based ranking system for these actions, allowing us to select only the most optimal actions for integration into our RL algorithm.

1.4 Explainable RL for DNR

While RL has shown remarkable success in solving complex optimization problems, its inherent opacity raises significant concerns regarding the interpretability of decision-making processes. The difficulty in understanding how and why RL models arrive at their decisions has led to skepticism and mistrust among users. The lack of transparency in RL algorithms creates a black box effect, making it challenging to comprehend the rationale behind the actions taken by the RL agents. As a result, the reliability and fairness of RL-based systems often come under scrutiny. To address this critical issue, researchers have been focusing on the development of explainable RL methods.

Explainable RL aims to elucidate the decision-making logic of RL agents, making their actions more interpretable and understandable to humans. By providing insights into the internal workings of RL models, explainable RL methods seek to enhance trustworthiness, acceptance, and broader adoption of RL in crucial domains such as autonomous systems, healthcare, finance, and more. Very few studies have focused on developing models to explain the rationale behind the RL agent's decision making process. These studies have utilized statistical analysis [8], Shapley additive explanations [113], and model trees [30] for this purpose.

Despite the progress in applying RL to DNR scenarios, a crucial research gap remains: the lack of comprehensive approaches to explain the underlying logic behind RL agents' decision-making in DNR. Addressing this gap is essential to gain a deeper understanding of how RL-based solutions can be leveraged to improve the reliability and efficiency of power distribution networks. To fill this void, this study proposes a novel methodology that involves generating a dataset based on the RL agent's decisions and training an explainer network to explain the decision-making process.

Attribution methods are later applied to this explainer network to find the relationship between its inputs and outputs, facilitating a better comprehension of RL agents' behavior in DNR scenarios. It is important to highlight that the main benefit of explaining the RL agent's actions is increasing humans' trust in suggested solutions and providing businesses with an AI-powered system that offers solid evidence to back up its decisions and outcomes.

1.5 Thesis Outline

The remaining sections of this thesis are organized as follows. Chapter 2 focuses on the development of an algorithm aimed at identifying the feasible action space for RL. We evaluate several RL algorithms and determine the most suitable ones for our purposes. This chapter is based on our study in [26]. In Chapter 3, we employ a novel graph analysis-based algorithm to discover the feasible action space in various test systems. Then, we utilize a power flow-based algorithm to extract the most optimal action space for the DNR agent. This chapter draws upon the research conducted in [27]. In Chapter 4, we focus on creating an explanation method that explains the decision-making process of the trained DNR agent as presented in [29]. The final chapter, Chapter 5, presents a comprehensive summary of our research findings and contributions. Additionally, we provide valuable insights into potential avenues for future research.

Chapter 2

A Comparative Study of Reinforcement Learning Algorithms for Distribution Network Reconfiguration with Deep Q-learning-based Action Sampling

2.1 Abstract

Distribution network reconfiguration (DNR) is one of the most important methods to cope with the increasing electricity demand due to the massive integration of electric vehicles. Most existing DNR methods rely on accurate network parameters and lack scalability and optimality. This study uses model-free reinforcement learning algorithms for training agents to take the best DNR actions in a given distribution system. Five reinforcement algorithms are applied to the DNR problem in 33- and 136-node test systems and their performances are compared: deep Q-learning, dueling deep Q-learning, deep Q-learning with prioritized experience replay, soft actor-critic, and proximal policy optimization. In addition, a new deep Q-learning-based action sampling method is developed to reduce the size of the action space and optimize the loss reduction in the system. Finally, the developed algorithms are compared against the existing methods in literature.

2.2 Introduction

Large-scale penetration of electric vehicles and integration of renewable energy resources have increased the complexity of power distribution networks [28]. As a result, more advanced control and optimization techniques are required to keep the system within operational constraints, reduce losses, and supply demands [109]. Distribution network reconfiguration (DNR) is the process of connecting and disconnecting different distribution lines in a way that the system loss is minimized and the voltage level is maintained. The non-linearity of AC power flow and the network radiality constraint make DNR optimization an NP-hard, mixed-integer, nonlinear problem [117]. Therefore, heuristic optimization algorithms [77] and mathematical simplifications [3] are typically used for DNR optimization in the literature.

Majority of DNR studies are based on heuristic optimization algorithms. To this end, a new social beetle swarm optimization algorithm considering two social behaviors is developed in [14] to solve the multiobjective DNR problem which minimizes network loss, load balance index, and maximum voltage deviation. The same method is coupled with grey target decision-making in [13] to improve the process of selecting the best beetle and solve the problem of conflicting objectives. Chaos disturbed beetle antennae search (CDBAS) [98] and Levy flight and chaos disturbed beetle antennae search (LDBAS) [97] algorithms are combined with grey target decision-making technique to solve the DNR problem while minimizing the load balancing index, active power loss, and the maximum node voltage deviation. A modified multi-objective Bayesian learning-based evolutionary algorithm is proposed in [119] to balance the voltage stability and the absorption rate of wind energy which is then combined by a technique for order preference by similarity to an ideal solution (TOPSIS) to solve DNR. In [49], a θ-modified bat algorithm is developed to solve the DNR problem while the uncertainties associated with multiobjective DNR are handled via cloud theory constructed based on fuzzy theory combined with the concept of probability. However, to solve the

DNR problem, all of these studies utilize heuristic or meta-heuristic algorithms which are time-consuming, have high computational cost, and do not guarantee finding the optimal solution.

In a different set of studies, mathematical models are utilized to solve the DNR optimization [4]. For example, a distributionally robust chance-constrained DNR approach for a three-phase unbalanced distribution network is proposed in [120]. It optimizes the switching cost and the expected power supply cost from an upstream grid. Later, the model is reduced to a mixed-integer, linear programming problem. A risk-averse two-stage mixed-integer conic programming model is presented in [11] for grid reconfiguration where the seasonal reconfiguration decisions of microgrids (MG) are made in the first stage, and validated under stochastic islanding scenarios in the second stage. In [85], Benders decomposition framework is combined with mixed-integer quadratic programming (MIQP) to solve DNR and reduce voltage volatility. MIQP and mixed-integer linear programming are used in [39] for operation mode adjustment minimizing the active power loss and for service restoration maximizing restored loads. The main drawback of these methods is that they simplify the powerflow equations. Hence, the obtained solutions are neither accurate nor optimal.

There have been other studies which use algorithmic or fuzzy logic approaches. Switch opening and exchange (SOE) method is developed in [112] for multi-hour stochastic DNR which is based on sequential opening of switches until no loops remain in the network and modifying the status of obtained branches. In [65], a multiagent weight-based self reconfiguration algorithm with distributed generators is presented for load sharing and reducing congestion in lines. An adaptively tunable fuzzy logic controller is proposed in [105] for network reconfiguration during power system restoration. Finally, intra-day dynamic reconfiguration [60] is solved using time period reduction and decimal coding strategy to maximize the accommodation revenue of distributed generation (DG) units and to minimize the operation cost of distribution network. These methods can improve the quality of the solutions, but as the system size grows, the number of decision variables increases and they cannot guarantee optimality. In addition, their execution time becomes exponentially longer with the increase in system size.

Recently, machine learning has also been successfully applied for DNR. By combining original neural network algorithm with chaotic local search and quasi-oppositional-based learning approaches, a new algorithm is developed in [91] to simultaneously solve DG allocation and DNR optimization. However, artificial intelligence algorithms used for optimization purpose still face the problems of poor convergence and falling into local optima [97]. A deep neural network is developed in [118] to adaptively learn the reference joint probability distributions (PD) of DG outputs and loads from historical data. Later, three-phase unbalanced DNR is solved using a modified column-and-constraint generation method under the worst-case PD scenarios. In this study, neural network is only developed for forecasting the PD of DG outputs. In [43], a deep convolutional neural network (CNN) is developed for DNR. It learns the relationship between network topology and short-term voltage stability performance from historical data. Since the model is only trained on historical network topologies, it is unable to perform new reconfigurations that result in unperceived topologies. Hence, the optimality of the reconfiguration actions is not guaranteed. In [58], a hybrid data-driven and model-based DNR framework is proposed which uses long short-term memory (LSTM) network to learn the mapping mechanism between load distribution and optimal reconfiguration strategies. The model presented in [44] was similar except that it also considered the switching cost in addition to the system loss in the objective function. The main disadvantage of these two models is that the dimension of the neural network needed for training grows with the size of the network and the computational cost increases exponentially. In addition, they do not search for optimal reconfiguration strategies for different load values as they are only trained on a few reconfiguration topologies.

Most DNR algorithms discussed so far are model-based controllers. This means that they need accurate values of distribution network parameters which is difficult to obtain due to the expansive structure of distribution networks and seasonal weather changes [24]. On the other hand, the computational burden of these algorithms increases exponentially with the size of the network [75] which makes them impractical for real-time control. Reinforcement learning (RL) is an area of machine learning which uses various algorithms to train agents for taking best actions in an environment [42]. This method allows building a model-free control approach for DNR since the agent does not need to directly interact with the distribution network and it does not need to know the transition probablities between different state and action pairs. Only very few studies have used RL for DNR process. To this end, deep Q-learning and NoisyNet deep Q-learning network (DQN) with automatic exploration is applied to DNR in [52] and [95], respectively. However, in neither of these two studies, the scalability of the proposed method is demonstrated on larger test systems. Moreover, these studies compare the performance of only several DQN methods. DQN was also applied to the DNR problem in [73]. However, DQN performance was only compared with the brute-force search and genetic algorithms, but not with any other RL algorithm. Batch-constrained soft actor-critic RL algorithm is developed in [25]. It learns a control policy from a finite historical operational dataset without interacting with the distribution network. The main drawback of this approach is that the agent is only trained on historical operational dataset which means that it does not search for new and more optimal reconfiguration strategies.

This study develops an RL model for DNR that samples from a very large action space. Hence, it is not only based on historical operation. In addition, this study compares the performance of five popular RL algorithms for DNR including DQN, policy gradient methods, and actor-critic methods. Although there were studies that implemented DQN and soft actor-critic (SAC) for DNR, no previous study implemented dueling DQN, DQN with prioritized experience replay, and proximal policy optimization (PPO) for this purpose. In addition, no study presented a comprehensive comparison of these algorithms. The main contributions of the first part of this thesis are as follows:

- Proposal of a new action search algorithm to find the feasible action space for RL algorithms.
- Development of a new DQN-based action sampling method to reduce the size of the action space and improve the optimality of the obtained DNR solutions by RL.
- Implementation of five RL algorithms for DNR (including DQN, dueling DQN, DQN with prioritized experience replay, SAC, and PPO) and comparison of their performances.

2.3 Problem Formulation

This section covers the problem formulation for DNR as well as the RL algorithms.

2.3.1 Preliminaries

There are two main components in a standard RL structure: environment and agent. As shown in Fig. 2.1, the agent selects an action a_t according to the environment state s_t and receives the reward r_{t+1} and next state s_{t+1} from the environment. The main goal of RL algorithm is to train the agent to select the actions in the environment that maximize its rewards. The environment for RL is modeled as a Markov decision process (MDP). An MDP is denoted by $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \mathcal{T})$ and consists of a state space $\mathcal{S}$ (where, $s_t \in \mathcal{S}$), action space $\mathcal{A}$ ($a_t \in \mathcal{A}$), transition probability function $\mathcal{P}$, reward function $\mathcal{R}$ ($r_t \in \mathcal{R}$), discount factor $\gamma \in [0, 1)$, and a time horizon $\mathcal{T}$.

The agent finds the optimal control policy π for its environment by maximizing the expected discounted return $G = \sum_{t=0}^{\mathcal{T}} \gamma^t r_{t+1}$. By calculating the action-values using Bellman equation, the agent decides which action to select in a given state.

Bellman equation, starting from state s_t and taking action a_t while following policy π, is determined as

$$Q_\pi(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1} \sim \pi}[r(s_t, a_t) + \gamma Q_\pi(s_{t+1}, \pi(s_{t+1}))], \tag{2.1}$$

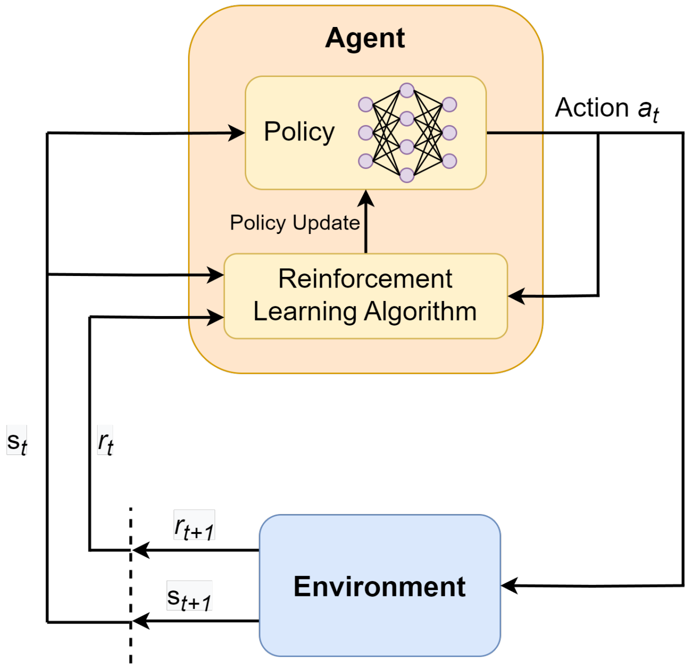

Figure 2.1: Reinforcement learning framework

2.3.2 DNR Environment

To apply RL algorithms to the DNR problem, DNR should be modeled as an MDP. For this purpose, the switching actions are defined as $a_t = [1,\ 1,\ 0,\ ...,\ 1]$ where the open and close commands of the lines are represented by zeros and ones, respectively. In this study, it is assumed that all lines have switches which can be opened or closed. Therefore, the action dimension is equal to the number of lines in the system. The state of the system is the switch states, active, and reactive power consumption at different buses represented by $s_t = [p_t, q_t, \alpha_t]$. Here, $\alpha_t = [\alpha_{1t}, \alpha_{2t}, ..., \alpha_{mt}]$ is the switch states, $p_t = [p_{1t}, p_{2t}, ..., p_{nt}]$ is the active power, and $q_t = [q_{1t}, q_{2t}, ..., q_{nt}]$ is the reactive power of different buses. Parameter n stands for the number of busses, and m for the number of lines in the system. The transition probability between time-steps, $\mathcal{P}$, is assumed to be the random process of power injections.

The reward function for this RL problem is the negative of network loss which needs to be maximized so that the loss will be minimized. It is defined as

$$\mathcal{R}(s_t, a_t) = -C^l p_t^l(s_{t+1}) - C^v(s_{t+1}), \tag{2.2}$$

where $p_t^l(s_{t+1})$ is the total line losses of the system at state s_{t+1} and C^l is the penalty for line losses in [1/kW]. The variable $C^v(s_{t+1})$ is the penalty of violating the network voltage constraint defined as

$$C^v(s_{t+1}) = \begin{cases} \lambda & \text{if } V^{\max} < \max(V_{t,n}) \text{ or} \\ & \quad \min(V_{t,n}) < V^{\min} \\ & \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

where λ is a large constant number to impose a highly negative reward for actions that cause voltage violation in the system. Load balance and radiality constraints are fulfilled while selecting the action space $\mathcal{A}$.

The complete action space for DNR is very large. For example, a system with 20 lines has a total of 2^{20} actions. However, most of these actions cause loops or disconnections in the system and hence, are infeasible to be performed in a real distribution system. To address this issue, Algorithm 1 is used to select only the feasible actions as the action space for RL. In this algorithm, first, all lines are closed to find all loops in the network (lines 3-4). Then, all of the switches that open these loops are found (line 5). Finally, all combinations between these switches are found in a way that no loops remain in the network and no disconnections occur (lines 6-23). It should be noted that action space is a pool from which the agents choose their actions. Algorithm 1 only changes the action space and it does not alter the action selection strategy of the agents in any of the RL algorithms.

Algorithm 1 Structure Identification

```
1  Input: Topology information of the network
2  Output: Feasible action space
   ▷ Construct the network with graph analysis tools
3  Close all lines
4  Find all loops in the network $l \in \{1, 2, ..., N\}$
5  Find the set of all switches $S_l$ that open each of these loops
6  Find the total number of combinations, $M$, between $S_l$ from each loop
   ▷ Find all combination of lines for opening loops
7  for $i \in \{1, 2, ..., M\}$ do
8      $flag = 0$
9      while $flag = 0$ do
10         for $j \in \{1, 2, ..., N\}$ do
11             if loop j exists then
12                 Open a line from loop $j$
13                 if Network is disconnected then
                       Close the line and open a different line
14                 if the line is already open then
15                     Open a different line from loop $j$
16                     if Network is disconnected then
17                         Close the line and open a different line
18     Find all the loops $l$ in the network
19     if $l = 0$ then
20         $flag = 1$
21         Save the current structure of the network as a feasible action
22     else
23         Repeat 5-21 for the new loops
```

2.3.3 RL Algorithms

RL algorithms are divided into two main categories: on-policy and off-policy. In on-policy RL, the behavior policy which agent uses to select its actions and the target policy which it tries to learn are the same. In off-policy RL, they are different. Therefore, on-policy algorithms are more likely to converge to a sub-optimal policy. This study implements and tests the performance of DQN, dueling DQN, DQN with prioritized experience replay, and SAC as off-policy algorithms, and PPO as an on-policy algorithm.

Algorithm 2 summarizes the DQN algorithm. First, the target neural network, q-network, replay buffer, and greedy policy are initialized (lines 3-5). Then, for N

number of episodes, the environment is reset, for T number of steps (episode length), an action is chosen by the policy and executed, the transitions are added to the replay buffer, and finally, the loss over a batch of transitions is computed and gradient descent is performed on the q-network to update its parameters and the policy (lines 6-15). Every M episodes, the target and q-network are synchronized (line 16). It should be noted that the q-network outputs actions values. The policy chooses the action with the highest value.

Dueling DQN is an improvement over DQN [101]. The key motivation behind this architecture is that, in some states, performing an action will not change the obtained reward so it is unnecessary to know the value of each action at every time step. An example of such condition is the Atari game Enduro where taking an action will not change the obtained reward until collision is imminent. In general, the action value is

Algorithm 2 Deep Q-Learning

```
1  Input: Learning rate $\alpha$, number of episodes $N$, discount factor $\gamma$,
   batch size, environment
2  Output: Optimal policy
   ▷ Initialization
3  Initialize target network with parameters $\theta_{targ}$ and q-network with parameters $\theta_{targ} \leftarrow \theta$
4  Initialize replay buffer $B$
5  Initialize greedy policy $\pi$ and $\epsilon$-greedy policy $\pi_\epsilon$
   ▷ Training
6  for $Episode \in \{1, 2, ..., N\}$ do
7      Reset the environment and record $s_0$
8      for $t \in \{0, 1, .., T-1\}$ do
9          Select action $a_t$ using $\pi_\epsilon$
10         Execute action $a_t$ and receive $s_{t+1}, r_{t+1}$
11         Add transition $(s_t, a_t, r_{t+1}, s_{t+1})$ into buffer $B$
12         Set $s_t = s_{t+1}$
13         Choose a batch, $b$, of transitions from buffer $B$
14         Compute loss function for the selected batch
           $y_i = \begin{cases} r_i & \textbf{For } \textit{terminal state} \\ r_i + \gamma \max(Q(s_i, a_i|\theta_{targ})) & \textbf{For } \textit{non-terminal state} \end{cases}$
           $L(\theta) = \frac{1}{|b|} \sum_{i=1}^{|b|} [y_i - Q(s_i, a_i|\theta)]^2$
15         Perform gradient descent for q-network and update its parameters $\theta$
16 Every $M$ episodes synchronize $\theta_{targ} \leftarrow \theta$
```

defined as

$$Q_\pi(s,a) = \mathbb{E}_\pi[\sum_{t=0}^{T-1} \gamma^t r_t | s_t = s, a_t = a], \tag{2.4}$$

and the state value is defined as

$$V_\pi(s) = \mathbb{E}_\pi[\sum_{t=0}^{T-1} \gamma^t r_t | s_t = s]. \tag{2.5}$$

The advantage value shows how advantageous selecting an action is relative to the other actions at a given state. The advantage function is obtained by subtracting the state value from the action value

$$A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s). \tag{2.6}$$

By separating $Q_\pi(s,a)$ into $V_\pi(s)$ and $A_\pi(s,a)$, the dueling architecture learns the state value without having to learn the effect of each action for each state. $V_\pi(s)$ and $A_\pi(s,a)$ are combined into one equation as follows

$$\begin{aligned} Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + (&A(s,a;\theta,\alpha) \\ &- \frac{1}{|A|}\sum_{a'} A(s,a;\theta,\alpha)). \end{aligned} \tag{2.7}$$

DQN with prioritized experience replay was first introduced by Google DeepMind in 2015 [81]. The basic idea behind this algorithm is to prioritize the transitions in the replay buffer that are rare but more helpful for the learning process. The metric to measure the importance of each transition is the magnitude of a transition's time difference (TD) error, δ. However, an offset value should be added to prevent the importance to become zero, i.e.

$$p_i = |\delta_i| + \epsilon. \tag{2.8}$$

Using the priority, the probability of selecting each transition is defined as

$$P_i = \frac{(p_i)^a}{\sum_{i=1}^{N} (p_i)^a}, \tag{2.9}$$

where a is a factor that determines how much prioritization is used. Prioritized replay introduces bias in training since it selects some transitions more frequently. To prevent this, the neural network update step is changed to αw_i instead of α, where w_i is defined as

$$w_i = (\frac{1}{N}.\frac{1}{P_i})^b. \tag{2.10}$$

The exponent b is used to control the bias correction which is more important later during the training than at the beginning. Therefore, b starts at a low value and gradually increases to 1 over time.

SAC is an off-policy reinforcement learning algorithm whose objective is not only to maximize the rewards but also to increase the entropy. Entropy is the likeliness of the algorithm to explore new actions. A high-entropy algorithm prevents premature convergence to a bad local optima and encourages the state space exploration improving the collected transition data. Therefore, the policy is trained to maximize the following objective

$$J(\theta) = \sum_{t=1}^{T} \mathbb{E}_{(s_t,a_t)\sim\rho_{\pi_\theta}}[r(s_t, a_t) + \alpha\mathcal{H}(\pi_\theta(.|s_t))], \tag{2.11}$$

where $\mathcal{H}(.)$ is the entropy measure, and the parameter α represents the entropy temperature and controls the randomness of the policy versus the reward. In general, there are three functions that SAC needs to learn: 1) the policy with parameter θ, 2) soft Q-value function with parameter w, and 3) soft state value function with parameter ψ. Soft Q-value and state value are determined, respectively, by

$$Q(s_t, a_t) = r(s_t, a_t) + \gamma\mathbb{E}_{s_{t+1}\sim\rho_\pi(s)}[V(s_{t+1})], \tag{2.12}$$

$$V(s_t) = \mathbb{E}_{a_t\sim\pi}[Q_{s_t,a_t} - \alpha log\pi(a_t|s_t)], \tag{2.13}$$

To obtain the soft state value function parameters, the following mean squared error is minimized

$$J_V(\psi) = \mathbb{E}_{s_t\sim\mathcal{D}}[\frac{1}{2}(V_\psi(s_t) - \mathbb{E}[Q_w(s_t, a_t) - log\pi_\theta(a_t|s_t)])^2], \tag{2.14}$$

where $\mathcal{D}$ is the replay buffer. The soft Q function parameters are obtained by minimizing the soft Bellman residual

$$J_Q(w) = \mathbb{E}_{(s_t,a_t)\sim\mathcal{D}}[\frac{1}{2}(Q_w(s_t,a_t) - (r(s_t,a_t) + \gamma\mathbb{E}_{s_{t+1}\sim\rho_\pi(s)}[V_{\overline{\psi}}(s_{t+1})]))^2], \tag{2.15}$$

where $\overline{\psi}$ is the target value function. Finally, the policy is optimized by minimizing the KL-divergence

$$J_\pi(\theta) = \mathbb{E}_{a_t\sim\pi}[log\pi_\theta(a_t|s_t) - Q_w(s_t,a_t) + logZ_w(s_t)],. \tag{2.16}$$

Parameter Z is the partition function to normalize the distribution. Finally, the SAC algorithm is implemented as Algorithm 3 [31]. First, the state value function, Q-value function, policy, target value function, and replay buffer parameters are initialized (lines 3-4). Then, for each episode, an action is selected by the policy and executed and the transition is added to the buffer (lines 5-9). If it is time to update, state value function, Q-value function, policy, and target value function are updated using the previously introduced equations (lines 10-14).

Algorithm 3 Soft Actor-Critic

```
1  Input: Learning rate $\alpha$, number of episodes $N$, discount factor $\gamma$,
   batch size, environment, entropy temperature $\alpha$
2  Output: Optimal policy
   ▷ Initialization
3  Initialize parameters $\theta, w, \psi$, and $\overline{\psi}$
4  Initialize replay buffer $\mathcal{D}$
   ▷ Training
5  for Episode $\in \{1, 2, ..., N\}$ do
6      for $t \in \{0, 1, .., T-1\}$ do
7          Select action $a_t$ using policy $\pi_\theta(a_t|s_t)$
8          Execute the action $a_t$ and receive $s_{t+1}, r_{t+1}$
9          Add transition $(s_t, a_t, r_{t+1}, s_{t+1})$ into buffer $\mathcal{D}$
10     for each gradient update step do
11         Update state value function with parameter $\psi$
12         Update Q-value function with parameter $w$
13         Update policy with parameter $\theta$
14         Update target value function with parameter $\overline{\psi}$
```

As previously mentioned, PPO is an on-policy RL algorithm. It is a type of policy gradient method that was introduced in 2017 [82] as an improvement over Trust

Region Policy Optimization (TRPO). PPO uses two neural networks, an actor network whose input is the state and output is the list of probabilities for each action, a critic network, whose input is the state and the output is state value. The objective function of vanilla policy gradient methods is defined as

$$L^{PG}(\theta) = \hat{\mathbb{E}}_t[log\pi_\theta(a_t|s_t)\hat{A}_t], \tag{2.17}$$

where $L^{PG}(\theta)$ is the policy loss, $log\pi_\theta(a_t|s_t)$ is the log of probabilities from the actor network, and $\hat{A}_t$ is the advantage function.

To prevent large policy updates, the policy gradient step is restricted by $r_t^s(\theta)$ in TRPO, defined as the probability ratio between the action under the current policy and the action under the previous policy

$$r_t^s(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \tag{2.18}$$

As a result, the objective function in TRPO is defined as

$$\text{Maximize} \;\; \hat{\mathbb{E}}_t[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}\hat{A}_t] \tag{2.19}$$

$$\text{subject to} \;\; \hat{\mathbb{E}}_t[KL[\pi_{\theta_{old}}(.|s_t), \pi_\theta(.|s_t)]] \leq \sigma$$

Similarly, to restrict the update step in PPO, clipped surrogate objective is defined as

$$L^{\text{CLIP}}(\theta) = \hat{\mathbb{E}}_t[\min(r_t^s(\theta)\hat{A}_t, \text{clip}(r_t^s(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)], \tag{2.20}$$

To minimize the value function error and ensure enough exploration of the agent, the final objective function is defined as

$$L^{\text{CLIP+VF+S}}(\theta) = \hat{\mathbb{E}}_t[L_t^{\text{CLIP}}(\theta) - c_1 L_t^{\text{VF}} + c_2 S[\pi_\theta](s_t)], \tag{2.21}$$

where L_t^{VF} is value function error, $S[\pi_\theta](s_t)$ is entropy bonus, and c_1 and c_2 are hyperparameters. PPO algorithm is given in Algorithm 4. First, the replay buffer, actor, and critic networks are initialized (lines 3-4). Then, for N number of episodes, each actor runs the policy for T time steps, stores the transitions, and computes the advantage functions (lines 5-9). Finally, for a batch of transitions every K epochs, the objective function in Eq. (2.21) is optimized and the actor and critic parameters are updated.

Algorithm 4 Proximal Policy Optimization

```
1  Input: Learning rate $\alpha$, number of episodes $N$, discount factor $\gamma$,
   batch size, environment, number of parallel agents $N^a$
2  Output: Optimal policy
   ▷ Initialization
3  Initialize actor and critic networks
4  Initialize replay buffer $B$
   ▷ Training
5  for $Episode \in \{1, 2, ..., N\}$ do
6      for $actor \in \{1, 2, ..., N^a\}$ do
7          Run policy $\pi_{\theta_{old}}$ in environment for $T$ timesteps
8          Store transitions in the replay buffer
9          Compute advantage estimates $\hat{A}_1, \hat{A}_2, ..., \hat{A}_T$
10     Optimize $L^{CLIP+VF+S}$ w. r. t. $\theta$, with $K$ epochs and batch size $M \leq N_a T$
11     $\theta_{old} \leftarrow \theta$
```

2.3.4 Action Reduction

The action space in DNR is quite large. Presented RL algorithms either would not converge to the optimal solution with such a large action space, or they would require a very long training time and computational resources. Therefore, the size of the action space needs to be reduced while keeping only feasible actions.

To improve the quality of the DNR solutions found by RL algorithms, in this study, the size of the action space is reduced in two stages. At the first stage, random sampling is used to define an action space for RL. Then, using the selected actions, DQN algorithm is implemented. At the end of the training, the agent has learned the action values. Therefore, it is used to only select the actions with the highest q-values to reduce the size of the action space at the second stage.

2.4 Simulation and Results

The proposed RL method is first applied to the IEEE 33-node system and then to a 136-node distribution system to demonstrate the scalability of the developed method. The data and parameters used for simulations are presented in this section, along with the obtained results.

2.4.1 Experimental Data Setup

The tuned parameters of the five described RL algorithms are given in Table 2.1. Since the number and range of hyperparameters for RL algorithms are very large, the hyperparameter tunning is performed manually, by changing the value of one hyperparameter, observing the result, and then deciding the next value. The daily load data is generated similar to [112] and then, considering a normal distribution and 15% standard deviation to the generated daily IEEE 33-node and 136-node network loads, the load data is generated for the required number of days. The minimum and maximum voltage limits are considered to be 0.95 p.u. and 1.05 p.u., respectively. Parameter C^l is set to 10,000 and λ to 100,000 based on empirical performance. The neural networks are coded using PyTorch and in Python environment. All RL algorithms use a four-layer neural network structure, with 1024 neurons in the first hidden layer and 512 neurons in the second hidden layer. NetworkX package is used as the graph analysis tool in Algorithm 1 and the power network is modeled using pandapower package.

Algorithm 1 finds 14,401 feasible actions for 33-node system and 56,996 actions for the 136-node system. The initial sampling randomly selects 300 and 1,000 actions for the 33- and 136-node systems, respectively. After applying the initial DQN, the convergence curve and action values for 33-node and 136-node systems are given in Fig. 2.2 and Fig. 2.3, respectively. Among these actions, the top 80 actions with the highest q-values are chosen as the action space for the 33-node test system and the

Table 2.1: Parameters of RL Algorithms

Algorithm	**Parameters**	**Values**
DQN	learning rate	10^{-3}
	number of hidden units	$\{1024, 512\}$
	batch size	512
	discount factor	0.99
	synchronize every	10
Dueling DQN	learning rate	10^{-3}
	number of hidden units	$\{1024, 512\}$
	batch size	512
	discount factor	0.99
	synchronize every	10
DQN with Prioritized Experience Replay	learning rate	10^{-3}
	number of hidden units	$\{1024, 512\}$
	batch size	512
	discount factor	0.99
	synchronize every	10
	parameter a	0.5
	parameter b	0.4
SAC	learning rate	10^{-4}
	batch size	512
	discount factor	0.99
	number of hidden units	$\{1024, 512\}$
	initial temperature parameter	1
	soft update interpolation factor	0.05
PPO	learning rate	$3 \cdot 10^{-4}$
	discount factor	0.99
	number of epochs (K)	512
	batch size	256
	clip rate	0.2
	number of hidden units	$\{1024, 512\}$
	L2 regularization coefficient	0.5
	entropy coefficient	10^{-3}
	entropy coefficient decay	0.99

top 400 are chosen as the action space of the 136-node system.

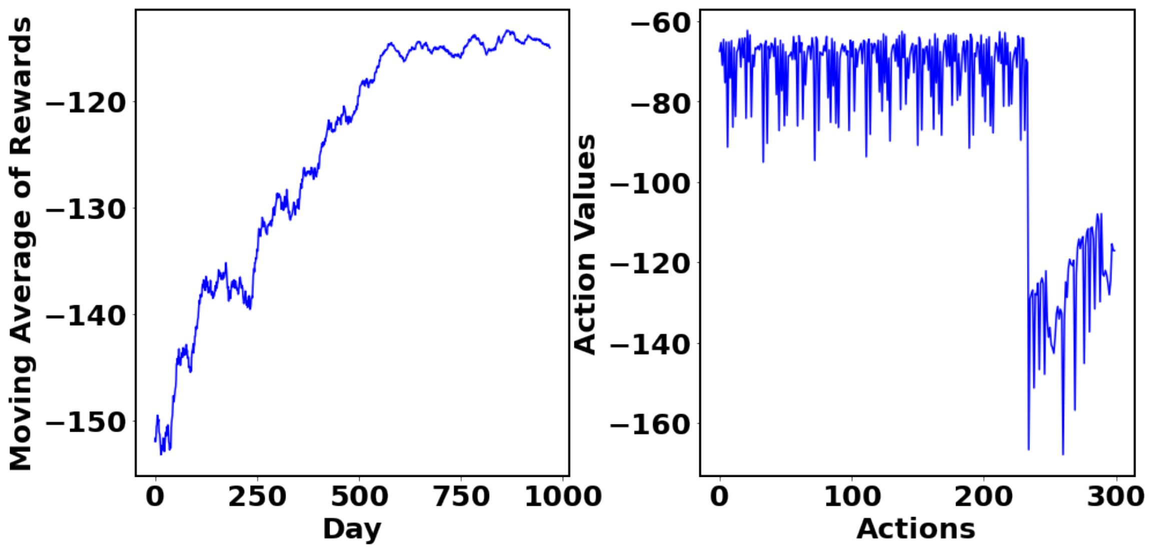

Figure 2.2: 30-step moving average of daily rewards and action values for 33-node system with 300 actions

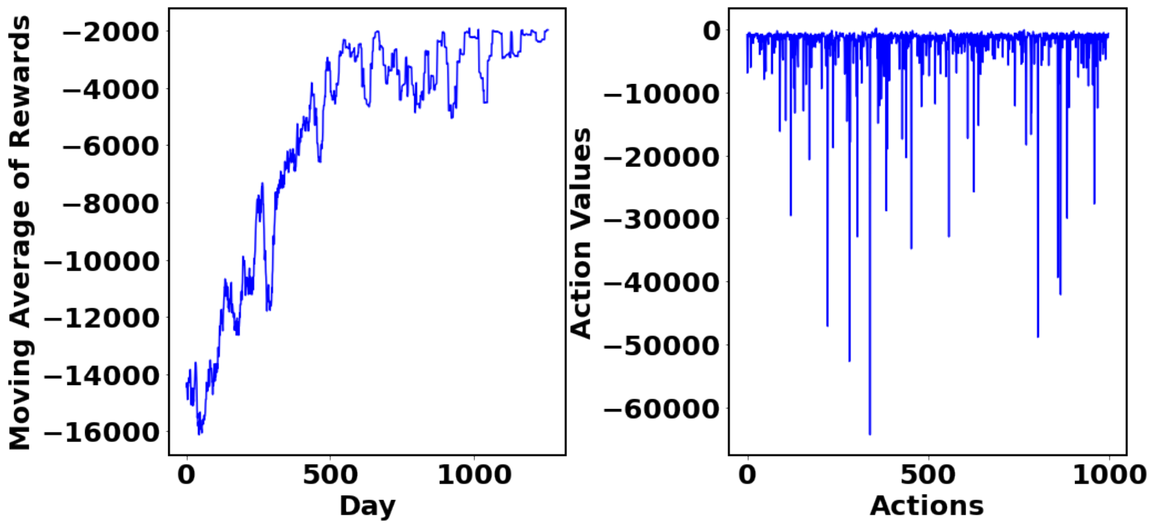

Figure 2.3: 30-step moving average of average daily rewards and action values for 136-node system with 1000 actions

2.4.2 33-Node Test System

The 33-node test system is depicted in Fig. 2.4. It is assumed that all lines can be opened and closed using switches and the dashed lines are initially open.

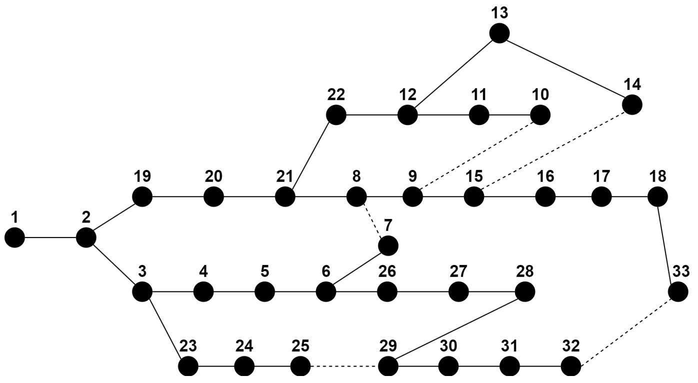

Figure 2.4: 33-node test system

Average daily rewards and their moving averages for the five RL algorithms are illustrated in Fig. 2.5 and Fig. 2.6, respectively. As can be seen, SAC has the fastest convergence. However, the final obtained rewards are lower compared to DQN and dueling DQN. DQN and dueling DQN have very similar performance and PPO has the weakest performance among these algorithms.

Since in on-policy RL algorithms the agent uses the same behavior and target policy, these algorithms are more likely to become trapped in a local minimum. As a result, PPO, which is the only on-policy algorithm in this study, has the worst performance among other algorithms. Policy gradient methods such as SAC and PPO search directly for the optimal policy instead of estimating the action-value. These methods require fresh samples from the environment obtained with the current policy. However, value methods such as DQN can take advantage of the old data obtained from an older policy stored in the memory. Therefore, policy methods usually require more interaction with the environment to reach the most optimal solution. On the other hand, since they search directly for the optimal policy, they might converge faster.

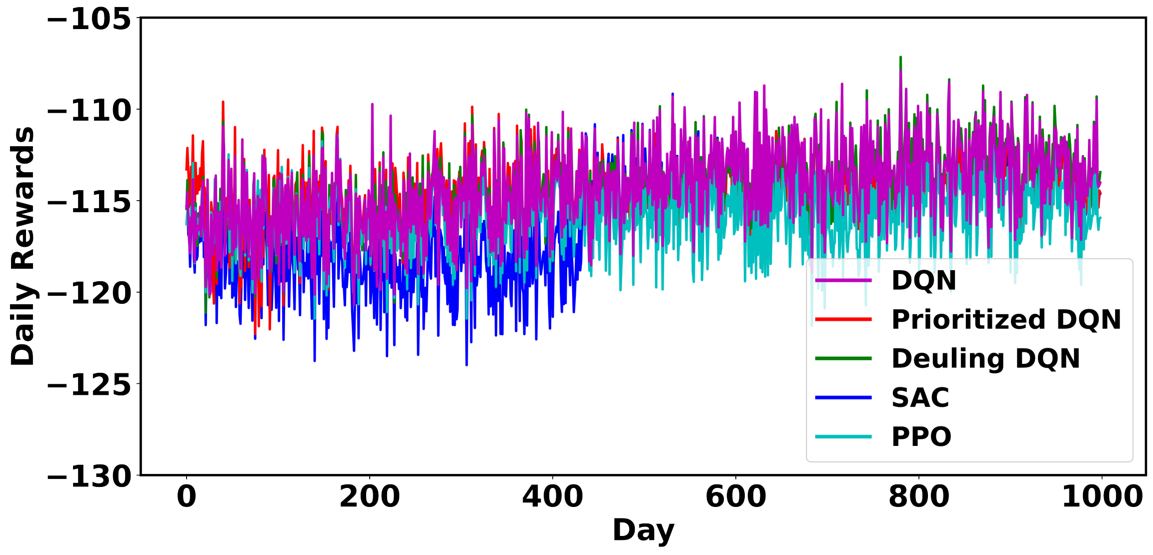

Figure 2.5: Average daily rewards for 33-node system during training using five RL algorithms

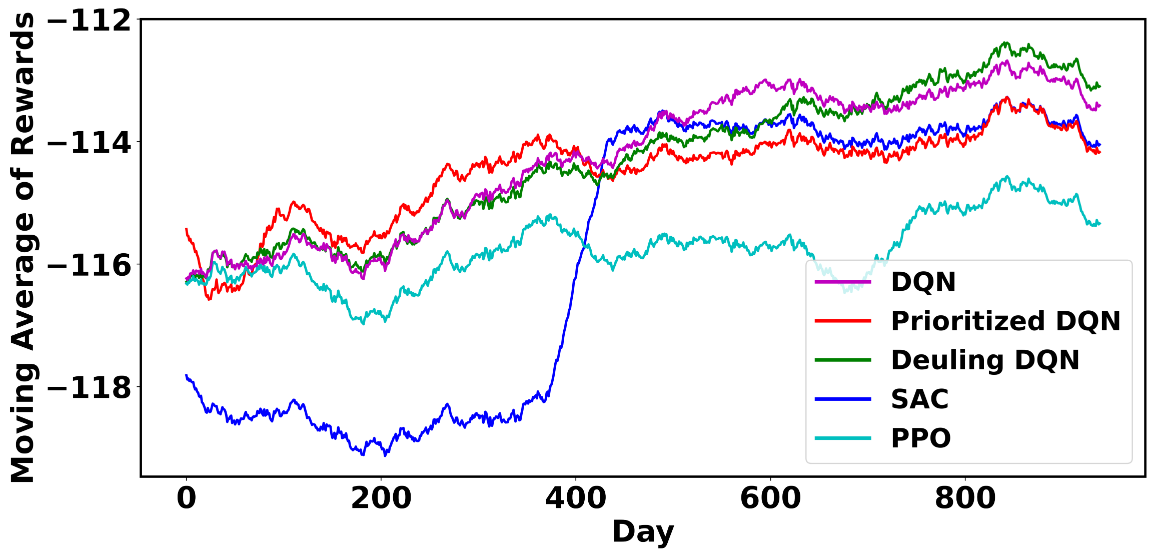

Figure 2.6: 60-step moving average of daily rewards for 33-node system using five RL algorithms

2.4.3 136-Node Test System

The 136-node distribution system has a relatively complex structure, as depicted in Fig. 2.9. The average daily rewards and their moving averages for the 136-node system are shown in Fig. 2.7 and Fig. 2.8, respectively. Similar to the 33-node system, SAC has the fastest convergence. However, the final obtained rewards are lower compared to DQN and DQN with prioritized experience replay which obtain similar rewards at the end of the training. However, DQN with prioritized experience replay converges slightly faster. By comparison to the 33-node system, it can be seen that for larger systems, the prioritized experience replay is more effective and converges faster.

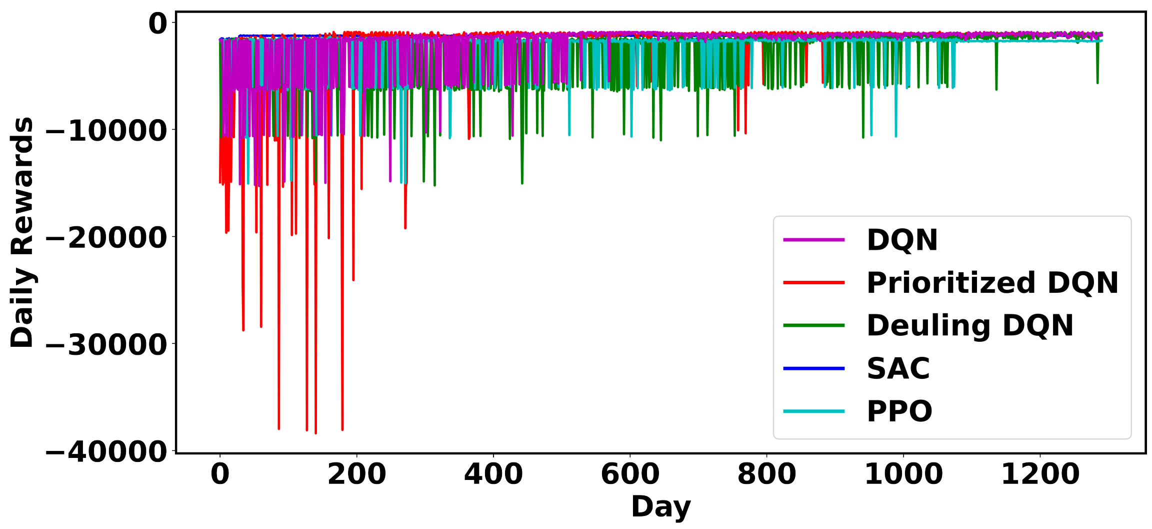

Figure 2.7: Average daily rewards for 136-node system during training using five RL algorithms

2.4.4 Comparison of Results

Table 2.2 shows the network loss after a single-hour DNR using each of the five RL algorithms, the SOE method [112], genetic algorithm (GA) [12], adaptive ant colony optimization (AACO) method [88], and cuckoo search algorithm (CSA) [72] from the literature. Among the compared algorithms, DQN attains the lowest network loss, outperforming other published methods. It can also be inferred that the DQN-based action reduction approach was effective and that the resulting action space was sufficient to reduce the system loss in the DQN, Dueling DQN, and SAC algorithms. It can be concluded that, in terms of stability and optimality, DQN is the best-performing

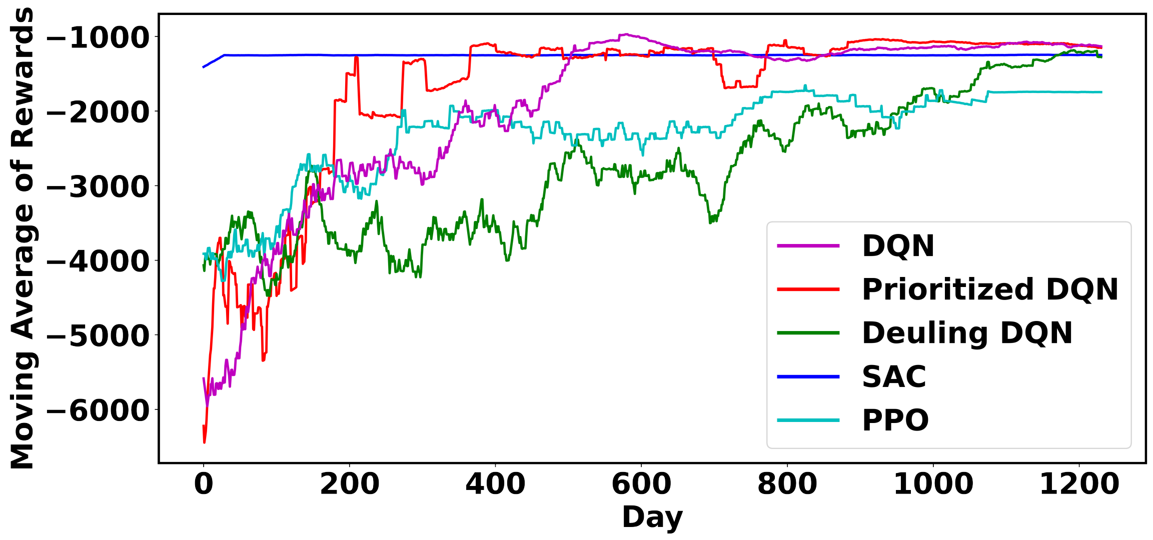

Figure 2.8: 60-step moving average of daily rewards for 136-node system using five RL algorithms

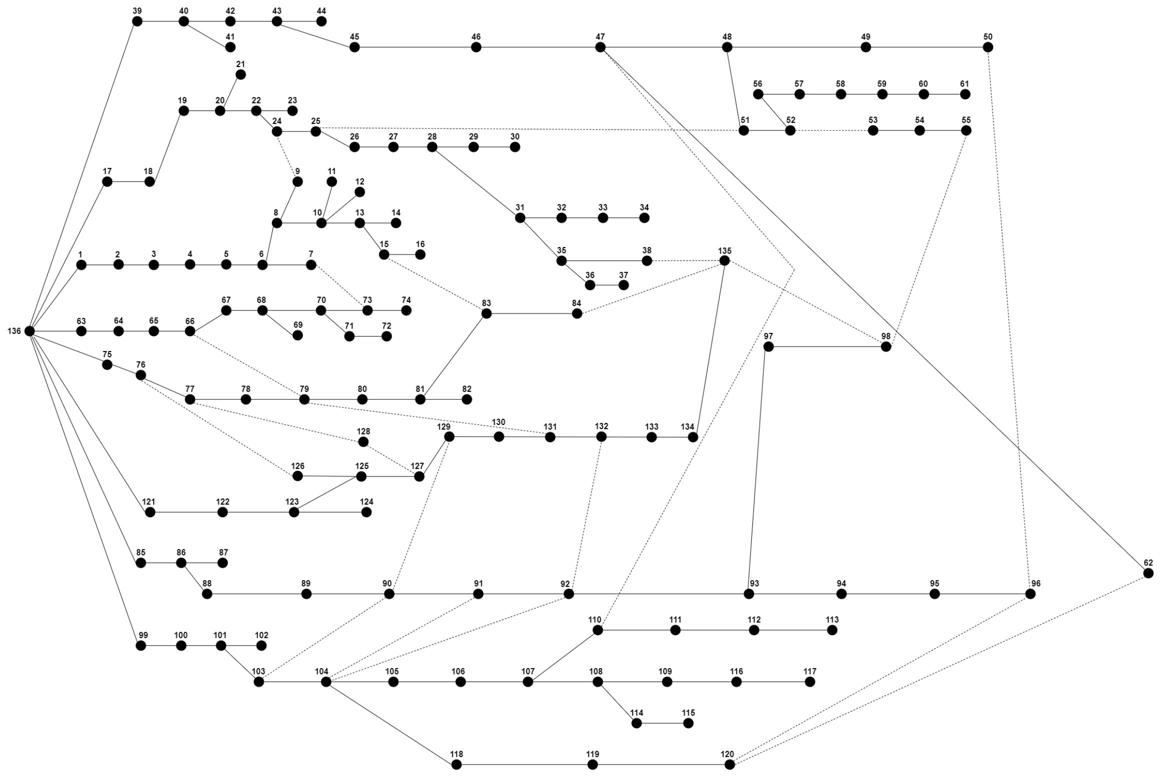

Figure 2.9: 136-node test system

algorithm. In terms of convergence speed, SAC performs the best.

Table 2.2: System Loss Obtained by Single-Hour DNR

Algorithm	**Test System**	
	33-node	136-node
DQN	6.201	51.606
Dueling DQN	6.188	72.401
Prioritized DQN	6.273	144.373
SAC	6.273	72.384
PPO	6.342	100.411
SOE [112]	6.740	51.651
GA [12]	6.740	51.911
AACO [88]	6.740	51.655
CSA [72]	6.740	-
Action Sampling DQN	6.266	90.197

The computational times of the RL methods are compared against SOE, AACO, and mathematical programming (MP) methods (including mixed-integer linear and mixed-integer nonlinear programming) [112] in Table 2.3. As it can be inferred from this table, RL methods are at least eight times faster than SOE, AACO, and MP methods for the 136-node system. The computational speed of the RL algorithms is even more signified as the size of the system grows. However, RL algorithms can have a very long training time, as presented in the same table.

Table 2.3: Execution Time Comparison

Algorithm	Execution Time [s]	
	33 bus	136 bus
PPO	0.206	0.776
DQN	0.207	0.786
Dueling DQN	0.502	1.911
SAC	0.453	1.916
Prioritized DQN	0.457	1.964
SOE [112]	1	16.9
AACO [88]	0.3	894.20
MP [112]	19	$1,236$
Approximate RL Training Time	$5,727$	$25,190$

2.5 Conclusion

In this chapter, the DNR problem is formulated as a Markov decision process to implement and compare five RL algorithms, including DQN, dueling DQN, DQN with prioritized experience replay, SAC, and PPO. The proposed RL-based approach was applied to the 33- and 136-node test systems. The results showed that among these algorithms, SAC had the fastest convergence, while DQN performed best in terms of stability and optimality. In addition, a new DQN-based action reduction method was developed to reduce the size of the action space. The effectiveness of the new reduction approach was tested by comparing the pre- and post-sampling results as well as the results from the literature.

Future studies will focus on developing new action space sampling methods for RL and performing sensitivity analysis for various action space selections. In addition, various objectives for the RL agent will be defined and tested. Finally, the effectiveness of the proposed method for increasing electric vehicle penetration will be investigated.

Chapter 3

A Generalised Deep Reinforcement Learning Model for Distribution Network Reconfiguration with Powerflow-based Action Space Sampling

3.1 Abstract

Distribution network reconfiguration (DNR) is used by utilities to enhance power system performance in various ways, such as reducing line losses. Conventional DNR algorithms rely on accurate values of network parameters and lack scalability and optimality. To tackle these issues, a new data-driven algorithm based on reinforcement learning is developed for DNR in this chapter. The proposed algorithm is comprised of two main parts. The first part, named action space sampling, aims at analyzing the network structure, finding all feasible reconfiguration actions, and reducing the size of action space to only most optimal actions. In the second part, deep Q-learning (DQN) and Dueling DQN methods are used to train an agent to take the best switching actions according to switch states and loads of the system. The results show that both DQN and Dueling DQN are effective in reducing system losses through grid reconfiguration. The proposed methods have faster execution time compared to the conventional methods and are more scalable.

3.2 Introduction

The primary goal of distribution network reconfiguration (DNR) is to reduce power losses in the system by altering the network's topology and switch settings [7]. DNR has become a fundamental part of distribution system operation due to power fluctuations caused by increasing penetration of distributed generation (DG) units and electric vehicles [70]. It can also increase the hosting capacity for distributed energy resources, improve the voltage profile, and minimize DG power curtailments.

3.2.1 Motivation and Literature Review

The DNR problem can be classified as a mixed-integer nonlinear problem that falls under the category of NP-hard problems, mathematically speaking [23]. The non-linearity of AC power flow and the radiality constraint of the network contribute to this classification. Therefore, it can only be optimized after simplifications [40] or through search algorithms [88]. To this end, the Lévy flight and chaos disturbed beetle antennae search (LDBAS) algorithm is used in [97] to solve a multi-objective dynamic reconfiguration model minimizing active power loss, load balancing index, and maximum node voltage deviation subject to stochastic power output from DG units and electric vehicles. Similarly, a multi-objective optimization problem is formulated in [13] and solved through a social beetle swarm algorithm to minimize network loss, load imbalance, and voltage deviation. In [69], the water cycle algorithm (WCA) is employed to address the network reconfiguration problem and to determine the optimal size and location of the DG units. Other relevant methods to solve network reconfiguration problem include molecular differential evolution algorithm [76], Harris Hawks optimization (HHO) algorithm [34], Bayesian learning-based evolutionary algorithm [119], modified particle swarm optimization (PSO) [22], and genetic algorithm (GA) [12]. However, none of these methods can guarantee the optimality of the obtained solutions since they are only search algorithms. Furthermore, the time taken to execute them increases exponentially as the system size grows.

In the second type of studies, mathematical programming and optimization methods are used to solve simplified formulations of the DNR problem. For instance, linear DistFlow equations are used in [85] to measure the voltage volatility of a bus and a Benders decomposition-based method is used to solve the proposed DNR problem as a mixed-integer quadratic program (MIQP). Similarly, an MIQP model is proposed in [3] for the DNR problem after linearizing the powerflow equations. The DNR problem was addressed in [21] by formulating it as a mixed-integer linear model and solving it with the commercial solver CPLEX. This was achieved through the linearization of power flow equations and load models. The major drawback of this approach is that, due to the linearization of powerflow equations and the use of simplified assumptions, the obtained results are not accurate.

Several recent studies have used machine learning techniques to perform the DNR procedure. The authors of [91] incorporated chaotic local search and quasi-oppositional-based learning techniques into the original neural network algorithm to tackle the challenges of network reconfiguration and distributed generation allocation concurrently. Nevertheless, optimization-focused artificial intelligence algorithms tend to have poor convergence and are highly susceptible to being trapped in local optima [97]. The authors of [58] used a long-short-term memory (LSTM) network to capture the mapping mechanism present between load distribution and the most effective reconfiguration strategies. The study presented in [43] involved the training of a deep convolutional neural network to establish the correlation between the network structure and the short-term voltage stability performance by analyzing historical data. Once trained, the network was used for the DNR process. The primary drawback of these two techniques is that the size of the neural network required for training tends to increase as the network size grows, resulting in an increase in computational expenses. Moreover, as these networks are exclusively trained on past operational data, they cannot find novel reconfiguration strategies. The study in [118] employed a deep neural network to construct a probability distribution (PD) forecasting network that could anticipate the joint PD for DG outputs and loads. Subsequently, a variant of column-and-constraint generation technique with efficient scenario decomposition solved the DNR problem in the worst-case scenario of PD for DG outputs and loads. In this instance, neural networks were only used to predict PD of DG outputs.

Most current studies adopt a model-based control strategy for DNR, which requires exact knowledge of distribution network parameters. However, obtaining accurate parameter estimates is challenging for electric utilities due to the complex nature of distribution networks and changes caused by weather-related fluctuations [24]. Moreover, as the network size grows, the computational complexity of model-based algorithms rapidly increases, making real-time control infeasible [75].

3.2.2 Related Work

Reinforcement learning (RL) is a machine learning technique based on the action and response process, which has been successful in solving many difficult sequential decision-making problems [116]. In the process of transforming DNR into a Markov decision process (MDP) for RL, each instance of manipulating various switches to either open or close is regarded as a singular action. As a consequence, the action space becomes notably expansive. Therefore, the primary challenge when using RL for DNR lies in effectively managing this extensive action space, while also ensuring the optimality of the achieved outcomes compared to alternative mathematical or heuristic methods.

In terms of addressing this challenge, RL studies can be divided into three groups. In the first set of studies, the RL agent is trained on only a very limited historical operational dataset. To this end, the method proposed in [25] for DNR involves training a batch-constrained soft actor-critic RL algorithm using a finite historical operational dataset. However, the performance of this algorithm largely depends on accurate historical DNR data which is difficult to obtain. In addition, the algorithm does not search for new reconfiguration strategies since it is trained only on previous strategies that might not be optimal.

In the second category of studies, a method for reducing the action space is devised to minimize its dimensions. For instance, NoisyNet deep Q-learning (DQN) method is adopted for DNR and coupled with a loop-based encoding to reduce the scale of the action space in [95]. However, it is important to emphasize that the loop-based encoding method finds a notably small set of configuration actions. Consequently, the resultant solutions by the DNR agent lack a definitive assurance of optimality or proximity to optimality. Additionally, the applicability of the method to large networks is not demonstrated.

The third set of studies utilize the multi-agent RL approach to manage the extensive action space. This is achieved by distributing the action space among multiple agents. In [54], a complex multi-objective DNR model is introduced, considering multiple factors, including renewable energy curtailment, voltage stability, power loss, and generation cost. It is solved using a DQN-assisted multi-objective bacterial foraging optimization algorithm. To reduce the action space size, the action space is divided into smaller subspaces, with each subspace being handled by a separate DQN. The study in [104] introduces a deep RL-based method to reconfigure distribution networks during extreme events, creating microgrids to restore critical services. The approach employs a multi-agent soft actor-critic (MA-SAC) strategy to efficiently control circuit breakers for isolating sections of the network and accommodating various system states and scales. In [46], a cloud-edge collaboration architecture is developed and combined with a deep RL model for real-time optimal reconfiguration of urban distribution networks. This approach involves multi-level dynamic reconfiguration, integrating feeder, transformer, and substation levels, using a multi-agent system. The model employs both offline and online learning phases, incorporating a Q-learning-based multi-agent conservative Q-learning algorithm for stability in offline learning, and a multi-agent deep deterministic policy gradient algorithm for exploration and experience pool updates in online learning. The primary limitation of this methodology stems from the extensive action space within large systems, such as the 136-node test system encompassing over three million feasible actions. Consequently, even after partitioning this action space among multiple agents, its sheer magnitude persists, presenting a challenge for comprehensive action space exploration. Furthermore, all of these studies lack direct comparisons with solutions documented in existing literature, thus, the efficacy of the proposed approach remains uncertain.

Additionally, there is a collection of studies that do not present any approach for reducing the dimensions of the action space. In such cases, it is not clear how the action space for the DNR agent is selected. These studies include a branching double DQN (BDDQN) and multi-policy soft actor-critic (MPSAC) for rapid decision-making in sequential reconfiguration with soft open points in [110], a DQN method to address DNR within both the IEEE 33-node system and a real-life, large-scale testing environment in [52], a DQN-based network reconfiguration to enhance the reliability of distribution networks, particularly when confronted with the fluctuating nature of distributed renewable energy resources in [73] and [63]. Finally, a dynamic network reconfiguration strategy aimed at minimizing operation costs and load shedding is introduced in [10]. It employs a three-stage deep RL approach to optimize the reconfiguration and set-points of distributed generators in real-time, enhancing distribution network stability and reliability by quickly adapting to events and uncertainties. Double DQN (DDQN) is used in the offline learning process for optimal reconfiguration and three RL algorithms including deep deterministic policy gradient (DDPG), soft actor-critic, and twin delayed DDPG (TD3) are evaluated to determine the real-time set-points of distributed generators and energy storage systems. However, this study does not describe any method to reduce the dimension of the action space.

3.2.3 Contributions

To address the limitations encountered in the existing literature regarding the effective management of extensive action spaces within the DNR problem, this chapter introduces an innovative approach for reducing the dimensions of action space. This method leverages a graph-theory-based procedure, known as the Yamada-Kataoka-Watanabe algorithm, in conjunction with powerflow analysis. Employing the DQN and dueling DQN methodologies, this approach trains agents to select optimal switching actions across a range of distribution systems. A comprehensive comparative analysis is conducted between these two advanced techniques and the conventional methodologies in the literature. Through this comparison, we elucidate the respective merits and drawbacks, as well as the optimality of the obtained solutions. The following are the key contributions of the second part of this thesis:

- Development of a novel method rooted in graph theory and power flow analysis, aimed at reducing the size of the action space which enhances the optimality and convergence of solutions generated by RL algorithms;
- Conducting a comprehensive sensitivity analysis of the action space dimensions, elucidating the impact of varying the size of the action space;
- Thorough comparative analysis between the proposed method and conventional DNR methods in terms of execution speed and optimality of the obtained solutions.

3.3 RL Foundations and Algorithms

This section focuses on the fundamental principles and key algorithms that form the basis of RL. It lays the groundwork by explaining the core concepts of RL, including the agent-environment interaction, states, actions, rewards, and the objective of learning optimal strategies. The section further explores various RL algorithms, providing insights into their workings, advantages, and applications.

3.3.1 RL Preliminaries

In a standard RL structure, an agent interacts with an environment E for a number of steps and receives an instant reward r_{t+1} for each action a_t that it performs inside E at time-step t. After performing the action a_t at time t according to state s_t, the environment arrives at a new state s_{t+1}. The process is illustrated in Fig. 3.1. The aim of the agent is to increase its cumulative reward over time. To use RL, it is necessary to create an environment that can be modeled as an MDP. This involves defining the action space $\mathcal{A}$, state space $\mathcal{S}$, reward function $\mathcal{R}$, transition probabilities $\mathcal{P}$, discount factor γ, and time horizon $\mathcal{T}$.

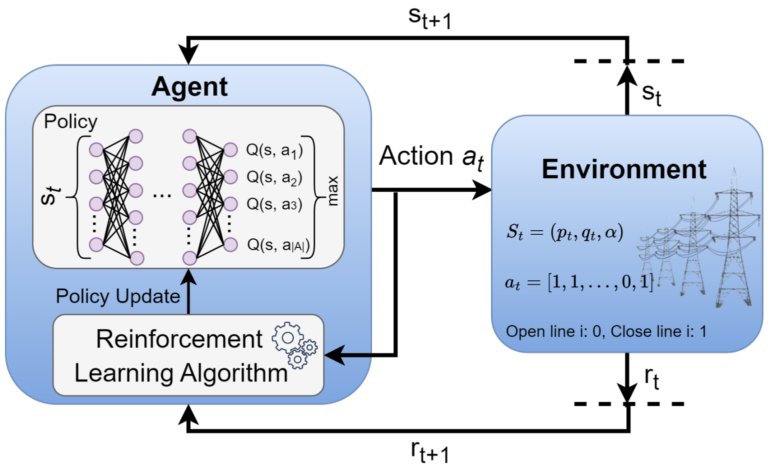

Figure 3.1: Schematic diagram showing how RL works

Based on the present state $s_t \in \mathcal{S}$, the agent chooses an action $a_t \in \mathcal{A}$. After the agent executes the selected action in the environment, it receives a reward r_{t+1} determined by the current state s_t and the action taken a_t. Subsequently, the agent transitions to a new state determined by the state transition probability function $\mathcal{P}$. The control process can either continue indefinitely with $\mathcal{T} \to \infty$, or it can end when the time step t reaches the final time $\mathcal{T}$. Finally, the agent determines the optimal control policy π that maximizes the expected discounted return G computed by taking the sum of the rewards obtained at each time step in episodic tasks and as $G_t = \sum_{j=0}^{\infty} \gamma^j r_{t+j+1}$ in continuous tasks. Bellman optimality equation is used in each state to find the optimal action value function denoted by $Q_\pi(s_t, a_t)$. In each state, the optimal policy is the one that maximizes the rewards, hence $\pi^* = \text{argmax}_a Q^*(s, a)$. However, at the beginning of the training, the optimal Q-values are unknown. In deep RL algorithms, the q-values are determined using a target neural network (Q^{targ}) and Bellman equation. Therefore, the policy neural network is updated by minimizing the following temporal difference error

$$\delta = Q_\pi(s_t, a_t) - (r(s_t, a_t) + \gamma \max_a Q^{targ}(s_{t+1}, a)). \tag{3.1}$$

3.3.2 RL Algorithms

The behavior policy refers to the policy adopted by the agent to make its action choices, while the target policy is the one that the agent aims to acquire through learning. In general, there are two types of RL algorithms: on-policy and off-policy. On-policy learning refers to a method where the behavior and target policies used by the agent are the same, whereas in off-policy learning, they are distinct. As a result, on-policy learning has a higher chance of being stuck in a suboptimal policy. Therefore, in this DNR study, two off-policy methods are implemented and tested: DQN and dueling DQN.

In DQN, the policy is determined by a neural network. The input of this neural network is the state of the agent and the output is the Q-value for each action in the action space. DQN overcomes unstable learning by using experience replay and target network. Using experience replay, previous transitions are stored in a memory as $(s_t, a_t, r_{t+1}, s_{t+1})$. Later, these transitions are sampled from the memory in mini-batches and used to update the neural network. This way, the correlation between experiences to update the neural network is reduced and catastrophic forgetting is prevented by reusing past transitions. Catastrophic forgetting or policy collapse is a common issue in policy gradient methods where the agent starts to forget a policy it had learned before.

Algorithm 5 briefly describes the general trainig process of RL algorithms. For a specified number of steps, an action is chosen by the policy and executed to get a full trajectory $(s_t, a_t, r_{t+1}, s_{t+1})$ (lines 3-5). The transitions are added to the memory and a batch of transitions are chosen from the memory to compute the loss and update the policy (lines 6-8).

Algorithm 5 RL Training Procedure

```
1 Input: Environment, $N$, learning rate $\alpha$, discount factor $\gamma$, batch size
2 Output: Optimal policy $\pi$
3 for $Step \in \{1, 2, ..., N\}$ do
4     Choose action $a_t$ based on policy
5     Perform the action $a_t$ in the environment
6     Append trajectory $(s_t, a_t, r_{t+1}, s_{t+1})$ into memory
7     Select a batch, $b$, of trajectories from the memory
8     Update the policy minimizing $Loss(\theta)$
      $y_i = \begin{cases} r_i & \textbf{If} \text{ terminal state} \\ r_i + \gamma \max(Q(s_i, a_i|\theta_{targ})) & \textbf{If} \text{ non-terminal state} \end{cases}$
      $Loss(\theta) = \frac{1}{|b|} \sum_{i=1}^{|b|} [y_i - Q(s_i, a_i|\theta)]^2$
```

Dueling DQN is an improved version of DQN. In some states, performing an action does not always increase the obtained reward. Therefore, in this method, it is unnecessary to know the value of each action at every time step instead, the state value and advantage function is used to find the optimal policy. The state value is the expected sum of discounted returns when starting from that state and is expressed as

$$V_\pi(s) = \mathbb{E}_\pi[\sum_{t=0}^{T-1} \gamma^t r_t | s_t = s]. \tag{3.2}$$

The difference between the action value and the state value is computed to obtain the advantage function, which can be expressed as

$$A_\pi(s, a) = Q_\pi(s, a) - V_\pi(s). \tag{3.3}$$

The advantage function is utilized to determine the relative benefit of choosing an action compared to other actions at a particular state. In dueling DQN method, one neural network with two streams is used to compute both the state value and advantage function. Then, the Q-value for the policy network is calculated according to

$$Q_\pi(s_t, a_t) = V(s_t) + (A(s_t, a_t) - \frac{1}{|A|} \sum_a A(s_t, a)). \tag{3.4}$$

Similarly to the DQN method, a target neural network (which is a copy of the policy neural network) is used to calculate the values of $V^{targ}(s_{t+1})$, $A^{targ}(s_{t+1}, a_{t+1})$, and eventually $Q^{targ}(s_t, a)$. The policy network is iteratively updated using the temporal difference error from Eq. 3.1.

3.4 Problem Formulation

In this section, the basics of RL are first discussed. Then, DNR problem is reformulated as an MDP. Finally, the powerflow-based action space sampling method is proposed and two RL algorithms for training the agents are presented.

3.4.1 System Modeling

In reconstructing the DNR as an MDP problem, the opening and closing of each line is considered an action. The system receives the action in binary format where a zero value shows an open line and one shows a closed line. In this study, an agent can open or close as many lines as needed in a single action. For example, changing the line states from [1, 1, 1, 0, 1, 0] to [1, 1, 1, 1, 0, 1] is considered one action. This form of representation allows avoiding loops or disconnections in the system while opening or closing lines. The state of the system shown as $s_t = [p_t, q_t, \alpha_t]$ is the real $p_t = [p_{1t}, p_{2t}, ..., p_{nt}]$ and reactive $q_t = [q_{1t}, q_{2t}, ..., q_{nt}]$ power consumption of the buses as well as the switch states $\alpha_t = [\alpha_{1t}, \alpha_{2t}, ..., \alpha_{mt}]$. Parameters n and m denote the number of busses and lines in the system, respectively. After each action, the time t is incremented by 1. The random process of power injections is assumed to represent the transition probability, $\mathcal{P}$, between time-steps.

Since the aim of DNR in this thesis is to minimize the total line losses, the reward function is defined as

$$\mathcal{R}(s_t, a_t) = -R^l p_t^l(s_{t+1}) - R^v(s_{t+1}). \tag{3.5}$$

In this equation, the penalty function for line losses in [1/kW] is represented by R^l, and $p_t^l(s_{t+1})$ denotes the total line losses at state s_{t+1}. The second term is defined as the penalty function $R^v(s_{t+1})$ for violating the voltage constraint of the system and is computed as

$$R^v(s_{t+1}) = \begin{cases} \lambda(\max(V_{t,n}) - 1) & \text{if } V^{\max} < \max(V_{t,n}) \\ \lambda(1 - \min(V_{t,n})) & \text{if } \min(V_{t,n}) < V^{\min} \\ 0 & \text{otherwise} \end{cases}. \tag{3.6}$$

Here, a significantly high value is assigned to the variable λ to represent a highly negative reward when an action causes voltage violation in the system. Other system constraints including radiality and load balance constraints are imposed while selecting the action space $\mathcal{A}$, as described in the following subsection.

3.4.2 Action Space Sampling

The original action space for DNR is extremely vast. If there are 33 lines in the system and all lines have switches to be opened and closed, the complete action space will contain 2^{33} actions. However, the majority of these actions are not practical to be executed in an actual distribution system because they can cause loops or disconnections in the network. Therefore, a graph-theory-based algorithm named Yamada-Kataoka-Watanabe [107] is used in this study to form the action space for RL by only selecting the feasible actions. The Yamada-Kataoka-Watanabe algorithm can identify every minimum spanning tree present in an undirected graph. All power networks can be considered undirected graphs where each bus is a node and the power lines are edges. A minimum spanning tree refers to a selection of edges from a graph, which connects all nodes without forming any cycles. Therefore, using this algorithm, a set of all reconfigurations and actions that cause no disconnections and loops in the power system can be found. This set is called feasible action space.

The feasible action space for power networks can still be fairly large. Therefore, a rational action space sampling method is needed to only select the most optimal actions as the final action space. For instance, consider the schematic diagram of IEEE 33-node distribution system [112] depicted in Fig. 3.2. Although opening the lines 2-19, 9-10, 15-14, 25-29, 32-22, and closing the line 7-8 and the rest of the lines is a feasible reconfiguration action, it cannot be considered an optimal action since it would open a line very close to the substation and would put too much load on other lines increasing the system loss.

To address the this issue, a new powerflow-based action space sampling method is developed in this thesis to only select the most optimal actions as the DNR action space. At the same time, this reduces the action space size and further improves the convergence of the RL algorithm. In this method, a powerflow is performed first, considering the standard load values and assuming that all lines are closed. Then, active power losses of each line are calculated. This is shown by red numbers in Fig. 3.2. Finally, the total system losses for each of the feasible actions are calculated and the actions are sorted based on the system loss. The most optimal actions are the ones with higher calculated system loss. This is because, according to Fig. 3.2, lines near main substation carry more current and have more active power losses. Therefore, disconnecting these lines will cause more reduction in overall system losses. As a result, the actions with higher calculated losses disconnect lines further away from the substation and are more optimal.

3.5 Simulation and Results

To verify the effectiveness of the proposed method, it is applied to solve DNR in 33-, 119-, and 136-node test systems. Section 3.5.1 presents the parameters of the algorithms used for the simulations. Section 3.5.2 presents the results of the action

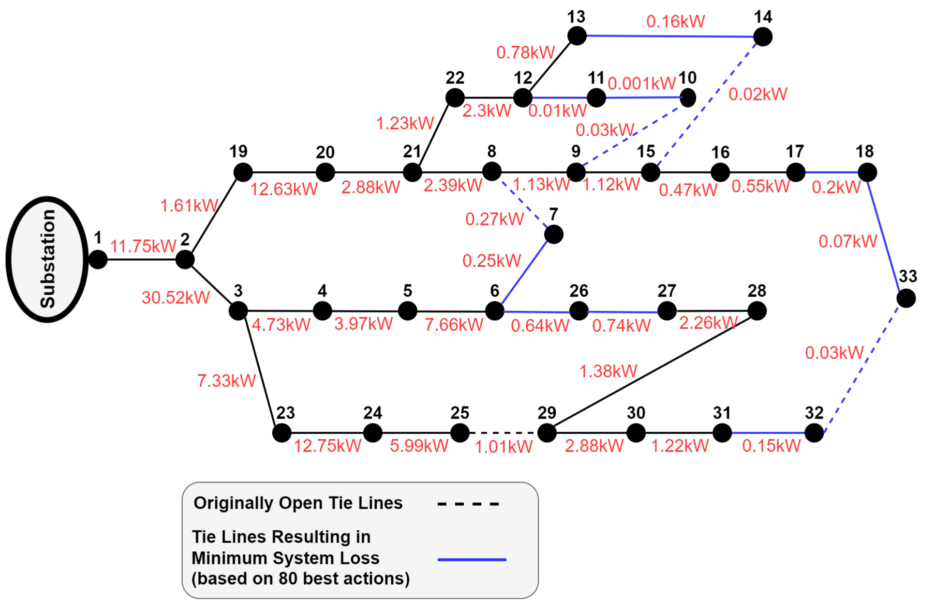
Substation
Originally Open Tie Lines
Tie Lines Resulting in Minimum System Loss (based on 80 best actions)

Figure 3.2: Active power losses for a fully connected 33-node test system

space sampling. Sections 3.5.3 to 3.5.5 describe the results obtained by applying the proposed method on test systems. Sensitivity analysis is discussed in Section 3.5.6. Finally, the proposed method is compared with other studies in Section 3.5.7.

3.5.1 Experimental Setup and Data

The lower and upper limits for the voltage levels in the constraint 3.6 are defined as 0.95 p.u. and 1.05 p.u., respectively. Based on empirical performance, the value of parameter R^l is assigned as 10,000 and λ as 1,000,000. The pandapower package in Python is used to create a model of the power network, and the Newton-Raphson method to perform power flow calculations. The graph analysis is performed using the NetworkX package and the Yamada-Kataoka-Watanabe algorithm. Hourly load data for 3000 days is generated by assuming a normal distribution with a standard deviation of 15% to the standard 33-, 119-, and 136-bus network loads. The tuned parameters of the DQN and Dueling DQN algorithms are given in Table 3.1.

Table 3.1: DQN and dueling DQN parameters

Parameters	Values
learning rate (α)	0.001
batch size (b)	512
discount factor (γ)	0.99
neural network structure	$\{1024, 1024\}$
experience replay memory size	$30,000,000$

3.5.2 Action Space Sampling

When applying the Yamada-Kataoka-Watanabe algorithm to the 33-node test system, 50,751 feasible actions are found. Applying the same algorithm to the 119- and 136-node test systems found more than 3,500,000 feasible actions. However, to prevent the memory overflow problem, the number of feasible actions is restricted to 3,500,000. Using the proposed action space sampling methodology, the sorted values of losses for the 33-node system are depicted in Fig. 3.3. The top illustration presents the sorted active power loss for all the feasible actions in the 33-node system. On the other hand, the lower illustration presents an enlarged perspective of the aforementioned upper illustration, focusing on a subset of 2000 actions from the right side. Since the active power reduction slope decreases after 300 actions, the top 300 actions are chosen as the optimal actions for the 33-node system.

The sorted values of active power losses for different actions in the 119- and 136-node test systems are illustrated in Figs. 3.4 and 3.5, respectively. Similarly, the top 600 actions are selected as the most optimal actions to form the action space for the RL algorithms in the 119- and 136-node systems, as it provides a trade-off between loss reduction and the action space size. It is important to note that a comprehensive sensitivity analysis on the action space size is conducted in Section 3.5.6.

This study makes a significant contribution by introducing a new action space sampling method, which greatly improves the efficacy of RL agents in achieving

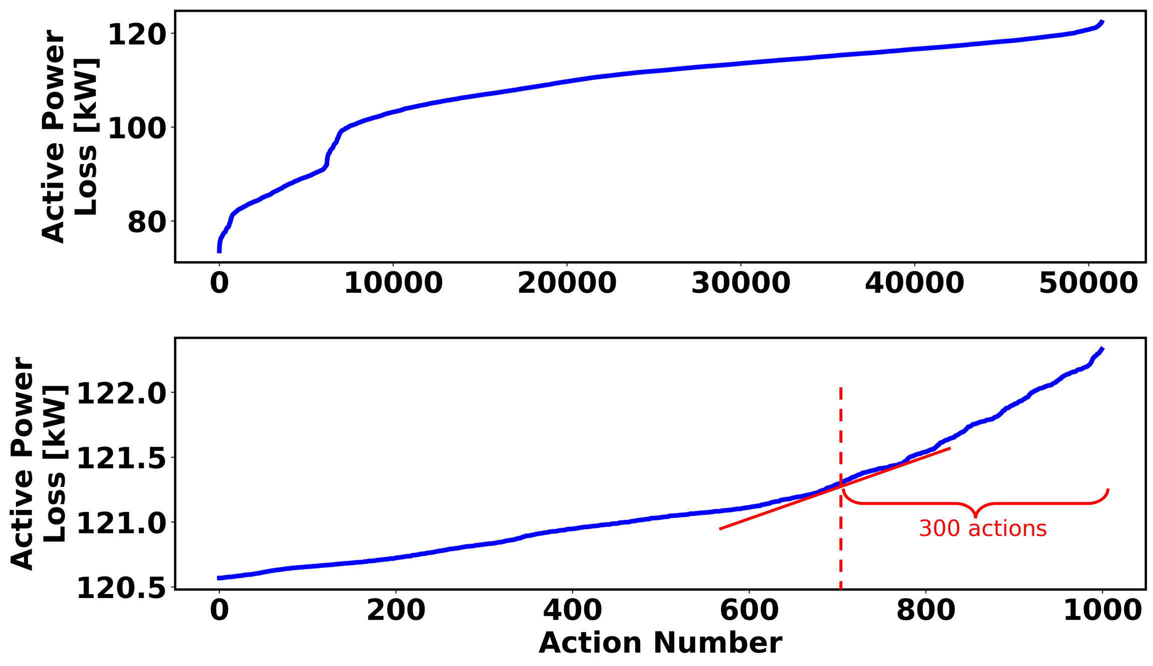

Figure 3.3: 33-node system sorted active power loss for 50,751 actions (top) and 1000 actions with maximum loss (bottom)

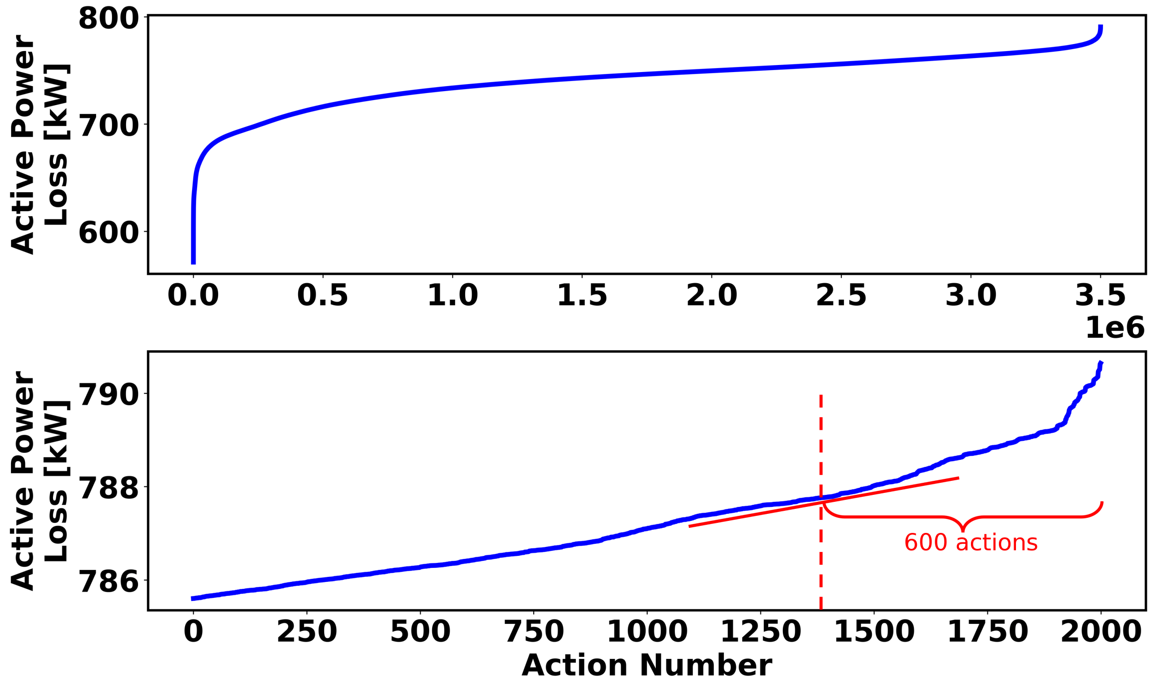

Figure 3.4: 119-node system sorted active power loss for 3,500,000 actions (top) and 2000 actions with maximum loss (bottom)

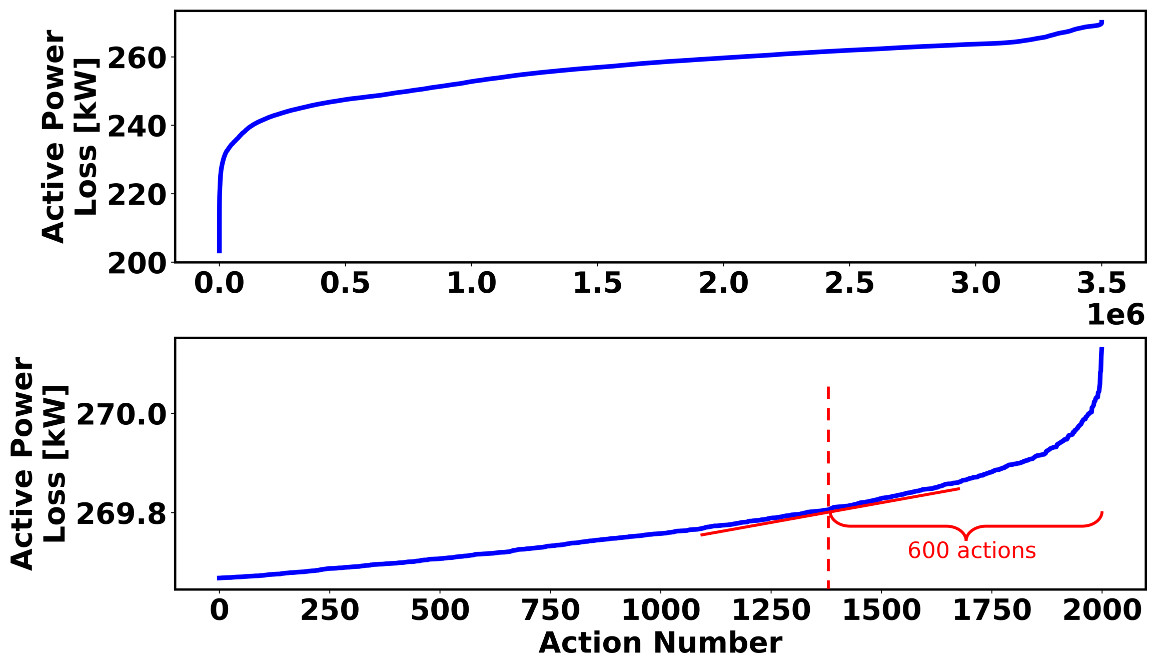

Figure 3.5: 136-node system sorted active power loss for 3,500,000 actions (top) and 2000 actions with maximum loss (bottom)

optimal solutions. Additionally, by careful hyperparameter tuning, factors such as the balance between exploration and exploitation, neural network size for function approximation, and learning rate are adjusted to ensure optimal rewards and solutions. In essence, ensuring the accuracy of DQN and dueling DQN involves a holistic approach that encompasses comprehension of their theoretical foundations, thorough analysis of components like function approximation and experience replay, strategic hyperparameter optimization, and rigorous empirical assessments to validate their performance across diverse tasks.

3.5.3 33-Node Test System

The IEEE 33-node test system is a very popular network for power system studies. This network is operated at 12.66 kV and has a total demand of 22,709 MW and 17,041 MVAr. Given the persistent interaction of RL agent with the environment spanning over 50,000 iterations, the obtained rewards tend to exhibit fluctuations when directly visualized. Therefore, a good strategy for assessing the progression of reward accumulation is to calculate the mean value of daily rewards. Additionally, to effectively represent the upward trend of these rewards, it is conventional in the field of RL to employ a technique known as the ”n-step moving average” of rewards. Applying these techniques to the 33-node system, Fig. 3.6 shows the average daily rewards and its 60-step moving average for the DQN and dueling DQN algorithms as they train and converge. When the obtained rewards stabilize, it is an indication of convergence of the RL algorithm and the training can be stopped. In the visualized figures, the higher the obtained rewards, the better the algorithm has performed. It appears from these figures that the RL agent can find a control policy that minimizes system losses. Therefore, the obtained rewards increase over time. In addition, DQN converges faster than dueling DQN.

The small fluctuations after convergence are mainly due to load fluctuations at different hours and the exploration property of the algorithms. Table 3.2 presents the DNR solution obtained using DQN and dueling DQN in the 33-node system for a single hour. The tie switches represent the open lines.

Table 3.2: Tie switches of the DNR result for a single hour in 33-node system

Algorithm	**Tie Switches**	**System Loss [kW]**
DQN	7-8, 10-11, 14-15, 28-29, 32-33	140.71
Dueling DQN	7-8, 9-10, 14-15, 28-29, 32-33	139.98

3.5.4 119-Node Test System

To demonstrate the scalability of the proposed method, it is applied to a large-scale 119-node test system [92]. The nominal voltage of this network is 11 kV, and the total demand is 22,709 MW and 17,041 MVAr. The average daily rewards using DQN and its 60-step moving average are depicted in Fig. 3.7. These curves show that the agent learns to choose better actions and maximize rewards over time. Also, the convergence

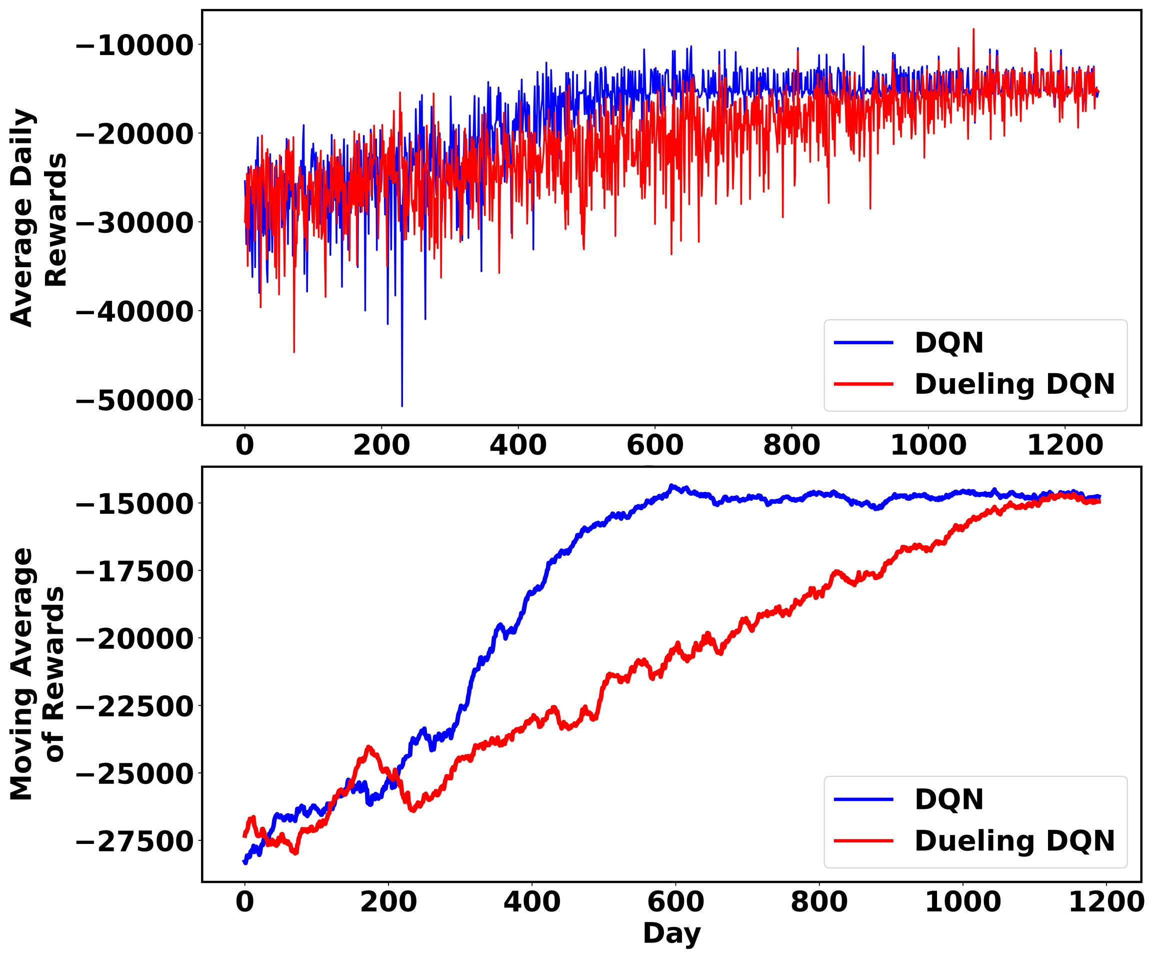

Figure 3.6: The mean daily rewards (top) and its 60-step moving average (bottom) for 33-node system during training

time is acceptable.

The DNR result for a single hour is given in Table 3.3. Overall, the results show that the proposed method performs well for a large-scale distribution system and that the convergence time is at an acceptable level.

3.5.5 136-Node Test System

To demonstrate the applicability of the proposed method on a real test system, it is tested on a 136-node system in the Midwest of Brazil [94]. This network has a relatively complex structure and operates at 13.8 kV. Figure 3.8 shows the average daily rewards and its moving average during training in the 136-node test system.

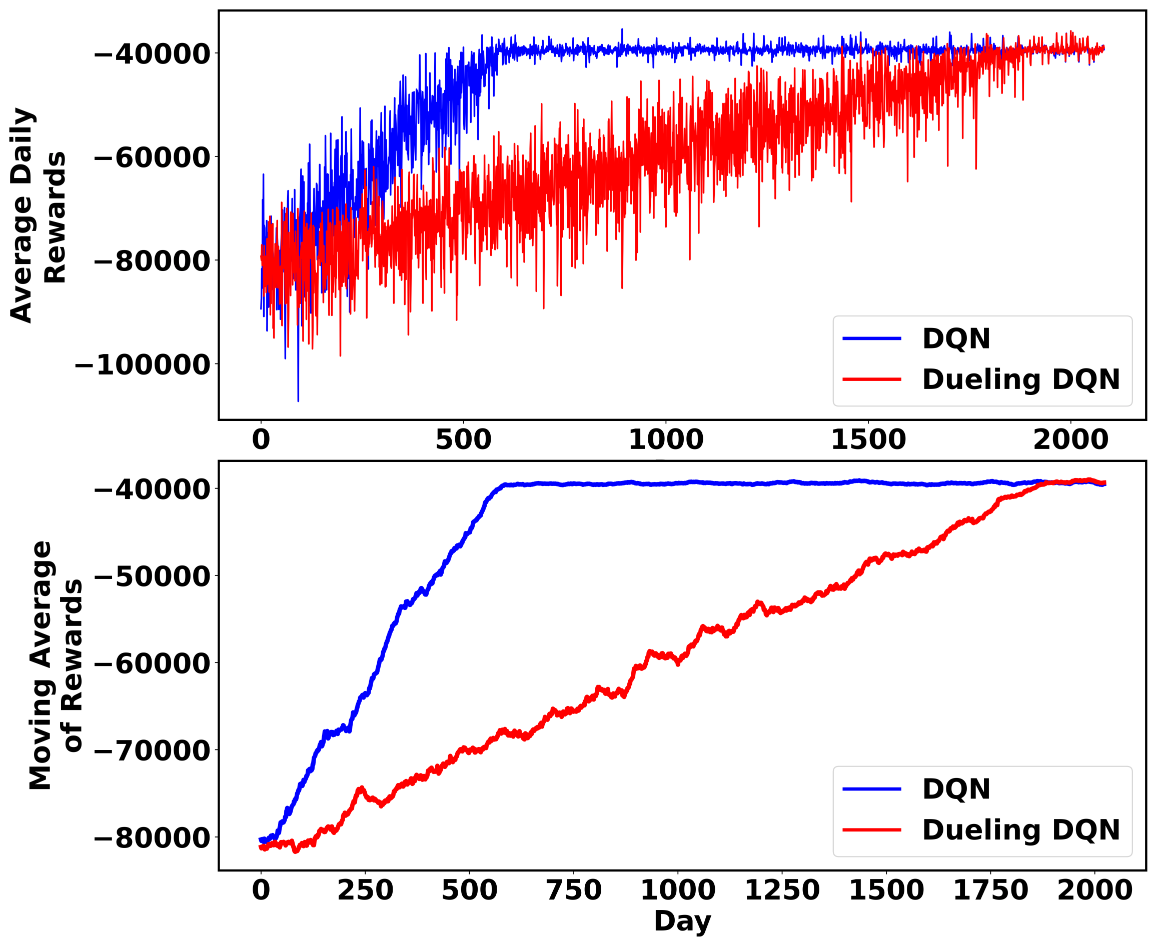

Figure 3.7: The mean daily rewards (top) and its 60-step moving average (bottom) for 119-node system during training

Table 3.3: Tie switches of the DNR result for a single hour in 119-node system

Algorithm	Tie Switches	System Loss [kW]
DQN	23-24, 32-33, 72-73, 109-110, 46-27, 17-27, 54-43, 62-49, 37-62, 9-40, 58-96, 88-75, 99-77, 108-83, 105-86	1025.98
Dueling DQN	20-21, 34-35, 72-73, 109-110, 46-27, 17-27, 54-43, 62-49, 37-62, 9-40, 58-96, 88-75, 99-77, 108-83, 105-86	1015.04

Similar to the other test cases, rewards increase over time, indicating agent learning, and DQN converges faster than dueling DQN. DNR results for a single operating hour are given in Table 3.4.

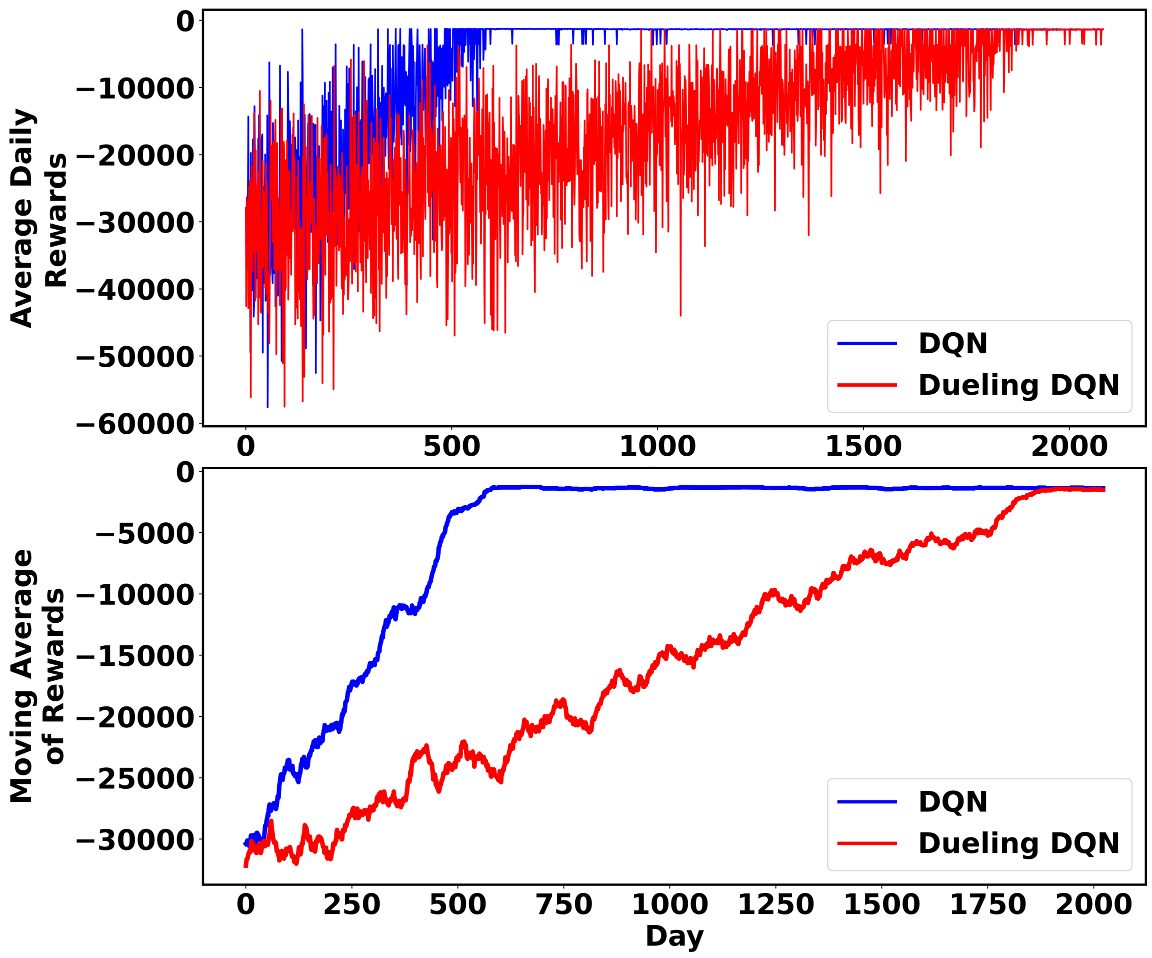

Figure 3.8: The mean daily rewards (top) and its 60-step moving average (bottom) for 136-node system during training

3.5.6 Sensitivity Analysis

To investigate the effect of selecting a different number of actions on the results, simulations are repeated with various sizes of action sets. The results are presented in Figs. 3.9–3.11. It can be observed from Fig. 3.9 that increasing the action space size from 200 to 500 does not have much impact on the rewards obtained in the 33-node

Table 3.4: Tie switches of the DNR result for a single hour in 136-node system

Algorithm	Tie Switches	System Loss [kW]
DQN	47-62, 89-90, 105-106, 134-135, 46-27, 7-73, 9-24, 15-83, 25-51, 50-96, 55-98, 66-79, 79-131, 84-135, 91-104, 90-129, 92-104, 92-132, 96-120, 126-76, 128-77, 135-98	288.84
Dueling DQN	31-35, 48-51, 89-90, 106-107, 7-73, 9-24, 15-83, 50-96, 55-98, 62-120, 66-79, 79-131, 84-135, 91-104, 90-129, 92-104, 92-132, 96-120, 126-76, 128-77, 135-98	294.33

system. However, choosing a much smaller action space (e.g. 100) significantly reduces the obtained rewards. Therefore, it can be concluded that the most optimal actions are within the first batch of 200 actions.

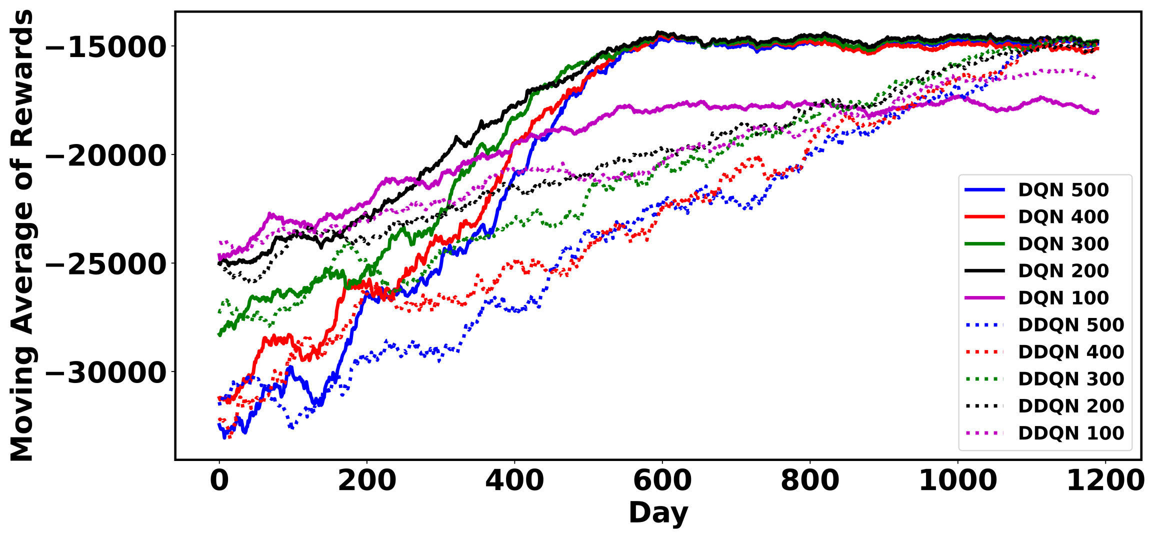

Figure 3.9: 60-step moving average of daily rewards for 33-node system with various number of actions

In the 119-node test system, action space sizes of 200, 400, 600, and 800 obtain similar rewards. However, increasing the size to 1000, significantly improves the obtained rewards. This indicates that the most optimal actions lie within the initial batch of 1000 actions. In the 136-node system, action-space sizes of 200–1000 all obtain similar rewards, since the best actions are within the first batch of 200 actions.

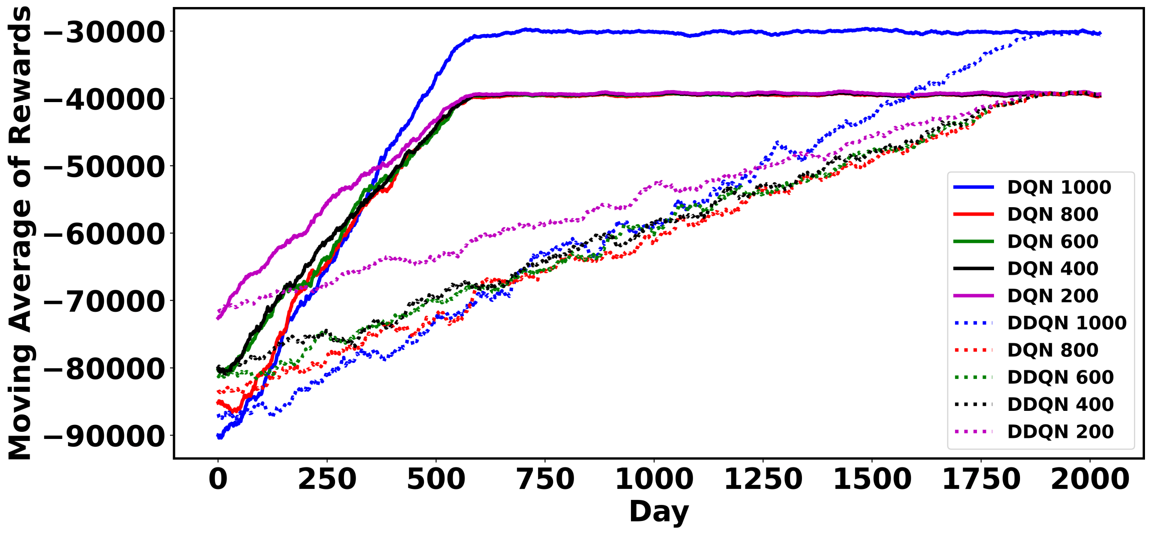

Figure 3.10: 60-step moving average of daily rewards for 119-node system with various number of actions

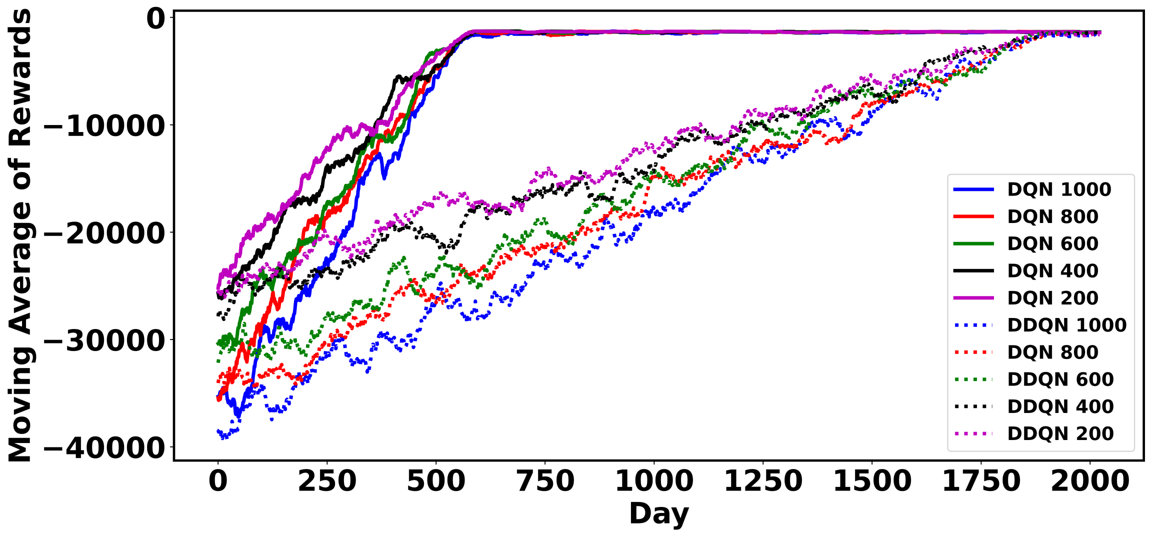

Figure 3.11: 60-step moving average of daily rewards for 136-node system with various number of actions

To show the robustness of the proposed method under extreme load fluctuation events, the load values are intentionally changed to significantly deviate from the average load on which the RL algorithms were trained. For this purpose, the standard deviation of these loads is set at 35% of the previous average loads. The system loss and minimum system voltage in this case are shown in Figs. 3.12–3.14 for the 33-, 119- and 136-node systems, respectively. Comparing the obtained results with the base case where there is no DNR in the system reveals that, using DQN and dueling DQN, the loss is lower and the voltage is higher, and the agent chooses the actions that minimize the system loss and improve the voltage profile. In the 119- and 136-node test systems, the dueling DQN sometimes chooses actions that are not optimal. Therefore, it can be concluded that DQN is more robust to unforeseen states or extreme events.

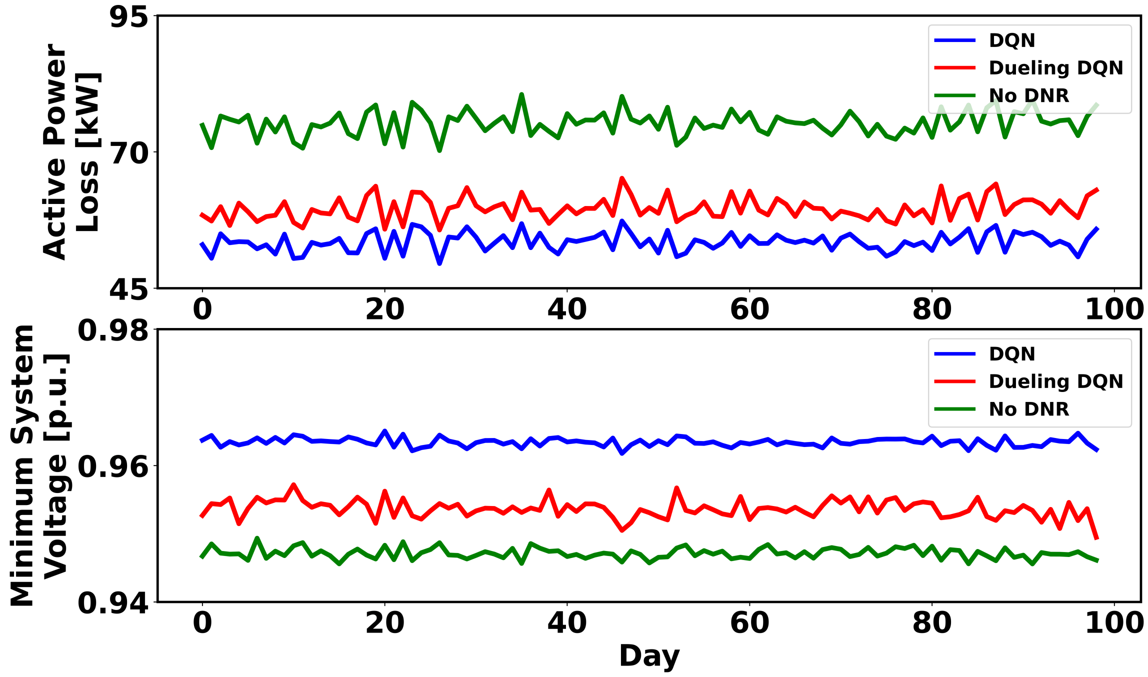

Figure 3.12: RL agent's response to unforeseen loads in the 33-node system

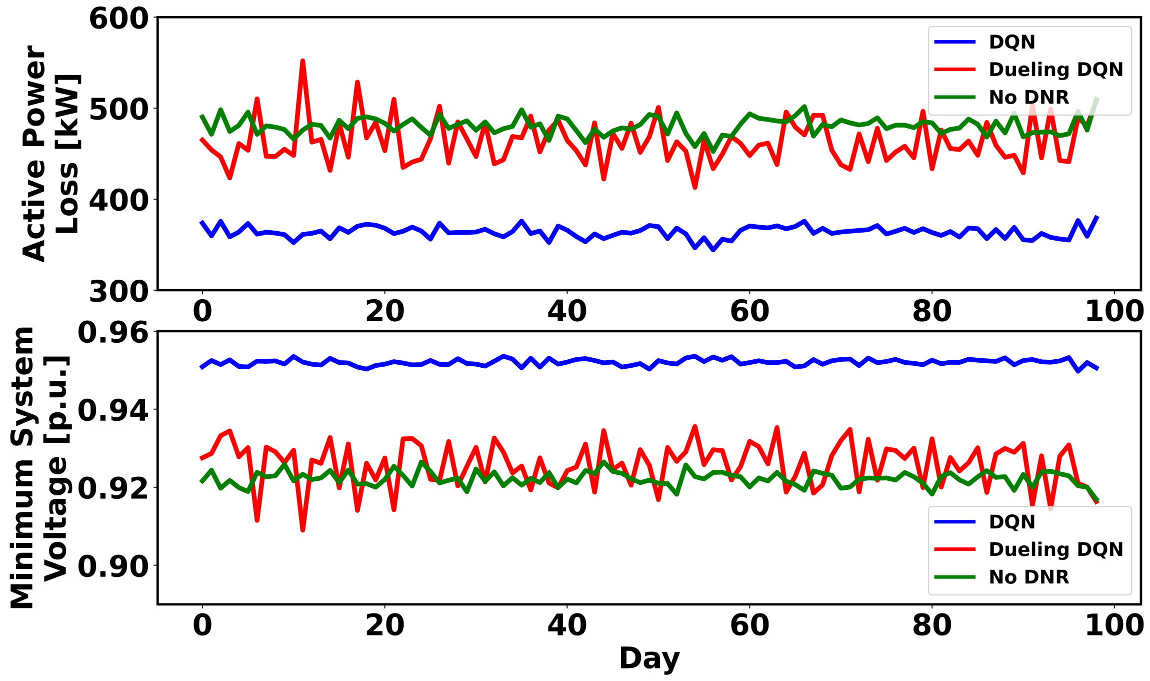

Figure 3.13: RL agent's response to unforeseen loads in the 119-node system

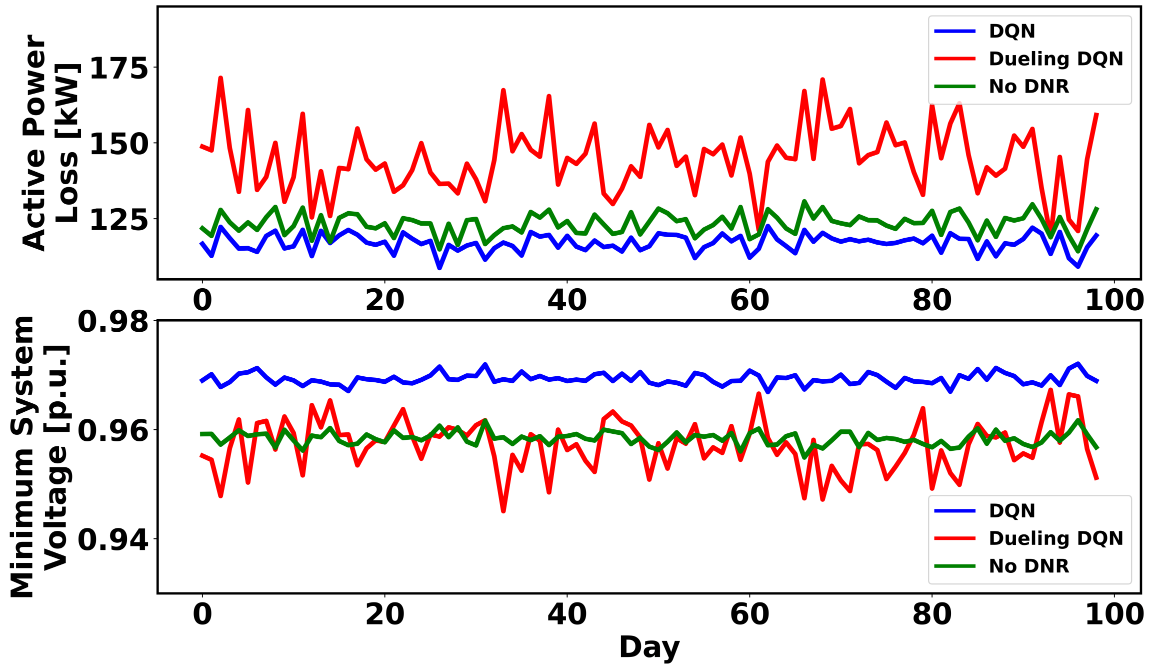

Figure 3.14: RL agent's response to unforeseen loads in the 136-node system

3.5.7 Comparative Study

In this section, the computational speed of the proposed method and the optimality of its results are compared with published results obtained using the switch opening and exchange (SOE) [112], mathematical programming (MP) methods (which encompass mixed-integer linear and mixed-integer nonlinear programming) [112], adaptive ant colony optimization (AACO) method [88], GA [12], MIQP [3], and two RL methods including BCSAC [25] and NoisyNet DQN [95]. The DNR execution times of all these methods are given in Table 3.5. As can be inferred from this table, the execution times for DQN and dueling DQN are quite similar, and both are significantly faster compared to other non-RL methods in the 136-node system, with a speed improvement factor of at least twenty. The efficiency of the RL algorithms becomes even more important as the system size increases. The offline training times for DQN and dueling DQN algorithms are given in the last row of the same table. Training was carried out on a system with an Intel i9-9820X 3.30GHz CPU and 100 GB of RAM.

Table 3.5: Computational Speed Comparison

Algorithm	Execution Time [s]		
	33 bus	119 bus	136 bus
DQN	0.21	0.88	0.78
Dueling DQN	0.21	0.88	0.79
BCSAC [25]	0.4	2.2	-
NoisyNet DQN [95]	0.48	–	–
SOE [112]	1	12.3	16.9
MIQP [3]	3.2	39.4	188.4
AACO [88]	0.3	430.74	894.20
MP [112]	19	4007	1236
Approximate RL Training Time	15145	45454	53000

Finally, the optimality of the results obtained by the proposed method is compared with other published results in Table 3.6. As can be seen, the results for the small 33-node system are within only 0.3% optimality gap of the best solution presented in the literature and the maximum optimality gap that occurs for the 119-node system is only 10.58%. In heuristic methodologies, adherence to system constraints is ensured by imposing large penalties when they are violated. As a consequence, these methods occasionally find solutions that effectively minimize the objective function; nonetheless, their feasibility may be compromised. Such challenges are often encountered in solutions documented in the existing literature. Conversely, the results obtained by the approach proposed in this study do not exhibit any constraint violations.

Table 3.6: Loss and Minimum Voltage Comparison by Different Methods

Algorithm	**Test System**					
	33-node		119-node		136-node	
	loss	voltage	loss	voltage	loss	voltage
DQN	140.70	0.94	958.90	0.93	285.34	0.96
Dueling DQN	139.98	0.94	954.62	0.93	286.32	0.96
SOE [112]	139.55	0.94	853.58	0.93	280.94	0.96
MIQP [3]	139.55	0.94	871.23	0.93	261.97	0.96
AACO [88]	139.55	0.94	865.86	0.93	279.75	0.96
MP [112]	139.55	-	853.58	-	280.14	-
No DNR	202.68	0.91	1298.09	0.87	320.36	0.93
Optimality Gap	0.3%		10.58%		1.95%	

3.6 Conclusion

This study employed a Markov decision process as a modeling technique for the DNR problem and leveraged two RL algorithms, namely DQN and dueling DQN, to tackle the problem. First, the feasible action space for DNR was obtained using the Yamada-Kataoka-Watanabe algorithm. Then, a powerflow-based action sampling method was applied to reduce the size of the action space. This was done to facilitate faster convergence of the RL algorithms and optimize the results. The proposed method was applied to 33-, 119-, and 136-node test systems to demonstrate its scalability.

The results indicated that the learning performances of DQN and dueling DQN are very similar. However, DQN has faster convergence. In addition, comparison with conventional methods proved that the proposed RL-based method is at least 20-times faster and that the obtained results are within a maximum 10.58% optimality gap of the best solution presented in the literature. This demonstrated that the proposed powerflow-based action space sampling method was effective and it only chose the most optimal actions. Future studies will focus on implementing the same method for semi-open lines to increase the electric vehicle hosting capacity and balance network loads.

Chapter 4

Explainable Reinforcement Learning for Distribution Network Reconfiguration

4.1 Abstract

Reinforcement learning methods have been successful in tackling a wide variety of control problems, owing to their remarkable ability to make fast decisions and learn effectively from data. Nevertheless, the lack of transparency regarding the decision-making process has resulted in a significant dearth of trust towards these models, subsequently limiting their utilization in critical decision-making applications. The use of reinforcement learning in distribution network reconfiguration is an inherently sensitive application due to the need to change the states of the switches, which can significantly impact the lifespan of the switches. Consequently, executing this process requires meticulous and deliberate consideration. This study presents a new methodology to analyze and elucidate reinforcement learning-based decisions in distribution network reconfiguration. The proposed approach involves the training of an explainer neural network based on the decisions of the reinforcement learning agent. The explainer network receives as input the active and reactive power of the buses at each hour and outputs the line states determined by the agent. To delve deeper into the inner workings of the explainer network, attribution methods are employed. These

techniques facilitate the examination of the intricate relationship between the inputs and outputs of the network, offering valuable insights into the agent's decision-making process. The efficacy of this novel approach is demonstrated through its application to both the 33- and 136-bus test systems, and the obtained results are presented.

4.2 Introduction

In the realm of machine learning, particularly in the context of reinforcement learning (RL), a significant concern arises from the opacity of decision-making processes. It is not uncommon for individuals to harbor reservations and a lack of trust in machine learning models, largely due to the inherent difficulty of understanding how and why these models arrive at their decisions. The complexity of these algorithms often creates a black box effect, leaving users and stakeholders questioning the reliability and fairness of the results [18]. Consequently, there is a growing imperative to demystify and shed light on these methods, as explaining the inner workings of machine learning models can significantly improve their trustworthiness and acceptance in critical domains [36].

To this end, Bellotti et al. [8] develop a framework consisting of three types of data analysis (including episode timeline, frame by frame, and aggregated statistical analysis) to quantitatively interpret the actions of an autonomous driving agent trained through deep RL in a highway-env simulation environment. Yun et al. [111] introduce an explainable multi-agent deep RL method, called decomposed multi-agent deep Q-network (DMADQN), which utilizes an analytical manufacturing system model to decompose the energy management objective and production requirement to the agent level. This results in interpretable actions and reduced energy costs in controlling a section of an automotive assembly line. Policy-guided RL is used by Yang et al. [108] to interpret the intent in multi-turn dialogue by identifying interpretable paths of inference based on specific characteristics of the dialogue text. To explain the actions of the RL agent in power system emergency control, Zhang et al. [113] propose a backpropagation deep explainer based on Shapley additive explanations. It quantifies the importance of input features and incorporates feature classification and probabilistic analysis for improved clarity. Gjærum et al. [30] examine and analyze various methods for constructing model trees, which serve as surrogate models to explain real-time decision-making processes of deep RL agents in robotic tasks.

In addition to these studies, explainable RL has been used for various other applications. One such application is mixed-mode ventilation in the tropics, where it helps uncover novel insights for enhanced ventilation control [17]. Furthermore, explainable RL has found application in recommendation systems by providing tangible paths and the underlying reasoning behind the recommendations [89]. Finally, it has been employed to explain the rationale behind RL decisions in automating docking procedures on fully actuated vessels [61], autonomous navigation for unmanned aerial vehicles [33], and the behavior of dopamine neurons [9].

RL has emerged as a new technique for solving complex optimization problems, such as distribution network reconfiguration (DNR) [41, 115]. It has proven to be beneficial in this domain due to its speed, model-free nature, and ability to provide near-optimal solutions. DNR is known to be an NP-hard mixed integer nonlinear problem [117], making it particularly challenging to solve for traditional optimization methods. These methods often struggle to find good solutions and suffer from slow computation times. In contrast, RL leverages its ability to learn and adapt through trial and error, enabling it to efficiently explore the problem space and discover effective reconfiguration strategies. By leveraging its speed and model-free nature, RL offers a promising avenue for addressing the challenges posed by DNR and obtaining high-quality solutions. To this end, the batch-constrained soft actor-critic RL algorithm is employed by Gao et al. [25] to learn DNR strategies from a finite historical operational dataset. The DNR problem has also been addressed using deep Q-learning (DQN) [52] and NoisyNet DQN [95]. An RL-based online DNR scheme is developed by Oh et al. [73] which alters the network's configuration to adjust power flow and address reliability issues in self-sufficient distribution networks with high variability in distributed renewable energy. A multi-agent soft actor-critic method is used by Wu et al. [104] to reconfigure distribution networks during extreme events, enabling critical service restoration through the formation of isolated sections within the distribution network. Other studies include an RL-based multilevel dynamic reconfiguration method in a cloud-edge collaboration architecture [46], a three-stage deep RL-based method to find the optimal reconfiguration and set points of distributed generators in real-time operation [10], and a DQN-based method to increase the reliability of the distribution network by reducing the failure rate [63].

Although numerous studies have explored the application of RL in DNR scenarios, an intriguing research gap exists: no previous study has presented a comprehensive approach to investigate the underlying logic behind RL agents' decision-making in DNR. To bridge this gap, this study introduces a novel methodology that involves generating a dataset based on the decisions made by the RL agent, which is then used to train an additional neural network called the explainer network. The explainer network takes the system state as its input and outputs the agent's decisions. In the next step, three distinct attribution methods, named Gradient*Input, Occlusion analysis, and Smoothgrad, are used to explain the relationship between the inputs and outputs of this explainer network. Using this approach, a deeper understanding can be achieved of the decision-making logic behind RL agents performing DNR. The main contributions of this study are as follows:

- Formulation, for the first time, of a new approach to explain actions of RL agents in various DNR scenarios;
- Development of an explainer network trained on RL agents' actions and corresponding states of the system;
- Examination of three different attribution methods to find the relationship between the explainer network's inputs and outputs, and to explain RL agents' actions.

This chapter is organized in four sections. Section 4.3 formualtes the addressed problem. The simulation results are presented and discussed in Section 4.4. Concluding remarks are given in Section 4.5, along with possible directions for future work.

4.3 Problem Formulation

This section presents a comprehensive description of the proposed method, offering a detailed explanation of the actions taken by the DNR agent. Furthermore, it provides a concise, yet informative overview of the methodologies employed throughout the process.

4.3.1 Foundations of RL in DNR

Every RL problem consists of an agent and an environment. The agent receives a reward for every action it performs in the environment, which is an indication of how good that action was. The agent's goal is to select actions that result in the highest rewards. In the DNR problem, the actions are the opening and closing of various lines, and the reward is the negative of line losses. This process results in the minimization of line losses. In addition, the agent receives a highly negative reward for violating any of the system constraints such as the voltage and demand balance constraints. The agent selects its actions based on the system state. In the DNR problem, the state of the system consists of the active and reactive power of the buses and the state of the lines. The DNR agents used in this study are trained using the DQN algorithm as it was concluded in [26] that DQN algorithm exhibits superior performance in DNR tasks.

4.3.2 General Structure

To provide a comprehensive understanding of the DNR agent's actions, a meticulous process is followed. It begins with the creation of a dataset that encompasses the active and reactive power of the buses, together with the configuration suggested by the DNR agent for the specific loading condition. Subsequently, a deep neural network is trained using the active and reactive power of the buses as input and the line states as the desired output. It should be noted that certain lines within the system remain unchanged (e.g., the first line after the substation) and, therefore, they are deliberately excluded from the output. This neural network is called the *explainer network* throughout this study. For a visual representation of the proposed neural network, refer to Fig. 4.1. When this neural network is trained, the intricate relationship between the active and reactive power of the buses and the proposed configuration of the DNR agent is revealed.

It is important to highlight that the explainer network's structure in this study significantly differs from that of the RL agent's deep neural network. In the RL agent's network, the output consists of action values, each corresponding to a specific action within a predefined action space. However, these actions are defined in a closed format, restricting direct access to the individual open/close states of each line involved in an action. For instance, in 'action 1', lines 33 and 36 are opened, while the remaining lines are closed, but the specific states of each line within this action cannot be individually accessed or modified. This limitation makes the RL agent's deep neural network unsuitable to be utilized as an explainer network in this study.

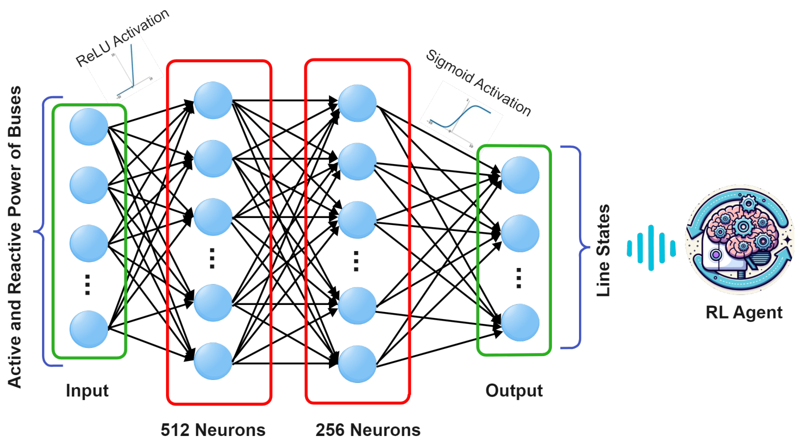

Figure 4.1: Structure of the explainer network

Attribution methods are considered among the most crucial explanation techniques for machine learning models. These methods play an important role in assigning attribution scores to individual input dimensions. These scores determine the contribution of each input element towards generating a specific output. By providing insights into the relative importance of different inputs, attribution methods help understand the intricate workings of machine learning models. Three of the most commonly used attribution methods are Gradient*Input, Occlusion analysis, and Smoothgrad methods.

After successfully training the explainer network, the mentioned attribution methods are leveraged to elucidate the relationship between the line states and the active and reactive power input. This comprehensive analysis uncovers the specific dependencies between line states and the active/reactive power originating from individual buses. The application of these attribution methods yields attribution scores, which can be effectively visualized using a heatmap, facilitating a more intuitive comprehension of the results. The workflow outlining this innovative approach is depicted in fig. 4.2.

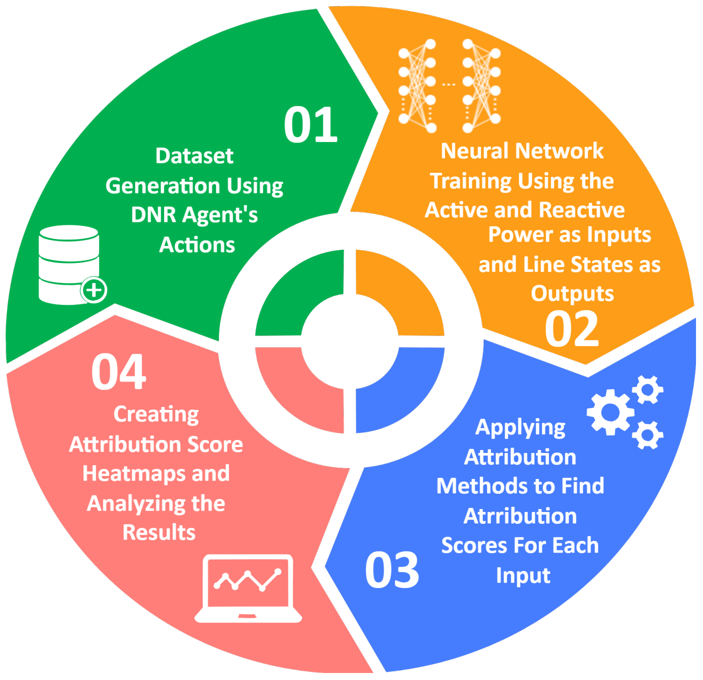
01
Dataset Generation Using DNR Agent's Actions
Neural Network Training Using the Active and Reactive Power as Inputs and Line States as Outputs
02
Applying Attribution Methods to Find Atrribution Scores For Each Input
03
04
Creating Attribution Score Heatmaps and Analyzing the Results

Figure 4.2: Steps of the proposed approach for explainable RL in DNR

4.3.3 Attribution Methods

Gradient*Input method is a fast and simple method to explain neural network models that are differentiable. Every differentiable function has a gradient which is a vector composed of the partial derivatives of the function with respect to each of its dimensions. The gradient provides information about how this function changes at any point when a small step is taken in any direction within the d-dimensional input space. Additionally, it indicates the direction in which the function will experience the greatest increase. Thus, when the gradient of the neural network model to a specific input is high, it means that small changes in that input will result in large changes in the output. In the Gradient*Input method, the gradient of the output with respect to each of the inputs is multiplied by the input value. This way, the gradient shows the importance of each input and the input value represents how strongly this input is expressed for a given data.

To find the importance of each input in Occlusion analysis, the model is assessed by excluding that specific input and noting the resulting alterations in the output. However, in machine learning, the models are trained over a specific data distribution, therefore, the changes in the input data distribution might result in an arbitrary output. As a result, in this study, different levels of change in input values are analyzed.

The Smoothgrad method aims to provide a smoother and more interpretable visualization of feature importance by reducing noise and emphasizing salient regions. In this method, the input data are perturbed multiple times by adding noise, and the gradients of the output with respect to the perturbed input are recorded and averaged over the number of perturbations. The averaging process reduces random noise and highlights consistent patterns. As can be inferred, this method has two important parameters which are the noise value and the number of samples. According to [84], 10%-20% noise level and a sample size of around 50 shows the best performance.

4.4 Results and Discussion

The proposed method was successfully applied to the 33- and 136-bus systems. However, to ensure clarity and a thorough understanding of the results, a more detailed explanation is provided for the 33-bus system. A schematic diagram of this system with labeled line numbers is shown in Fig. 4.3. This focused examination sheds light on the specific nuances, patterns, and implications that emerged during the analysis of this system. By emphasizing the findings of this particular system, we aim to facilitate a more comprehensive understanding of the capabilities of the proposed method and the insights gained from its application. The explainer network used in both systems consists of two hidden layers with 512 neurons in the first layer and 256 neurons in the second. The learning rate of Adam optimizer in both networks is 10^{-5}. The activation function for the last layer is Sigmoid while for other layers it is ReLU.

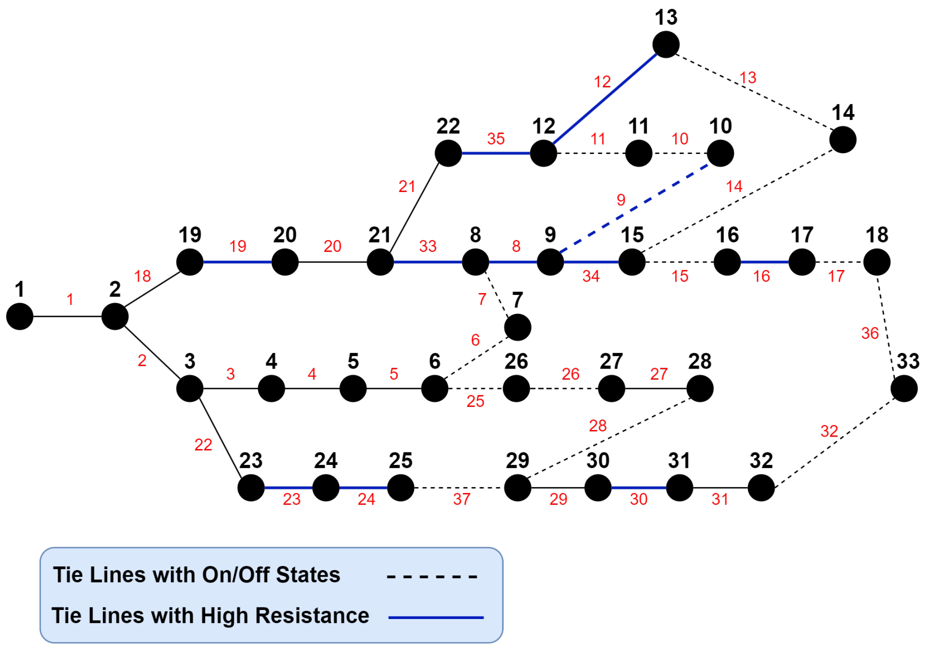

Figure 4.3: Single line diagram of the 33-bus system and its line numbers in red

4.4.1 33-Bus System

The RL agent employed to optimize DNR in this system underwent rigorous training, incorporating a dataset comprising 200 feasible actions. For more information on this approach and training results, see our previous study [26]. The dashed lines in the diagram of Fig. 4.3 represent the lines that undergo status changes. The proposed explainer network was trained considering the status of these tie lines as its output. The binary cross-entropy (BCE) loss of this network during the training process is depicted in Fig. 4.4. The graph clearly shows a consistent downward trend, which serves as compelling evidence of a highly effective training procedure. This decreasing pattern signifies that the explainer network has progressively improved its performance, steadily reducing the errors encountered during the training process. Thus, it can be concluded that the training of this neural network has been remarkably successful.

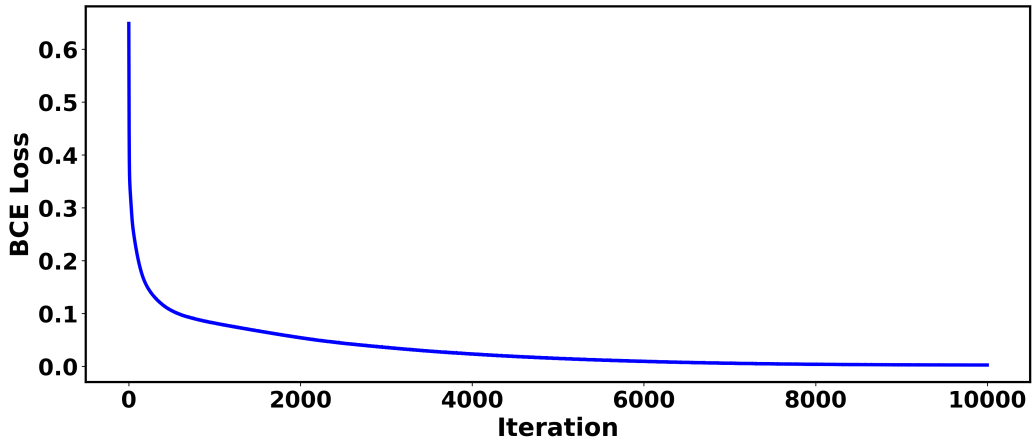

Figure 4.4: BCE loss of the 33-bus explainer network over training

In this study, the effectiveness of the proposed method is assessed by analyzing three distinct time periods, namely high consumption, average consumption, and low consumption within a 24-hour interval. Conducting the evaluations hourly would be an arduous task, as numerous heatmaps would need to be generated for each hour. Taking into account the space constraints of this thesis, it is impractical to include such a voluminous amount of data. The output of the explainer network for the

24-hour duration was carefully analyzed, ensuring that every prediction was accurate, ultimately yielding a zero BCE loss. As the load patterns tend to remain consistent from day to day, this approach can be considered sufficient.

The total active and reactive power consumptions for the 24-hour interval are given in Fig. 4.5. Based on the data presented in the figure, it is evident that hour 17 exhibits the highest consumption rate, while hour 12 reflects an average consumption level, and hour 5 has the lowest consumption during the specified period. The status of the tie lines for the 24-hour interval is visualized in Fig. 4.6, employing a barplot for better comprehension. The y-axis represents the line numbers, and the open lines are signified using a dark blue color.

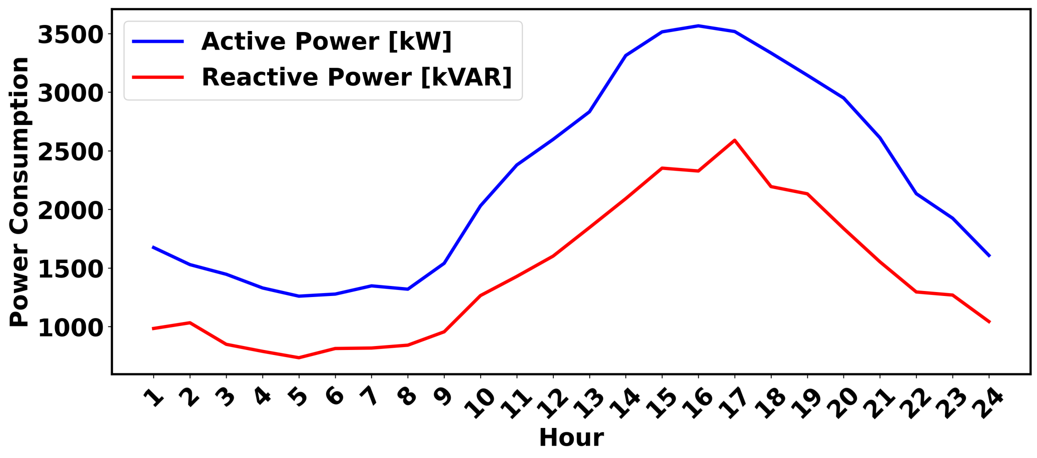

Figure 4.5: Total active and reactive power consumption of the 33-bus system in 24 hours

The 24-hour boxplot in Fig. 4.7 provides valuable insight into the active and reactive power of the 33-bus system. This visual representation allows easier identification of buses that exhibit significant variations in their active and reactive power levels. Buses with pronounced fluctuations in their power profiles can be recognized by examining this figure. Eventually, the heat maps depicted in Figs. 4.8 and 4.9, respectively, provide a comprehensive visualization of the active and reactive power consumption at every hour across all bus locations.

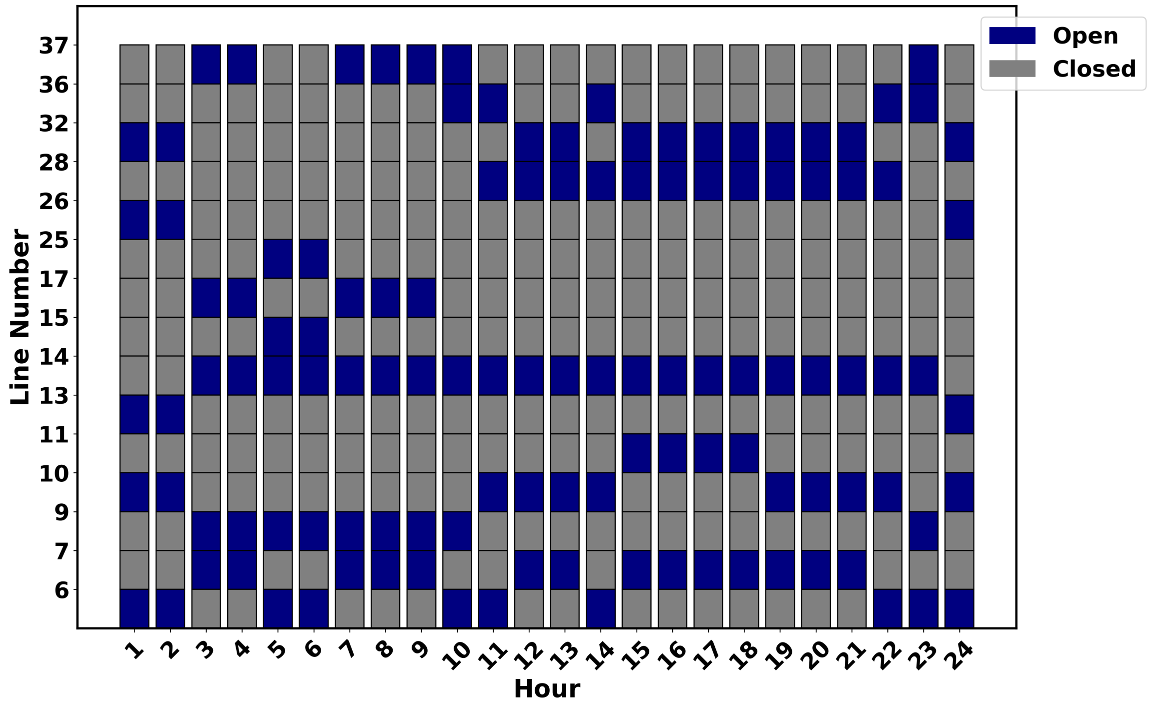

Figure 4.6: Bar plot representing open and closed lines in the 33-bus system during 24 hours

First, the highest consumption hour, hour 17, is given to the explainer network. Fig. 4.10 illustrates the heatmap of the attribution scores generated by the Gradient*Input method. Notably, on the x-axis, the first set of numbers (ranging from 2 to 33) represents the active power of the corresponding buses, while the second set of numbers represents the reactive power of the buses. Observing the heatmap, it becomes evident that lines 32, 36, and 37 are most reliant on the reactive power of bus 30 when employing this method. However, it is important to note that, despite being the highest value in the heatmap, this dependency is still extremely low and almost negligible. Consequently, it can be inferred that no significant changes occurred in the system as a result of the consumption variation during this hour. In fact, a careful examination of Fig. 4.6 makes it evident that the configuration of the system between hours 15 and 18 remains the same.

Based on the Occlusion analysis, only decreasing the reactive power of bus 30 by 90% will alter the line states and trigger the opening of lines 6, 10, and 36,

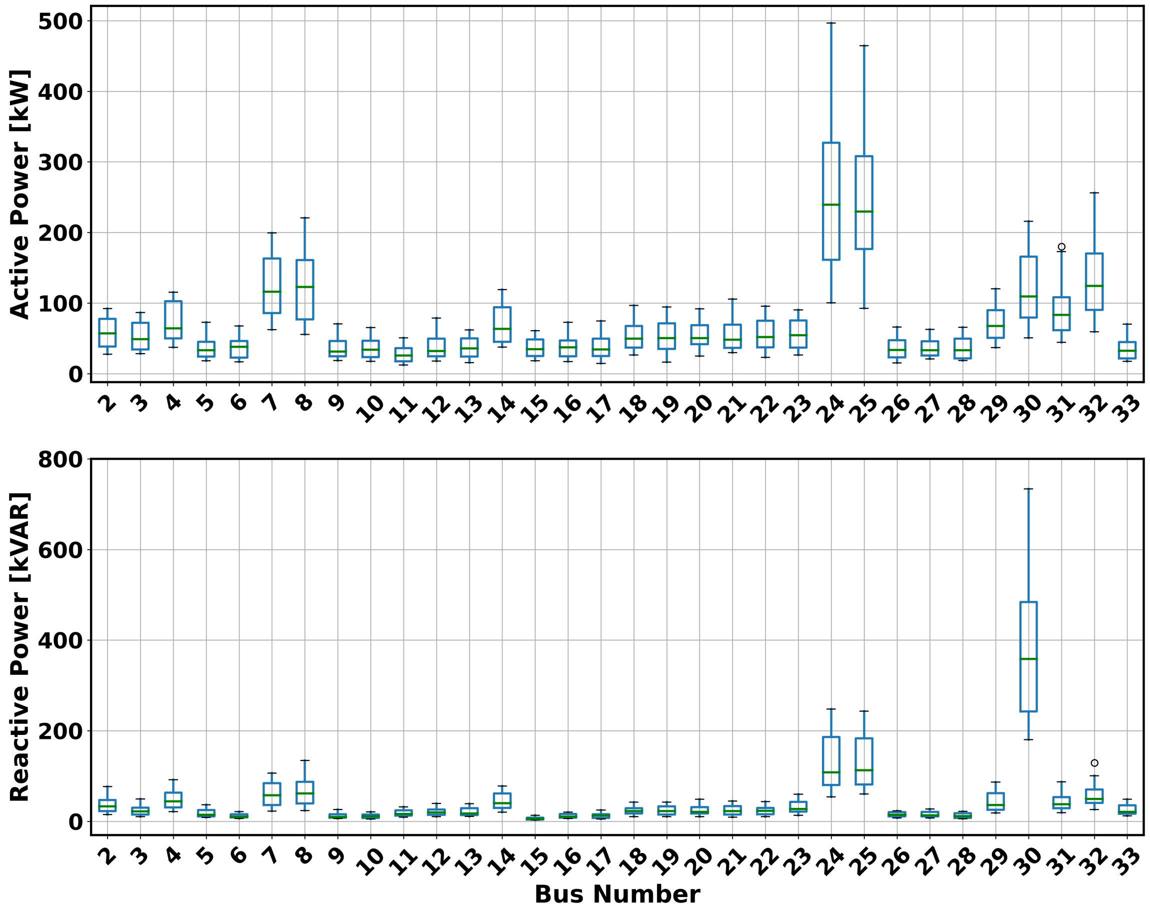

Figure 4.7: Active (top) and reactive (bottom) power consumption boxplot for the 33-bus system in 24 hours

while simultaneously causing the closure of lines 7 and 32. The attribution scores generated by the Smoothgrad method are depicted as a heatmap in Fig. 4.11 and show similar results as the Gradient*Input method. Akin to the Gradient*Input method, the heatmap illustrates that even the highest value recorded remains close to zero, emphasizing that dependencies observed in this hour are negligible.

The heatmaps illustrating the results of the Gradient*Input and Smoothgrad methods for the average consumption period can be observed in Fig. 4.12 and Fig. 4.13, respectively. As can be seen, none of the lines exhibit sensitivity to variations in the active and reactive power of the buses. In Occlusion analysis, even reducing the input values to zero does not change the output at this hour.

Finally, the heatmaps for the Gradient*Input and Smoothgrad methods during the

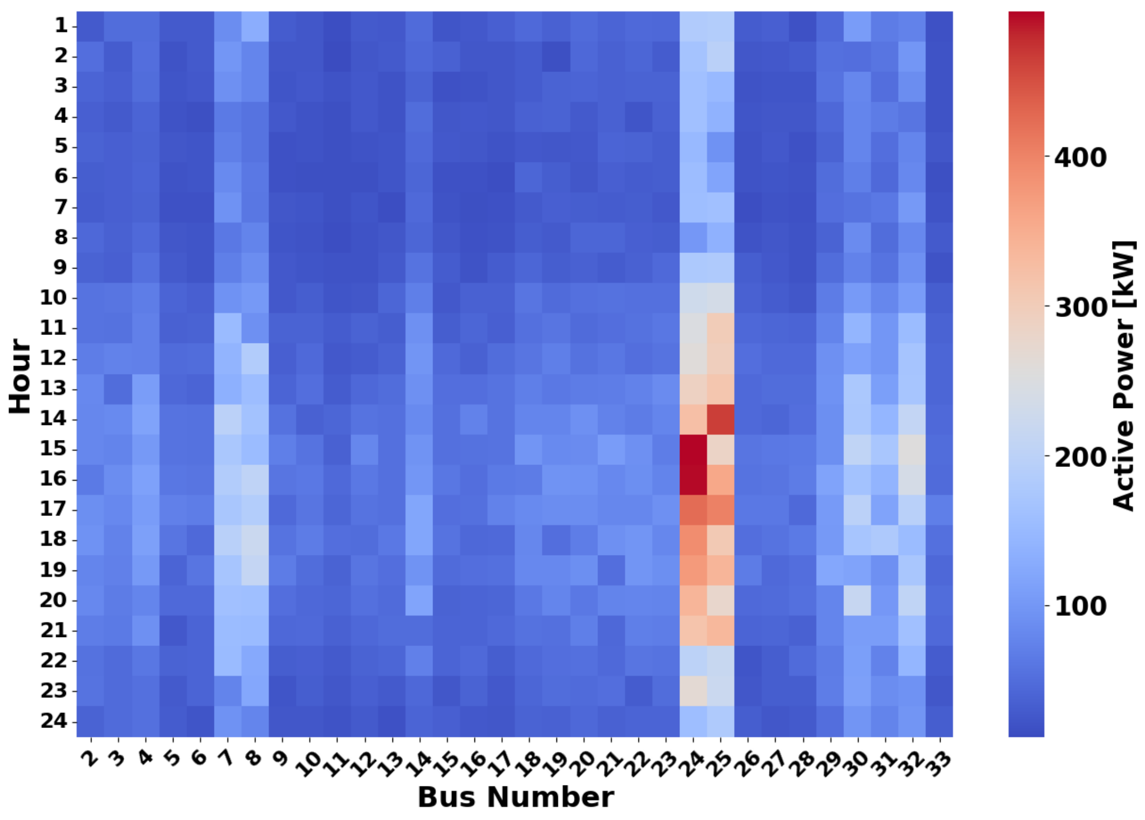

Figure 4.8: Heatmap representing the active power consumption in the 33-bus system during 24 hours

low consumption period are given in Fig. 4.14 and Fig. 4.15, respectively. According to Fig. 4.14, the Gradient*Input method indicates that all lines have the highest reliance on the reactive power of bus 30. They also have some dependence on the active power of buses 32 and 25. On the other hand, based on the Smoothgrad results given in Fig. 4.15, lines 25, 26, 28, 32, 36 highly depend on the reactive power of bus 30. In addition, this method reveals that line 15 also has some dependence on the reactive power of bus 30. It should be noted that lines 25, 26, 28, 32, and 36 also exhibit a certain level of dependence on the active power of bus 32.

Through occlusion analysis, it becomes apparent that altering the reactive power of bus 30 by 50% exclusively affects the states of the lines. Moreover, if the input values are decreased by 60%, only the variations in the active power of buses 23 and 30 and the reactive power of bus 30, will alter the original output. Modifying the

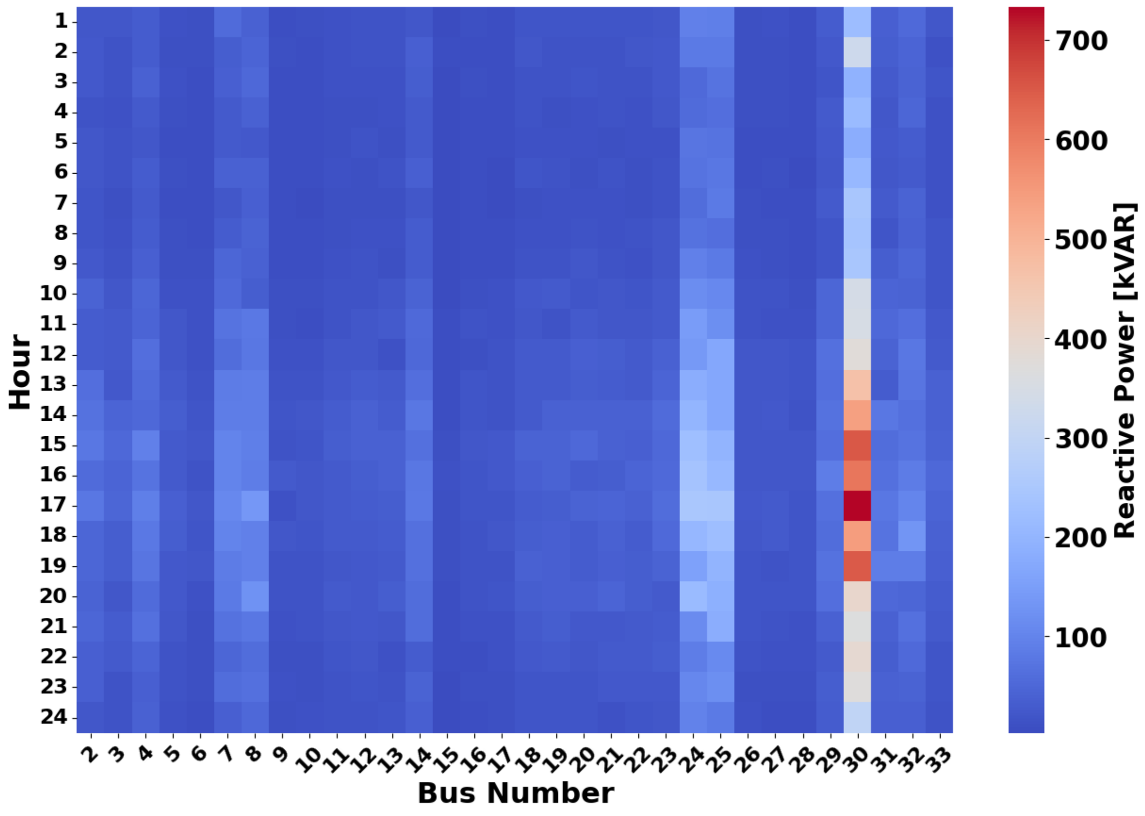

Figure 4.9: Heatmap representing the reactive power consumption in the 33-bus system during 24 hours

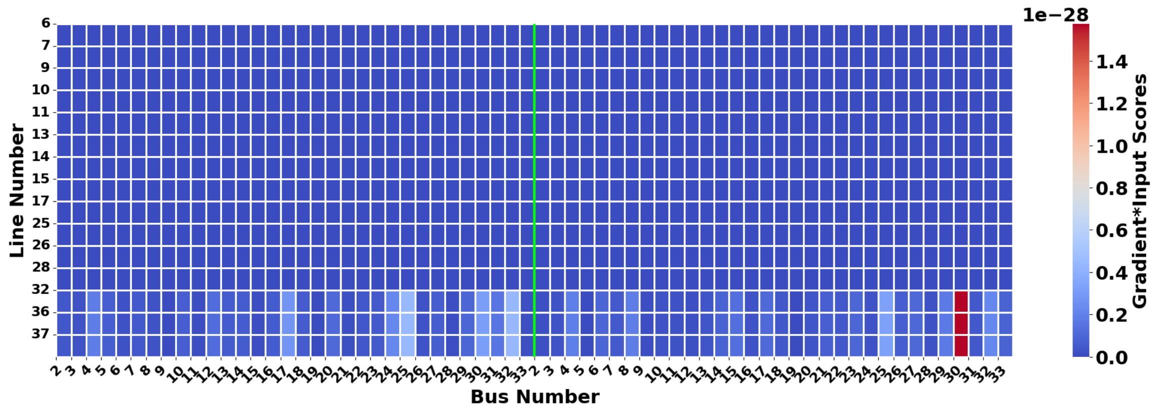

Figure 4.10: Heatmap representing the scores obtained by the Gradient*Input method during high consumption hour in the 33-bus system (numbers 2-33 on the left represent active power and the numbers on the right represent reactive power of the buses)

active power of buses 22, 23, 30, and the reactive power of bus 30 by 70% results in an alteration of the output. Finally, when setting the inputs to zero individually, only

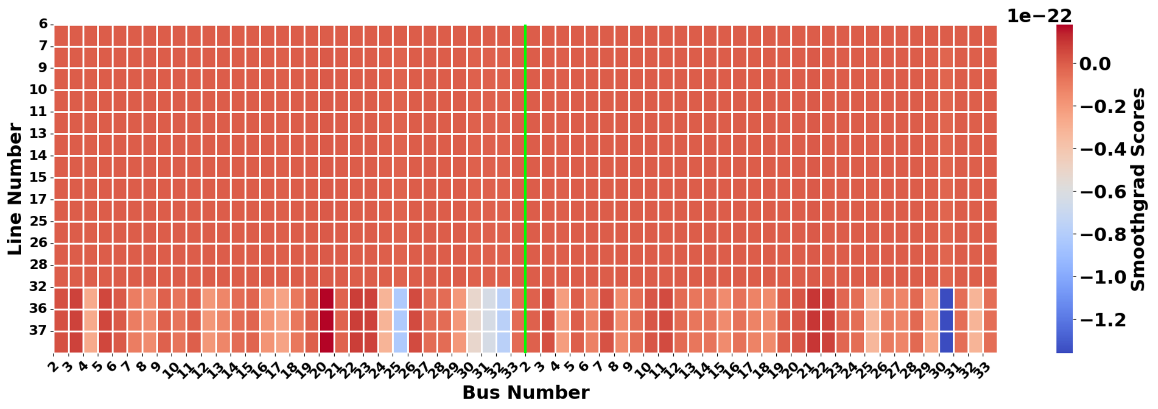

Figure 4.11: Heatmap representing the scores obtained by the Smoothgrad method during high consumption hour in the 33-bus system (numbers 2-33 on the left represent active power and the numbers on the right represent reactive power of the buses)

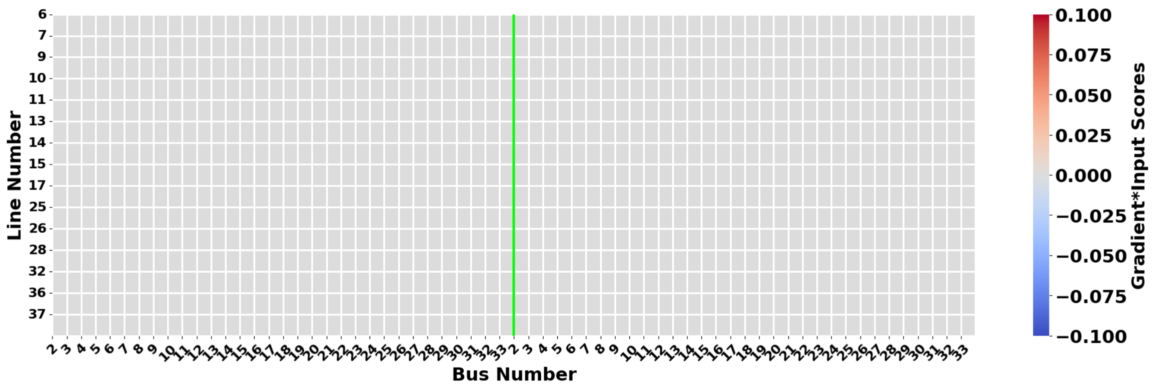

Figure 4.12: Heatmap representing the scores obtained by the Gradient*Input method during average consumption hour in the 33-bus system (numbers 2-33 on the left represent active power and the numbers on the right represent reactive power of the buses)

the active power of buses 6, 22, 23, 30, along with the reactive power of bus 30, affect the line states.

Figure 4.16 illustrates the structure of the system in the three distinct time periods. A notable observation is that the system's structure remains largely consistent between the high- and average-consumption periods, indicating minimal dependence on any specific input. Consequently, the attribution scores associated with these periods were nearly negligible. However, during the low consumption hour, the system structure deviates significantly, which explains the significant attribution scores obtained during

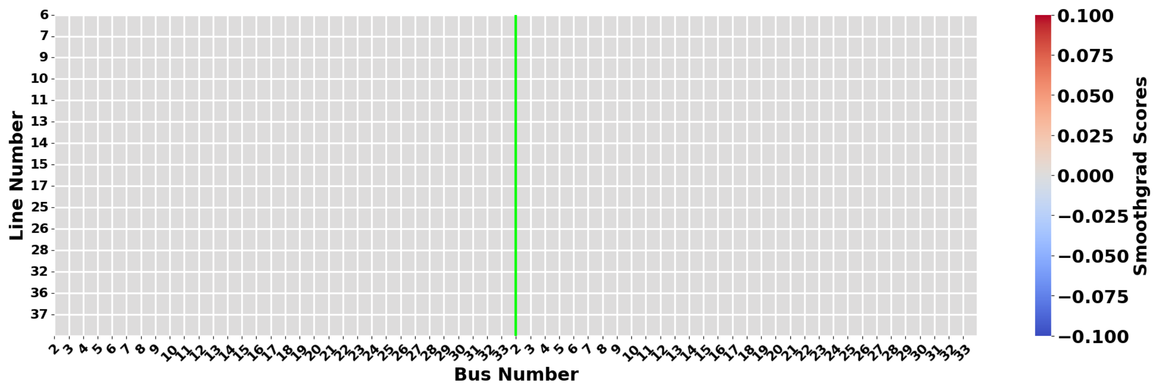

Figure 4.13: Heatmap representing the scores obtained by the Smoothgrad method during average consumption hour in the 33-bus system (numbers 2-33 on the left represent active power and the numbers on the right represent reactive power of the buses)

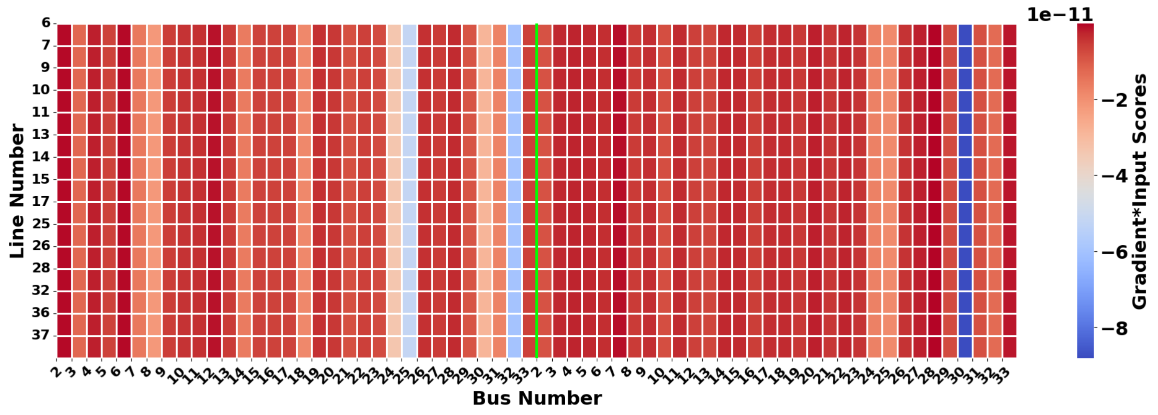

Figure 4.14: Heatmap representing the scores obtained by the Gradient*Input method during low consumption hour in the 33-bus system (numbers 2-33 on the left represent active power and the numbers on the right represent reactive power of the buses)

this time.

An analysis of the system's structure under the highest possible load condition reveals that the network's configuration during hour 17 proves to be the most optimal configuration. Consequently, this finding elucidates why the attribution scores were considerably low, as there exists no alternative optimal configuration for the system. Regardless of the increase in consumption, the network configuration remains unchanged.

Finally, to ascertain the effectiveness of the proposed method, a critical evaluation

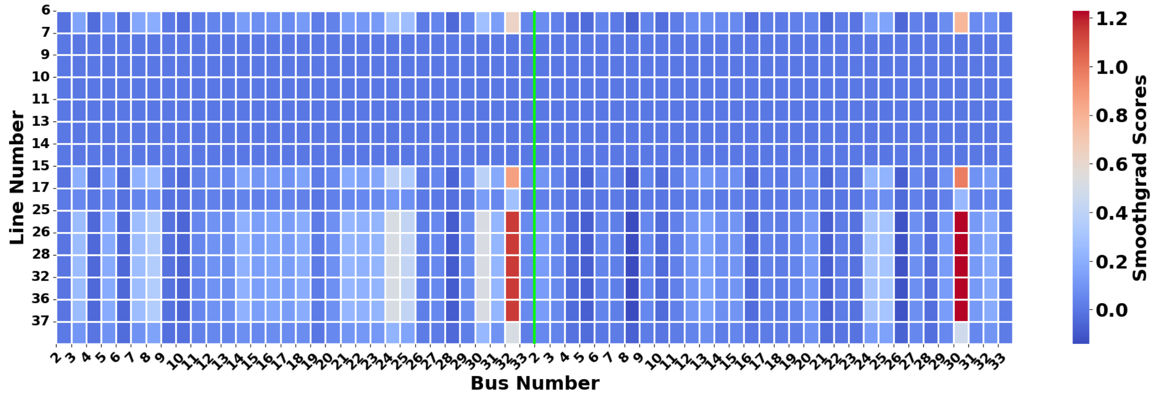

Figure 4.15: Heatmap representing the scores obtained by the Smoothgrad method during low consumption hour in the 33-bus system (numbers 2-33 on the left represent active power and the numbers on the right represent reactive power of the buses)

is conducted at hour 14, just before the system undergoes a transition into a high-load configuration. The Occlusion analysis shows that the active power of buses 25 and 32 and the reactive power of bus 30 have the highest impact on line states. Additionally, the attribution scores obtained by the Gradient*Input and Smoothgrad methods are illustrated in Fig. 4.17 and Fig. 4.18, respectively. According to the Smoothgrad method, which is more reliable, line 6 depends the most on the active power of bus 25 and the reactive power of bus 30. Additionally, line 32 depends on the reactive power of bus 30 and the active power of bus 25. Lines 36 and 37 also show some reliance on these buses.

To combine the results from the three attribution methods, the Smoothgrad method is repeated multiple times by varying the sample size from 25 to 200 and the noise level from 5% to 25% and the attribution scores are averaged. Furthermore, the results of the occlusion analysis are translated into scores. For instance, if altering input i by 30% affects output j, a score of 0.7 is given to the score map between input i and output j. The scores from Gradient*Input, Smoothgrad, and Occlusion methods are brought to a same scale by normalizing the values to be between -1 and 1. Subsequently, a weighted average of these normalized scores is calculated. In this computation, the Smoothgrad and Occlusion analysis methods are deemed more reliable, and therefore, assigned a

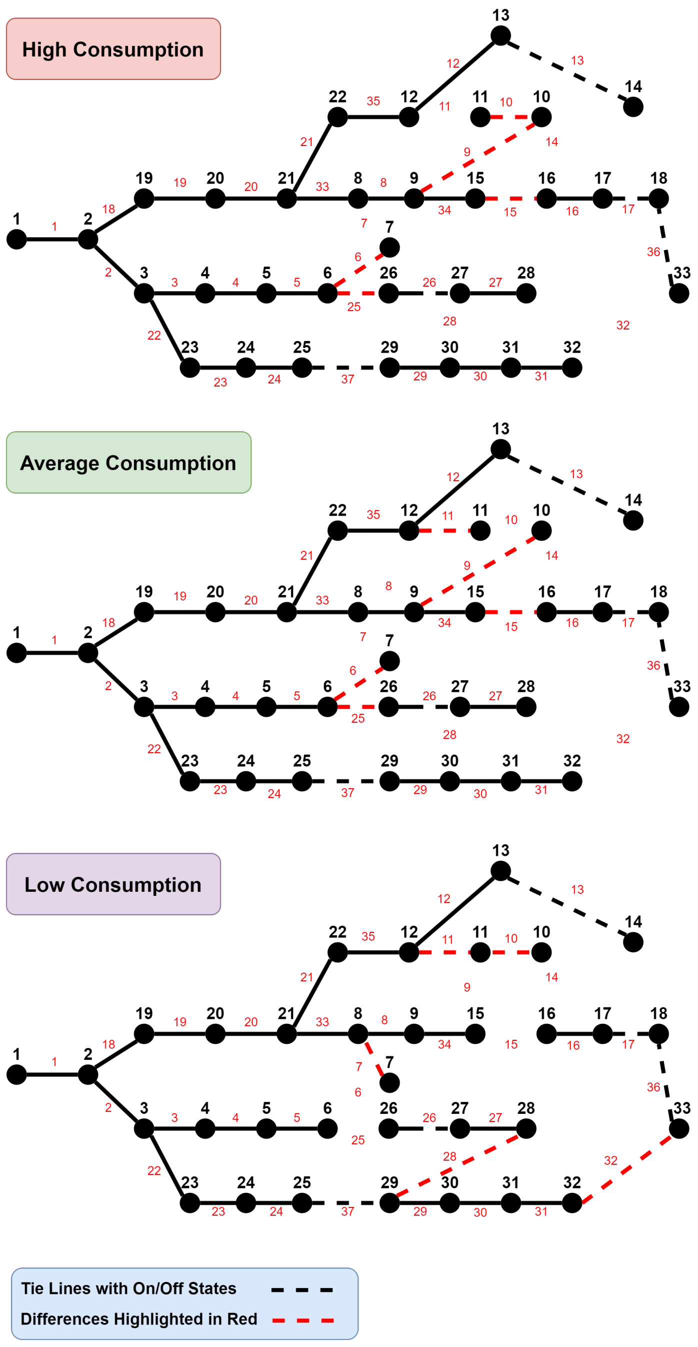

Figure 4.16: 33-bus system structure in high, average and low power consumption times

higher weight of 2 each. On the other hand, the Gradient*Input method is allocated a weight of 1 as it is considered less reliable [5]. This weighting rationale stems from the observation that the Gradient*Input method can occasionally yield noisy outcomes by capturing minor fluctuations. Moreover, the attribution scores from Gradient*Input

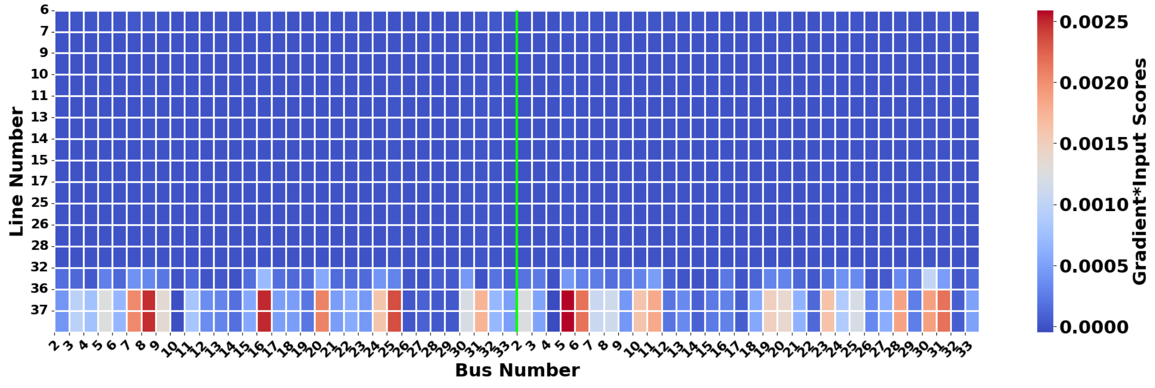

Figure 4.17: Heatmap representing the scores obtained by the Gradient*Input method during hour 14 in the 33-bus system (numbers 2-33 on the left represent active power and the numbers on the right represent reactive power of the buses)

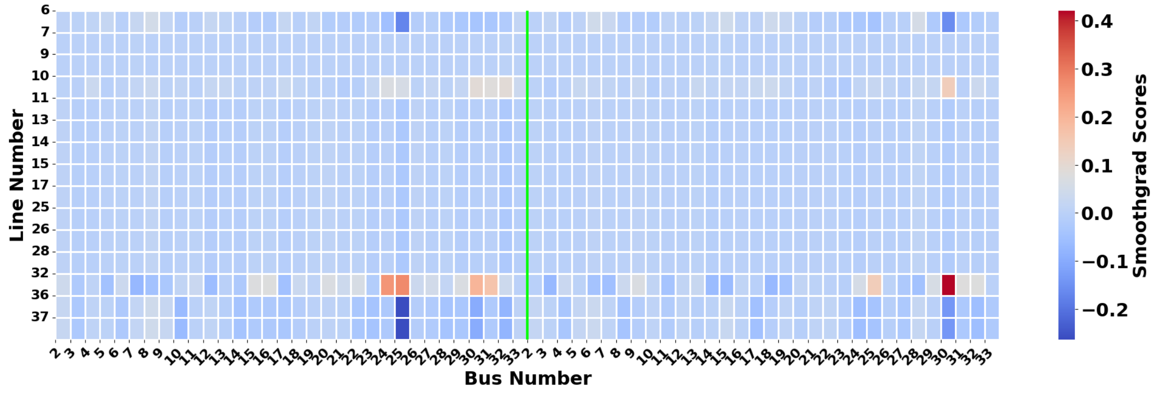

Figure 4.18: Heatmap representing the scores obtained by the Smoothgrad method during hour 14 in the 33-bus system (numbers 2-33 on the left represent active power and the numbers on the right represent reactive power of the buses)

tend to be less stable in comparison to methods like Smoothgrad, which employs smoothing techniques to create more consistent scores. Additionally, the Occlusion analysis facilitates more interpretable results by systematically eliminating features to assess their impact, thereby simplifying the interpretation and justification of the outcomes. The final scores for minimum load and transition to high load condition are illustrated in Fig. 4.19 and Fig. 4.20, respectively. Based on this, the logic behind the DNR agent's decisions for the 33-bus system can be summarized as shown in Table 4.1.

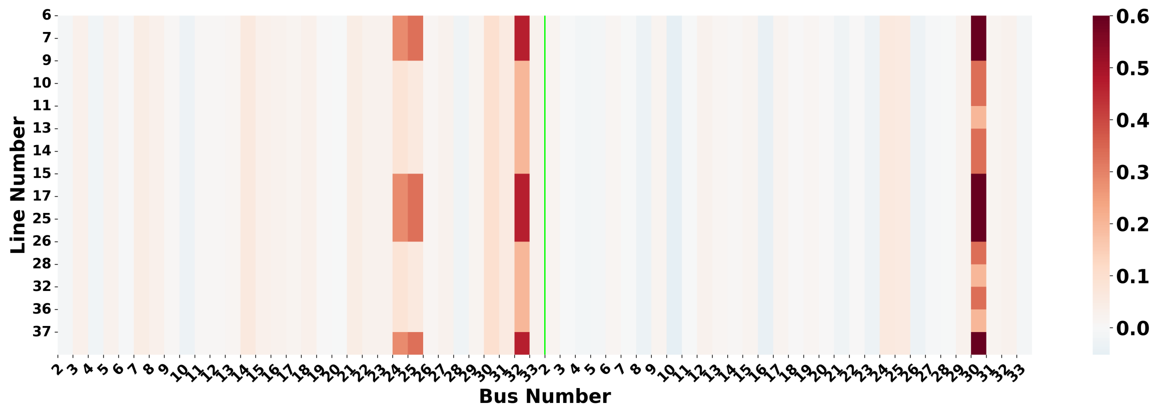

Figure 4.19: Heatmap representing the final scores by combining the result of three attribution methods during low consumption hour in the 33-bus system (numbers 2-33 on the left represent active power and the numbers on the right represent reactive power of the buses)

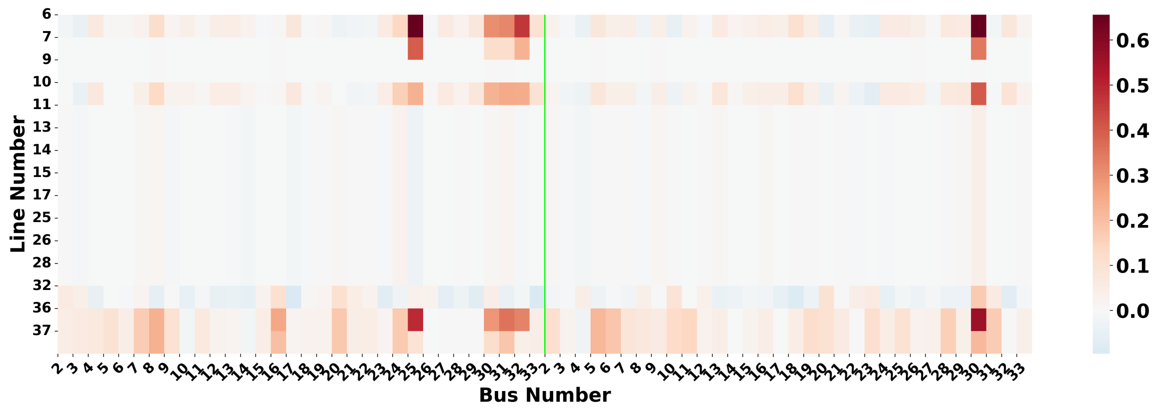

Figure 4.20: Heatmap representing the final scores by combining the result of three attribution methods during hour 14 in the 33-bus system (numbers 2-33 on the left represent active power and the numbers on the right represent reactive power of the buses)

Table 4.1: Operational Strategies of the DNR Agent in the 33-Bus System

Time	**Action Explanation**
Peak load (h=17)	Line states do not change in response to small variations in bus loads.
Transition to high-load condition (h=14)	Line states change mainly in response to variations in active power of buses 25, 30, 31, 32 and the reactive power of bus 30.
Average load (h=12)	Line states do not change in response to small variations in bus loads.
Minimum load (h=5)	Line states change mainly in response to variations in active power of buses 24, 25, 32 and the reactive power of bus 30.

4.4.2 136-Bus System

The BCE loss of the explainer network for the 136-bus system, illustrated in Fig. 4.21, significantly decreases during the training process, indicating a successful learning. Similarly to the 33-bus system, a 24-hour interval is selected for analysis, focusing first on evaluating the performance during the peak-load hour using the proposed approach. The 24-hour active and reactive demand are illustrated in Fig. 4.22. Due to the extensive number of inputs in this system, totaling 270 (comprising active and reactive power), it is not feasible to visually represent the results using a heatmap for this particular test system. Consequently, only the most significant observations are presented here.

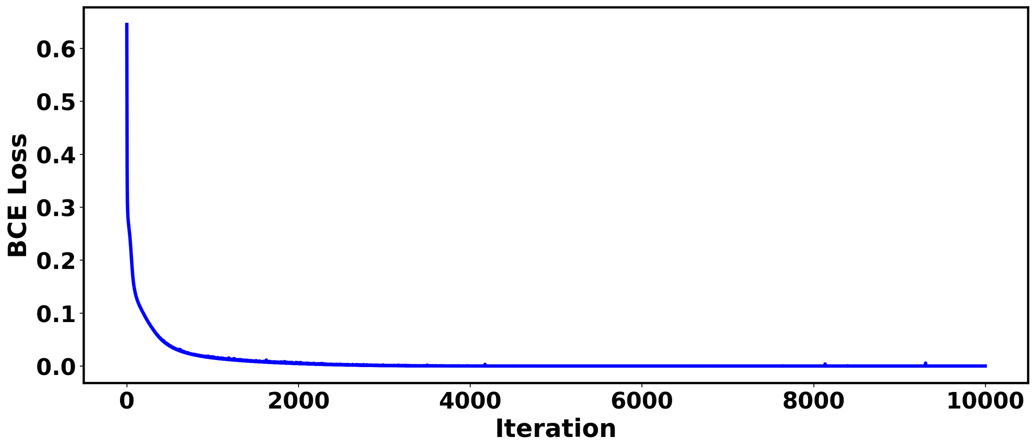

Figure 4.21: BCE loss of the 136-bus explainer network over training

During the highest load condition, both the Gradient*Input and Smoothgrad methods yield zero attribution scores. Furthermore, through Occlusion analysis, it was determined that setting any individual input value to zero would have no impact on the system's output. These findings were also consistent across the average consumption hours.

During the low-demand hour, Gradient*Input method indicates that line 155 exhibits a minor dependency on the active power of bus 105. However, this dependence is extremely insignificant and almost zero. Through occlusion analysis, it is revealed

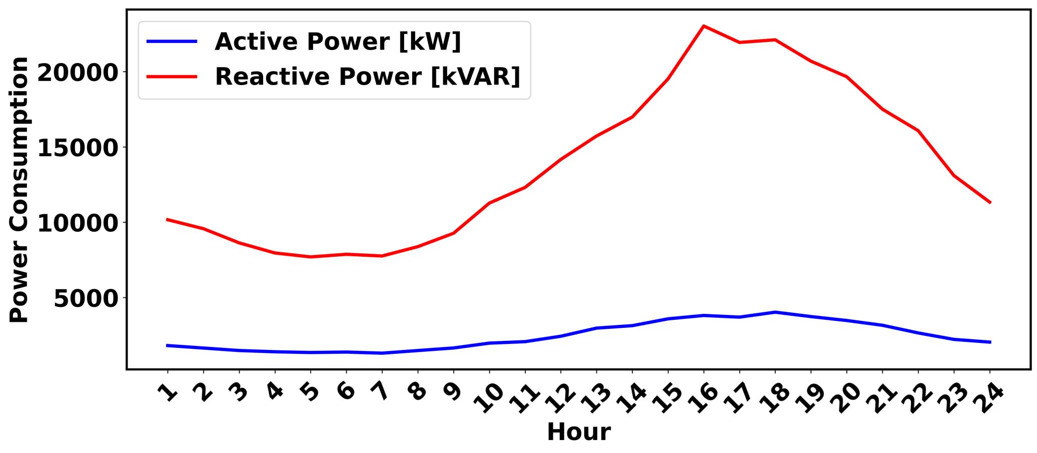

Figure 4.22: Total active and reactive power consumption of the 136-bus system in 24 hours

that the active power of buses 34 and 105, as well as the reactive power of buses 80 and 105, play a role in the line states. Moreover, employing the Smoothgrad method, it has been determined that various lines, specifically lines 2, 51, 54, 62, 75, 99, 119, 136, 140, 141, 142, 143, and 154, exhibit a dependency on the active power of bus 105. These line numbers are colored red in Figure 4.23.

Finally, the last hour before the high-demand period (hour 14) is analyzed. The application of the Gradient*Input method failed to reveal any notable patterns or insights. However, through Occlusion analysis, it was discovered that the output exhibits changes solely in response to variations in the active power of bus 105. This finding highlights the pivotal role played by bus 105's active power in driving alterations in the output. Furthermore, applying the Smoothgrad method, it was determined that lines 49 and 54 are particularly dependent on the active power of bus 105, further emphasizing its significance in shaping the outcome. By combining the results from three attribution methods, the logic behind the DNR agent's decisions for the 136-bus system is summarized in Table 4.2.

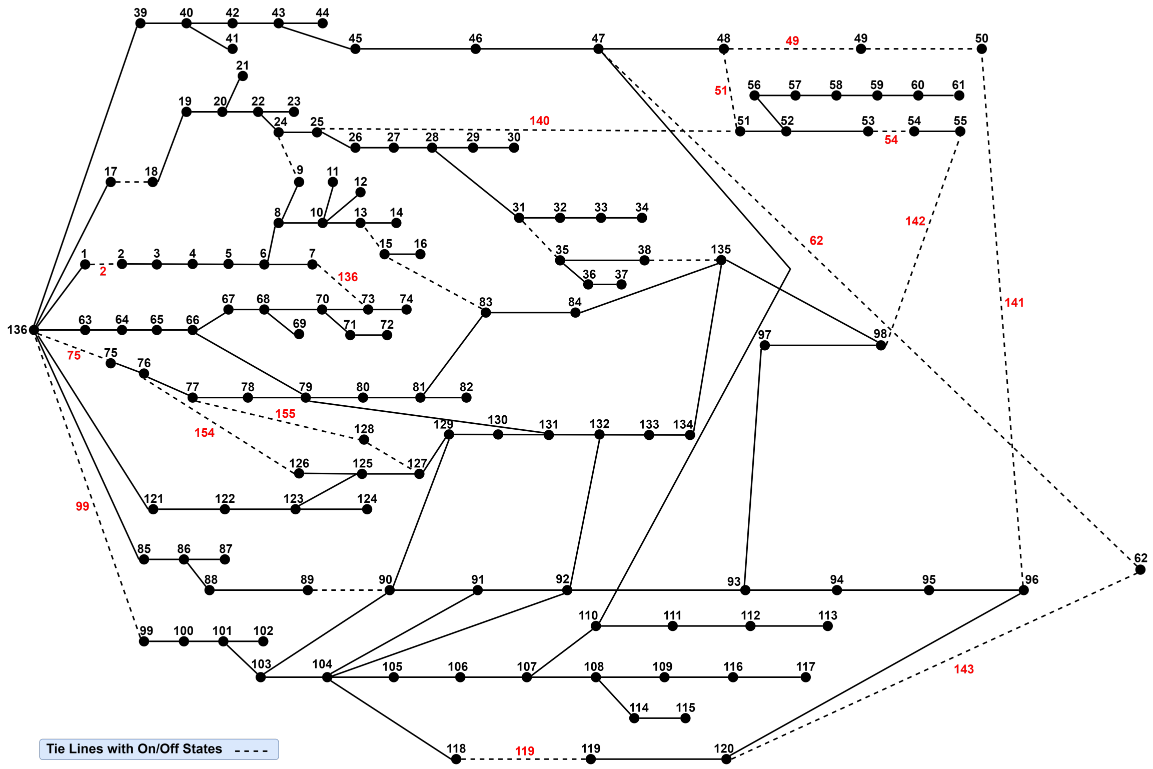

Figure 4.23: 136-bus system with lines numbers

Table 4.2: Operational Strategies of the DNR Agent in the 136-Bus System

Time	**Action Explanation**
Peak load (h=16)	Line states do not change in response to small variations in bus loads.
Transition to high-load condition (h=14)	Line states change mainly in response to variations in active power of buses 62, 105 and the reactive power of buses 74, 134.
Average load (h=12)	Line states do not change in response to small variations in bus loads.
Minimum load (h=5)	Line states change mainly in response to variations in active and reactive power of bus 105.

4.5 Conclusion

This study describes a novel approach to explain the decisions made by an RL agent for DNR. The proposed methodology involves training an explainer neural network that takes the active and reactive power of the buses as input and the line states determined by the DNR agent as output. Next, three different attribution methods are used to explain the relationship between the inputs and outputs of this network. This approach was applied to the 33- and 136-bus test systems. The results demonstrate that the reactive power of bus 30 and the active power of bus 105 play a crucial role in determining the line states in the 33- and 136-bus systems, respectively.

The knowledge gained from this study can be applied to optimize power grid operations and improve grid stability and efficiency. By identifying key factors that influence the state of the lines, power grid operators can make informed decisions during different demand scenarios, leading to more effective and reliable power management. Additionally, this technique is scalable and can be applied to any test system.

Possible future directions for this study include dynamic analysis and examining how the importance of input features and their relationships change over time. This dynamic perspective would be particularly relevant in the context of managing power systems where conditions constantly change. Additionally, integrating the explainer neural network into the RL agent's training process is a promising future direction. By doing so, the RL agent can potentially benefit from the insights gained from the explainer network, leading to improved performance and interpretability.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In conclusion, this research endeavors to address the limitations in the field of distribution network reconfiguration (DNR) optimization by harnessing the potential of reinforcement learning (RL). The existing studies in this area lack comprehensive analysis, scalability, and flexibility, which motivates this research to develop a flexible RL framework for DNR. By applying RL to DNR, power system operators can leverage RL's learning capabilities to expedite reconfiguration execution and avoid the DNR's NP-hard optimization problem without using any simplifications for radiality and AC power flow constraints, thus, potentially contributing to improved efficiency and reliability in power system operations.

The key objectives of this research have been successfully achieved through three insightful studies. In the first study, we formulated the DNR problem as a Markov decision process. The state of the system was defined as the active and reactive power consumption at system buses and the line states while the actions were defined as the opening or closing of each line. Next, we developed an algorithm capable of identifying the feasible actions within the distribution network, addressing the challenge posed by the vast switching action space in power systems. The introduction of a new deep Q-learning-based action reduction method demonstrated promising results in terms of convergence and optimality. This study also performed a comparative analysis of five

RL algorithms for DNR, highlighting their respective strengths and capabilities. It was found that deep Q-learning and dueling deep Q-learning have the best performance in terms of optimality of the obtained solutions while the soft actor-critic method has the fastest convergence.

The second study contributed to the advancement of RL-based DNR by proposing a generalized deep RL model and a new power flow-based action space sampling method. A graph theory-based algorithm named Yamada-Kataoka-Watanabe was utilized to find significantly more feasible actions in the 33-, 119-, and 136-bus test systems. Using this approach, it was discovered that the DNR agent successfully found solutions with optimality gaps of only 0.3%, 10.58%, and 1.95% for the 33-, 119-, and 136-bus test networks, respectively. Additionally, comparing the proposed RL method with traditional methods for DNR revealed that the RL-based method had at least 20 times faster execution time.

Furthermore, the third study tackled the inherent opacity of RL algorithms, which can lead to skepticism and mistrust among users. By developing an explanation model for the RL agents' decision-making process, this research enhanced the interpretability and trustworthiness of RL-based control in DNR. The proposed explanation approach involved training of an explainer neural network based on the actions of the trained DNR agent as its output and the active and reactive system loads as the input. In the next step, three attribution methods were used to assign importance scores to each input and explain the relationship between the explainer network's inputs and outputs. This method was applied to the 33- and 136-bus test systems and it was found that the reactive power of bus 30 had a crucial role in determining the 33-bus network's configuration. In the 136-bus system, the active power of bus 105 was the most important factor in changing network's configuration. The insights gained from this study have the potential to optimize power grid operations, improve grid stability, and facilitate effective decision-making during different demand scenarios. By recognizing crucial elements that impact the condition of the lines, power grid operators can make well-informed choices in response to various demand situations, resulting in more efficient and dependable power management. Moreover, this method is adaptable and can be implemented in any test system.

In summary, the motivation and main contributions of each study are as follows:

Research Study 1: "A Comparative Study of Reinforcement Learning Algorithms for Distribution Network Reconfiguration with Deep Q-learning-based Action Sampling"

Motivation: No previous study presented a comprehensive comparison of RL algorithms for DNR. Additionally, no prior research has introduced an approach to reduce the action space size in DNR.

Contributions: A new DQN-based method was developed to reduce the size of the action space for RL in DNR. The performance of five on-policy and off-policy RL algorithms were tested and compared for the DNR problem.

Research Study 2: "A Generalised Deep Reinforcement Learning Model for Distribution Network Reconfiguration with Powerflow-based Action Space Sampling"

Motivation: The feasible action space size in a DNR problem is very large. Although a method was developed in our previous study to reduce the action space size, the initial sampling of the actions in this method remained largely random. None of the previous studies, have focused on developing an action space sampling method for DNR while selecting the most optimal actions.

Contributions: A new method for sampling the action space in DNR problem was devised which was based on the Yamada-Kataoka-Watanabe algorithm for feasible action extraction and was based on the power flow study for determining the optimality of each action. A comprehensive sensitivity analysis and comparison with the traditional methods was conducted at the end of this study.

Research Study 3: "Explainable Reinforcement Learning for Distribution Network Reconfiguration"

Motivation: The inherent opacity in RL agent's decision making leads to skepticism and mistrust among users. However, no prior research has studied the explainability of the RL agent's decisions in DNR.

Contributions: An explanation model for RL agent's decisions in DNR was developed for the first time to understand the logic behind the agent's decision making and increase trust.

Collectively, these research studies have made significant contributions to the application of RL in DNR. The developed flexible RL framework, efficient action space sampling methods, and the introduction of explanation models have paved the way for future investigations in the area of RL-based power system optimization. By addressing the identified limitations and challenges, this research has set a solid foundation for achieving improved efficiency, reliability, and trust in power system operations, ultimately benefiting utilities through enhanced power grid management and operation. As the field of RL continues to evolve, the findings from this research will remain invaluable for researchers and power system operators alike in their pursuit of innovative solutions for the dynamic challenges of modern power systems.

5.2 Future Work

Our research successfully addressed several fundamental gaps in applying RL to DNR. Nevertheless, there remain exciting and promising avenues for future research that deserve thorough exploration.

- **Designing an efficient action space exploration method:** In our study, we tackled the challenge of identifying the action space in the system by finding the whole feasible action space and sampling from it. Although this approach identified the most optimal actions in the network, searching for the feasible action space, especially in the case of large systems like the 119- or 136-bus systems (containing over 3,500,000 actions), was extremely memory-intensive.

This significant memory requirement and computational burden necessitate a new method to identify optimal actions directly without having to search the entire feasible action space. Future research could focus on developing innovative techniques that efficiently find optimal actions for DNR without exhaustive exploration.

- **Integrating explainer neural networks to the RL agent's training:** One promising future research direction involves integrating the explainer neural networks into the training process of the RL agent. By making the model's decisions more interpretable, explainable RL can help identify which state features are most influential in making decisions. This information can be used to simplify the state representation or modify the feature engineering process.

- **Developing hybrid RL and metaheuristic algorithms:** Combining RL with metaheuristic algorithms, such as genetic algorithms or particle swarm optimization, could offer a powerful approach for solving large-scale DNR problems efficiently. This hybridization may leverage the exploration capabilities of RL while exploiting the optimization strengths of metaheuristic algorithms, leading to improved convergence and reduced computational burden.

- **Transfer learning for DNR:** Investigating transfer learning techniques for DNR could enable leveraging knowledge gained from one distribution network to improve the learning process on another network. This approach can potentially accelerate convergence and reduce the data requirements for training RL agents.

- **Scalable state representation:** Finding an efficient and compact representation of the distribution network state is essential to handle large-scale systems. Future research could explore dimensionality reduction techniques or graph-based embeddings to represent complex networks more efficiently.

By addressing these future research directions, the application of RL to DNR can be further advanced, contributing to more reliable, sustainable, and efficient power distribution systems.

Bibliography

[1] M. Abdelaziz, “Distribution network reconfiguration using a genetic algorithm with varying population size,” *Electric Power Systems Research*, vol. 142, pp. 9–11, 2017, ISSN: 0378-7796. DOI: https://doi.org/10.1016/j.epsr.2016.08.026.

[2] M. Abdelaziz, “Distribution network reconfiguration using a genetic algorithm with varying population size,” *Electric Power Systems Research*, vol. 142, pp. 9–11, 2017, ISSN: 0378-7796. DOI: https://doi.org/10.1016/j.epsr.2016.08.026.

[3] H. Ahmadi and J. R. Martí, “Distribution system optimization based on a linear power-flow formulation,” *IEEE Transactions on Power Delivery*, vol. 30, no. 1, pp. 25–33, 2015. DOI: 10.1109/TPWRD.2014.2300854.

[4] P. Akaber, B. Moussa, M. Debbabi, and C. Assi, “Automated post-failure service restoration in smart grid through network reconfiguration in the presence of energy storage systems,” *IEEE Systems Journal*, vol. 13, no. 3, pp. 3358–3367, 2019. DOI: 10.1109/JSYST.2019.2892581.

[5] M. Ancona, E. Ceolini, C. Öztireli, and M. Gross, “Gradient-based attribution methods,” in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K.-R. Müller, Eds. Cham: Springer International Publishing, 2019, pp. 169–191, ISBN: 978-3-030-28954-6. DOI: 10.1007/978-3-030-28954-6_9.

[6] D. Anteneh, B. Khan, O. P. Mahela, H. H. Alhelou, and J. M. Guerrero, “Distribution network reliability enhancement and power loss reduction by optimal network reconfiguration,” *Computers & Electrical Engineering*, vol. 96, p. 107 518, 2021, ISSN: 0045-7906. DOI: https://doi.org/10.1016/j.compeleceng.2021.107518.

[7] A. Azizi, B. Vahidi, and A. F. Nematollahi, “Reconfiguration of active distribution networks equipped with soft open points considering protection constraints,” *Journal of Modern Power Systems and Clean Energy*, vol. 11, no. 1, pp. 212–222, 2023.

[8] F. Bellotti, L. Lazzaroni, A. Capello, M. Cossu, A. De Gloria, and R. Berta, “Explaining a deep reinforcement learning (drl)-based automated driving agent in highway simulations,” *IEEE Access*, vol. 11, pp. 28 522–28 550, 2023.

[9] M. Bertin, N. Schweighofer, and K. Doya, “Multiple model-based reinforcement learning explains dopamine neuronal activity,” *Neural Networks*, vol. 20, no. 6, pp. 668–675, 2007, ISSN: 0893-6080.

[10] V.-H. Bui and W. Su, "Real-time operation of distribution network: A deep reinforcement learning-based reconfiguration approach," *Sustainable Energy Technologies and Assessments*, vol. 50, p. 101 841, 2022, ISSN: 2213-1388.

[11] X. Cao, J. Wang, J. Wang, and B. Zeng, "A risk-averse conic model for networked microgrids planning with reconfiguration and reorganizations," *IEEE Transactions on Smart Grid*, vol. 11, no. 1, pp. 696–709, 2020. DOI: 10.1109/TSG.2019.2927833.

[12] J. C. Cebrian and N. Kagan, "Reconfiguration of distribution networks to minimize loss and disruption costs using genetic algorithms," *Electric Power Systems Research*, vol. 80, no. 1, pp. 53–62, 2010, ISSN: 0378-7796. DOI: https://doi.org/10.1016/j.epsr.2009.08.005.

[13] Q. Chen, W. Wang, H. Wang, J. Wu, X. Li, and J. Lan, "A social beetle swarm algorithm based on grey target decision-making for a multiobjective distribution network reconfiguration considering partition of time intervals," *IEEE Access*, vol. 8, pp. 204 987–205 013, 2020. DOI: 10.1109/ACCESS.2020.3036898.

[14] Q. Chen, W. Wang, H. Wang, J. Wu, and J. Wang, "An improved beetle swarm algorithm based on social learning for a game model of multiobjective distribution network reconfiguration," *IEEE Access*, vol. 8, pp. 200 932–200 952, 2020. DOI: 10.1109/ACCESS.2020.3035791.

[15] M. Cikan and B. Kekezoglu, "Comparison of metaheuristic optimization techniques including equilibrium optimizer algorithm in power distribution network reconfiguration," *Alexandria Engineering Journal*, vol. 61, no. 2, pp. 991–1031, 2022, ISSN: 1110-0168. DOI: https://doi.org/10.1016/j.aej.2021.06.079.

[16] M. Cui, J. Wang, and M. Yue, "Machine learning-based anomaly detection for load forecasting under cyberattacks," *IEEE Transactions on Smart Grid*, vol. 10, no. 5, pp. 5724–5734, 2019. DOI: 10.1109/TSG.2018.2890809.

[17] X. Dai, S. Cheng, and A. Chong, "Deciphering optimal mixed-mode ventilation in the tropics using reinforcement learning with explainable artificial intelligence," *Energy and Buildings*, vol. 278, p. 112 629, 2023, ISSN: 0378-7788.

[18] P. M. Dassanayake, A. Anjum, A. K. Bashir, J. Bacon, R. Saleem, and W. Manning, "A deep learning based explainable control system for reconfigurable networks of edge devices," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 1, pp. 7–19, 2022. DOI: 10.1109/TNSE.2021.3083990.

[19] R. Dobbe, O. Sondermeijer, D. Fridovich-Keil, D. Arnold, D. Callaway, and C. Tomlin, "Toward distributed energy services: Decentralizing optimal power flow with machine learning," *IEEE Transactions on Smart Grid*, vol. 11, no. 2, pp. 1296–1306, 2020. DOI: 10.1109/TSG.2019.2935711.

[20] A. M. Eldurssi and R. M. O'Connell, "A fast nondominated sorting guided genetic algorithm for multi-objective power distribution system reconfiguration problem," *IEEE Transactions on Power Systems*, vol. 30, no. 2, pp. 593–601, 2015. DOI: 10.1109/TPWRS.2014.2332953.

[21] J. F. Franco, M. J. Rider, M. Lavorato, and R. Romero, “A mixed-integer lp model for the reconfiguration of radial electric distribution systems considering distributed generation,” *Electric Power Systems Research*, vol. 97, pp. 51–60, 2013, ISSN: 0378-7796. DOI: https://doi.org/10.1016/j.epsr.2012.12.005.

[22] Y.-Y. Fu and H.-D. Chiang, “Toward optimal multiperiod network reconfiguration for increasing the hosting capacity of distribution networks,” *IEEE Transactions on Power Delivery*, vol. 33, no. 5, pp. 2294–2304, 2018. DOI: 10.1109/TPWRD.2018.2801332.

[23] H. Gao *et al.*, “Multi-objective dynamic reconfiguration for urban distribution network considering multi-level switching modes,” *Journal of Modern Power Systems and Clean Energy*, vol. 10, no. 5, pp. 1241–1255, 2022.

[24] Y. Gao, B. Foggo, and N. Yu, “A physically inspired data-driven model for electricity theft detection with smart meter data,” *IEEE Transactions on Industrial Informatics*, vol. 15, no. 9, pp. 5076–5088, 2019. DOI: 10.1109/TII.2019.2898171.

[25] Y. Gao, W. Wang, J. Shi, and N. Yu, “Batch-constrained reinforcement learning for dynamic distribution network reconfiguration,” *IEEE Transactions on Smart Grid*, vol. 11, no. 6, pp. 5357–5369, 2020. DOI: 10.1109/TSG.2020.3005270.

[26] N. Gholizadeh, N. Kazemi, and P. Musilek, “A comparative study of reinforcement learning algorithms for distribution network reconfiguration with deep q-learning-based action sampling,” *IEEE Access*, vol. 11, pp. 13 714–13 723, 2023. DOI: 10.1109/ACCESS.2023.3243549.

[27] N. Gholizadeh and P. Musilek, “A generalized deep reinforcement learning model for distribution network reconfiguration with powerflow-based action space sampling,” *Journal of Modern Power Systems and Clean Energy*, 2023, under review.

[28] N. Gholizadeh and P. Musilek, “Distributed learning applications in power systems: A review of methods, gaps, and challenges,” *Energies*, vol. 14, no. 12, 2021, ISSN: 1996-1073. DOI: 10.3390/en14123654.

[29] N. Gholizadeh and P. Musilek, “Explainable reinforcement learning for distribution network reconfiguration,” *Applied Energy*, 2023, under review.

[30] V. B. Gjærum, I. Strümke, J. Løver, T. Miller, and A. M. Lekkas, “Model tree methods for explaining deep reinforcement learning agents in real-time robotic applications,” *Neurocomputing*, vol. 515, pp. 133–144, 2023, ISSN: 0925-2312.

[31] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, *Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor*, 2018. DOI: 10.48550/ARXIV.1801.01290. [Online]. Available: https://arxiv.org/abs/1801.01290.

[32] A. Hagberg, P. Swart, and D. S Chult, “Exploring network structure, dynamics, and function using networkx,” Los Alamos National Lab.(LANL), Los Alamos, NM (United States), Tech. Rep., 2008.

[33] L. He, N. Aouf, and B. Song, "Explainable deep reinforcement learning for uav autonomous path planning," *Aerospace Science and Technology*, vol. 118, p. 107 052, 2021, ISSN: 1270-9638.

[34] A. M. Helmi, R. Carli, M. Dotoli, and H. S. Ramadan, "Efficient and sustainable reconfiguration of distribution networks via metaheuristic optimization," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 1, pp. 82–98, 2022. DOI: 10.1109/TASE.2021.3072862.

[35] G. Henri and N. Lu, "A supervised machine learning approach to control energy storage devices," *IEEE Transactions on Smart Grid*, vol. 10, no. 6, pp. 5910–5919, 2019. DOI: 10.1109/TSG.2019.2892586.

[36] A. Heuillet, F. Couthouis, and N. Díaz-Rodríguez, "Explainability in deep reinforcement learning," *Knowledge-Based Systems*, vol. 214, p. 106 685, 2021, ISSN: 0950-7051.

[37] H. Hizarci, O. Demirel, and B. E. Turkay, "Distribution network reconfiguration using time-varying acceleration coefficient assisted binary particle swarm optimization," *Engineering Science and Technology, an International Journal*, p. 101 230, 2022, ISSN: 2215-0986. DOI: https://doi.org/10.1016/j.jestch.2022.101230.

[38] H. Hizarci, O. Demirel, and B. E. Turkay, "Distribution network reconfiguration using time-varying acceleration coefficient assisted binary particle swarm optimization," *Engineering Science and Technology, an International Journal*, vol. 35, p. 101 230, 2022, ISSN: 2215-0986. DOI: https://doi.org/10.1016/j.jestch.2022.101230.

[39] H. Hong, Z. Hu, R. Guo, J. Ma, and J. Tian, "Directed graph-based distribution network reconfiguration for operation mode adjustment and service restoration considering distributed generation," *Journal of Modern Power Systems and Clean Energy*, vol. 5, no. 1, pp. 142–149, 2017. DOI: 10.1007/s40565-016-0198-3.

[40] H. Hong, Z. Hu, R. Guo, J. Ma, and J. Tian, "Directed graph-based distribution network reconfiguration for operation mode adjustment and service restoration considering distributed generation," *Journal of Modern Power Systems and Clean Energy*, vol. 5, no. 1, pp. 142–149, 2017.

[41] W. Hua, B. Stephen, and D. C. Wallom, "Digital twin based reinforcement learning for extracting network structures and load patterns in planning and operation of distribution systems," *Applied Energy*, vol. 342, p. 121 128, 2023, ISSN: 0306-2619.

[42] C. Huang, H. Zhang, L. Wang, X. Luo, and Y. Song, "Mixed deep reinforcement learning considering discrete-continuous hybrid action space for smart home energy management," *Journal of Modern Power Systems and Clean Energy*, vol. 10, no. 3, pp. 743–754, 2022. DOI: 10.35833/MPCE.2021.000394.

[43] W. Huang, W. Zheng, and D. J. Hill, "Distribution network reconfiguration for short-term voltage stability enhancement: An efficient deep learning approach," *IEEE Transactions on Smart Grid*, vol. 12, no. 6, pp. 5385–5395, 2021. DOI: 10.1109/TSG.2021.3097330.

[44] X. Ji, Z. Yin, Y. Zhang, B. Xu, and Q. liu, "Real-time autonomous dynamic reconfiguration based on deep learning algorithm for distribution network," *Electric Power Systems Research*, vol. 195, p. 107 132, 2021, ISSN: 0378-7796. DOI: https://doi.org/10.1016/j.epsr.2021.107132.

[45] X. Ji, Z. Yin, Y. Zhang, B. Xu, and Q. liu, "Real-time autonomous dynamic reconfiguration based on deep learning algorithm for distribution network," *Electric Power Systems Research*, vol. 195, p. 107 132, 2021, ISSN: 0378-7796. DOI: https://doi.org/10.1016/j.epsr.2021.107132.

[46] S. Jiang, H. Gao, X. Wang, J. Liu, and K. Zuo, "Deep reinforcement learning based multi-level dynamic reconfiguration for urban distribution network: A cloud-edge collaboration architecture," *Global Energy Interconnection*, vol. 6, no. 1, pp. 1–14, 2023, ISSN: 2096-5117.

[47] J. Jose and A. Kowli, "Path-based distribution feeder reconfiguration for optimization of losses and reliability," *IEEE Systems Journal*, vol. 14, no. 1, pp. 1417–1426, 2020.

[48] M. Kaur and S. Ghosh, "Network reconfiguration of unbalanced distribution networks using fuzzy-firefly algorithm," *Applied Soft Computing*, vol. 49, pp. 868–886, 2016, ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2016.09.019.

[49] A. Kavousi-Fard, T. Niknam, and M. Fotuhi-Firuzabad, "A novel stochastic framework based on cloud theory and θ -modified bat algorithm to solve the distribution feeder reconfiguration," *IEEE Transactions on Smart Grid*, vol. 7, no. 2, pp. 740–750, 2016. DOI: 10.1109/TSG.2015.2434844.

[50] H. M. Khodr, J. Martinez-Crespo, M. A. Matos, and J. Pereira, "Distribution systems reconfiguration based on opf using benders decomposition," *IEEE Transactions on Power Delivery*, vol. 24, no. 4, pp. 2166–2176, 2009. DOI: 10.1109/TPWRD.2009.2027510.

[51] N. C. Koutsoukis, D. O. Siagkas, P. S. Georgilakis, and N. D. Hatziargyriou, "Online reconfiguration of active distribution networks for maximum integration of distributed generation," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 2, pp. 437–448, 2017. DOI: 10.1109/TASE.2016.2628091.

[52] O. B. Kundačina, P. M. Vidović, and M. R. Petković, "Solving dynamic distribution network reconfiguration using deep reinforcement learning," *Electrical Engineering*, vol. 104, 1487–1501, 2022. DOI: https://doi.org/10.1007/s00202-021-01399-y.

[53] L.-L. Li, J.-L. Xiong, M.-L. Tseng, Z. Yan, and M. K. Lim, "Using multi-objective sparrow search algorithm to establish active distribution network dynamic reconfiguration integrated optimization," *Expert Systems with Applications*, vol. 193, p. 116 445, 2022, ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2021.116445.

[54] Y. Li, G. Hao, Y. Liu, Y. Yu, Z. Ni, and Y. Zhao, "Many-objective distribution network reconfiguration via deep reinforcement learning assisted optimization algorithm," *IEEE Transactions on Power Delivery*, vol. 37, no. 3, pp. 2230–2244, 2022.

[55] T. Lin and C. Shang, "Reliability evaluation on a joint machine learning and optimization framework," *IEEE Transactions on Power Systems*, vol. 36, no. 1, pp. 49–57, 2021. DOI: 10.1109/TPWRS.2020.3007618.

[56] B. Liu, K. Meng, Z. Y. Dong, P. K. C. Wong, and X. Li, "Load balancing in low-voltage distribution network via phase reconfiguration: An efficient sensitivity-based approach," *IEEE Transactions on Power Delivery*, vol. 36, no. 4, pp. 2174–2185, 2021. DOI: 10.1109/TPWRD.2020.3022061.

[57] J. Liu and H.-D. Chiang, "Maximizing available delivery capability of unbalanced distribution networks for high penetration of distributed generators," *IEEE Transactions on Power Delivery*, vol. 32, no. 3, pp. 1196–1202, 2017. DOI: 10.1109/TPWRD.2014.2359291.

[58] N. Liu, C. Li, L. Chen, and J. Wang, "Hybrid data-driven and model-based distribution network reconfiguration with lossless model reduction," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 5, pp. 2943–2954, 2022. DOI: 10.1109/TII.2021.3103934.

[59] Y. Liu, J. Li, and L. Wu, "Coordinated optimal network reconfiguration and voltage regulator/der control for unbalanced distribution systems," *IEEE Transactions on Smart Grid*, vol. 10, no. 3, pp. 2912–2922, 2019. DOI: 10.1109/TSG.2018.2815010.

[60] Z. Liu, Y. Liu, G. Qu, X. Wang, and X. Wang, "Intra-day dynamic network reconfiguration based on probability analysis considering the deployment of remote control switches," *IEEE Access*, vol. 7, pp. 145 272–145 281, 2019. DOI: 10.1109/ACCESS.2019.2944917.

[61] J. Løver, V. B. Gjærum, and A. M. Lekkas, "Explainable ai methods on a deep reinforcement learning agent for automatic docking∗∗this work was supported by the research council of norway through the exaigon project, project number 304843.," *IFAC-PapersOnLine*, vol. 54, no. 16, pp. 146–152, 2021, 13th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles CAMS 2021, ISSN: 2405-8963.

[62] M. Mahdavi, H. H. Alhelou, A. Bagheri, S. Z. Djokic, and R. A. V. Ramos, "A comprehensive review of metaheuristic methods for the reconfiguration of electric power distribution systems and comparison with a novel approach based on efficient genetic algorithm," *IEEE Access*, vol. 9, pp. 122 872–122 906, 2021. DOI: 10.1109/ACCESS.2021.3109247.

[63] S. Malekshah, A. Rasouli, Y. Malekshah, A. Ramezani, and A. Malekshah, "Reliability-driven distribution power network dynamic reconfiguration in presence of distributed generation by the deep reinforcement learning method," *Alexandria Engineering Journal*, vol. 61, no. 8, pp. 6541–6556, 2022, ISSN: 1110-0168.

[64] M. A. Mejia, L. H. Macedo, G. Muñoz-Delgado, J. Contreras, and A. Padilha-Feltrin, "Active distribution system planning considering non-utility-owned electric vehicle charging stations and network reconfiguration," *Sustainable Energy, Grids and Networks*, vol. 35, p. 101 101, 2023, ISSN: 2352-4677. DOI: https://doi.org/10.1016/j.segan.2023.101101.

[65] R. K. Mishra and K. S. Swarup, "Adaptive weight-based self reconfiguration of smart distribution network with intelligent agents," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 6, pp. 464–472, 2018. DOI: 10.1109/TETCI.2018.2822832.

[66] M. J. H. Moghaddam, A. Kalam, J. Shi, S. A. Nowdeh, F. H. Gandoman, and A. Ahmadi, "A new model for reconfiguration and distributed generation allocation in distribution network considering power quality indices and network losses," *IEEE Systems Journal*, vol. 14, no. 3, pp. 3530–3538, 2020.

[67] P. Moghari, R. M. Chabanloo, and H. Torkaman, "Distribution system reconfiguration based on milp considering voltage stability," *Electric Power Systems Research*, vol. 222, p. 109 523, 2023, ISSN: 0378-7796. DOI: https://doi.org/10.1016/j.epsr.2023.109523.

[68] F. Mohammadi, M. Sahraei-Ardakani, D. N. Trakas, and N. D. Hatziargyriou, "Machine learning assisted stochastic unit commitment during hurricanes with predictable line outages," *IEEE Transactions on Power Systems*, vol. 36, no. 6, pp. 5131–5142, 2021. DOI: 10.1109/TPWRS.2021.3069443.

[69] M. A. Muhammad, H. Mokhlis, K. Naidu, A. Amin, J. F. Franco, and M. Othman, "Distribution network planning enhancement via network reconfiguration and dg integration using dataset approach and water cycle algorithm," *Journal of Modern Power Systems and Clean Energy*, vol. 8, no. 1, pp. 86–93, 2020. DOI: 10.35833/MPCE.2018.000503.

[70] M. Naguib, W. A. Omran, and H. E. A. Talaat, "Performance enhancement of distribution systems via distribution network reconfiguration and distributed generator allocation considering uncertain environment," *Journal of Modern Power Systems and Clean Energy*, vol. 10, no. 3, pp. 647–655, 2022.

[71] T. T. Nguyen, T. T. Nguyen, N. A. Nguyen, and T. L. Duong, “A novel method based on coyote algorithm for simultaneous network reconfiguration and distribution generation placement,” *Ain Shams Engineering Journal*, vol. 12, no. 1, pp. 665–676, 2021, ISSN: 2090-4479. DOI: https://doi.org/10.1016/j.asej.2020.06.005.

[72] T. T. Nguyen and A. V. Truong, “Distribution network reconfiguration for power loss minimization and voltage profile improvement using cuckoo search algorithm,” *International Journal of Electrical Power & Energy Systems*, vol. 68, pp. 233–242, 2015, ISSN: 0142-0615. DOI: https://doi.org/10.1016/j.ijepes.2014.12.075.

[73] S. H. Oh, Y. T. Yoon, and S. W. Kim, “Online reconfiguration scheme of self-sufficient distribution network based on a reinforcement learning approach,” *Applied Energy*, vol. 280, p. 115 900, 2020, ISSN: 0306-2619. DOI: https://doi.org/10.1016/j.apenergy.2020.115900.

[74] V. B. Pamshetti, S. Singh, and S. P. Singh, “Combined impact of network reconfiguration and volt-var control devices on energy savings in the presence of distributed generation,” *IEEE Systems Journal*, vol. 14, no. 1, pp. 995–1006, 2020.

[75] R. A. Pegado and Y. P. M. Rodriguez, “Distribution network reconfiguration with the opendss using improved binary particle swarm optimization,” *IEEE Latin America Transactions*, vol. 16, no. 6, pp. 1677–1683, 2018. DOI: 10.1109/TLA.2018.8444386.

[76] C. Peng, L. Xu, X. Gong, H. Sun, and L. Pan, “Molecular evolution based dynamic reconfiguration of distribution networks with dgs considering three-phase balance and switching times,” *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 1866–1876, 2019. DOI: 10.1109/TII.2018.2866301.

[77] E. Quintana and E. Inga, “Optimal reconfiguration of electrical distribution system using heuristic methods with geopositioning constraints,” *Energies*, vol. 15, no. 15, 2022, ISSN: 1996-1073. DOI: 10.3390/en15155317.

[78] S. Rasheed and A. R. Abhyankar, “Efficient operational planning of active distribution network by embedding uncertainties and network reconfiguration,” *Electric Power Systems Research*, vol. 216, p. 109 036, 2023, ISSN: 0378-7796. DOI: https://doi.org/10.1016/j.epsr.2022.109036.

[79] S.-M. Razavi, H.-R. Momeni, M.-R. Haghifam, and S. Bolouki, “Multi-objective optimization of distribution networks via daily reconfiguration,” *IEEE Transactions on Power Delivery*, vol. 37, no. 2, pp. 775–785, 2022. DOI: 10.1109/TPWRD.2021.3070796.

[80] S. F. Santos, M. Gough, D. Z. Fitiwi, J. Pogeira, M. Shafie-khah, and J. P. S. Catalão, “Dynamic distribution system reconfiguration considering distributed renewable energy sources and energy storage systems,” *IEEE Systems Journal*, vol. 16, no. 3, pp. 3723–3733, 2022. DOI: 10.1109/JSYST.2021.3135716.

[81] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, *Prioritized experience replay*, 2015. DOI: 10.48550/ARXIV.1511.05952. [Online]. Available: https://arxiv.org/abs/1511.05952.

[82] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, *Proximal policy optimization algorithms*, 2017. DOI: 10.48550/ARXIV.1707.06347. [Online]. Available: https://arxiv.org/abs/1707.06347.

[83] H. Shi, M. Xu, and R. Li, "Deep learning for household load forecasting—a novel pooling deep rnn," *IEEE Transactions on Smart Grid*, vol. 9, no. 5, pp. 5271–5280, 2018. DOI: 10.1109/TSG.2017.2686012.

[84] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, *Smoothgrad: Removing noise by adding noise*, 2017. arXiv: 1706.03825 [cs.LG].

[85] Y. Song, Y. Zheng, T. Liu, S. Lei, and D. J. Hill, "A new formulation of distribution network reconfiguration for reducing the voltage volatility induced by distributed generation," *IEEE Transactions on Power Systems*, vol. 35, no. 1, pp. 496–507, 2020. DOI: 10.1109/TPWRS.2019.2926317.

[86] B. Stojanović, T. Rajić, and D. Šošić, "Distribution network reconfiguration and reactive power compensation using a hybrid simulated annealing – minimum spanning tree algorithm," *International Journal of Electrical Power & Energy Systems*, vol. 147, p. 108 829, 2023, ISSN: 0142-0615. DOI: https://doi.org/10.1016/j.ijepes.2022.108829.

[87] H.-Y. Su and H.-H. Hong, "An intelligent data-driven learning approach to enhance online probabilistic voltage stability margin prediction," *IEEE Transactions on Power Systems*, vol. 36, no. 4, pp. 3790–3793, 2021. DOI: 10.1109/TPWRS.2021.3067150.

[88] A. Swarnkar, N. Gupta, and K. Niazi, "Adapted ant colony optimization for efficient reconfiguration of balanced and unbalanced distribution systems for loss minimization," *Swarm and Evolutionary Computation*, vol. 1, no. 3, pp. 129–137, 2011, ISSN: 2210-6502. DOI: https://doi.org/10.1016/j.swevo.2011.05.004.

[89] S. Tao, R. Qiu, Y. Ping, and H. Ma, "Multi-modal knowledge-aware reinforcement learning network for explainable recommendation," *Knowledge-Based Systems*, vol. 227, p. 107 217, 2021, ISSN: 0950-7051.

[90] L. Thurner *et al.*, "Pandapower—an open-source python tool for convenient modeling, analysis, and optimization of electric power systems," *IEEE Transactions on Power Systems*, vol. 33, no. 6, pp. 6510–6521, 2018. DOI: 10.1109/TPWRS.2018.2829021.

[91] T. V. Tran, B.-H. Truong, T. P. Nguyen, T. A. Nguyen, T. L. Duong, and D. N. Vo, "Reconfiguration of distribution networks with distributed generations using an improved neural network algorithm," *IEEE Access*, vol. 9, pp. 165 618–165 647, 2021. DOI: 10.1109/ACCESS.2021.3134872.

[92] T. T. Tran, K. H. Truong, and D. N. Vo, "Stochastic fractal search algorithm for reconfiguration of distribution networks with distributed generations," *Ain Shams Engineering Journal*, vol. 11, no. 2, pp. 389–407, 2020, ISSN: 2090-4479.

[93] T. T. Tran, K. H. Truong, and D. N. Vo, "Stochastic fractal search algorithm for reconfiguration of distribution networks with distributed generations," *Ain Shams Engineering Journal*, vol. 11, no. 2, pp. 389–407, 2020, ISSN: 2090-4479. DOI: https://doi.org/10.1016/j.asej.2019.08.015.

[94] T. S. UNESP-FEIS Electrical Energy Systems Planning Laboratory Homepage, Accessed: Feb 10, 2022. [Online]. Available: https://www.feis.unesp.br/#!/lapsee.

[95] B. Wang, H. Zhu, H. Xu, Y. Bao, and H. Di, "Distribution network reconfiguration based on noisynet deep q-learning network," *IEEE Access*, vol. 9, pp. 90 358–90 365, 2021. DOI: 10.1109/ACCESS.2021.3089625.

[96] H.-J. Wang, J.-S. Pan, T.-T. Nguyen, and S. Weng, "Distribution network reconfiguration with distributed generation based on parallel slime mould algorithm," *Energy*, vol. 244, p. 123 011, 2022, ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2021.123011.

[97] J. Wang, W. Wang, H. Wang, and H. Zuo, "Dynamic reconfiguration of multiobjective distribution networks considering dg and evs based on a novel ldbas algorithm," *IEEE Access*, vol. 8, pp. 216 873–216 893, 2020. DOI: 10.1109/ACCESS.2020.3041398.

[98] J. Wang, W. Wang, Z. Yuan, H. Wang, and J. Wu, "A chaos disturbed beetle antennae search algorithm for a multiobjective distribution network reconfiguration considering the variation of load and dg," *IEEE Access*, vol. 8, pp. 97 392–97 407, 2020. DOI: 10.1109/ACCESS.2020.2997378.

[99] W. Wang, J. Xu, G. Zhang, M. Yang, X. Xu, and H. Armghan, "A novel chance constrained joint optimization method under uncertainties in distribution networks," *International Journal of Electrical Power & Energy Systems*, vol. 147, p. 108 849, 2023, ISSN: 0142-0615. DOI: https://doi.org/10.1016/j.ijepes.2022.108849.

[100] X. Wang, Y. Gu, Y. Cheng, A. Liu, and C. L. P. Chen, "Approximate policy-based accelerated deep reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 6, pp. 1820–1830, 2020. DOI: 10.1109/TNNLS.2019.2927227.

[101] Z. Wang, T. Schaul, M. Hessel, H. van Hasselt, M. Lanctot, and N. de Freitas, *Dueling network architectures for deep reinforcement learning*, 2015. DOI: 10.48550/ARXIV.1511.06581. [Online]. Available: https://arxiv.org/abs/1511.06581.

[102] L. Wells and T. Bednarz, "Explainable ai and reinforcement learning—a systematic review of current approaches and trends," *Frontiers in Artificial Intelligence*, vol. 4, 2021, ISSN: 2624-8212. DOI: 10.3389/frai.2021.550030. [Online]. Available: https://www.frontiersin.org/articles/10.3389/frai.2021.550030.

[103] H. Wu, P. Dong, and M. Liu, "Distribution network reconfiguration for loss reduction and voltage stability with random fuzzy uncertainties of renewable energy generation and load," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 5655–5666, 2020. DOI: 10.1109/TII.2018.2871551.

[104] T. Wu, J. Wang, X. Lu, and Y. Du, "Ac/dc hybrid distribution network reconfiguration with microgrid formation using multi-agent soft actor-critic," *Applied Energy*, vol. 307, p. 118 189, 2022, ISSN: 0306-2619.

[105] N. Xia *et al.*, "Fuzzy logic based network reconfiguration strategy during power system restoration," *IEEE Systems Journal*, vol. 16, no. 3, pp. 4735–4743, 2022. DOI: 10.1109/JSYST.2021.3123325.

[106] J. Xin, *PPO-discrete-pytorch*, https://github.com/XinJingHao/PPO-Discrete-Pytorch, 2022.

[107] T. Yamada, S. Kataoka, and K. Watanabe, "Listing all the minimum spanning trees in an undirected graph," *International Journal of Computer Mathematics*, vol. 87, no. 14, pp. 3175–3185, 2010.

[108] K. Yang, X. Kong, Y. Wang, J. Zhang, and G. De Melo, "Reinforcement learning over knowledge graphs for explainable dialogue intent mining," *IEEE Access*, vol. 8, pp. 85 348–85 358, 2020.

[109] T. Yang, Y. Guo, L. Deng, H. Sun, and W. Wu, "A linear branch flow model for radial distribution networks and its application to reactive power optimization and network reconfiguration," *IEEE Transactions on Smart Grid*, vol. 12, no. 3, pp. 2027–2036, 2021. DOI: 10.1109/TSG.2020.3039984.

[110] Z. Yin, S. Wang, and Q. Zhao, "Sequential reconfiguration of unbalanced distribution network with soft open points based on deep reinforcement learning," *Journal of Modern Power Systems and Clean Energy*, vol. 11, no. 1, pp. 107–119, 2023.

[111] L. Yun, D. Wang, and L. Li, "Explainable multi-agent deep reinforcement learning for real-time demand response towards sustainable manufacturing," *Applied Energy*, vol. 347, p. 121 324, 2023, ISSN: 0306-2619.

[112] J. Zhan, W. Liu, C. Y. Chung, and J. Yang, "Switch opening and exchange method for stochastic distribution network reconfiguration," *IEEE Transactions on Smart Grid*, vol. 11, no. 4, pp. 2995–3007, 2020. DOI: 10.1109/TSG.2020.2974922.

[113] K. Zhang, J. Zhang, P.-D. Xu, T. Gao, and D. W. Gao, "Explainable ai in deep reinforcement learning models for power system emergency control," *IEEE Transactions on Computational Social Systems*, vol. 9, no. 2, pp. 419–427, 2022.

[114] T. Zhang, M. Sun, J. L. Cremer, N. Zhang, G. Strbac, and C. Kang, "A confidence-aware machine learning framework for dynamic security assessment," *IEEE Transactions on Power Systems*, vol. 36, no. 5, pp. 3907–3920, 2021. DOI: 10.1109/TPWRS.2021.3059197.

[115] X. Zhang, W. Hua, Y. Liu, J. Duan, Z. Tang, and J. Liu, “Reinforcement learning for active distribution network planning based on monte carlo tree search,” *International Journal of Electrical Power & Energy Systems*, vol. 138, p. 107 885, 2022, ISSN: 0142-0615.

[116] Z. Zhang, D. Zhang, and R. C. Qiu, “Deep reinforcement learning for power system applications: An overview,” *CSEE Journal of Power and Energy Systems*, vol. 6, no. 1, pp. 213–225, 2020. DOI: 10.17775/CSEEJPES.2019.00920.

[117] W. Zheng, W. Huang, D. J. Hill, and Y. Hou, “An adaptive distributionally robust model for three-phase distribution network reconfiguration,” *IEEE Transactions on Smart Grid*, vol. 12, no. 2, pp. 1224–1237, 2021. DOI: 10.1109/TSG.2020.3030299.

[118] W. Zheng, W. Huang, D. J. Hill, and Y. Hou, “An adaptive distributionally robust model for three-phase distribution network reconfiguration,” *IEEE Transactions on Smart Grid*, vol. 12, no. 2, pp. 1224–1237, 2021. DOI: 10.1109/TSG.2020.3030299.

[119] T. Zhong, H.-T. Zhang, Y. Li, L. Liu, and R. Lu, “Bayesian learning-based multi-objective distribution power network reconfiguration,” *IEEE Transactions on Smart Grid*, vol. 12, no. 2, pp. 1174–1184, 2021. DOI: 10.1109/TSG.2020.3027290.

[120] A. Zhou, H. Zhai, M. Yang, and Y. Lin, “Three-phase unbalanced distribution network dynamic reconfiguration: A distributionally robust approach,” *IEEE Transactions on Smart Grid*, vol. 13, no. 3, pp. 2063–2074, 2022. DOI: 10.1109/TSG.2021.3139763.