

DEEP LEARNING FOR REMOTE STATE ESTIMATION UNDER PHYSICAL LAYER ATTACKS

By
Ke Wang

A THESIS SUBMITTED IN FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY IN ENGINEERING
AT
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING
THE UNIVERSITY OF SYDNEY

OCTOBER 2024

© Copyright by **Ke Wang**, 2024

*To the spirit, resilience, and wisdom
of my mother.*

Acknowledgements

This thesis would not have been possible without the support, guidance, and encouragement of many individuals to whom I am deeply grateful.

First and foremost, I would like to express my profound gratitude to my supervisors, Professor Teng Joon Lim and Doctor Wanchun Liu. Their unwavering support, insightful feedback, and constant encouragement have been invaluable throughout this journey. Their expertise and dedication have not only guided my research but also inspired me to strive for excellence. I am deeply thankful for their patience and for always being available to discuss ideas, provide constructive criticism, and offer words of wisdom. Their mentorship has been a cornerstone of my academic development, and I am honored to have had the opportunity to work under their guidance.

I am also immensely grateful to my wife, Yusen Gan. Her love, patience, and unwavering support have been my anchor throughout this challenging journey. She has been my greatest source of strength, always believing in me even when I doubted myself. Her sacrifices, understanding, and encouragement have made it possible for me to dedicate myself fully to this research. I am deeply thankful for her constant presence, her comforting words, and her ability to lift my spirits during the toughest times. This thesis is as much hers as it is mine, and I am forever indebted to her for her unwavering support and love.

To my father, Xianran Wang, I owe a debt of gratitude that words cannot fully express. His wisdom, guidance, and unconditional support have been a constant source of inspiration. He has always encouraged me to pursue my dreams and has been a pillar of strength throughout my life. His belief in my abilities has given me the confidence to overcome obstacles and strive for success. I am deeply thankful for his sacrifices, his patience, and his unwavering faith in me. His love and support have been instrumental in my academic and personal growth, and I am profoundly grateful for everything he has done for me.

I would also like to extend my heartfelt thanks to my colleagues and friends who have supported me throughout this journey. Their camaraderie, encouragement, and insightful discussions have enriched my research experience and made this journey more enjoyable. I am grateful for their friendship and for the many memorable moments we have shared.

Lastly, I would like to thank everyone who has contributed to this thesis in one way or another again. Your support, encouragement, and belief in me have been invaluable, and I am deeply grateful to each and every one of you.

This thesis is a testament to the collective support and encouragement of all these wonderful individuals. Thank you all for being a part of this journey.

Sydney, Australia
October, 2024

Ke Wang

Statement of Originality

I hereby certify that the work embodied in the thesis is my own work, conducted under normal supervision. The thesis contains no material which has been accepted, or is being examined, for the award of any other degree or diploma in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made. I give consent to the final version of my thesis being made available worldwide when deposited in the University's Digital Repository, subject to the provisions of the Copyright Act 1968 and any approved embargo.

Ke Wang
School of Electrical and Computer Engineering
The University of Sydney
October, 2024

Acknowledgment OF Authorship

I hereby certify that the work embodied in this thesis contains published/submitted research of which I am a joint author. I have included as part of the thesis a written declaration endorsed in writing by my supervisor, attesting to my contribution to the joint published/submitted research.

Chapter 3 of this thesis is published as [J1] and [C1], as listed in Section 1.4.

Chapter 4 of this thesis contain the materials in submitted paper [J2], as listed in Section 1.4.

This research reported in this thesis was supported by the award of a Faculty of Engineering scholarship from the University of Sydney to the PhD Candidate.

Ke Wang
October, 2024

By signing below I confirm that Ke Wang contributed to all publications embodied in this thesis

Name of Supervisor: Teng Joon Lim

Signature of Supervisor:

Date :

Abstract

With the rapid developments in sensing and communication technologies, cyber-physical systems (CPSs) have been widely applied in various engineering fields, playing a crucial role in the era of Industry 4.0. CPSs typically integrate spatially distributed plants, sensors, machines, and controllers to achieve the desired performance of physical processes. However, the integration of communication networks in CPSs makes them susceptible to malicious attacks, such as denial of service (DoS) attacks. These attackers not only disrupt communication networks but also pose significant threats to physical systems. Therefore, it is essential to design efficient control approaches to ensure the security of the CPSs in the presence of such attacks. Conventional security systems often lack effectiveness against these sophisticated attackers, whereas machine learning (ML) techniques have shown great promise in various cyber-security applications. This thesis focuses on addressing remote state estimation problems under physical layer DoS attacks using deep learning algorithms.

Firstly, we focus on a wireless resource allocation problem of remote state estimation in wireless networked CPSs, where information from multiple sensors is transmitted to a central gateway over a wireless network with fewer channels than sensors. Channel and power allocation are performed jointly, in the presence of a denial of service (DoS) attack where one or more channels are jammed by an attacker transmitting spurious signals. The attack policy is unknown and the central gateway has the objective of minimizing state estimation error with maximum energy efficiency. The problem involves a combination of discrete and continuous action spaces. In addition, the state and action spaces have high dimensionality, and the channel states are not fully known to the defender. We propose an innovative model-free deep reinforcement learning (DRL) algorithm to address the problem. In addition, we develop a deep learning-based method with a novel deep neural network (DNN) structure for detecting changes in the attack policy post-training. The proposed online policy change detector accelerates the adaptation of the defender to a new attack policy and also saves computational resources compared to continuous training. In short, a complete system featuring a DRL-based defender that is trained initially and adapts continually to changes in attack policy has been developed. Our numerical results show that the proposed intelligent system can significantly enhance the resilience of the system to DoS attacks.

Next, we further consider a wireless resource allocation problem in a CPS where the control channel, carrying resource allocation commands, is subjected to DoS attacks. We propose a novel concept of collaborative distributed and centralized (CDC) resource allocation to effectively mitigate the impact of these attacks. To optimize the CDC resource allocation policy, we develop a new CDC-DRL algorithm, whereas existing DRL frameworks only formulate either centralized or distributed

decision-making problems. Simulation results demonstrate that the CDC-DRL algorithm significantly outperforms state-of-the-art DRL benchmarks, showcasing its ability to address resource allocation problems in large-scale CPSs under control channel attacks.

Lastly, we explore a wireless resource allocation problem in a two-hop relay system, which is vulnerable to intelligent data channel DoS attackers. In this scenario, we propose a DRL agent that leverages domain knowledge to enhance learning efficiency and optimize the joint resource allocation problem for both the sensor-relay and relay-server segments. The proposed architecture demonstrates superior performance compared to state-of-the-art DRL benchmarks, significantly improving the system's robustness and overall performance. Our results highlight the potential of DRL-based approaches in addressing complex resource allocation problems under DoS attack, paving the way for more resilient and efficient systems.

Table of Contents

Acknowledgements	iii
Statement of Originality	v
Acknowledgment OF Authorship	vii
Abstract	ix
Table of Contents	xi
List of Tables	xv
List of Figures	xvii
List of Acronyms	xix
List of Symbols and Notations	xxi
1 Introduction	1
1.1 Background of Cyber-Physical Systems (CPSs)	1
1.1.1 Definition of CPSs	1
1.1.2 Scope of CPSs	2
1.1.3 Remote State Estimation	3
1.1.4 CPS Security	4
1.1.5 Challenges and Opportunities	5
1.2 Related Work	6
1.2.1 Security Problems	6
1.2.2 Resource Allocation Problems	8
1.2.3 Deep Learning Applications	11
1.3 Research Problems	12
1.3.1 Problem 1	13
1.3.2 Problem 2	13
1.3.3 Problem 3	13
1.4 List of Publications	13
1.5 Thesis Outline	14

2	Preliminaries of Related Methodologies	15
2.1	Chapter Introduction	15
2.2	Machine Learning	15
2.2.1	Supervised Learning	16
2.2.2	Unsupervised learning	17
2.2.3	Reinforcement Learning	17
2.3	Deep Learning	18
2.3.1	Feedforward Neural Networks	19
2.3.2	Recurrent Neural Networks	20
2.4	Deep Reinforcement Learning	21
2.4.1	Deep Q-Networks (DQNs)	21
2.4.2	Deep Deterministic Policy Gradient (DDPG)	23
2.5	Chapter Conclusions	25
3	Deep Learning for Radio Resource Allocation Under DoS Attack	27
3.1	Chapter Introduction	27
3.2	System Model	31
3.2.1	Sensing Model	32
3.2.2	Scheduling and Data Channel Model under DoS Attack	33
3.2.3	Remote State Estimation at Gateway	35
3.3	Defense and Attack Policy Design	35
3.3.1	Gateway Design	35
3.3.2	Attacker Design	36
3.3.3	DRL-based Solution for Defense Policy Design	37
3.3.4	DRL-Based Solution for Attack Policy Design	40
3.4	Feedback Channel DoS Attack	40
3.5	DNN-based Detector of Attack Policy Changes	42
3.5.1	Conceptual Problem Formulation	43
3.5.2	DNN-Based Detector Design	44
3.6	Experiment Results	46
3.6.1	Dynamic Resource Allocation under Data Channel DoS Attack	46
3.6.2	Resource Allocation under Feedback Channel DoS Attack	52
3.6.3	Data Channel DoS Attack Policy Change Detector	53
3.7	Chapter Conclusions	55
4	Wireless Resource Allocation with Collaborative Distributed and Centralized DRL under Control Channel Attacks	57
4.1	Chapter Introduction	57
4.2	System Model	59
4.2.1	Remote State Estimation with Smart Sensors	60
4.2.2	NOMA-based Data Transmission Channel	61
4.2.3	DoS Attacker at Broadcast Control Channel	63
4.2.4	Collaborative Distributed and Centralized Power Allocation	64
4.3	Collaborative Distributed-Centralized DRL	64
4.3.1	Network Architecture of CDC-DRL	65
4.3.2	Two-stage Training of CDC-DRL	66
4.4	Numerical Results	69

4.4.1	Simulation setting	69
4.4.2	Testing Error Comparison	71
4.4.3	Convergence Analysis	73
4.5	Chapter Conclusions	73
5	Wireless Resource Allocation of a two-hop relay network under DoS attack	77
5.1	Chapter Introduction	77
5.2	System Model	79
5.2.1	Local State Estimation with Smart Sensors	79
5.2.2	Cooperative Transmission of Two-hop Relay Network	80
5.2.3	Remote State Estimation	81
5.2.4	NOMA-based Data Transmission Channel	81
5.3	Defending and Attacking Policies	85
5.3.1	Defender design	85
5.3.2	Attacker design	85
5.3.3	Problem setup	86
5.4	Deep Reinforcement Learning Algorithms	87
5.4.1	Defender's Algorithm	87
5.4.2	DoS Attacker's Algorithm	91
5.5	Numerical Results	92
5.6	Chapter Conclusions	94
6	Conclusions and Future Work	97
6.1	Chapter Introduction	97
6.2	Summary of Results	97
6.3	Future Work	98
	Bibliography	99

List of Tables

2	Summary of notations.	xxii
3.1	Simulation result of average reward (r_k) under fixed attack policy	48
3.2	Joint testing result after defender's and attacker's training	50
3.3	Simulation result of average reward (r_k) under feedback channel attack	53
3.4	Average testing error rate of detectors	53
4.1	Summary of simulation parameters for CDC-DRL	70
4.2	Control channel attacker dynamics	70
5.1	Summary of simulation parameters for knowledge-based HDDPG	93
5.2	Summary of neural network parameters of knowledge-based HDDPG	94

List of Figures

1.1	Building blocks of CPS. [1]	2
2.1	Taxonomy of machine learning.	16
2.2	A simple feedforward neural network with two hidden layers.	19
3.1	Remote state estimation with sensor scheduling jammed by DoS attack.	31
3.2	The HDDPG architecture.	39
3.3	Integrated online attack policy change detection and defender policy re-training.	42
3.4	Online sequential change detection mechanism.	43
3.5	DNN-based attack policy change detector.	44
3.6	Training process and performance of HDDPG.	49
3.7	Training process and performance of HDDPG: effect of truncated state space and pre-training	50
3.8	Testing result of scalability: N and M	51
3.9	Testing result of scalability: $E_{\max}^s$ and $E_{\max}^a$	51
3.10	Convergence distribution	52
3.11	Estimated PDF of online policy change detection delay.	54
3.12	HDDPG training with and without the proposed policy change detector.	55
4.1	Multi-sensor remote state estimation under control channel DoS attack. Actor and critic modules (in blue) will be explained in Section 4.3.	59
4.2	The network architecture of CDC-DRL	66
4.3	Comparison of average testing costs in small-scale systems.	71
4.4	Comparison of average testing costs in large-scale systems.	72
4.5	Comparison of average testing costs across the systems.	72
4.6	Convergence analysis of CDC-DRL in small scale systems.	74
4.7	Convergence analysis of CDC-DRL in large scale systems.	75
5.1	Multi-sensor remote state estimation in a two-hop relay network under DoS attack.	79
5.2	HDDPG architecture for resource allocation in a two-hop relay network.	89
5.3	Performance comparison of the algorithms	95

List of Acronyms

CPS	Cyber-Physical System
WSN	Wireless Sensor Network
IoT	Internet of Things
FDI	False Data Injection
DoS	Denial-of-Service
UAV-BS	Unmanned Aerial Vehicle Base Station
AI	Artificial Intelligence
ML	Machine Learning
PCA	Principal Component Analysis
t-SNE	t-distributed Stochastic Neighbor Embedding
SVM	Support Vector Machine
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
MAB	Multi-Armed Bandit
CUSUM	Cumulative Sum Procedure
RL	Reinforcement Learning
MDP	Markov Decision Process
MAB	Multi-Armed Bandit
DL	Deep Learning
DNN	Deep Neural Network
FFNN	Feedforward Neural Network
RNN	Recurrent Neural Network
BPTT	Backpropagation Through Time
CNN	Convolutional Neural Network
LSTM	Long Short-Term Memory
GRU	Gated Recurrent Unit
DRL	Deep Reinforcement Learning

DQN	Deep Q-Network
DDPG	Deep Deterministic Policy Gradient
HDDPG	Hybrid Deep Deterministic Policy Gradient
MADDPG	Multi-Agent Deep Deterministic Policy Gradient
CDC-DRL	Collaborative Distributed-Centralized Deep Reinforcement Learning
TD	Temporal Difference
MSE	Mean-Square Error
CUSUM	Cumulative Sum Procedure
IID	Independent and identically distributed
OU	Ornstein-Uhlenbeck
OFDMA	Orthogonal Frequency-Division Multi-Access
NOMA	Non-Orthogonal Multiple Access
SINR	Signal-to-Interference-plus-Noise Ratio
AWGN	Additive White Gaussian Noise
IRC	Interference Rejection Combining
SIC	Successive Interference Cancellation

List of Symbols and Notations

Table 2: Summary of notations.

Notation		Description	
Sensing Model			
	$\mathbf{x}_{i,k}$		State of process i at time k
	$\mathbf{A}_i$		State transition matrix
	$\mathbf{y}_{i,k}$		Measurement of process
	$\hat{\mathbf{x}}_{i,k}^s$		Local state estimate
	$\mathbf{P}_{i,k}^s$		Local estimation error co-variance
OFDMA Data Channel Model			
	$\rho_{m,k}$		SINR of channel m at time k
	$\gamma_{m,k}$		Packet transmission outcome of channel m
Sensor		Attacker	
$E_{m,k}^s$		$E_{m,k}^a$	Allocated transmission power
$G_{m,i}^s$		G_m^a	Channel gain to the gateway
NOMA Data Channel Model			
Sensor	Relay	Attacker	
$\mathbf{E}_{i,k}$	$\tilde{\mathbf{E}}_k$	$\mathbf{a}_k^a$	Allocated transmission power
$\mathbf{G}_{i,k}$	$\tilde{\mathbf{G}}_k$	$\mathbf{G}_k^a$ and $\tilde{\mathbf{G}}_k^a$	Channel state vector
Broadcast Control Channel Model			
	$\rho_{i,k}^d$		SINR of control channel for sensor i at time k
	β_k		Control signal transmission outcome
Server		Attacker	
E^d		E_k^a	Allocated transmission power
$g_{i,k}^d$		$g_{i,k}^a$	Channel gain to sensor i
Remote Estimation			
Server		Relay	
$\hat{\mathbf{x}}_{i,k}$		$\tilde{\mathbf{x}}_{i,k}$	Remote state estimate
$\mathbf{P}_{i,k}$		n.a.	Remote estimation error co-variance
$\tau_{i,k}$		$\lambda_{i,k}$	Age of information of sensor i 's packet
$\gamma_{i,k}$		$\phi_{i,k}$	Packet transmission outcome of sensor i 's packet
DRL			
Centralized defender	Distributed defender at sensor i	Attacker	
π or μ	κ_i	π^a	Policy
A_k^c and A_k^p	n.a.	A_k^a	Actions for OFDMA channel
$\mathbf{a}_k^c$ or $\mathbf{A}_k$	$\mathbf{a}_k^d$	$\mathbf{a}_k^a$	Actions for NOMA channel
O_k or $\mathbf{O}_k^c$	$\mathbf{O}_{i,k}^d$	$\mathbf{O}_k^a$	Observation state
r_k	n.a.	r_k^a	Per-step reward
c_k	c_k	n.a.	Per-step cost
Detector			
	$\kappa_{k'},k$		Binary label for attacker policy change
	$\hat{\kappa}_{k'},k$		Predicted output of detector
	O_k^d		Defender's state-action pair
	$\mathbf{L}_{k'}$		Anchored sequence
	$\mathbf{L}_k$		Sliding sequence
	$\mathbf{O}_{g,h}^d$		Time series of observation under policy pair (π_g, π_h^a)
	$\mathbf{L}_{k'}^f$		Anchored sequence in offline training
	$\mathbf{L}_k^f$		Sliding sequence in offline training
	$\kappa_{k'},k^f$		Label for offline training data $(\mathbf{L}_k^f, \mathbf{L}_{k'}^f)$

Chapter 1

Introduction

1.1 Background of Cyber-Physical Systems (CPSs)

1.1.1 Definition of CPSs

Cyber-Physical Systems (CPSs) represent a revolutionary integration of computation, networking, and physical processes. These systems interact with the physical world through sensors, actuators, and computational algorithms, enabling real-time monitoring, control, and optimization of complex processes [2]. CPSs facilitate the inter-connection of physical and virtual environments, but may require significant computational power, communication channels, and storage for effective monitoring and control. At a high level, CPSs consist of five major components: sensors, communication, actuators, storage, and computing, as illustrated in Fig. 1.1. These components are deeply integrated to form a distributed control system capable of operating across various dimensions. The convergence of these domains has led to significant advancements, establishing advanced connectivity for real-time data acquisition, management, processing, and intelligent analysis.

Rapid advances in computing and communication technologies have enabled seamless and intelligent interfacing between the cyber and physical worlds, where information is exchanged in a real-time manner. The interaction among computers, networks, and physical systems necessitates fundamentally new design technologies. These technologies span multiple disciplines, including embedded systems, computing, and communications, with software embedded in devices whose primary function extends beyond computation alone [1]. This multidisciplinary approach ensures that CPSs can effectively manage and optimize the interactions between the physical and digital realms.

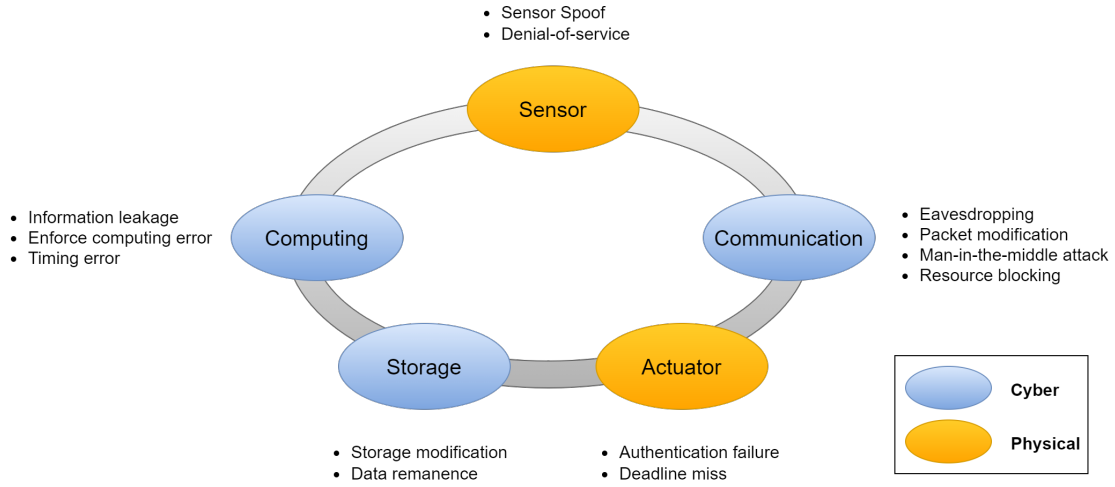


Figure 1.1: Building blocks of CPS. [1]

1.1.2 Scope of CPSs

Unlike traditional embedded systems designed as stand-alone devices, CPSs focus on networking multiple devices. This trend aligns with the increasing demand for remote information sharing and control, even in household appliances. Applications such as remote heating or cooling systems and programmable washing machines exemplify daily uses of CPS concepts. In recent years, CPSs have become ubiquitous across various fields [3]. We will briefly cover their applications in manufacturing, medical care, and intelligent transportation.

In the manufacturing domain, CPSs improve and automate processes such as maintenance [4], production planning [5], and energy consumption management [6]. These systems integrate physical processes with computing and networking components to provide real-time monitoring and control [7]. By matching capabilities to real-time conditions, CPSs optimize manufacturing processes, enhancing productivity, flexibility, and precision in operations. This integration allows for more efficient resource utilization and better response to dynamic changes in the manufacturing environment.

In the medical field, CPSs have significantly impacted healthcare by integrating advanced computational tools with medical devices for real-time patient monitoring, diagnosis, and treatment [8]. These systems use Wireless Sensor Networks (WSNs) and Internet of Things (IoT) technologies

to continuously gather patient data, which is then processed for clinical decision-making. Applications range from wearable devices for remote health monitoring to implantable medical devices that track vital signs, improving patient care both in hospitals and at home. The integration of medical devices with cloud computing further enhances scalability, supporting advanced diagnostics and remote care services. This continuous data flow and analysis enable more accurate and timely medical interventions, ultimately improving patient outcomes.

In the domain of intelligent transportation systems, CPSs enhance safety, control, and efficiency [9]. CPS frameworks integrate vehicle dynamics with advanced data-driven systems to enable real-time communication and decision-making between connected and automated vehicles [10]. Each vehicle operates as an independent CPS, leveraging cloud-based data for optimal control, such as adaptive cruise control and collision avoidance. CPS applications in autonomous driving ensure vehicles can predict and react to traffic conditions, merge safely, and maintain optimal performance. This system improves overall traffic flow and safety by utilizing dynamic models and vehicle behavior prediction. The integration of CPSs in transportation not only enhances individual vehicle performance but also contributes to the broader goal of creating smarter, more efficient transportation networks.

1.1.3 Remote State Estimation

WSNs play a critical role in CPSs. The decreasing cost of WSNs has enabled their widespread deployment for monitoring and controlling various environments. Due to physical constraints, cost, or safety concerns, direct sensor measurement or monitoring is often challenging, giving rise to remote state estimation problems. Remote state estimation, where sensor nodes communicate with a remote estimator through a wireless channel, is pivotal in CPSs. Timely and accurate monitoring of dynamic processes in a remote state estimation problem is essential for maintaining system performance, ensuring safety, and facilitating automated decision-making. These CPSs rely on remote state estimation to provide the real-time data that is required for closed-loop control, guaranteeing the reliability and efficiency of operations. Without reliable estimation methods, system operators risk delayed or inaccurate responses, leading to inefficiencies, resource wastage, or even critical system failures.

A prominent example of remote estimation can be found in smart cities, which is characterized by

diverse, distributed sensor networks monitoring infrastructure, transportation, energy consumption, and environmental factors [11]. Sensors collect massive amounts of data across spatially distributed areas in real time, to ensure the efficient functioning of city services and infrastructure. However, physical limitations and the sheer scale of urban environments often prevent direct access to all sensing locations, making remote estimation techniques essential for providing comprehensive, city-wide insights. For instance, in smart traffic management systems, remote estimation predicts vehicle flow and congestion levels based on partial data from sensors embedded in roads or vehicles. This allows city managers to optimize traffic signals and reroute traffic dynamically, minimizing delays and reducing fuel consumption. In energy systems, remote estimation helps monitor and predict power usage patterns across different city districts, enabling better load balancing and the integration of renewable energy sources.

1.1.4 CPS Security

The rapid growth of CPS and its safety-critical applications have generated a surge of interest in CPS security in recent years [12]. Since the measurement and control data in CPS are commonly transmitted through unprotected communication networks, such systems are vulnerable to cyber threats. Any successful CPS attack may lead to a variety of severe consequences, including customer information leakage, damages to national economy, destruction of infrastructure, and even endangering of human lives [13].

A cyber-physical attack is defined as the exploitation or disruption of a CPS by inflicting harm on networks that connect computing and physical systems [14]. The cyber-physical attack space can be divided according to the adversary's system knowledge, disclosure resources and disruption resources. Attack models, such as replay attack, zero dynamic attack, false data injection (FDI) attack and denial-of-service (DoS) attack are analyzed in [15].

Replay attack refers to an adversary recording legitimate data transmissions and replaying them at a later time to deceive the system into accepting false data as genuine [16]. This type of attack can be particularly damaging in control systems, as it can cause the system to operate based on outdated and incorrect information, leading to potential system failures or inefficiencies.

Zero dynamic attack exploits the inherent dynamics of the system to remain undetected while causing disruption [17]. In this attack, the adversary injects signals that align with the system's

zero dynamics, effectively nullifying the system's ability to detect the intrusion. This type of attack requires detailed knowledge of the system's model and can result in severe consequences, as the system continues to operate under the false assumption that it is functioning normally.

False injection attack targets the integrity of the data used by the control system [18]. In this scenario, the attacker introduces a constant bias into the sensor or actuator data, causing the system to make incorrect adjustments based on skewed information. This type of attack can be particularly insidious, as it may not trigger immediate alarms but can lead to long-term degradation of system performance and reliability.

Lastly, DoS attack aims to disrupt the availability of the system by overwhelming it with excessive data traffic or by jamming communication channels [19]. This attack prevents legitimate data from reaching its intended destination, effectively paralyzing the system's ability to function. DoS attacks can be particularly devastating in CPSs, where timely and reliable data transmission is crucial for maintaining control and safety.

1.1.5 Challenges and Opportunities

Our interest in remote estimation stems from its critical role in enhancing the functionality and reliability of CPSs. However, many challenges exist due to the scale, heterogeneity, and dynamic nature of the systems involved. Within the communication network of a remote estimation system, challenges include transmission delays, data packet dropouts, and even malicious attacks, which can significantly degrade system performance. Addressing these challenges requires the development of novel algorithms capable of managing limited wireless radio resources for remote state estimation. Additionally, countering malicious attacks to achieve optimal estimation performance is essential for maintaining the effectiveness and robustness of CPSs.

This thesis aims to address some of these problems by developing advanced deep learning algorithms tailored for the research problems introduced in section 1.3. Our research will explore novel approaches to algorithm design, focusing on improving the robustness and quality of remote estimation systems. Before delving into our problems of interest, we will first discuss related work in this field in the following section.

1.2 Related Work

1.2.1 Security Problems

This section reviews the existing literature on CPS security, focusing on three main aspects: attacker design, detection strategies, and prevention and resilience strategies.

Attacker Design

Designing effective and novel attack strategies is crucial for understanding CPS vulnerabilities. Simulating potential attack scenarios helps identify weaknesses in defense systems and develop mitigation strategies. This anticipation of adversarial methods leads to more robust and secure systems. Additionally, understanding attacker design aids in creating realistic threat models, essential for testing security measures and improving CPS resilience. From the perspective of cyber-physical attacks, existing works in control theory often consider these attacks as mathematical modifications of feedback control system signals [20]. Below are some notable examples.

Guo *et al.* [21] formulated an optimal linear FDI attack on remote state estimation, undetectable by a residue-based false data detector. They provided a closed-form expression for the optimal attack strategy among all linear attacks and analyzed the degradation of system performance by deriving the evolution of the estimation error covariance at the remote estimator under such attacks. Similarly, Zhou *et al.* [22] proposed a different innovation-based linear FDI attack policy that fuses all available information. Building on this, Guo introduced an event-based stealthy FDI attack mechanism to selectively inject false data while evading a residual-based anomaly detector [23]. This mechanism dynamically launches FDI attacks based on real-time residuals, maximizing the trace of the state estimation error covariance matrix under attack stealthiness constraints. Furthermore, Xu *et al.* established a new model of joint FDI attacks, injecting attacks into both the remote estimator and the communication channels [24].

In another line of research, Zhang *et al.* constructed optimal DoS attack scheduling schemes on remote state estimation, considering two performance indexes: the expected average estimation error and the expected terminal estimation error [25]. They also proposed optimal attack schedules to avoid detection mechanisms. Expanding on their initial findings, they addressed a power allocation problem for attackers to degrade remote state estimation performance [26]. They derived a sufficient

condition for the attacker's strategy in a static attack energy allocation problem and formulated an attack power allocation algorithm for general systems. Additionally, Ye *et al.* proposed a sufficient condition to guarantee the optimality of the attack allocation strategy, which maximally degrades system performance, and provided a closed-form optimal attack sequence in a remote state estimation system where sensors send data under a round-robin protocol [27]. Extending this field further, Ding *et al.* considered asymmetric information between the sensor and the DoS attacker, investigating solutions for the attacker in both open-loop and closed-loop cases [28].

Detection Strategies

Detection strategies are vital for identifying and responding to cyber attacks on CPSs in real-time. The ability to detect intrusions promptly can prevent or minimize damage, ensuring the continued safe and efficient operation of these systems. Existing literature often employs statistical models or machine learning techniques to recognize abnormal behavior or unauthorized access in CPS. Effective detection strategies serve as the first line of defense against cyber threats, enabling timely interventions and mitigating potential impacts. Below are some notable examples of cyber attack detection.

Gao *et al.* proposed a distributed fusion detection strategy to improve the speed of alarm response in FDI attacks on CPSs [29]. They utilized a residual-based detector, optimally weighted through a convex optimization problem. Similarly, Biron *et al.* studied a DoS attack detection and estimation problem in both vehicle-to-vehicle and vehicle-to-infrastructure communication within a connected vehicle system [30]. Their detector, organized using sliding mode and adaptive estimation theory, was analyzed via Lyapunov's stability theory. Extending this field, Guan *et al.* addressed the joint problem of distributed attack detection and secure estimation for networked CPSs [31]. They assumed the attacker could launch FDI attacks at the physical layer and DoS attacks to block wireless transmission channels between the sensor and the remote estimator. Their proposed solution included a two-step attack detection mechanism for FDI attacks and a model-based DoS attack detector. Raghuvamsi studied the FDI attack and DoS attack detection in distribution system state estimation [32]. Their approach combines the advantages of the attention mechanism and temporal convolutional network to capture spatio-temporal information, and outperforms other machine learning approaches like denoising autoencoder.

Hosseinzadeh *et al.* analyzed the effect of replay attacks on constrained cyber-physical systems subject to linear probabilistic constraints and proposed a detection scheme [33]. Additionally, Zhao *et al.* proposed a replay attack detection scheme for CPS, combining dynamical system characteristics with data-driven implementation, which only needs to be implemented on the control and monitoring side [34]. Naha *et al.* used a cumulative sum test to address replay attack detection in CPS, where the physical process is modeled by a discrete-time linear time-invariant system [35].

Prevention and Resilient Strategies

In contrast to attack analysis and detection, prevention and resilience strategies focus on fortifying systems against potential attacks and maintaining functionality during disruptions. Prevention strategies aim to address vulnerabilities and implement robust security protocols, while resilient strategies ensure quick recovery and continued operation. This approach includes a wide range of methods, from data encryption [36] to fault isolation techniques [37]. Below are some notable examples.

Wang *et al.* investigated the remote robust state estimation problems for nonlinear cyber-physical systems under DoS attack [38]. They proposed Fauser *et al.* demonstrated that standard encryption schemes cannot prevent sophisticated attacks such as covert attacks [39]. They proposed an improved encryption scheme to mask communication and cancel the influence of attack signals on the system. Building on this, Zhou *et al.* studied defense strategies for remote state estimation under deception attacks [40]. They proposed an encoding-decoding mechanism combining linear transformation and artificial noise to mitigate the effect of stealthy attackers without losing estimation performance in the absence of attacks. Pajic *et al.* proposed an attack-resilient state estimation method to cope with FDI attackers without assuming the exact system model is known [41]. They further considered the problem of attack-resilient state estimation in the presence of bounded-size noise, where FDI attackers can inject any signal via compromised sensors [42].

1.2.2 Resource Allocation Problems

Wireless spectrum is a scarce communication resource, and it can be very expensive [43]. The high price of wireless spectrum is partly due to the sharp growth and rapid development of CPSs, which required their incorporation into 5G/6G standards. Consequently, wireless radio resources in

CPSs are usually limited, making CPS resource allocation a critical area of research. This section reviews existing works on resource allocation in CPSs to optimize and enhance communications and networking capabilities, focusing on two main aspects: channel allocation problems and power allocation problems.

Channel Allocation

Given that frequency channels are typically limited resources, the channel allocation problem, equivalent to the sensor scheduling problem in the CPS context, is a well-studied problem in sensor networks. Gao *et al.* considered a single sensor single channel scheduling and remote estimation problem with additive communication channel noise, where the sensor's decision is whether to transmit its observation to the estimator or not [44]. They formulated the optimal solution to minimize the expected value of the sum of communication cost and estimation cost over a defined time horizon. Similarly, Vasconcelos *et al.* studied a channel allocation problem for remote state estimation in wireless networks, considering both unicast and broadcast networks [45]. They posed the empirical risk minimization problem as a difference-of-convex optimization problem and proposed a local optimal solution.

Further extending this field, Forootani *et al.* addressed a multi-sensor-multi-channel power allocation problem and proposed a Least Squares Temporal Difference Approximate Dynamic Programming approach to approximate the value function in the Bellman equation [46]. Leong *et al.* considered the channel allocation problem for energy harvesting sensors and proposed an optimal threshold policy structure [47]. Their subsequent work formulated an associated Markov decision process for the channel allocation problem and proposed a DQN-based solution to the dynamic channel allocation problem under a simple Gilbert–Elliott packet drop model [48]. Liu *et al.* extended this algorithm to a distributed networked control scenario over Markov fading channels [49]. This work further extended multisensor scheduling to temporally correlated communication channels and formulated a decentralized dynamic programming algorithm to address issues of unknown channel statistics and the curse of dimensionality in the MDP [50]. Additionally, Han *et al.* proposed both open-loop and closed-loop stochastic event-triggered sensor schedules for remote state estimation, addressing the challenge of nonlinear filtering problems caused by the destruction of the Gaussian property of the innovation process [51].

Transmission Power Allocation

As sensor energy is also a limited resource, the transmission power allocation problem is another common research area. In [52], a multi-sensor transmission power allocation problem for remote state estimation over a single interference channel is considered. In the shared packet-dropping channel, the packet drop rate is related to transmission power. By utilizing an approximated SINR quantity, a sub-optimal solution to the optimal power control problem is obtained by solving a standard convex optimization problem. Similarly, in [53], the power allocation problem is treated as a scheduling problem where sensors support discrete power modes. They first consider the Partially Observable MDP problem for known Gilbert-Elliott channels and derive conditions for the optimality of a stationary schedule. Next, a Bayesian inference approach is used to approximately learn the channel statistics when initially unknown. Li *et al.* [54] extended the multi-sensor transmission power scheduling problem under an Additive White Gaussian Noise channel using Quadrature Amplitude Modulation.

Mei *et al.* investigated multi-sensor power allocation over semi-Markov fading channels, finding the double-threshold structure of the optimal strategy and providing the necessary and sufficient conditions for remote estimation stability [55]. Peng *et al.* [56] explored optimal transmission power allocation for sensors equipped with energy-harvesting technology for remote state estimation in wireless cyber-physical systems. They formulated the problem as an optimal transmission power allocation to minimize remote estimation error covariances, modeled as an MDP subject to energy constraints. The optimal policy was found to be deterministic and stationary with a threshold structure. Pang *et al.* considered power allocation in a remote state estimation system with practical wireless models over the orthogonal multiple-access and non-orthogonal multiple-access schemes [57]. The system's stability was analyzed and novel action compression methods for DRL was proposed to solve the problem.

Some existing works even consider joint channel and power allocation problems. For instance, Gao *et al.* [58] addressed single sensor single channel scheduling and power allocation, formulating an optimal decision strategy under certain technical assumptions. Their subsequent work [59] extended the problem formulation to a more general setting where the encoder has a constraint on the total

encoding power over the time horizon, instead of the average power case. They proposed threshold-based communication scheduling policies along with affine encoding and decoding policies.

1.2.3 Deep Learning Applications

With the rapid advances in machine learning, especially deep learning, increasing opportunities for novel methods in enhancing system security and resource allocation have emerged. This section reviews existing literature on CPS where deep learning is applied, divided into two categories of methodology: deep reinforcement learning, and supervised deep learning.

Deep Reinforcement Learning (DRL)

Many research problems in the existing literature were formulated as complex decision-making problems, for which DRL has emerged as a powerful tool. Yang *et al.* [60] utilized Deep Q Network (DQN) for solving the channel allocation in a remote state estimation problem where a single communication channel was considered. Leong *et al.* extended the DQN's application to the channel allocation of a multi-channel scenario[48]. In [61], a single Markov fading channel was considered. To overcome the curse of dimensionality, a Deep Deterministic Policy Gradient (DDPG) agent was proposed to solve the stochastic channel allocation problem. Zhao *et al.* propose a co-design DRL framework for controller and scheduler optimization in remote state estimation [62]. Their proposed architecture utilized both model-free and model-based data, under novel schemes for enhancing the stability of joint training. Cao *et al.* proposed an actor-critic framework for channel and power allocation in a UAV-enabled IoT system [63]. Their method made the Unmanned Aerial Vehicle Base Station (UAV-BS) able to intelligently allocate both channels and transmit power for uplink transmissions of IoT nodes to maximize the minimum energy-efficiency among all the IoT nodes.

Supervised Deep Learning

A deep learning based multi-label classification approach is presented in [64] to detect the location of the FDI attack in a smart grid. This approach has employed the convolutional neural network (CNN) as a classifier to detect the structured FDI attack and the conventional measurement residual-based bad data detection algorithm to identify the unstructured FDI attack. In [32] present a attention-based temporal convolutional denoising autoencoder to address the FDI and DoS attack detection in distribution system state estimation. They use an attention structure in the model

facilitates the extraction of important features from the long sequence input data, and propose a modified Huber-loss function by considering the standard deviation of measurements. Kukadiya *et al.* [65] study the DoS attack mitigation on state estimation in power systems. Apart from analysis on the system performance of weighted least square and cubature Kalman filter -based hybrid state estimation, they propose a method to improve the performance of CKF under long-duration attacks by accurately predicting the synchrophasor measurements using CNN and long short-term memory (LSTM). Napoleon *et al.* [66] presented a scheme that learned the implicit power system measurement distribution and a surrogate of the unknown state estimator model to defend the power systems against FDI attack.

1.3 Research Problems

As discussed in section 1.2.2, existing research on wireless resource allocation has primarily focused on data-oriented communications, targeting transmission throughput, latency, and reliability [67]. However, in remote estimation systems, resource allocation should be task-oriented, aiming to minimize the long-term average remote estimation mean-square error (MSE) of dynamic physical states [67]. While advanced data-driven machine learning approaches, such as supervised learning and reinforcement learning, have been applied to resource allocation problems [68], the joint problem of channel allocation and transmission power allocation remains under-explored. This joint problem introduces a hybrid action space for a MDP, which existing DRL frameworks, designed for either discrete or continuous action spaces, cannot directly address.

Moreover, most studies on attack detection, discussed in section 1.2.1, focus on identifying the onset of an attack rather than detecting changes in attack behavior post-launch. If a malicious attacker alters its policy over time, it is crucial for the defender to detect these changes to adapt accordingly. A resilient system must implement not only attack detection but also attack policy change detection and mitigation. Traditional detection algorithms, such as the Cumulative Sum Procedure (CUSUM) [69], are ineffective due to the non-independent and identically distributed (IID) distribution and strong time correlations of attacking actions, which are often not directly observable. Additionally, the vulnerability of control channels to DoS attacks has not been thoroughly investigated, despite the significant impact on centralized control policies in CPSs. Therefore, a mechanism to counteract

the effects of imperfect control channels caused by DoS attacks is necessary.

Furthermore, the literature discussed in 1.2.2 rarely addresses remote state estimation in two-hop relay networks under DoS attacks. As sensors are typically physically located far away from the remote estimator, they face significant packet errors when transmitting directly. A two-hop relay network can form a cooperative system to mitigate this issue. However, introducing a relay complicates the communication dynamics, especially under DoS attacks, necessitating specialized decision-making processes.

This thesis addresses the following fundamental problems:

1.3.1 Problem 1

How can we manage limited wireless radio resources for remote state estimation of numerous distributed sensors to achieve optimal estimation performance under various multiuser access schemes?

1.3.2 Problem 2

How can we mitigate the impact of different types of DoS attackers who may jam communication channels?

1.3.3 Problem 3

How can we manage cooperation between segments in a CPS to enhance overall system performance?

1.4 List of Publications

The following is a list of publications in refereed journals and conference proceedings produced during my Ph.D. candidature. In some cases, the conference papers contain material overlapping with the journal publications.

Journal Papers

[J1]. K. Wang, W. Liu and T. J. Lim, "Deep Learning for Radio Resource Allocation Under DoS Attack," in IEEE Transactions on Machine Learning in Communications and Networking, vol. 2, pp. 703-716, 2024.

[J2]. K. Wang, W. Liu and T. J. Lim, "Wireless Resource Allocation with Collaborative Distributed and Centralized DRL under Control Channel Attacks," Submitted to IEEE Wireless Communication Letter, 2024.

Conference Papers

[C1]. K. Wang, W. Liu and T. J. Lim, "Deep Reinforcement Learning for Joint Sensor Scheduling and Power Allocation under DoS Attack," ICC 2022 - IEEE International Conference on Communications, Seoul, Korea, Republic of, 2022, pp. 1968-1973.

1.5 Thesis Outline

The remainder of this thesis is organized as follows:

In Chapter 2, we introduce the preliminary methodologies applied in this thesis.

Chapter 3 focuses on joint channel and power allocation for a remote estimation problem in CPS, considering the presence of a data channel DoS attacker. An advanced DRL algorithm is proposed to handle the hybrid action space, and a deep learning-based attacker policy change detector is developed to adapt to time-variant attacking policies. The content in this chapter is based on [J1] and [C1].

Chapter 4 addresses power allocation for remote state estimation, where a DoS attacker targets the feedback control channel. A collaborative distributed-centralized DRL algorithm is proposed for joint training and deployment of the defending agent. This chapter is based on [J2].

Chapter 5 studies power allocation in a two-hop relay network under data channel DoS attack. A knowledge-based DRL framework is proposed to solve this problem. The content in this chapter is based on research work intended for future publication.

Finally, Chapter 6 concludes the thesis and suggests directions for future research.

Chapter 2

Preliminaries of Related Methodologies

2.1 Chapter Introduction

In this chapter, we outline the preliminary methodology employed for this thesis. We begin by introducing machine learning concepts, followed by examples of deep learning methods utilizing neural networks. Additionally, we discuss deep reinforcement learning techniques, providing detailed descriptions of two popular algorithms.

2.2 Machine Learning

At its core, a learning algorithm is any algorithm that learns to accomplish a specific goal given some data to learn from. A common formalism of this definition is given in [70]: “A Computer program is said to learn from experience E with respect to some class of tasks T and performance measure P , if its performance at tasks T , as measured by P , improves with experience E .” Machine learning (ML) is a pivotal branch of artificial intelligence (AI) that focuses on the development of algorithms enabling computers to perform tasks without explicit instructions. By leveraging vast amounts of data, machine learning algorithms utilize mathematics, statistics, optimization, and knowledge discovery methods to identify intricate patterns, make informed decisions, and enhance their performance over time. This capability is instrumental in a myriad of applications, ranging from classical binary classification to predictive analytics and autonomous systems, thereby revolutionizing various industries. Most introductory texts in ML categorize the field into three primary

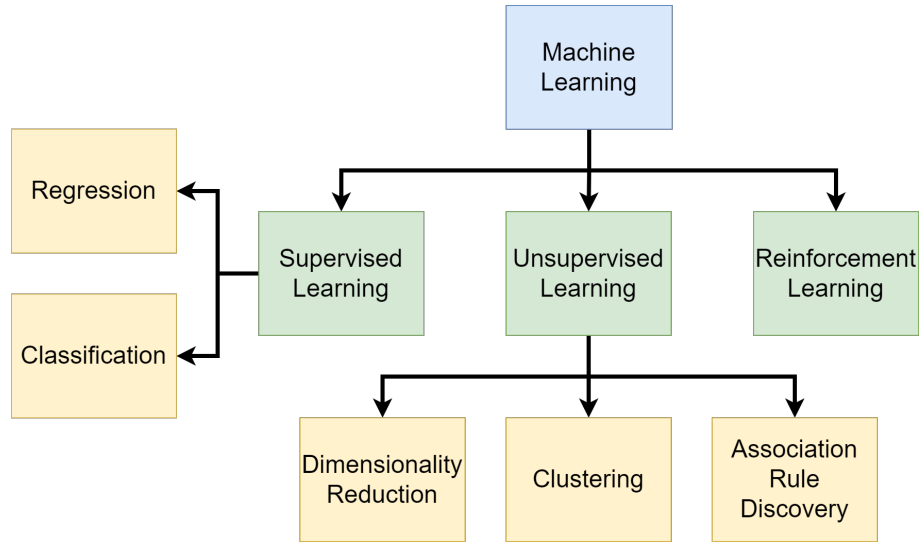


Figure 2.1: Taxonomy of machine learning.

subdivisions: supervised learning, unsupervised learning, and reinforcement learning [71]. The basic taxonomy of ML is illustrated in Figure 2.1. In this section, we will provide a brief introduction to each subdivision, highlighting their distinct characteristics and applications.

2.2.1 Supervised Learning

Supervised learning is one of the most prevalent and extensively researched paradigms in machine learning [72]. In supervised learning, the algorithm is trained on a labeled dataset, where each input is associated with a corresponding output. The primary objective is to learn a best mapping function from inputs x to outputs y , i.e., $f : x \rightarrow y$ that is applied at the end of the training process and can be generalized to new, previously unseen data. This paradigm encompasses two main types of tasks: classification and regression. Classification involves assigning inputs to discrete predefined categories, such as identifying spam emails or diagnosing diseases from medical images. Algorithms like logistic regression, support vector machines (SVMs), and naïve Bayes are commonly employed for classification tasks. Regression, on the other hand, aims to predict continuous values, such as forecasting stock prices or estimating house prices based on various features. Linear regression, polynomial regression, decision trees, and ensemble methods like random forests are some example algorithms used for regression tasks.

2.2.2 Unsupervised learning

Unsupervised learning, in contrast, deals with raw data inputs without available target variables. The goal of unsupervised learning is to uncover hidden patterns or structures within the data without any prior knowledge of the outcomes. This paradigm is particularly useful for exploratory data analysis and dimensionality reduction. Clustering, dimensionality reduction and association rule mining are three primary tasks in unsupervised learning. Clustering algorithms, such as k-means, hierarchical clustering, and Density-Based Spatial Clustering of Applications with Noise (DBSCAN), group similar data points together based on their inherent characteristics. These techniques are widely used in market segmentation, image compression, and anomaly detection. Dimensionality reduction techniques, like principal component analysis (PCA) and t-distributed stochastic neighbor embedding (t-SNE), extract information from the variables and reduce the variable complexity, facilitating visualization and interpretation of high-dimensional data. These methods are crucial for pre-processing data, enhancing computational efficiency, and mitigating the curse of dimensionality. Association rule mining aims at deriving the association relationships between features. This technique is widely applied in various domains, such as market basket analysis, where it helps in identifying products that frequently co-occur in transactions, and recommendation systems, where it aids in suggesting items based on user behavior patterns. Association rule mining provides insights into the underlying structure of the data, revealing correlations and dependencies that can inform decision-making and strategy development. By uncovering these hidden relationships, such techniques enable a deeper understanding of complex datasets, driving innovation and enhancing the effectiveness of data-driven applications.

2.2.3 Reinforcement Learning

Reinforcement learning (RL) is a distinct and dynamic paradigm within machine learning, inspired by behavioral psychology. In reinforcement learning, an agent interacts with an environment and learns to make decisions by receiving feedback in the form of rewards. The agent's objective is to maximize the cumulative discounted reward over time, which involves balancing exploration (trying new actions) and exploitation (choosing actions that are known to yield high rewards). This approach is particularly well-suited for problems involving sequential decision-making. A fundamental

framework for modeling reinforcement learning problems is the Markov Decision Process (MDP). An MDP is defined by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$, where:

1. $\mathcal{S}$ is the set of states, containing all possible observations,
2. $\mathcal{A}$ is the set of actions, containing all possible actions that can be taken,
3. $\mathcal{P}$ is the set of state transition probabilities, where $P(s'|s, a) \in \mathcal{P}$ represents the probability of transitioning to state s' from state s after taking action a ,
4. $\mathcal{R}$ is the set of reward functions, where $R(s, a) \in \mathcal{R}$ gives the immediate reward received after taking action a in state s .

An MDP provides a structured mathematical model for decision-making in situations where the state transitions are determined by only the current state and action, and not by previous states. Key concepts in reinforcement learning include (a) the environment, which provides the reward and new state feedback after receiving the agent's action; (b) the policy, which defines the agent's behavior in response to a reward, state and action; (c) the value function or the Q function, which estimates the expected reward for each state or for each state-action combination. RL algorithms have achieved remarkable success in complex tasks, such as robotics controlling, game playing, and autonomous driving.

In summary, machine learning encompasses a diverse array of techniques and methodologies that enable computers to learn from data and improve their performance over time. Supervised learning, unsupervised learning, and reinforcement learning each offer unique approaches to solving different types of problems, contributing to the broad applicability and transformative potential of machine learning in numerous domains. As the field continues to evolve, ongoing research and advancements promise to further expand the capabilities and impact of machine learning technologies.

2.3 Deep Learning

Deep learning, a subset of machine learning, has become a powerful method for modeling complex patterns in large datasets. It utilizes multi-layered artificial neural networks to automatically learn representations from data, uncovering intricate structures that traditional algorithms might overlook. These deep learning models, often called deep neural networks, consist of multiple layers

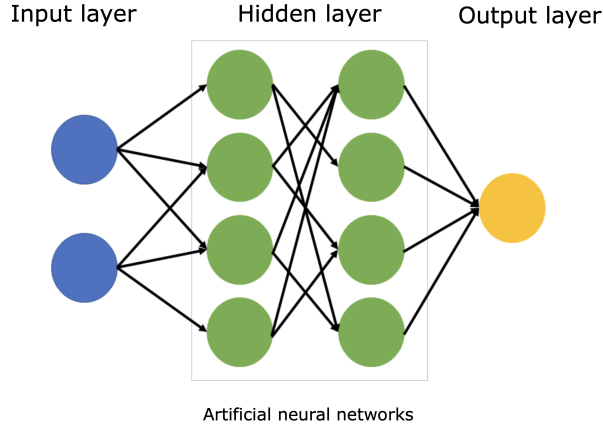


Figure 2.2: A simple feedforward neural network with two hidden layers.

of interconnected neurons. Each layer transforms the input data into increasingly abstract representations, enabling the algorithm to capture complex dependencies between input x and output y . According to the universal approximation theorem, any continuous function on a bounded set can be approximated by an appropriate neural network within a given accuracy [73]. The success of deep learning is largely due to advancements in computational power, the availability of large-scale datasets, and innovations in network architectures and training techniques. Consequently, deep learning has become a cornerstone of modern artificial intelligence, driving significant progress across various scientific and industrial domains.

2.3.1 Feedforward Neural Networks

Feedforward Neural Networks (FFNN) are a fundamental class of artificial neural networks where connections between nodes do not form cycles. An FFNN consists of an input layer, one or more hidden layers, and an output layer. Each layer is composed of neurons, the basic units of computation. Neurons in each layer are fully connected to those in the subsequent layer, forming a directed acyclic graph. A simple FFNN is illustrated in Figure 2.2. The formulation of the n th layer in an FFNN is given by:

$$\mathbf{y}_n = f_n(\mathbf{W}_n \mathbf{x}_n + \mathbf{b}_n), \quad (2.3.1)$$

where $\mathbf{y}_n \in \mathbb{R}^{n^y}$ is the output vector, $\mathbf{x}_n \in \mathbb{R}^{n^x}$ is the input feature vector, $\mathbf{W}_n \in \mathbb{R}^{n^y \times n^x}$ is the weight matrix, $\mathbf{b}_n \in \mathbb{R}^{n^y}$ is the bias vector, and $f_n : \mathbb{R}^{n^y} \rightarrow \mathbb{R}^{n^y}$ is the activation function.

The activation function typically introduces non-linearity to the neural network, enabling it to approximate complex functions as described by the universal approximation theorem [73].

Training an FFNN involves adjusting the weights and biases to minimize a loss function, which measures the approximation quality of the network. This is typically achieved using the backpropagation algorithm, a generalization of the delta rule to multi-layered networks, which employs the chain rule to iteratively compute gradients for each layer [74].

FFNNs have been successfully applied to a variety of tasks, due to their straightforward architecture and ability to approximate complex functions. However, FFNNs can struggle with tasks that require understanding temporal dependencies or sequential data, as they lack mechanisms to maintain context over time. Additionally, training deep FFNNs can be challenging due to issues such as vanishing and exploding gradients. Despite these challenges, advancements in training techniques and network architectures have significantly improved their performance, making FFNNs a foundational tool in the field of deep learning.

2.3.2 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a class of artificial neural networks designed to recognize patterns in sequences of data, such as time series, text, or speech. Unlike feedforward neural networks, RNNs have connections that form directed cycles, allowing them to maintain a memory of previous inputs. This capability makes RNNs particularly well-suited for tasks where context and sequential information are crucial.

An RNN processes an input sequence one element at a time, maintaining a hidden state vector that captures information about previous elements. The hidden state $\mathbf{h}_t \in \mathbb{R}^{n^h}$ at time step t is updated based on the current input $\mathbf{x}_t \in \mathbb{R}^{n^x}$ and the previous hidden state $\mathbf{h}_{t-1} \in \mathbb{R}^{n^h}$, as described by the following equation:

$$\mathbf{h}_t = f(\mathbf{W}_h \mathbf{h}_{t-1} + \mathbf{W}_x \mathbf{x}_t + \mathbf{b}), \quad (2.3.2)$$

where $\mathbf{W}_h \in \mathbb{R}^{n^h \times n^h}$ and $\mathbf{W}_x \in \mathbb{R}^{n^h \times n^x}$ are weight matrices, $\mathbf{b} \in \mathbb{R}^{n^h}$ is the bias vector, and $f : \mathbb{R}^{n^h} \rightarrow \mathbb{R}^{n^h}$ is a non-linear activation function. The output $\mathbf{y}_t \in \mathbb{R}^{n^y}$ at each time step can be computed from the hidden state:

$$\mathbf{y}_t = g(\mathbf{W}_y \mathbf{h}_t + \mathbf{c}), \quad (2.3.3)$$

where $\mathbf{W}_y \in \mathbb{R}^{n^y \times n^h}$ is the output weight matrix, $\mathbf{c} \in \mathbb{R}^{n^y}$ is the output bias vector, and $g : \mathbb{R}^{n^y} \rightarrow \mathbb{R}^{n^y}$ is an activation function.

Training RNNs involves adjusting the weights and biases to minimize a loss function, typically using the backpropagation through time (BPTT) algorithm [75]. BPTT is an extension of the backpropagation algorithm that accounts for the temporal dependencies in the data by unrolling the network through time and computing gradients for each time step.

RNNs have been successfully applied to a wide range of sequence modeling tasks, including language modeling, machine translation, and speech recognition. However, they can suffer from issues such as vanishing and exploding gradients, which can hinder their ability to learn long-term dependencies. To address these challenges, variants such as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) have been developed, offering improved performance in capturing long-range dependencies. As LSTM and GRUs are not covered in our methodology sections in the following chapters, the technical details of these algorithms are omitted.

2.4 Deep Reinforcement Learning

Deep Reinforcement Learning (DRL) combines the principles of reinforcement learning and deep learning to create powerful algorithms capable of solving complex decision-making problems. In DRL, an agent uses deep neural networks to approximate value functions or policies, enabling it to handle high-dimensional state and action spaces. This approach allows DRL to excel in tasks such as game playing, robotic control, and autonomous driving, where the observation features can be in image format and traditional RL techniques are not applicable. Despite its successes, DRL faces challenges such as sample inefficiency and stability issues during training. Ongoing research focuses on addressing these challenges through techniques like experience replay, target networks, and advanced exploration strategies, further enhancing the capabilities and applications of DRL.

2.4.1 Deep Q-Networks (DQNs)

Deep Q-Networks (DQNs) [76] are a class of deep reinforcement learning algorithms that combine Q-learning with deep neural networks to handle high-dimensional state spaces. Unlike traditional Q-learning, which uses a table to store Q-values, DQNs use a neural network to approximate the Q-function, enabling them to scale to problems with continuous or very large state spaces.

The DQN takes the state vector as input and outputs the Q-value of each state-action combination. Specially, DQN is applicable only to MDPs with discrete action spaces. The optimal Q-function should satisfy the Bellman equation as:

$$Q^*(s, a) = r + \delta \max_{a'} Q^*(s', a'), \quad (2.4.1)$$

where r is the reward, δ is the discount factor, and s' is the next state. The neural network is trained based on the Bellman equation, using techniques such as the $\{\epsilon_t\}$ -greedy action exploration, experience replay and target networks to stabilize training.

The $\{\epsilon_t\}$ -greedy action exploration is a process where the agent selects a random action with probability ϵ_t and the action that maximizes the Q-value with probability $1 - \epsilon_t$. The variable ϵ_t is a behavior distribution parameter that typically decays over time to balance exploration and exploitation

DQN utilize experience replay to enhance the training process. This involves storing the agent's interactions with the environment, known as state transitions, in a replay buffer. These state transitions are denoted as $\langle s, A, r, s' \rangle$, where s is the current state, A is the action taken, r is the reward received, and s' is the next state. Mini-batches of $\mathcal{N}$ state transitions are then randomly sampled from the replay buffer to minimize the loss function. This random sampling helps to break the correlation between consecutive state transitions, thereby stabilizing and improving the learning process. Since the policy used to explore these state transitions may differ from the policy under training, DQN is classified as an off-policy DRL technique.

To provide stable target Q-values, DQN employs target networks, which are periodically updated with the weights of the main network. Let the target network be denoted as Q' . The temporal difference (TD) loss function of DQN is represented as

$$TD^{DQN} = \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} \left(r_n + \delta \max_{a'} Q'(s'_n, a') - Q(s_n, a_n | \theta) \right)^2, \quad (2.4.2)$$

which is minimizing the deference between the left-hand-side and the right-hand-side of the Bellman equation within a mini-batch of size $\mathcal{N}$. The training process of DQN is summarized in the Algorithm 1.

Algorithm 1: DQN Training Algorithm

```

1 Initialize DQN  $Q(s, a|\theta)$  with random weights  $\theta$ ;
2 Initialize target network  $Q'$  with weights  $\theta' \leftarrow \theta$ ;
3 Initialize replay buffer  $\mathcal{R}$ ;
4 Initialize behavior distribution  $\{\epsilon_t\}$ ;
5 for  $episode = 1$  till convergence do
6   Observe  $s_0$ ;
7   for  $t = 1$  to  $T$  do
8     Obtain  $a_t = \begin{cases} \text{a random action} & \text{with probability } \epsilon_t, \\ \arg \max_a Q(s_t, a|\theta) & \text{otherwise;} \end{cases}$ 
9     Execute action  $a_t$  and observe reward  $r_t$  and next state  $s_{t+1}$ ;
10    Store transition  $\langle s_t, a_t, r_t, s_{t+1} \rangle$  in replay buffer  $\mathcal{R}$ ;
11    Sample a random mini-batch of  $\mathcal{N}$  transitions  $\langle s_n, a_n, r_n, s_{n+1} \rangle$  from  $\mathcal{R}$ ;
12    Set  $y_n = \begin{cases} r_n & \text{if episode terminates at step } n+1 \\ r_n + \delta \max_{a'} Q'(s_{n+1}, a'; \theta') & \text{otherwise;} \end{cases}$ 
13    Perform a gradient descent step on  $\frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} (y_n - Q(s_n, a_n|\theta))^2$  with respect to the
      network parameters  $\theta$ ;
14    Every  $C$  steps, soft update target network:  $\theta' \leftarrow \lambda\theta + (1 - \lambda)\theta'$ .
15  end
16 end

```

2.4.2 Deep Deterministic Policy Gradient (DDPG)

Deep Deterministic Policy Gradient (DDPG) [77] is a model-free, off-policy actor-critic DRL algorithm designed for continuous action spaces. It combines deterministic policy gradients with elements of DQN to address complex environments with large state-action space where traditional Q-learning is infeasible.

In DRL, policy optimization methods are categorized into value-based and policy-based approaches. Policy-based methods directly parameterize the policy π , mapping state s to action a , and optimize it to maximize the expected reward [78]. In contrast, value-based methods estimate the value function or Q-function, representing the expected return of taking action a in state s and following the policy thereafter [78]. The combination of value-based and policy-based optimization produces the actor-critic structure, to which DDPG belongs. DDPG consists of two main components: the actor network $\mu(s|\theta^\mu)$, which determines the action to take given a state, and the critic network $Q(s, a|\theta^Q)$, which evaluates the action by approximating the Q-function. The actor-critic approach leverages the strengths of both policy-based and value-based methods, enabling efficient

learning in complex environments.

Exploration in DDPG is facilitated by Ornstein-Uhlenbeck (OU) noise [79], which is added to the actions taken by the actor network. The OU noise process is designed to generate temporally correlated noise, which is beneficial for exploring continuous action spaces. The noise is generated according to the following stochastic differential equation:

$$dx_t = \theta(\mu - x_t)dt + \sigma dW_t, \quad (2.4.3)$$

where x_t is the noise value at time t , θ is the rate of mean reversion, μ is the long-term mean, σ is the volatility, and W_t is a Wiener process.

DDPG algorithm optimizes a policy by maximizing the estimated Q-value provided by the critic network. In this framework, the actor network is updated to produce actions that increase the Q-value predictions of the critic. Specifically, within each mini-batch of size $\mathcal{N}$, the actor network is adjusted by performing gradient ascent on a deterministic policy gradient as follows:

$$\nabla_{\theta^\mu} J \approx \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} \left[\nabla_a Q(s, a | \theta^Q) \Big|_{s=s_n, a=\mu(s_n)} \nabla_{\theta^\mu} \mu(s_n | \theta^\mu) \right], \quad (2.4.4)$$

where $J(\cdot) \triangleq \mathbb{E} \left[\sum_{k=1}^{\infty} \delta^{k-1} r_k \Big| a_k = \mu(s_k | \theta^\mu) \right] \approx Q(s_1, \mu(s_1 | \theta^\mu) | \theta^Q)$ denotes the performance objective [80]. Here, the critic network provides an estimate of this objective by approximating the Q-value for the current policy. By maximizing the Q-values over sampled states in the mini-batch, the algorithm effectively approximates the maximization of the performance objective. This iterative process enables the actor network to refine its policy to select actions that increasingly align with the critic's assessment of optimal behavior.

Similar to DQN, the critic network is updated by minimizing the TD loss defined below:

$$TD^{DDPG} = \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} \left(r_n + \delta \max_{a'} Q'(s'_n, \mu'(s'_n | \theta^\mu)) - Q(s_n, a_n | \theta^Q) \right)^2, \quad (2.4.5)$$

DDPG has been successfully applied to various continuous control tasks, demonstrating its ability to learn complex policies in high-dimensional action spaces. However, it can be sensitive to hyperparameters and requires careful tuning to achieve optimal performance. Despite these challenges, DDPG remains a foundational algorithm in the field of deep reinforcement learning, driving advancements in continuous control and policy optimization.

Algorithm 2: DDPG Training Algorithm

```

1 Initialize actor network  $\mu(s|\theta^\mu)$  and critic network  $Q(s, a|\theta^Q)$  with random weights  $\theta^\mu$  and  $\theta^Q$ ;
2 Initialize target networks  $\mu'$  and  $Q'$  with weights  $\theta^{\mu'} \leftarrow \theta^\mu$  and  $\theta^{Q'} \leftarrow \theta^Q$ ;
3 Initialize replay buffer  $\mathcal{R}$ ;
4 for  $episode = 1$  till convergence do
5   Initialize a Ornstein-Uhlenbeck process  $\{N_t\}$  for action exploration;
6   Observe initial state  $s_0$ ;
7   for  $t = 1$  to  $T$  do
8     Select action  $a_t = \mu(s_t|\theta^\mu) + N_t$ ;
9     Execute action  $a_t$  and observe reward  $r_t$  and next state  $s_{t+1}$ ;
10    Store transition  $\langle s_t, a_t, r_t, s_{t+1} \rangle$  in replay buffer  $\mathcal{R}$ ;
11    Sample a random mini-batch of  $\mathcal{N}$  transitions  $\langle s_n, a_n, r_n, s_{n+1} \rangle$  from  $\mathcal{R}$ ;
12    Set  $y_n = r_n + \delta Q'(s_{n+1}, \mu'(s_{n+1}|\theta^{\mu'})|\theta^{Q'})$ ;
13    Perform a gradient descent step on  $\frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} (y_n - Q(s_n, a_n|\theta^Q))^2$  with respect to the critic network parameters  $\theta^Q$ ;
14    Update the actor network using the sampled policy gradient:


$$\nabla_{\theta^\mu} J \approx \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} \nabla_a Q(s, a|\theta^Q)|_{s=s_n, a=\mu(s_n)} \nabla_{\theta^\mu} \mu(s_n|\theta^\mu);$$


    Every  $C$  steps, soft update the target networks:


$$\theta^{Q'} \leftarrow \lambda \theta^Q + (1 - \lambda) \theta^{Q'};$$


$$\theta^{\mu'} \leftarrow \lambda \theta^\mu + (1 - \lambda) \theta^{\mu'};$$

15   end
16 end

```

2.5 Chapter Conclusions

In this chapter, we have reviewed the methodology employed in this thesis. We have described the algorithmic frameworks and the theoretical details of the methods covered. The following chapter will study a radio resource allocation problem in remote state estimation, under DoS attack.

Chapter 3

Deep Learning for Radio Resource Allocation Under DoS Attack

This chapter considers a wireless resource allocation problem of remote state estimation in wireless networked CPS, where a DoS attacker jams the data communication channel. A complete defending system is proposed to mitigate the attacker.

3.1 Chapter Introduction

As explained in Chapter 1, CPS have been employed broadly in many sectors such as energy distribution and management, industrial robotics, healthcare, transportation, and entertainment. Rapid developments in CPS system design, network architecture, and device implementation have occurred in recent years. The resulting systems offer precise control based on real-time feedback and are poised for tremendous growth. Generally, a CPS observes and controls physical processes through sensors, communication networks, local and distributed computers, controllers, and actuators. Their realization faces substantial challenges though, with the diversity of applications requiring customized designs of multiple system components, ranging from communications and control to computing.

Wireless resource allocation in CPS is commonly modeled within a decision-making framework. Within this, problems characterized by a static state are often formulated through a Multi-Armed

Bandit (MAB) paradigm. MAB considers a single-state scenario and the optimal decision is fundamentally rooted in selecting the arm with the highest expected reward [81]. For instance, Alipour-Fanid *et al.* have formulated the cognitive base station scheduling problem with stationary frequency channels as a non-stochastic combinatorial MAB problem [82]. Work in [83] has investigated a sniffer-channel assignment problem with fixed packet capture probabilities. Guo *et al.* has explored a sensor scheduling problem considering energy efficiency, with the assumption of a static communication channel [84].

Conversely, decision-making problems that incorporate dynamic states are often formulated as a MDP. MDP considers Markovian state-transition dynamics and can be solved theoretically by the classic policy and value iteration methods [85–87]. However, due to the “curse of dimensionality” [88], such methods cannot be applied to problems with high-dimensional state and action spaces, and thus do not scale with the number of devices. DRL using DNNs as function approximators has emerged as a powerful alternative technique for solving large-scale MDP problems. In particular, the DQN is widely adopted for solving MDPs with discrete action spaces [89], while the deep deterministic policy gradient (DDPG) method is explicitly designed to operate over continuous action spaces [77].

There are many existing works applying DRL to optimize and enhance communications and networking capabilities in CPS environments [48, 49, 63, 90, 91]. Leong *et al.* [48] has proposed a DQN-based solution to the dynamic scheduling problem under a simple Gilbert–Elliott packet drop model. The follow-up work [49] extended the algorithm to a distributed networked control scenario over Markov fading channels. To handle continuous power allocation problems, a commonly adopted approach is to first discretize the action space and then apply the DQN algorithm [63, 90]. Nguyen *et al.* [91] has applied the DDPG algorithm for communication power allocation of a connected vehicle network. Relatively little work though has been done in scheduling and power allocation in a single policy. Difficulties arise from the scheduling action space being discrete and the power allocation action space being continuous. Joint optimization of both becomes challenging as a result.

In parallel, security and reliability in CPS have risen rapidly in prominence in recent years. With the increasing reliance of control systems on stable and reliable communication networks, cyber-attacks on communication networks and software systems can be expected to have wide-ranging consequences in the physical world, not just in cyber-space [12]. Much research effort has gone

into the detection, mitigation and analysis of cyber-attacks, including DoS [92, 93] and deception (integrity) attacks. As a deception attack attempts to manipulate the transmitted data, it typically requires a highly sophisticated attacker able to compromise the system software. On the other hand, DoS attacks only attempt to disrupt communications, e.g. by jamming a communication channel. Given its relative ease of implementation, a DoS attack is possibly more likely in practice than a deception attack [92]. A dynamic binary channel selection scheme has been proposed for a multi-sensor remote estimation system defending against DoS attacks [94]. However, to the best of our knowledge, transmission scheduling and power allocation co-design of a distributed system under DoS attack has not been investigated in the literature.

The detection of DoS attacks has attracted substantial research interest in the past two decades [95–98]. Classic change point detection algorithms, such as the Cumulative Sum Procedure (CUSUM) [69], have been adopted to solve the problem to some extent. However, they require the knowledge of the probability distributions of various signals before and after the DoS attack occurs, and also make the assumption of independent and identically distributed (IID) observations both before and after the attack [95–97]. To address these concerns, data-driven approaches have been proposed for DoS attack detection, which does not require explicit statistical knowledge [98].

It is notable that most works on attack detection stop at discovering the onset of an attack [95–98], rather than on detecting changes in attack behavior after the launch of an attack. If a malicious attacker alters its policy over time, then it is important for the defender to know when such changes occur in order to adapt to the new attack. A resilient system would have to implement not just attack detection, but attack policy change detection, as well as attack mitigation. However, attack policies are not known to the legitimate system and the attacking actions from a large action space can have strong time correlations and be non-IID distributed. Furthermore, the attacking actions in practice are not observable directly, e.g., a packet dropout may or may not be caused by a DoS attack. All these properties make attack policy changes difficult to detect. Due to these challenges, attack policy change detection has not been thoroughly studied in the literature.

In this chapter, we develop a complete method for countering DoS attacks launched at the physical layer of a remote estimation system. It does not merely detect the existence of an attack but can automatically adapt a defensive policy to time-varying attack policies. The proposed system

consists of a DRL-based joint channel and power allocation method, aided by a DNN-based attack policy change detector.

The main contributions are summarized below.

1. *In-depth problem formulation.* We have formulated a comprehensive problem that tackles not only sensor scheduling and power allocation under a stationary DoS attack but also adeptly accounts for any changes in the attack policy. The sensor scheduling and power allocation components are formulated as MDP problems with partial observations, where the sensor scheduling operates in a discrete action space and the power allocation is continuous. This multifaceted problem is further compounded by the attacker's aim of maximizing channel disruption. Detecting changes in attack policy is treated as a change detection problem, assuming a highly sophisticated DRL-based attacker that can adapt its attack policy in response to the defending actions.
2. *Unified DRL framework.* We have developed an advanced DRL framework that bridges the gap between the discrete action space of DQN and continuous action space of DDPG. This DRL architecture employing a single policy effectively reduces the overall complexity of the resource allocation challenges and leverages the synergies between the sub-problems. In addition to this novel policy integration, we have also introduced training stabilization techniques to stabilize the complicated neural network training procedure.
3. *Specialized DNN-based detector.* We have introduced a tailored DNN detector for discerning changes in the attack policy. This detector processes two time-correlated and Markovian sequences, whereas traditional change detection methods typically consider an IID scenario. It employs a parallel offline training framework to improve data utilization. By integrating this detector, our approach enhances the legitimate system's ability to adapt to unknown and dynamically evolving attack behaviors, thereby improving its resilience against attacks. Furthermore, our DRL framework does not require continual training if no policy change is detected, which reduces the computational burden associated with training.

The remainder of the chapter is organized as follows: Section 3.2 describes the remote estimation system under DoS attack on data channel. Section 3.3 formulates a sensor scheduling and channel

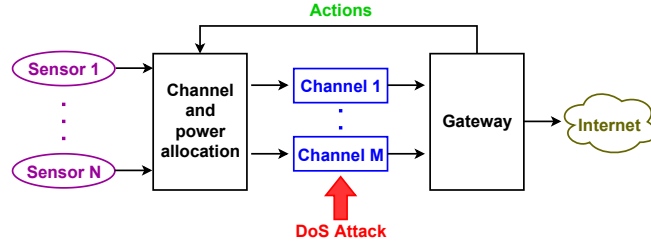


Figure 3.1: Remote state estimation with sensor scheduling jammed by DoS attack.

power allocation problem for the defender, as well as an attacking power allocation problem from the attacker's perspective. Innovative DRL-based algorithms have been proposed to solve these problems. Section 3.4 extends the defender's scale toward heuristic feedback channel DoS attack. Section 3.5 proposes a novel DNN-based detection algorithm to detect the change in the attack policy. Section 3.6 numerically evaluates the performance of our designed defender and detector, compared with different benchmarks. Some of the key notations of the chapter are summarized in Table 2.

3.2 System Model

We focus on remote state estimation under a physical-layer DoS attack, as illustrated in Fig. 3.1. N different processes are measured by N sensors, respectively, and the sensors transmit the local estimate of the state to a gateway via a shared wireless network consisting of M channels. The gateway is responsible for organizing the sensor scheduling on the channels, as well as the power allocated to the channels. The gateway is not assumed to have knowledge of the channel states or channel statistics, and thus scheduling and power allocation is performed in a model-free manner. It is assumed that the scheduling and power allocation actions determined by the gateway are sent to the sensors via error-free control channels.

3.2.1 Sensing Model

We consider N independent, linear and discrete-time processes with the following state transition equation [99]

$$\mathbf{x}_{i,k+1} = \mathbf{A}_i \mathbf{x}_{i,k} + \omega_{i,k}, \quad i = 1, \dots, N, \quad (3.2.1)$$

where $\mathbf{x}_{i,k} \in \mathbb{R}^{n_i^x}$ is the state of process i at time k , $\mathbf{A}_i \in \mathbb{R}^{n_i^x \times n_i^x}$ is the state transition matrix, and $\omega_{i,k}$ is stationary zero-mean Gaussian process noise with co-variance matrix $\mathbf{W}_i$, IID in both i and k . The state $\mathbf{x}_{i,k}$ cannot be measured directly and instead, we have the measurements $\mathbf{y}_{i,k} \in \mathbb{R}^{n_i^y}$ given by

$$\mathbf{y}_{i,k} = \mathbf{C}_i \mathbf{x}_{i,k} + \nu_{i,k}, \quad i = 1, \dots, N, \quad (3.2.2)$$

where $\mathbf{C}_i$ is the measurement matrix, and $\nu_{i,k}$ is IID zero-mean Gaussian measurement noise with co-variance matrix $\mathbf{V}_i$. For all i and j , $\omega_{i,k}$ and $\nu_{j,k}$ are assumed to be mutually independent. We assume that each sensor is capable of running a Kalman filter to compute local state estimates and estimation error covariance matrices, similar to [48, 93, 99]. At each time step k , we use the classic Kalman filter update equations to compute the filtered state estimate $\hat{\mathbf{x}}_{i,k}^s$,

$$\hat{\mathbf{x}}_{i,k|k-1}^s = \mathbf{A}_i \hat{\mathbf{x}}_{i,k-1|k-1}^s, \quad (3.2.3a)$$

$$\mathbf{P}_{i,k|k-1}^s = \mathbf{A}_i \mathbf{P}_{i,k-1|k-1}^s \mathbf{A}_i^\top + \mathbf{W}_i, \quad (3.2.3b)$$

$$\mathbf{K}_{i,k} = \mathbf{P}_{i,k|k-1}^s \mathbf{C}_i^\top (\mathbf{C}_i \mathbf{P}_{i,k|k-1}^s \mathbf{C}_i^\top + \mathbf{V}_i)^{-1}, \quad (3.2.3c)$$

$$\hat{\mathbf{x}}_{i,k|k}^s = \hat{\mathbf{x}}_{i,k|k-1}^s + \mathbf{K}_{i,k} (\mathbf{y}_{i,k} - \mathbf{C}_i \hat{\mathbf{x}}_{i,k|k-1}^s), \quad (3.2.3d)$$

$$\mathbf{P}_{i,k|k}^s = (\mathbf{I}_i - \mathbf{K}_{i,k} \mathbf{C}_i) \mathbf{P}_{i,k|k-1}^s, \quad (3.2.3e)$$

where $\mathbf{I}_i$ denotes the $n_i^y \times n_i^y$ identity matrix, $\hat{\mathbf{x}}_{i,k|k-1}^s$ represents the prior state estimate with error covariance $\mathbf{P}_{i,k|k-1}^s$, $\hat{\mathbf{x}}_{i,k|k}^s$ is the posterior state estimate with error covariance $\mathbf{P}_{i,k|k}^s$, and $\mathbf{K}_{i,k}$ is the Kalman gain. The first two equations above present the prediction steps while the subsequent three equations correspond to the updating (or filtering) steps. For notational convenience, $\hat{\mathbf{x}}_{i,k|k}^s$ will be abbreviated as $\hat{\mathbf{x}}_{i,k}^s$, and $\mathbf{P}_{i,k|k}^s$ as $\mathbf{P}_{i,k}^s \triangleq \mathbb{E}[(\mathbf{x}_{i,k} - \hat{\mathbf{x}}_{i,k}^s)(\mathbf{x}_{i,k} - \hat{\mathbf{x}}_{i,k}^s)^\top]$. We assume that every $(\mathbf{A}_i, \mathbf{C}_i)$ is observable and every $(\mathbf{A}_i, \sqrt{\mathbf{W}_i})$ is controllable [99]. Under these assumptions, the local Kalman filter at every sensor is stable. Specifically, $\mathbf{P}_{i,k}^s$ converges to a finite constant matrix $\bar{\mathbf{P}}_i$ as k approaches infinity for all i . It is further assumed that the local estimators have reached the

steady state with constant estimation error co-variances, i.e., $\mathbf{P}_{i,k}^s = \overline{\mathbf{P}}_i, \forall i = 1, \dots, N, \forall k$, where $\overline{\mathbf{P}}_i$ is determined by $\mathbf{A}_i, \mathbf{C}_i, \mathbf{W}_i$ and $\mathbf{V}_i$ [48, 93, 99].

3.2.2 Scheduling and Data Channel Model under DoS Attack

Sensors need to transmit their local state estimates $\hat{\mathbf{x}}_{i,k}^s$ via M shared orthogonal wireless fading channels to the gateway, which creates a scheduling problem. For instance, one could adopt the well-known orthogonal frequency-division multi-access (OFDMA). At each time slot, each channel would be allocated to one sensor, eliminating intra-cell interference and simplifying the scheduling process. This results in more predictable and stable communication channels, which is crucial for maintaining the integrity of remote state estimation in real-time applications. There is an attacker in the network who has launched a DoS attack, thus impairing the quality of the channels under attack and reducing their packet reception probabilities.¹ We assume the transmission power of a sensor allocated on channel m at time k , denoted as $E_{m,k}^s$, is adjustable and subject to $0 \leq E_{m,k}^s \leq E_{\max}^s$. Lastly, we assume that the sum of power used on all the channels by the attacker is constrained as $\sum_{m=1}^M E_{m,k}^a \leq E_{\max}^a$. This both reflects the practical reality that the attacker does not have access to infinite energy resources, and does not want to be easily detected.

Define decision variables $a_{m,k} \in \{0, 1, \dots, N\}$, $m = 1, \dots, M$, $k = 1, 2, \dots$. If no sensor is scheduled on channel m at time k , $a_{m,k} = 0$; if sensor i is scheduled to transmit on channel m at time k , $a_{m,k} = i$. If the communication channel m is attacked and $a_{m,k} \neq 0$, its signal-to-interference-plus-noise ratio (SINR) at time k is

$$\rho_{m,k} = \frac{E_{m,k}^s G_{m,i,k}^s}{E_{m,k}^a G_{m,k}^a + \sigma^2}, \quad (3.2.4)$$

where $G_{m,i,k}^s$ is the channel gain of channel m from sensor i to the gateway, and $G_{m,k}^a$ is the channel gain of channel m from the attacker to the gateway, at time k . We assume that $G_{m,i,k}^s$ is known to the gateway using standard channel estimation methods. σ^2 is the receiver additive white Gaussian noise power.

¹It is assumed that the attacker primarily targets the sensor-gateway data transmission channels, while the gateway-sensor feedback channels remain attack-free. This assumption is based on the fact that feedback channels typically have a much lower data rate, making them less attractive and less vulnerable to attacks. Additionally, the lower data rate of feedback channels reduces the potential impact of any successful attack. In the experimental results section, we have briefly considered the scenario of feedback channel attacks. A comprehensive analysis of feedback channel attacks and their mitigation strategies is provided in Chapter 4 of the thesis.

To capture the time-correlated nature of wireless fading channels, we consider time-homogeneous, ergodic Markov block-fading channels with q quantization states, in which each channel power gain remains constant within a time slot but may change from one slot to the next [99, 100]. Let $\mathbf{B}^s \triangleq \{b_1, \dots, b_q\}$, $\mathbf{B}^a \triangleq \{b_1^a, \dots, b_q^a\}$ denote the set of q -channel gain states from sensors and the attacker respectively. Then we have $G_{m,i,k}^s \in \mathbf{B}^s$, $G_{m,k}^a \in \mathbf{B}^a$. The channel transition probability of sensor i at channel m from state u to state v is

$$p_{u,v,m,i}^s \triangleq \mathbb{P}[G_{m,i,k+1}^s = b_v | G_{m,i,k}^s = b_u], \forall u, v \in \mathbf{Q}, \quad (3.2.5)$$

where $\mathbf{Q} \triangleq \{1, \dots, q\}$. Similarly, the channel transition probability of the attacker at channel m is

$$p_{u,v,m}^a \triangleq \mathbb{P}[G_{m,k+1}^a = b_v^a | G_{m,k}^a = b_u^a], \forall u, v \in \mathbf{Q}. \quad (3.2.6)$$

The corresponding channel transition probability matrices are $\mathbf{T}_{m,i}^s$ and $\mathbf{T}_m^a$ respectively.

We adopt finite-blocklength information theory [101] to approximate packet drop probabilities. The channel capacity $C_{m,k}$ and the channel dispersion $v_{m,k}$ of channel m at time k are defined as

$$C_{m,k} = \log_2(1 + \rho_{m,k}), \quad (3.2.7)$$

and

$$v_{m,k} = \rho_{m,k} \frac{2 + \rho_{m,k}}{(1 + \rho_{m,k})^2} (\log_2 e)^2, \quad (3.2.8)$$

where $e \approx 2.718$ is Euler's number. As in [102], each of the M channels is an Additive White Gaussian Noise (AWGN) channel with a finite-alphabet input. Let the transmission rate be R , and the Shannon capacity be $C_{m,k}$, as defined in equation (3.2.7). The block length of a codeword is n , and $\hat{x}_{i,k}^s$ is transmitted in a single codeword when sensor i is scheduled to transmit. Then the probability of successful packet transmission is approximated according to [102] as

$$\mu_{m,k} \approx 1 - Q\left(\sqrt{\frac{n}{v_{m,k}}} (C_{m,k} - R)\right), \quad (3.2.9)$$

where $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{t^2}{2}} dt$.

We introduce the binary variable $\gamma_{m,k}$ as an indicator of successful packet transmission on channel m at time k :

$$\gamma_{m,k} \triangleq \begin{cases} 1 & \text{if use of channel } m \text{ succeeds at time } k \\ 0 & \text{otherwise.} \end{cases}$$

3.2.3 Remote State Estimation at Gateway

As the gateway will not receive all sensors' local estimates in each time slot, it adopts an optimal [103] minimum mean-square error (MMSE) remote estimator for each process, given by

$$\hat{\mathbf{x}}_{i,k} = \begin{cases} \hat{\mathbf{x}}_{i,k}^s & \text{if } \exists m \text{ s.t. } a_{m,k} = i \text{ and } \gamma_{m,k} = 1 \\ \mathbf{A}_i \hat{\mathbf{x}}_{i,k-1} & \text{otherwise.} \end{cases} \quad (3.2.10)$$

The estimation error co-variances of each process can be obtained directly as

$$\begin{aligned} \mathbf{P}_{i,k} &\triangleq \mathbb{E}[(\mathbf{x}_{i,k} - \hat{\mathbf{x}}_{i,k})(\mathbf{x}_{i,k} - \hat{\mathbf{x}}_{i,k})^\top] \\ &= \begin{cases} \bar{\mathbf{P}}_i & \text{if } \exists m \text{ s.t. } a_{m,k} = i \text{ and } \gamma_{m,k} = 1 \\ h_i(\mathbf{P}_{i,k-1}) & \text{otherwise,} \end{cases} \end{aligned} \quad (3.2.11)$$

where $h_i(\mathbf{X}) \triangleq \mathbf{A}_i \mathbf{X} \mathbf{A}_i^\top + \mathbf{W}_i$.

The estimation mean-square error (MSE) at the gateway of process i at time k is defined as $\mathbb{E}[(\mathbf{x}_{i,k} - \hat{\mathbf{x}}_{i,k})^\top (\mathbf{x}_{i,k} - \hat{\mathbf{x}}_{i,k})] = \text{tr} \mathbf{P}_{i,k}$, where tr is the trace operator. $\text{tr} \mathbf{P}_{i,k}$ characterizes the instantaneous remote estimation quality of process i .

3.3 Defense and Attack Policy Design

3.3.1 Gateway Design

The gateway dynamically determines the channel scheduling and power allocation action based on the state vector:

$$\mathbf{O}_k = (\text{tr} \mathbf{P}_{1,k-1}, \dots, \text{tr} \mathbf{P}_{N,k-1}, G_{1,1,k}^s, \dots, G_{M,N,k}^s), \quad (3.3.1)$$

which includes the computed estimation quality of all processes in the previous time slot and the current channel states. Preceding the dispatch of the centralized allocation command to the sensors, the gateway is assumed to estimate the data channel through standard pilot-based methods [99, 104]. Let $\mathbf{A}_k^c = (a_{1,k}, \dots, a_{M,k})$ and $\mathbf{A}_k^p = (E_{1,k}^s, \dots, E_{M,k}^s)$ denote the scheduling (channel allocation) action and the defending power allocation action, respectively. We define the observation state

space, and the action spaces for scheduling and power allocation as $\mathcal{O}$, $\mathcal{A}^c$ and $\mathcal{A}^p$, respectively, where $\mathcal{A}^c$ is finite and discrete while $\mathcal{A}^p$ is continuous.

The gateway policy $\pi(\cdot)$ is a mapping from the state O_k to the action pair $A_k \triangleq (A_k^c, A_k^p)$ in each time slot. To find the policy that optimally balances remote estimation performance and energy consumption, we formulate the infinite horizon discounted optimization problem:

$$\begin{aligned} \max_{\pi(\cdot)} \quad & \mathbb{E} \left[\sum_{k=1}^{\infty} \delta^{k-1} r_k \right] \\ \text{s.t.} \quad & A_k = \pi(O_k), A_k \in \mathcal{A}^c \times \mathcal{A}^p, O_k \in \mathcal{O}, \end{aligned} \quad (3.3.2)$$

where $\delta \in (0, 1)$ is the discount factor, r_k is the per-step reward function defined as

$$r_k \triangleq - \left(\sum_{i=1}^N \text{tr} \mathbf{P}_{i,k} + \frac{1-\beta}{\beta} \sum_{m=1}^M E_{m,k}^s \right), \quad (3.3.3)$$

and $\beta \in (0, 1)$ is a design parameter reflecting the relative importance of energy consumption and estimation error. The reward function is a weighted linear combination of the sum of the mean squared remote estimation error and the sum of the sensors' energy consumption.

3.3.2 Attacker Design

The attacking power allocation action at time k is $A_k^a = (E_{1,k}^a, \dots, E_{M,k}^a)$, which satisfies the constraint $\sum_{m=1}^M E_{m,k}^a \leq E_{\max}^a$. The attacker's power budget is assumed to be unknown to the defender. The action space is denoted as $\mathcal{A}^a$.

We assume that the attacker has access to the observed states at the gateway but not the gateway actions. Realistically, such knowledge is not available and hence attacks will not be as sophisticated as this "genie-aided" one. The attacker state is defined in a similar way to the gateway's as

$$O_k^a = (\text{tr} \mathbf{P}_{1,k-1}, \dots, \text{tr} \mathbf{P}_{N,k-1}, G_{1,k}^a, \dots, G_{M,k}^a), \quad (3.3.4)$$

and the corresponding state space is defined as $\mathcal{O}^a$.

The attacker aims to find the policy that maximally degrades the remote estimation performance under its power constraint, by solving the problem

$$\begin{aligned} \max_{\pi^a(\cdot)} \quad & \mathbb{E} \left[\sum_{k=1}^{\infty} \delta^{k-1} r_k^a \right] \\ \text{s.t.} \quad & A_k^a = \pi^a(O_k^a), A_k^a \in \mathcal{A}^a, O_k^a \in \mathcal{O}^a, \end{aligned} \quad (3.3.5)$$

where r_k^a is the attacker's per-step reward function defined as

$$r_k^a = \sum_{i=1}^N \text{tr} \mathbf{P}_{i,k}. \quad (3.3.6)$$

It can be proved that both problems (3.3.2) and (3.3.5) have the Markov property, i.e., given the current state and action, the current reward and the optimal next state are independent of all previous states and actions [88], and thus they can be seen as MDPs. Classic MDPs are commonly solved by the value or policy iteration method, which requires explicit model information, i.e., the state transition probabilities. However, it is easy to show that in problems (3.3.2) and (3.3.5), the state transition probability $\mathbb{P}[O_{k+1}|O_k, A_k]$ and $\mathbb{P}[O_{k+1}^a|O_k^a, A_k^a]$ are dependent on the adversary's policy, which is completely unknown. This underscores the limitations of a model-based approach due to potential modeling inaccuracies and inherent uncertainties. Thus, we resort to model-free solutions for the defender's policy.

3.3.3 DRL-based Solution for Defense Policy Design

DRL is a widely applied model-free method for solving decision-making problems. The key difference between DRL and classic MDP is that DRL does not assume knowledge of the state transition probabilities, but records and exploits many sampled "state transitions", each consisting of the current state O , action A , reward r and next state O' as $\langle O, A, r, O' \rangle$. The state transitions are utilized to train neural networks for optimal approximation of the policy that maximizes the discounted long-term average reward.

In the following, we first propose four basic DRL algorithms for the gateway defender, building on the classic DQN and DDPG frameworks to solve (3.3.2); then we develop an advanced DRL algorithm; finally, we present an algorithm for the DoS attacker to solve (3.3.5).

Gateway Defender – Basic Algorithms

Algorithm A: Decoupled DQN and DDPG Traditional DRL algorithms DQN and DDPG are designed for solving discrete and continuous problems respectively, thus the easiest (but naïve)

method is to consider a *decoupled* approach – using DQN and DDPG separately for generating discrete (A_k^c) and continuous (A_k^p) actions.

DQN for discrete action. Q-value under the policy π represents the average discounted (sum) future reward given the current state-action pair (O, A^c) . Then the optimal Q-values should satisfy the Bellman equation:

$$Q^*(O, A^c) = \mathbb{E} \left[r + \delta \max_{A^{c'} \in \mathcal{A}^c} Q^*(O', A^{c'}) \right]. \quad (3.3.7)$$

DQN approximates these optimal Q-values and updates by minimizing a squared temporal-difference (TD) error.

DDPG for Continuous Action: DDPG utilizes two neural networks: the critic for Q-value estimation and the actor for policy approximation. The critic updates similarly to DQN, and the actor is updated by maximizing the Q-value from the critic. Note that the output layer of the actor is followed by the sigmoid activation function with a scalar $E_{\max}^s$ to meet the power constraint.

Then, the optimal policy $\pi^*(\cdot)$ is approximated by the decoupled discrete policy from DQN and continuous policy from DDPG. The former/latter is obtained without taking into account the action A^p/A^c . However, the actions are interrelated and need to be jointly designed. Next, we consider a joint design approach in Algorithm B1-2.

Algorithm B1: DQN with power allocation input Here, the input to DQN is the power allocation action A^p , allowing the scheduling action A^c to depend on it.

Algorithm B2: DDPG with sensor scheduling input In this variation, both the critic and actor of DDPG use the scheduling action A^c as input, making power allocation A^p dependent on it.

From our simulations, none of the basic algorithms have stable training of DQN nor DDPG, and thus no reasonable (converged) policy can be obtained. It is mainly due to the following issues: a) *Inefficient network structure with redundancy.* In the three neural networks both DQN and the critic generate Q-values. However, the training of DQN and critic are decoupled. b) *Unbounded state space.* From (3.2.11), the estimation quality indicator $\text{tr} \mathbf{P}_{i,k-1}$ contained in state O_k can grow arbitrarily large if sensor i 's packet has not been received by the gateway for a long time. c) *Large*

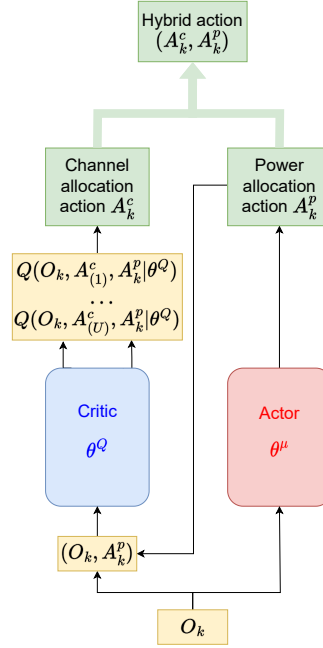


Figure 3.2: The HDDPG architecture.

action space exploration. Our DRL algorithm has a large hybrid discrete and continuous action space $\mathcal{A}^c \times \mathcal{A}^p$, which easily leads to unstable training.

Gateway Defender – Advanced Algorithm

This motivates us to propose an advanced algorithm, hybrid deep deterministic policy gradient (HDDPG) as illustrated in Fig. 3.2, to tackle the issues above. It has the following features.

a) *Economical network architecture.* Unlike the basic algorithms, the HDDPG has only two neural networks: the critic θ^Q and the actor θ^μ . The actor is identical to Algorithms A and B1, though the training method is different and will be discussed later. Define U as the cardinality of the discrete action space, i.e., $U \triangleq |\mathcal{A}^c|$, and $A_{(i)}^c$ as the i th element of $\mathcal{A}^c$. The critic takes both the state O and the continuous action A^p as its input and has U outputs of Q-values $Q(O, A_{(1)}^c, A^p | \theta^Q), \dots, Q(O, A_{(U)}^c, A^p | \theta^Q)$. The discrete action A^c that leads to the highest Q-value is considered the best action given O and A^p . Based on the sampled transitions $\langle O, A^c, A^p, r, O' \rangle$, the critic parameter θ^Q is updated by minimizing the TD error

$$\left[r + \delta \max_{A^{c'} \in \mathcal{A}^c} Q(O', A^{c'}, \mu(O' | \theta^\mu) | \theta^Q) - Q(O, A^c, A^p | \theta^Q) \right]^2 \quad (3.3.8)$$

and the actor parameter θ^μ is updated by maximizing the highest Q-value of the critic output

$$\max_{A^{c'} \in \mathcal{A}^c} Q(O, A^{c'}, \mu(O|\theta^\mu)|\theta^Q). \quad (3.3.9)$$

b) *Truncated state space.* We restrict the estimation quality state by a constant ϵ for HDDPG training, i.e. $\text{tr}\mathbf{P}_{i,k} \leq \epsilon, i = 1, \dots, N$, and the truncated state space is denoted as $\bar{\mathcal{O}}$. From the MDP theory [88], the difference between the optimal Q-values of the original MDP problem and that of the truncated state space problem is bounded by a constant, thereby limiting the performance loss due to state-space truncation. Note that the original state space will be used for HDDPG testing.

c) *Effective pre-training.* Instead of training the actor and critic from randomly generated parameters over the entire hybrid action space, we propose a novel heuristic policy-driven pre-training process for effective policy exploration. The general idea is to use heuristic policies (rather than the actor) for data sampling to pre-train the neural networks first as a warm-up. Then, these pre-trained parameters are used for the actor and critic initialization of the training process. In particular, the pre-training has two phases: in the first l_1 episodes, we use a maximum power allocation policy for A_k^p as $E_{m,k}^s = E_{\max}^s$ for all m ; in the following l_2 episodes, we use the uniform random power allocation policy. In Section 3.6, we will show that the pre-training initialized training process is stable and the training performance is improved.

The details of HDDPG algorithm are given in Algorithm 3.

3.3.4 DRL-Based Solution for Attack Policy Design

To solve (3.3.5) with the continuous action space $\mathcal{A}^a$, we use the classical DDPG algorithm consisting of the critic $Q^a(O^a, A^a|\tilde{\theta}^{Q^a})$ and the actor $\mu^a(O^a|\theta^{\mu^a})$. Here the output layer of the actor is passed through the softmax activation function with a scaling of $E_{\max}^a$ to meet the power constraint.

3.4 Feedback Channel DoS Attack

In addition to addressing DoS attacks on the data channel, this study also explores the impact of DoS attacks on the gateway-sensor feedback or control channel. Let the attacker's action on the feedback channel be defined as $A_k^b \in \{0, 1\}$, where $A_k^b = 1$ signifies that the attacker is obstructing the feedback channel, thereby preventing the sensors from receiving commands from the gateway at

Algorithm 3: HDDPG for joint sensor scheduling and power allocation at the gateway

```

1 Randomly initialize critic network  $Q(O, A^c, A^p|\theta^Q)$ , actor network  $\mu(O|\theta^\mu)$ , and initialize
  replay buffer  $\mathcal{R}$ ;
2 Initialize target network  $Q'$ , and  $\mu'$  with weights  $\theta^{Q'} \leftarrow \theta^Q$ ,  $\theta^{\mu'} \leftarrow \theta^\mu$ ;
3 Initialize a uniform distribution for  $A^c$  exploration and a Ornstein-Uhlenbeck process  $N^p$ 
  for  $A^p$  exploration;
4 for  $episode = 1$  to  $n$  till convergence do
5   Initialize observation state  $O_1$ ;
6   for  $k = 1$  to  $K$  do
7     Obtain the defending action
      
$$A_k^p = \begin{cases} (E_{\max}^s, \dots, E_{\max}^s) & \text{episode} < l_1 \text{ (pre-training)} \\ \text{random } A_k^p & l_1 \leq \text{episode} < l_2 \text{ (pre-training)} \\ \mu(O_k|\theta^\mu) + N_k^p & \text{others} \end{cases}$$

8     Select the scheduling action  $A_k^c$  by the critic network using epsilon-greedy
      exploration;
9     Execute the action pair  $(A_k^c, A_k^p)$ , observe the reward  $r_k$  and the new state  $O_{k+1}$ ;
10    Store transition  $\langle O_k, A_k^c, A_k^p, r_k, O_{k+1} \rangle$  in  $\mathcal{R}$ ;
11    Sample a random mini batch of  $\mathcal{N}$  transitions  $\langle O_i, A_i^c, A_i^p, r_i, O_{i+1} \rangle$  from  $\mathcal{R}$ ;
12    Set  $y_i = r_i + \gamma \max_{A^c} Q'(O_{i+1}, A^c, \mu'(O_{i+1}|\theta^{\mu'})|\theta^{Q'})$ ;
13    Update the critic by minimizing the loss:  $L = \frac{1}{\mathcal{N}} \sum_i (y_i - Q(O_i, A_i^c, A_i^p|\theta^Q))^2$ ;
14    Update the actor by applying the chain rule for calculating the gradient of (3.3.9):
15     $\nabla_{\theta^\mu} J \approx \frac{1}{\mathcal{N}} \sum_i \nabla_{A^p} \max_{A^c} Q(O_i, A^c, A^p|\theta^Q)|_{A^p=\mu(O_i|\theta^\mu)} \nabla_{\theta^\mu} \mu(O_i|\theta^\mu)$ ;
16    Update the target networks  $\theta^{Q'} \leftarrow \tau \theta^Q + (1 - \tau) \theta^{Q'}$ ,  $\theta^{\mu'} \leftarrow \tau \theta^\mu + (1 - \tau) \theta^{\mu'}$ .
17  end
18 end

```

time k . The blocking decision A_k^b has a frequency constraint

$$\lim_{K \rightarrow \infty} \frac{1}{K} \mathbb{E} \left[\sum_{k=1}^K A_k^b \right] \leq \lambda, \quad (3.4.1)$$

and the attacker's blocking outcome $\zeta_{i,k} \in \{1, 0\}$ on sensor i at time k is of a Bernoulli distribution IID in k ,

$$\mathbb{P}[\zeta_{i,k} = x | A_k^b = 1] = \begin{cases} p_i^b & \text{if } x = 1 \\ 1 - p_i^b & \text{otherwise.} \end{cases} \quad (3.4.2)$$

The feedback channel is assumed to be perfect when $A_k^b = 0$. Given that the defender's strategy is centralized and local sensors rely solely on their individual observations, we have developed a contingency strategy that encompasses both local sensor scheduling and data channel power allocation. The first backup mechanism uses a stand-by strategy which does not transmit when the gateway's command is not received and the second mechanism employs a stochastic approach for channel and

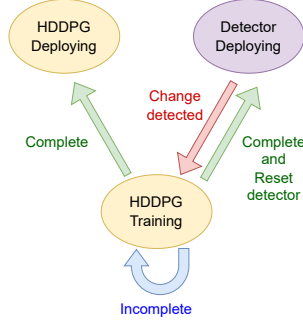


Figure 3.3: Integrated online attack policy change detection and defender policy re-training.

power allocation.

3.5 DNN-based Detector of Attack Policy Changes

An intelligent attacker may dynamically adjust its policy to either counteract the defense mechanisms or introduce confusion to improve its future rewards. In the presence of such time-varying attack policies $\{\pi^a\}$, an HDDPG defender directly implemented in an online training setting is susceptible to performance degradation due to policy deviations induced by action exploration. Moreover, the iterative learning process intrinsic to neural network training consumes substantial computational resources, predominantly due to gradient computation, and thus continual training should be avoided. Therefore, we propose distinct training and deployment phases, which collectively contribute to system optimization. In the training phase, the attacker’s policy is assumed to remain unchanged. This ensures that the defender’s training focuses on countering a specific, well-defined threat, allowing for a more controlled and effective learning process. In the deployment phase, a defense mechanism capable of quickly detecting these changes will be introduced in this section. We encapsulate these considerations within an integrated framework, as illustrated in Fig. 3.3. This framework facilitates real-time detection of shifts in attack policies and triggers immediate re-training of the defense strategy. Upon retraining, the central gateway clears the existing replay memory, which was formulated based on the previous attack strategy, and initiates a fresh round of HDDPG policy training. Following the completion of this training phase, the newly developed defense policy is deployed and the detection mechanism is re-initialized to identify subsequent shifts in attack policy.

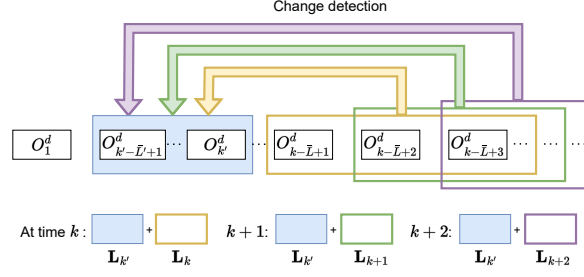


Figure 3.4: Online sequential change detection mechanism.

3.5.1 Conceptual Problem Formulation

To capture potential changes in attack policies between two time-indices $k' < k$, we introduce a binary variable $\kappa_{k',k} \in \{0,1\}$ which is 1 if and only if there is a change in $\{\pi^a\}$. We introduce two sequences $\mathbf{L}_{k'}$ and $\mathbf{L}_k$, as collections of gateway's observation state-action pair $O_k^d \triangleq (O_k, A_k^p)$:

$$\begin{aligned} \mathbf{L}_{k'} &= (O_{k'-\bar{L}'+1}^d, \dots, O_{k'}^d), \\ \mathbf{L}_k &= (O_{k-\bar{L}+1}^d, \dots, O_k^d), \end{aligned} \tag{3.5.1}$$

where $\bar{L}'$ refers to the number of observations of the previous attack policy and $\bar{L}$ signifies the minimum time required for detection. Given these sequences, the detector $d(\cdot, \cdot)$ generates a detection output $\hat{\kappa}_{k',k} = d(\mathbf{L}_{k'}, \mathbf{L}_k) \in \{0,1\}$ at each time k , aiming to identify changes in the attack policy by examining the two sequences $\mathbf{L}_{k'}$ and $\mathbf{L}_k$. It is assumed that the defender's policy has successfully adapted to π^a at time k' , using the DRL algorithm in Section 3.3. In this sliding detection mechanism illustrated in Fig. 3.4, the anchored sequence $\mathbf{L}_{k'}$ is recorded once following the defender's last training session, while $\mathbf{L}_k$ is continuously updated to enable real-time detection.

In Section 3.3, we highlighted that the gateway lacks direct knowledge of the attack policy and is unable to observe the attacker's actions. Consequently, the gateway can only infer shifts in the attacker's policy based on its own state observations. This constraint necessitates a model-free approach for detecting changes in attack strategies. It is pertinent to note that traditional change detection methods, such as CUSUM, typically require comprehensive knowledge of the statistical model and are generally limited to non-Markovian scenarios (e.g., for change detection of i.i.d sequences). In the subsequent section, we will elaborate on our proposed deep-learning-based model-free algorithm designed specifically for the detection of changes in attack policies.

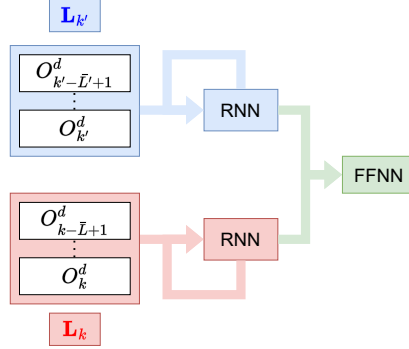


Figure 3.5: DNN-based attack policy change detector.

3.5.2 DNN-Based Detector Design

DNN design. In the realm of neural network architectures, the vanilla Feed-Forward Neural Network (FFNN) serves as a foundational model. Characterized by acyclic connections between neurons, the FFNN is principally utilized for function approximation with one-shot data inputs. On the other hand, Recurrent Neural Networks (RNNs) introduce cyclic connections between neurons, facilitating the capture of temporal dynamics in multi-dimensional sequential data [105].

Building upon the strengths of both FFNN and RNN architectures, we propose a hybrid network structure specifically designed for the detection of changes in attack policies, as depicted in Fig. 3.5. Our innovative design employs two separate RNN layers to extract pertinent features from two observation sequences: the anchored sequence, $\mathbf{L}_{k'}$, and the sliding sequence, $\mathbf{L}_k$. These extracted features are subsequently input into an FFNN layer to robustly identify modifications to attack policies.

Data generation and offline training. To train the DNN-based detector across diverse combinations of defender and attacker policies, we need a method for generating training data. Toward this end, we introduce a parallel time-series sampling approach. Initially, we establish a set of H distinct attack policies denoted as Π_{train}^a and employ the HDDPG algorithm to derive a corresponding defense policy for each attack policy. This yields a collection of H defense policies represented as Π_{train} . Given these sets, we produce observation data time-series, $\mathbf{O}_{g,h}^d$ under specific defense-attack policy pair (π_g, π_h^a) over $\mathcal{K}$ time slots

$$\mathbf{O}_{g,h}^d = (O_1^d, O_2^d, \dots, O_{\mathcal{K}}^d | \pi = \pi_g, \pi^a = \pi_h^a), \quad (3.5.2)$$

Algorithm 4: Data generation and training process of the detector

```

1 Collect the series of observation  $\mathbf{O}_{g,h}^d, \forall g, h \in \{1, \dots, H\}$ ;
2 Randomly initialize detector  $d(\cdot, \cdot | \theta^d)$  with parameter  $\theta^d$ , which has the structure illustrated
  in Fig. 3.5;
3 for  $episode = 1$  to  $n$  till convergence do
4   for  $i = 1$  to  $\mathcal{N}$  do
5     Randomly sample  $g, h$  and  $h'$  in  $\{1, \dots, H\}$ ;
6     Randomly sample  $k'$  and  $k$  subject to  $k' < k \leq \mathcal{K}$ ;
7     Collect  $\mathbf{L}_{k'}^f$  from  $\mathbf{O}_{g,h'}^d$  and  $\mathbf{L}_k^f$  from  $\mathbf{O}_{g,h}^d$  based on (3.5.3);
8     Calculate the label of  $(\mathbf{L}_{k'}^f, \mathbf{L}_k^f)$  as  $y_i = \kappa_{k',k}^f$  based on (3.5.4);
9     Calculate the detector output  $\hat{y}_i = d(\mathbf{L}_{k'}^f, \mathbf{L}_k^f | \theta^d)$ ;
10  end
11  Update  $\theta^d$  by minimizing the mean-square loss:  $\frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} (y_i - \hat{y}_i)^2$ ;
12 end

```

where $g, h \in \{1, 2, \dots, H\}$, $\pi_g \in \Pi_{train}$, and $\pi_h^a \in \Pi_{train}^a$. To sample the training data, we randomly select a sliding time series $\mathbf{O}_{g,h}^d$ and an anchored time series $\mathbf{O}_{g,h'}^d$ that share the same defense policy. We then extract two sequences of length $\bar{L}$ and $\bar{L}'$ from $\mathbf{O}_{g,h}^d$ and $\mathbf{O}_{g,h'}^d$, respectively, with random time indices $k' < k$ as

$$\begin{aligned} \mathbf{L}_{k'}^f &= (O_{k'-\bar{L}'+1}^d, \dots, O_{k'}^d | \pi = \pi_g, \pi^a = \pi_{h'}^a). \\ \mathbf{L}_k^f &= (O_{k-\bar{L}+1}^d, \dots, O_k^d | \pi = \pi_g, \pi^a = \pi_h^a), \end{aligned} \quad (3.5.3)$$

The corresponding label for a training data pair $(\mathbf{L}_{k'}^f, \mathbf{L}_k^f)$ is defined as

$$\kappa_{k',k}^f = \begin{cases} 1 & \text{if } h' \neq h \\ 0 & \text{otherwise.} \end{cases} \quad (3.5.4)$$

We subsequently employ this labeled data to construct a MSE loss function for the detector training, defined as $\left(d(\mathbf{L}_{k'}^f, \mathbf{L}_k^f) - \kappa_{k',k}^f\right)^2$.

The process offline training algorithm is outlined in Algorithm 4. Utilizing a classic supervised learning paradigm, we optimize the detector's performance under various policy combinations.

Performance testing and online detector implementation To evaluate the performance of the detector, we employ distinct offline and online testing approaches. For these approaches, we generate specific sets of policies not experienced in the training phase, for both the defender and attacker, denoted as Π_{test} and Π_{test}^a respectively. In the offline testing approach, we adhere to the parallel series sampling scheme previously described. A detection error is flagged whenever the detector's output differs from the true label.

Conversely, the online testing method seeks to mimic a realistic environment by generating time-series data that incorporates random attack policy changes within a specified time horizon. In this context, a premature change detection is considered a false alarm, while a delayed change detection is categorized as a detection delay. To reduce the likelihood of false alarms, we implement a median filter on the detector's output $\hat{\kappa}_{k',k}$. The median filter is a well-established nonlinear signal processing technique to eliminate spurious spike errors from the real-time detection output [106]. The filtered detection prediction, denoted as $\tilde{\kappa}_{k',k}$, is computed as the median of $\hat{\kappa}_{k',k_f}$ for k_f within the range $[k - \bar{L}_w + 1, k]$. Here $\bar{L}_w$ represents the window size, which is a crucial design choice that trades off detection accuracy and response time. A larger window size offers improved robustness at the cost of increased delay. Results from both testing approaches will be presented in Section 3.6.

3.6 Experiment Results

3.6.1 Dynamic Resource Allocation under Data Channel DoS Attack

Under fixed attack policy. We constructed simulated environments of varying levels of complexity to test our algorithms. Firstly, we test our algorithms under fixed attack policies:

1. Equal-power attack policy: $E_{m,k}^a = \frac{E_{\max}^a}{M}$ for all m .
2. Greedy channel gain attack policy:

$$E_{m,k}^a = \begin{cases} E_{\max}^a & m = \operatorname{argmax}_{m'} G_{m',k}^a \\ 0 & \text{otherwise} \end{cases}$$
3. Random attack policy: $E_{m,k}^a$ is randomly allocated.
4. Round robin attack policy: allocate $E_{\max}^a$ to one channel at a time in a fixed order.

Under each attack policy, we consider 10 examples with $N = 6$ sensors and $M = 3$ channels. Each process has a state dimension of 2 and a measurement dimension of 1. The process parameters $\mathbf{A}_i, \mathbf{C}_i, \mathbf{W}_i, \mathbf{V}_i, i = 1, \dots, N$ are randomly generated. The eigenvalues of $\mathbf{A}_i$ are drawn uniformly from the range $[0, 1.3]$. The entries of $\mathbf{C}_i$ are drawn uniformly from range $[0, 1]$, and $\mathbf{W}_i$ and $\mathbf{V}_i$ are generated by random orthogonal transformations of a diagonal matrix with random diagonal entries drawn uniformly from range $[0.2, 1.0]$. The sensors' and the

attacker's channel gains take values from $\mathbf{B}^s = \{0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5\}$ and $\mathbf{B}^a = \{0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7\}$, respectively. These values are chosen to reflect the moderate fading scenario for the sensor and the improved conditions for the attacker in the worst-case analysis. We note that the proposed framework is general and can be extended to accommodate other channel conditions as needed. The channel gain transition probability matrices $\mathbf{T}_{m,i}^s$ and $\mathbf{T}_m^a$ are randomly generated, representing different channel statistics. The receiver noise power is set as $\sigma^2 = 0.0001$. The blocklength of a codeword is set as $n = 128$ and the data rate as $R = 0.5$. The data rate of all sensors is uniform, as they generate their local estimates at the same rate. We set $E_{\max}^s = E_{\max}^a = 1$, and all the neural networks used have two hidden layers with 1024 nodes each, using the Relu activation function.

We use an ADAM optimizer with an initial learning rate of 10^{-4} for the DQN and the critic network, and 10^{-5} for the actor network. The learning rate is under manually adjusted decay while training. The experience replay memory has a size of 10^5 . 400 training episodes in total are used. The length of an episode is set as $K = 500$. We would reset all the processes at the end of each episode. The size of each mini-batch is set as 256. The target networks are updated once every 10 time steps with the soft update parameter $\tau = 0.01$. The discount factor δ is set to be 0.95. The normalization parameter β is set to be 0.05. After the DRL defender has been trained, we evaluate its performance on a testing episode of 100. We note that the basic algorithms (Algorithms A, B1, B2) and HDDPG all adopt the truncated state space for training with $\epsilon = 1000$ and the pre-training with episode length $l_1 = l_2 = 50$.

The simulated results of average reward over 100 testing episodes are presented in Table 3.1. In our first experimental setup under the equal attack policy, all the basic algorithms are unstable and fail to converge to a policy with reasonable performance. In addition, HDDPG outperforms all of them in the ten different environments. Therefore, we only present the result of the proposed HDDPG algorithm, compared with two benchmarks that we describe presently in the following simulation. According to [48], the greedy error co-variance benchmark for scheduling problems is the optimal policy compared with the random scheduling policy, random round-robin policy, and greedy holding times policy. However, the channel quality is assumed unknown to our gateway defender. Therefore, we build our benchmark 1 with a greedy error co-variance scheduling policy

and a full power allocation policy. Benchmark 2 uses a greedy error co-variance scheduling policy and a random power allocation policy.

Table 3.1: Simulation result of average reward (r_k) under fixed attack policy

	Equal attack policy			Greedy channel gain attack policy		
	HDDPG	BM1	BM2	HDDPG	BM1	BM2
1	-31.51	-70.51	-52.06	-20.57	-90.03	-69.28
2	-34.71	-77.20	-55.46	-25.80	-93.75	-85.69
3	-29.13	-73.51	-69.91	-24.15	-117.01	-113.43
4	-38.30	-69.98	-59.15	-56.88	-85.77	-79.23
5	-30.40	-72.19	-58.90	-71.23	-98.74	-93.14
6	-32.33	-70.55	-50.59	-35.87	-94.65	-101.86
7	-42.30	-77.73	-69.53	-50.40	-108.23	-103.11
8	-44.80	-79.33	-63.59	-39.89	-85.54	-66.18
9	-37.83	-76.98	-55.69	-36.51	-82.68	-60.94
10	-34.75	-69.39	-54.35	-37.25	-98.84	-100.96

	Random attack policy			Round robin attack policy		
	HDDPG	BM1	BM2	HDDPG	BM1	BM2
1	-30.90	-70.87	-46.03	-19.41	-72.03	-45.26
2	-33.29	-77.41	-51.05	-25.15	-77.62	-50.05
3	-30.26	-73.85	-52.85	-22.46	-76.05	-51.14
4	-43.57	-71.10	-51.67	-31.57	-73.05	-50.05
5	-32.82	-72.44	-48.58	-22.48	-73.70	-47.74
6	-33.17	-70.81	-45.36	-22.20	-71.37	-44.69
7	-44.24	-78.14	-55.89	-33.32	-79.21	-53.33
8	-49.29	-80.06	-58.18	-35.06	-81.07	-56.05
9	-38.62	-77.28	-52.41	-27.78	-77.87	-51.82
10	-36.84	-70.25	-47.77	-29.82	-71.45	-45.41

1. BM stands for benchmark. 2. Algorithms A, B1, B2 all fail to stably converge to a reasonable policy within the training episodes.

According to our simulation, under all four different attack policies and all ten different environments, our HDDPG defender outperforms the two benchmarks. The average testing performance of the HDDPG defender is 41.61% better than the best-performing benchmark. Within four different attack policies, HDDPG has an optimal performance under the round robin policy, with an average testing r_k of -26.92 . HDDPG under the greedy channel gain attack policy has the weakest performance, with an average testing r_k of -39.86 . This is expected as it is more profitable for the attacker to jam the channel with the highest gain using maximum power.

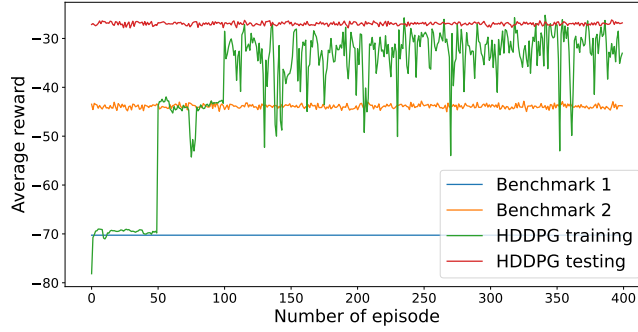


Figure 3.6: Training process and performance of HDDPG.

An example of the HDDPG training and the performance comparison to the benchmark algorithms are shown in Fig. 3.6, where the defenders are exposed to the equal attack policy. As mentioned earlier, the first 100 episodes are dedicated to pre-training, after which the policy smoothly approaches a steady state. We see significant performance gaps between the HDDPG and the benchmarks. In Fig. 3.7, we demonstrate the criticality of the two core features of the HDDPG agent: state-space truncation and effective pre-training. The evaluation of the training process is conducted ten times in each subplot, showcasing the resultant mean and variances. When both features are deactivated simultaneously (i.e., HDDPG' in Figure 3.7), the stability and efficacy of the training evidently deteriorate. Furthermore, HDDPG' struggles to achieve convergence towards a viable policy. In contrast, HDDPG exhibits a consistent and smooth convergence trajectory after 100 training episodes. The crucial role of these features in enhancing training is further validated by the notable difference in the average training reward of -53.95 for HDDPG and -1003.05 for HDDPG', and also a marked difference in variance.

Under intelligent attack policy. In this section, we evaluate the performance of our proposed RL-based genie-aided attacker. We adopt a turn-based training approach, where the gateway defender and the attacker train after the other converges to a counter policy. We start by training the HDDPG defender under an attacker with a randomly initialized policy DDPG. Prior to each training turn, we reset the memory of the agent to adapt to the opponent's new policy. In practice, we detect policy changes using the proposed DNN-based detector, which we evaluate in the next section. When a change is detected, re-training is triggered. The results of testing after two training phases are presented in Table 3.2. Our findings demonstrate that an intelligent attacker can significantly

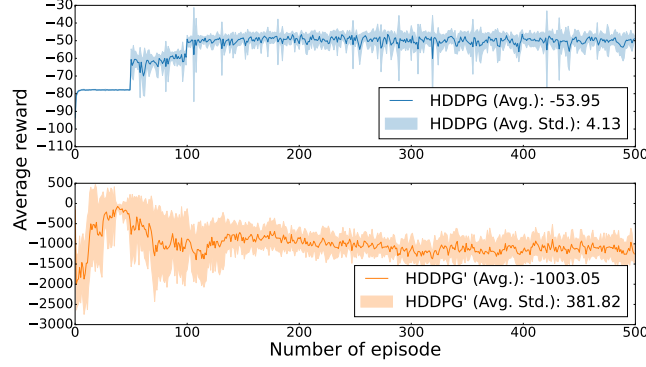


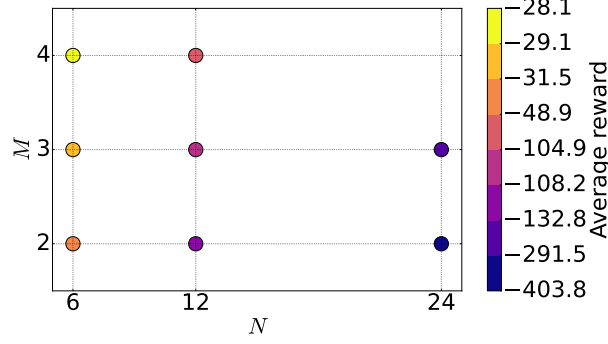
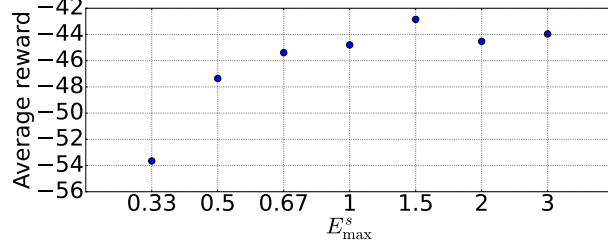
Figure 3.7: Training process and performance of HDDPG: effect of truncated state space and pre-training

compromise the network. The attacker learns to effectively jam the weakest communication links, thereby maximizing the remote estimation error. With each training iteration, the attacker inflicts greater damage on the system and increasingly adapts to the defender's policies. Conversely, an intelligent DRL defender can rapidly adapt to new attacks and restore the communication system. This is in stark contrast to a non-intelligent defender, which relies on a fixed policy and is less effective in responding to dynamic threats.

Table 3.2: Joint testing result after defender's and attacker's training

	1st defender training		1st attacker training		2nd defender training		2nd attacker training	
	Avg.	Avg.	Avg.	Avg.	Avg.	Avg.	Avg.	Avg.
	r_k	r_k^a	r_k	r_k^a	r_k	r_k^a	r_k	r_k^a
1	-30.80	21.71	-47.63	33.80	-25.46	24.07	-882.31	877.04
2	-35.30	29.47	-64.56	52.90	-24.30	23.99	-529.57	516.03
3	-31.01	22.66	-44.44	33.28	-20.67	20.50	-924.20	918.16
4	-38.16	22.02	-50.32	32.53	-20.50	20.21	-180.95	162.87
5	-30.02	23.38	-50.24	39.55	-23.18	21.27	-608.01	603.11
6	-33.16	23.00	-47.50	32.87	-23.67	19.58	-594.53	586.62
7	-43.15	29.37	-51.25	35.73	-28.21	27.16	-872.31	866.69
8	-45.65	29.13	-55.74	37.62	-34.41	33.05	-163.87	151.31
9	-39.32	32.05	-52.27	41.43	-28.03	26.63	-880.71	875.99
10	-37.40	24.44	-53.70	37.40	-29.99	29.34	-820.98	814.23

Scalability and convergence analysis. First, we evaluate the gateway's estimation performance across varying network scales (i.e., different N and M) and diverse power constraints (i.e., different $E_{\max}^s$ and $E_{\max}^a$). The results are presented in Figures 3.8 and 3.9, where the system parameters for

Figure 3.8: Testing result of scalability: N and M Figure 3.9: Testing result of scalability: $E_{\max}^s$ and $E_{\max}^a$

the same sensors and channels in different setups are identical and the attacker's power constraint is fixed at $E_{\max}^a = 1$. It can be observed that a higher number of channels compared to sensors results in a higher average reward (i.e., a better estimation performance). Similarly, a higher $E_{\max}^s$ relative to $E_{\max}^a$ yields a higher average reward up to a point. This upward trajectory flattens out when the channel transmission success rate is already very high, at which point increasing transmit power fails to improve the reward. Our study also indicates that the HDDPG algorithm is scalable to some extent in optimizing the defender's policy. However, the algorithm's computation complexity grows exponentially with M and N . Therefore, the proposed HDDPG algorithm for centralized resource allocation is suitable for moderate values of N and M . We will investigate decentralized algorithms for enhancing scalability in large-scale settings in our future work.

Second, we assess the convergence efficiency of the proposed algorithm, defining convergence as the point where the average reward r_k across the most recent five episodes is lower than that of

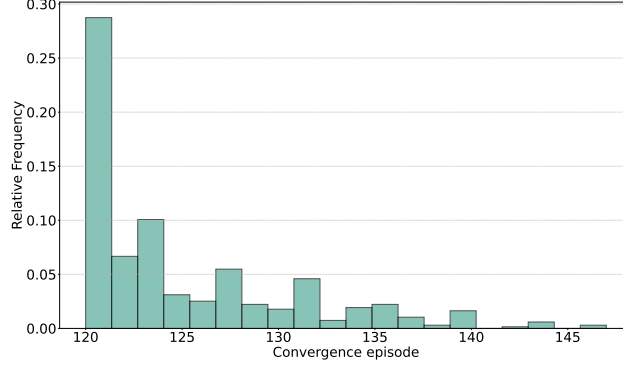


Figure 3.10: Convergence distribution

the preceding twenty episodes. To this end, we generate 500 distinct sets of system and channel parameters and illustrate the distribution of convergence episodes via a histogram in Fig. 3.10. Our findings indicate that the HDDPG algorithm achieves convergence within an average of 125 episodes, with the training process consistently completing within 150 episodes. This underscores HDDPG’s capability to address the challenge outlined in (3.3.2) effectively.

3.6.2 Resource Allocation under Feedback Channel DoS Attack

We consider briefly the scenario of feedback channel attacks here. It is assumed that the frequency of attacks targeting the feedback channel, i.e. λ as defined in Section 3.4, is constrained. Our assessment encompasses two distinct attack strategies employed on the feedback channel: a) a sequential attack approach focused on the initial time slots and b) a strategy based on randomly timed attacks. As shown in Table 3.3, sequential attacks are more likely to induce a pronounced decline in system performance. Unsurprisingly, a defense policy based on HDDPG trained in an environment devoid of attacks on the feedback channel exhibits suboptimal performance. However, by training the defender’s policy to recognize feedback attack patterns, labeled as HDDPG* in Table 3.3, we observe enhanced resilience. Remarkably, the proposed contingency strategy in conjunction with both versions of HDDPG consistently ensures system stability in scheduling and power allocation tasks without significant performance degradation. This investigation serves as a preliminary step in developing strategies to counter feedback channel attacks, and a more detailed analysis will be pursued in future work.

Table 3.3: Simulation result of average reward (r_k) under feedback channel attack

Attack Mode	Attack Frequency	Defense Mode	HDDPG (Testing)	HDDPG* (Training)	HDDPG* (Testing)
Sequential	0.25	Stand-by	-71.49	-53.38	-43.60
Sequential	0.50	Stand-by	-88.88	-56.93	-45.54
Sequential	0.25	Random	-58.13	-48.44	-38.60
Sequential	0.50	Random	-59.07	-48.10	-39.91
Random	0.25	Stand-by	-58.63	-47.69	-38.25
Random	0.50	Stand-by	-60.88	-47.23	-39.01
Random	0.25	Random	-58.27	-47.43	-37.89
Random	0.50	Random	-58.78	-47.54	-39.15

3.6.3 Data Channel DoS Attack Policy Change Detector

Offline training and testing. The proposed DNN-based detector is trained using Algorithm 4. We compare our detector with two benchmark methods: random forest, and logistic regression, where the stationary observation length $\bar{L}' = 200$ and detector operation lengths $\bar{L} = 10$. The FFNN has two 1024-node hidden layers, and each RNN has 128 nodes. The training set of attack policies Π_{train}^a consists of all the fixed policies in Section 3.6.1 and the RL-based policies after the 1st and 2nd round of defender training in the example of Table 3.2, while the testing set Π_{test}^a contains RL-based attack policies after 3 and 4 rounds of defender training, extending to the example of Table 3.2. The end of frame $\mathcal{K}$ is set as 10000.

Table 3.4: Average testing error rate of detectors

	Proposed	Proposed (Filtered)	Logistic Regression	Random Forest
1	0.0004	0.0000	0.0100	0.0380
2	0.0170	0.0000	0.0230	0.0290
3	0.0099	0.0002	0.0150	0.0050
4	0.0063	0.0021	0.0030	0.0050
5	0.0136	0.0025	0.2189	0.3090
6	0.0137	0.0013	0.0300	0.0360
7	0.0118	0.0045	0.0100	0.0310
8	0.0105	0.0017	0.0120	0.0100
9	0.0039	0.0008	0.0440	0.0360
10	0.0098	0.0000	0.0080	0.0000

In Table 3.4, the average detection error rates of the proposed DNN detector are compared with the benchmark detectors across 100 episodes, with each episode encompassing 500 testing records. Among the benchmarks, logistic regression has the lowest detection error rate. It also provides

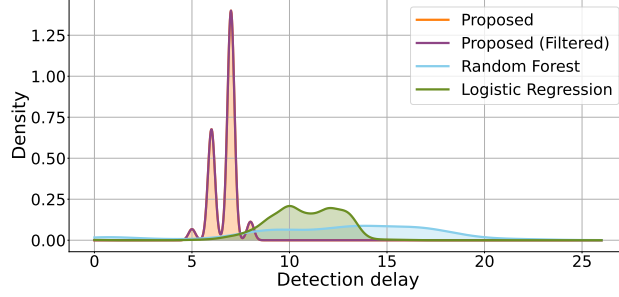


Figure 3.11: Estimated PDF of online policy change detection delay.

interpretability of the detection decision and necessitates fewer computational resources compared to the proposed detector. However, the proposed detector often reduces the error rate by a factor of 10 – 100 compared with logistic regression, thus convincingly demonstrating the effectiveness of our design.

Online testing. In Fig. 3.11, we present the estimated detection delay using kernel density estimation. The detection delay D is defined as

$$D \triangleq \min\{k - k^c \mid \hat{\kappa}_k = 1, k \geq k^c\} \quad (3.6.1)$$

The attack policy π^a transitions from π_3^a to π_4^a at time slot k^c , thereby $\kappa_{1,k} = \begin{cases} 0 & \text{if } k \leq k^c \\ 1 & \text{otherwise} \end{cases}$. Here π_3^a and π_4^a refer to the DRL-based attack policy after the 3rd and the 4th defender training rounds, extended from the examples in Table 3.2. Although both logistic regression and random forest yield low error rates of 0.01 and 0.038 in offline testing (shown in the first row of Table 3.4), they exhibit greater detection delay compared to our approach. The application of a median filter with a window size of $\bar{L}_w = 9$ to suppress spurious spike error refines the DNN detector output, and centers the distribution of delay around 7.

Fig. 3.12 showcases ten trials of online HDDPG training across episodes 1 to 1000, both with and without the policy change detector. The attack policy π^a changes at the beginning of episode 500 from π_3^a to π_4^a . Without our detector, the benchmark HDDPG continues its training after the policy changes based on its replay memory. Due to conflicting observations generated from two different attack policies, the benchmark HDDPG suffers significant performance degradation and variance at episodes 500 to 700. In contrast, when a policy change is detected by the detector-integrated

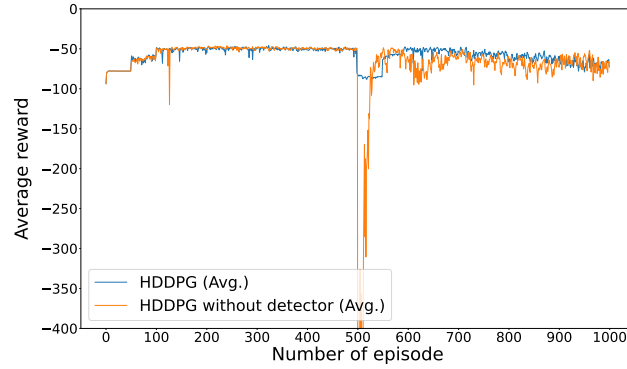


Figure 3.12: HDDPG training with and without the proposed policy change detector.

HDDPG agent at episode 500, it discards old replay memory and commences a new training session. Therefore, the HDDPG agent promptly and stably adapts to the new attack policy through the pre-training process. This strategic adaptation results in a markedly enhanced training robustness, exemplified by its ability to achieve rapid and stable convergence. DRL methods generally exhibit higher computational complexity compared to heuristic approaches, while HDDPG shares the same deployment complexity as DDPG or DQN. Our experiment utilizes an RTX 3070 GPU and an 11th Gen Intel i7-11700F CPU. Training the policy for 400 episodes requires 22 minutes. If we employ a standard edge server with 8 NVIDIA A40 GPUs [107], the training time is about 1.5 minutes.

3.7 Chapter Conclusions

In this paper, we investigated the problem of dynamic sensor scheduling and channel power allocation in the presence of a physical-layer DoS attack on a remote estimation system. The problem formulation involved a discrete action space for sensor scheduling and a continuous action space for power allocation, intending to minimize long-term estimation error and transmission energy consumption.

To address this problem, we proposed a novel model-free HDDPG algorithm designed to manage a hybrid action space effectively. Simulated results consistently demonstrate the superior performance of our proposed algorithm against all benchmark strategies, including those rooted in DRL. Notably, the proposed HDDPG defender also exhibits robust performance against a genie-aided DRL-based DoS attacker, which is even with a hypothetical advantage of complete knowledge. This emphasizes

the defender's potential utility in even the most adversarial conditions. Additionally, we incorporated a DNN-based attack policy change detector within the HDDPG resource allocation framework to achieve fast adaptation to attack policy changes and reductions in computational complexity. This algorithm can have significant impacts on practical CPS applications, such as mission-critical state monitoring.

In future work, we plan to extend the proposed intelligent resource allocation framework under DoS attacks to wireless networked control scenarios. Potential applications include autonomous driving, connected car platooning, and drone swarming. Additionally, we aim to formulate a game-theoretic approach to study the outcomes of the competition between attackers and defenders.

The following chapter will study a wireless resource allocation problem in remote state estimation, under control channel DoS attack.

Chapter 4

Wireless Resource Allocation with Collaborative Distributed and Centralized DRL under Control Channel Attacks

This chapter considers a wireless resource allocation problem of remote state estimation in wireless networked CPS where the control channel carrying resource allocation commands, is subjected to DoS attacks. A collaborative distributed and centralized DRL framework is proposed to mitigate this type of attacker.

4.1 Chapter Introduction

CPSs seamlessly integrate physical operations with computational and communication capabilities, enabling advanced control, decision-making, and sensing. As integral components of industrial monitoring systems, wireless sensors and remote estimators face significant network security challenges. The susceptibility of wireless channels to cyberattacks necessitates the development of robust and

secure estimation and control strategies to enhance the performance, resilience, and security of CPS [108].

Dynamic wireless resource allocation in CPS is frequently modeled as a MDP [48, 57, 94, 109]. As device networks expand, the complexity of MDPs increases correspondingly. DRL, which employs neural networks as function approximators, has become a pivotal technique for addressing large-scale MDP challenges within CPS. DRL can be implemented through centralized or distributed frameworks, each with its own advantages and drawbacks. Centralized DRL, with access to comprehensive information, enhances coordination and optimization but incurs higher computational costs and security risks. Conversely, distributed DRL promotes scalability by treating a problem as a collection of independent sub-problems. However, not making use of global information leads to suboptimal decisions, and coordinating multiple distributed agents remains complex. Recent studies have applied DRL to various aspects of remote estimation, adopting both centralized and distributed strategies. Centralized DRL has been used to optimize sensor scheduling [48] and power allocation for remote estimation [57]. Distributed DRL has addressed sensor scheduling under data transmission channel DoS attacks [94]. A centralized DRL framework has been proposed in [109], enhancing system resilience against DoS attacks on data channels.

In addition to data channel attacks, control channel attacks, where commands for the management and operation of the network are disrupted, also pose a significant challenge to maintaining the integrity and reliability of CPS. Various studies have explored methods to mitigate these attacks. Chan *et al.* tackled the problem of broadcast control channel jamming using random mapping with cryptographic functions and deterministic key assignment schemes [110]. Tague *et al.* expanded on this by introducing a framework that employs random cryptographic key assignments to obscure control channel locations [111]. Lo *et al.* presented a multiagent DRL approach to dynamically allocate control channels in response to primary user activity and jamming attempts [112]. While these studies focus on maintaining the functionality and security of the control channel under attack conditions, optimal wireless resource allocation problems in the presence of control channel attacks have not been investigated in the open literature.

In this paper, we tackle this open challenge, and our main contributions are as follows:

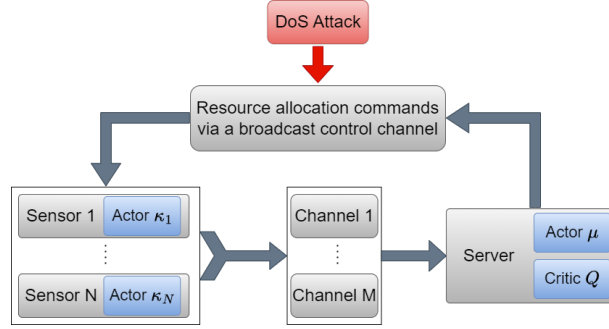


Figure 4.1: Multi-sensor remote state estimation under control channel DoS attack. Actor and critic modules (in blue) will be explained in Section 4.3.

- We formulate a new wireless resource allocation problem of a multi-sensor remote state estimation system under control channel DoS attacks, requiring both centralized and distributed resource allocation strategies to effectively mitigate the impact of these attacks.
- We propose a novel DRL framework, referred to as Collaborative Distributed and Centralized DRL (CDC-DRL), for the collaborative distributed-centralized power allocation problem, which has not been explored in the literature.
- We demonstrate that CDC-DRL outperforms state-of-the-art centralized and distributed DRL benchmarks, reducing the overall remote estimation error by 17% to 95% across various scenarios tested.

4.2 System Model

We consider a remote estimation system consisting of N sensors and a server (i.e., a remote estimator), as illustrated in Fig. 4.1. The system includes M data channels for sensor information transmission to the server and a broadcast control channel for transmitting resource allocation commands to the sensors.

4.2.1 Remote State Estimation with Smart Sensors

The physical process measured by sensor n is modeled as a discrete-time linear system [48] as defined in section 3.2.1:

$$\begin{aligned}\mathbf{x}_{i,k+1} &= \mathbf{A}_i \mathbf{x}_{i,k} + \omega_{i,k}, \\ \mathbf{y}_{i,k} &= \mathbf{C}_i \mathbf{x}_{i,k} + \nu_{i,k},\end{aligned}\tag{4.2.1}$$

where non-negative integer $k \in \mathbb{Z}^+$ is the time index. $\mathbf{x}_{i,k}$ is the system state. $\mathbf{A}_i$ is the state transition matrix. $\mathbf{y}_{i,k}$ is measurement of sensor i . $\mathbf{C}_i$ is the measurement matrix. $\omega_{i,k}$ and $\nu_{i,k}$ represent i.i.d. Gaussian measurement noise with covariance matrix $\mathbf{W}_i$ and $\mathbf{V}_i$, respectively. Instead of sending the raw measurement $y_{i,k}$ directly to the server, each sensor with computational capability (i.e., a “smart” sensor) pre-processes its measurement data by running a local Kalman filter and sends its local state estimates $\hat{\mathbf{x}}_{i,k}^s$. As we focus on the remote estimation scenario, we assume that each local Kalman filter is stable and working at its steady state with the estimation error covariance matrix $\overline{\mathbf{P}}_i \triangleq \mathbb{E}[(\mathbf{x}_{i,k} - \hat{\mathbf{x}}_{i,k}^s)(\mathbf{x}_{i,k} - \hat{\mathbf{x}}_{i,k}^s)^\top], \forall k \in \mathbb{Z}^+$ [57, 94, 109].

Due to limited wireless communication resources and potential packet errors, the server may not receive all local sensor estimates in each time slot and thus employs a Minimum Mean-Square Error (MMSE) estimator for each process [103]. We define the binary variable $\gamma_{i,k} \in \{0, 1\}$ such that successful transmission of sensor n 's packet is indicated by $\gamma_{i,k} = 1$. The remote state estimate and the error covariance matrix are given by

$$\begin{aligned}\hat{\mathbf{x}}_{i,k} &= \begin{cases} \hat{\mathbf{x}}_{i,k}^s & \text{if } \gamma_{i,k} = 1 \\ \mathbf{A}_i \hat{\mathbf{x}}_{i,k-1} & \text{otherwise,} \end{cases} \\ \mathbf{P}_{i,k} &= \begin{cases} \overline{\mathbf{P}}_i & \text{if } \gamma_{i,k} = 1 \\ h_i(\mathbf{P}_{i,k-1}) & \text{otherwise,} \end{cases}\end{aligned}\tag{4.2.2}$$

where $h_i(X) \triangleq \mathbf{A}_i X \mathbf{A}_i^\top + \mathbf{W}_i$ is the Lyapunov operator. Define the age of information (AoI) at sensor i at time k as

$$\tau_{i,k} \triangleq \min\{t \geq 0 : \gamma_{i,k-t} = 1\} = \begin{cases} 0 & \text{if } \gamma_{i,k} = 1 \\ \tau_{i,k-1} + 1 & \text{otherwise,} \end{cases}\tag{4.2.3}$$

allowing $P_{i,k}$ to be rewritten as

$$P_{i,k} = h_i^{\tau_{i,k}}(\overline{\mathbf{P}}_i).\tag{4.2.4}$$

Now we define the cost function below as the expected sum estimation errors of all processes to measure the remote estimation performance:

$$c_k \triangleq \sum_{i=1}^N \text{tr} \mathbf{P}_{i,k}, \quad (4.2.5)$$

where tr is the trace operator. The cost c_k is a function of the sensors' AoI, $\tau_{1,k}, \dots, \tau_{N,k}$.

4.2.2 NOMA-based Data Transmission Channel

We consider M orthogonal channels (e.g., subcarriers) for uplink sensor data transmissions. To model the time-varying and correlated nature of wireless fading channels, we use a time-homogeneous, ergodic Markov block-fading channel model with ψ quantization levels [100]. Each channel's power gain is constant within a packet transmission time slot but may vary between packets [99]. Let $\mathbf{G}_{i,k} \triangleq [G_{1,k}^i, \dots, G_{M,k}^i]^\top$ denote the channel power gain from sensor i to the server across M channels, where $G_{m,k}^i \in \mathbb{G} \triangleq \{h_1, \dots, h_\psi\}$ represents the discrete state of the channel power gain on channel m . The sensor's channel state vector $\mathbf{G}_{i,k} \in \{\mathbf{h}_1, \dots, \mathbf{h}_\psi\}$ is modeled as a multi-state Markov chain with $\bar{\psi}$ states. Assuming independent Markov channel states for different sensors due to their unique locations and varying radio propagation conditions, the channel state transition matrix $\mathbf{M}_i \in \mathbb{R}^{\bar{\psi} \times \bar{\psi}}$ for sensor i has the $\bar{i}, \bar{j}$ -th element representing the transition probability from state $\mathbf{h}_{\bar{i}}$ to $\mathbf{h}_{\bar{j}}$.

We adopt an advanced Non-Orthogonal Multi-Access (NOMA) scheme based on [57], due to its high spectral efficiency. Each channel can accommodate multiple concurrent sensors, and each sensor can send its packet across one or more of the M orthogonal channels given a total power budget E^s . The transmission power of sensor i in all M channels at time slot k is captured in the vector

$$\mathbf{E}_{i,k} \triangleq [E_{1,k}^i, \dots, E_{M,k}^i]^\top, \quad (4.2.6)$$

where $\sum_{m=1}^M E_{m,k}^i \leq E^s$. If $\sum_{m=1}^M E_{m,k}^i = 0$, then sensor i is not scheduled for transmission at time k . The aggregate transmission power of all sensors is represented by $\mathbf{E}_k \triangleq [\mathbf{E}_{1,k}, \dots, \mathbf{E}_{N,k}]$.

The server utilizes an N -round Interference Rejection Combining (IRC) and Successive Interference Cancellation (SIC) algorithm to recover packets from N sensors based on the superimposed signals across M channels. The received signal at the M channels is a combination of signals from

all sensors, expressed as:

$$\mathbf{u}_k = \sum_{i=1}^N (\sqrt{\mathbf{E}_{i,k}} \odot \mathbf{G}_{i,k}) u_{i,k} + \mathbf{z}, \quad (4.2.7)$$

where $u_{i,k}$ is the transmitted signal from sensor i at time k , and $\mathbf{z}$ represents receiver noise with covariance $\sigma^2 \mathbf{I}_M$, where $\mathbf{I}_M$ is an $M \times M$ identity matrix. The operator $\odot$ denotes the Hadamard product. For simplicity, let $\mathbf{E}_{i,k} \triangleq \sqrt{\mathbf{E}_{i,k}} \odot \mathbf{G}_{i,k}$. To detect sensor i 's signal, equation (4.2.7) is rewritten as:

$$\begin{aligned} \mathbf{u}_k &= \sum_{i=1}^N \mathbf{E}_{i,k} u_{i,k} + \mathbf{q}_{i,k}, \\ \mathbf{q}_{i,k} &= \sum_{n \neq i}^N \mathbf{E}_{n,k} u_{n,k} + \mathbf{z}, \end{aligned} \quad (4.2.8)$$

where $\mathbf{q}_{i,k}$ is the colored interference to $u_{i,k}$ due to correlated interference across channels. The covariance of $\mathbf{q}_{i,k}$ is:

$$\begin{aligned} \mathbf{R}_{i,k} &\triangleq \mathbb{E} [\mathbf{q}_{i,k} \mathbf{q}_{i,k}^H] \\ &= \sigma^2 \mathbf{I}_M + \sum_{n \neq i}^N \mathbf{E}_{n,k} \mathbf{E}_{n,k}^H, \end{aligned} \quad (4.2.9)$$

where $(\cdot)^H$ denotes the conjugate transpose. We employ the optimal linear combining scheme, IRC, to calculate the minimum SINR for each sensor's signal [113]. The IRC combiner is defined as:

$$\mathbf{c}_{i,k}^\top \triangleq \mathbf{E}_{i,k}^H (\mathbf{R}_{i,k})^{-1}. \quad (4.2.10)$$

Thus, the combined signal for detecting $u_{i,k}$ is $\mathbf{c}_{i,k}^\top \mathbf{u}_k$, and the SINR is:

$$\rho_{i,k} = \mathbf{E}_{i,k}^H (\mathbf{R}_{i,k})^{-1} \mathbf{E}_{i,k}. \quad (4.2.11)$$

The remote estimator decodes the sensor packet with the highest SINR and subtracts the successfully detected signal from $\mathbf{u}_k$. The SINR order of the received signals may change after SIC. In each iteration, the server computes the SINR for the undecoded sensors using IRC and executes SIC to detect the packet with the highest SINR. Let the decoding sequence of the sensors at time k be represented in the bijective mapping $D_k(\cdot) : i \in \{1, \dots, N\} \rightarrow j \in \{1, \dots, N\}$, where $D_k(i) = j$ indicates that sensor i 's packet is decoded in the j th round. Denote the remaining sensor signal index set as $\mathcal{N}_{i,k} \triangleq \{n : D_k(n) > D_k(i)\}$. The SINR for sensor i 's packet detection is:

$$\tilde{\rho}_{i,k} = \mathbf{E}_{i,k}^H (\tilde{\mathbf{R}}_{i,k})^{-1} \mathbf{E}_{i,k}, \quad (4.2.12)$$

where

$$\tilde{\mathbf{R}}_{i,k} = \sigma^2 \mathbf{I}_M + \sum_{n \in \mathcal{N}_{i,k}} \mathbf{E}_{n,k} \mathbf{E}_{n,k}^H. \quad (4.2.13)$$

Assuming each sensor packet has a block length of L and data rate R , the packet drop probabilities for each sensor can be accurately approximated based on its SINR and finite-blocklength information theory, as shown in equation (3.2.9) of Section 3.2.1. Let the single-round probability of successful packet transmission of sensor i at time k be denoted by $\mu_{i,k}$, which is determined by its SINR $\tilde{\rho}_{i,k}$. The overall decoding success probability of sensor i in a multi-round IRC-SIC scheme is:

$$\tilde{\mu}_{i,k} = \prod_{D_k(n)=1}^{D_k(i)} \mu_{n,k}. \quad (4.2.14)$$

The decoding process for all sensors terminates when either a packet detection error occurs or all packets are successfully detected.

4.2.3 DoS Attacker at Broadcast Control Channel

The server generates transmit power allocation commands for all sensors, packages this information into a single packet, and sends it over the broadcast control channel. A DoS attacker with Markovian behavior targets the broadcast control channel by transmitting a wireless jamming signal. Let the attacker's energy budget be E^a and its actual energy expended at time k be E_k^a . To minimize the likelihood of detection, the attacker intermittently alternates between attack ($E_k^a = E^a$) and silence ($E_k^a = 0$). The 2-by-2 attacker's action transition matrix is denoted by T^a .

Let $g_{i,k}^d$ and $g_{i,k}^a \in \mathbb{G}$ denote the control channel gain and the jamming channel (i.e., from attacker to sensor) gain for sensor i , respectively. The signal-to-interference-plus-noise ratio (SINR) on the broadcast control channel for sensor i is defined as

$$\rho_{i,k}^d = \frac{g_{i,k}^d E^d}{g_{i,k}^a E_k^a + \sigma^2}, \quad (4.2.15)$$

where σ^2 is the sensor's receiving noise and E^d is the server's transmit power on the control channel. The packet drop probability of the control channel can be approximated with the same method as the data channel.

4.2.4 Collaborative Distributed and Centralized Power Allocation

We propose a collaborative distributed and centralized power allocation scheme to mitigate the impact of broadcast control channel attacks. Let $\mathbf{a}_k^c \triangleq [\mathbf{a}_{1,k}^c, \dots, \mathbf{a}_{N,k}^c]$ denote the centralized power allocation action generated by the server. The distributed power allocation action generated by sensor n is denoted as $\mathbf{a}_{i,k}^d$, and $\mathbf{a}_k^d \triangleq [\mathbf{a}_{1,k}^d, \dots, \mathbf{a}_{N,k}^d]$ represents the aggregated distributed power allocation.

If sensor i receives the server's packet, the power allocation action $\mathbf{a}_{i,k}^c$ will be applied; otherwise, the local action $\mathbf{a}_{i,k}^d$ will be in effect. Let $\beta_{i,k} \in \{1, 0\}$ denote whether sensor i receives the server's packet over the control channel at time k , as affected by the DoS attacker. Thus, the collaborative distributed and centralized data channel power allocation is formulated as

$$\mathbf{a}_k = \mathbf{a}_k^c \beta_k + \mathbf{a}_k^d (\mathbf{I} - \beta_k), \quad (4.2.16)$$

where $\mathbf{I}$ is the N -by- N identity matrix, and the diagonal matrix $\beta_k \triangleq \text{diag}\{\beta_{1,k}, \dots, \beta_{N,k}\}$ serves as an **action collaboration indicator**, managing the blend of centralized and distributed actions determined by the control channel and attack status.

4.3 Collaborative Distributed-Centralized DRL

We aim to determine the optimal collaborative power allocation action $(\mathbf{a}_k^d, \mathbf{a}_k^c)$. For the distributed actions, sensor i generates $\mathbf{a}_{i,k}^d$ based on its local observation of its AoI and channel power gains:

$$\mathbf{O}_{i,k}^d = [\tau_{i,k-1}, \mathbf{g}_{i,k}^\top]. \quad (4.3.1)$$

The distributed policy at sensor i , $\kappa_i(\cdot)$ maps its local observation $\mathbf{O}_{i,k}^d$ to local action $\mathbf{a}_{i,k}^d$.

For the centralized actions, the server has the global observation

$$\mathbf{O}_k^c = [\mathbf{O}_{1,k}^d, \dots, \mathbf{O}_{N,k}^d]. \quad (4.3.2)$$

The centralized policy $\mu(\cdot)$ is a mapping from the global state $\mathbf{O}_k^c$ to the centralized action $\mathbf{a}_k^c$.

We formulate the collaborative power allocation problem under control channel DoS attacks as

a long-term discounted optimization problem as:

$$\begin{aligned}
& \min_{\kappa_1(\cdot), \dots, \kappa_N(\cdot); \mu(\cdot)} \mathbb{E} \left[\sum_{k=1}^{\infty} \delta^{k-1} c_k \right] \\
& \text{s.t. } \mathbf{a}_{i,k}^d = \kappa_i(\mathbf{O}_{i,k}^d), \mathbf{1}^\top \mathbf{a}_{i,k}^d \leq E^s, \forall i, \forall k \in \mathbb{Z}^+, \\
& \mathbf{a}_k^c = \mu(\mathbf{O}_k^c), \mathbf{1}^\top \mathbf{a}_{i,k}^c \leq E^s, \forall i, \forall k \in \mathbb{Z}^+,
\end{aligned} \tag{4.3.3}$$

where $\mathbf{1} = [1, \dots, 1]^\top \in \mathbb{R}^M$, c_k is the per-step cost defined in (4.2.5) and $\delta \in (0, 1)$ is the discount factor. This is a decision-making problem, as the current actions depend on the current states (i.e., the observations) and will influence the next states due to (4.2.3).

To solve the problem, we resort to DRL, which is recognized for its effectiveness in handling complex decision-making scenarios with high-dimensional spaces. However, existing DRL methods typically employ either a centralized or distributed approach, which is not suitable for our problem that requires a collaborative design of both distributed and centralized policies. To tackle this, we propose the collaborative distributed and centralized (CDC)-DRL framework.

4.3.1 Network Architecture of CDC-DRL

CDC-DRL has three types of deep neural networks (DNNs) below, as illustrated in Fig. 4.1:

1. A centralized actor μ with parameter θ^μ represents the central policy as $\mathbf{a}_k^c = \mu(\mathbf{O}_k^c | \theta^\mu)$;
2. N distributed actors $\kappa_1, \dots, \kappa_N$ with parameters $\theta^{\kappa_1}, \dots, \theta^{\kappa_N}$, respectively, represent the distributed policy as $\mathbf{a}_{i,k}^d = \kappa_n(\mathbf{O}_{i,k}^d | \theta^{\kappa_i}), \forall i$;
3. A centralized critic Q with parameter θ^Q approximates the Q function $Q(\mathbf{O}_k^c, \mathbf{a}_k | \theta^Q) \approx \mathbb{E} \left[\sum_{k'=k}^{\infty} \delta^{k'-k} c_{k'} \middle| \mathbf{O}_k^c, \mathbf{a}_k \right]$, which represents the expected future cost given the state and action under the CDC policies $\kappa_1(\cdot), \dots, \kappa_N(\cdot)$ and $\mu(\cdot)$. This estimation will guide the actors during training, optimizing their policies to minimize expected costs, and it will not be needed for deployment.

In particular, to meet the power allocation constraints in (4.3.3), each distributed actor's output layer is designed with two heads to determine the sensor's transmission power. The first head outputs a scalar, which is passed through a sigmoid activation function and then scaled by the energy budget E^s . The second head outputs a vector of size M using a softmax activation function. The product

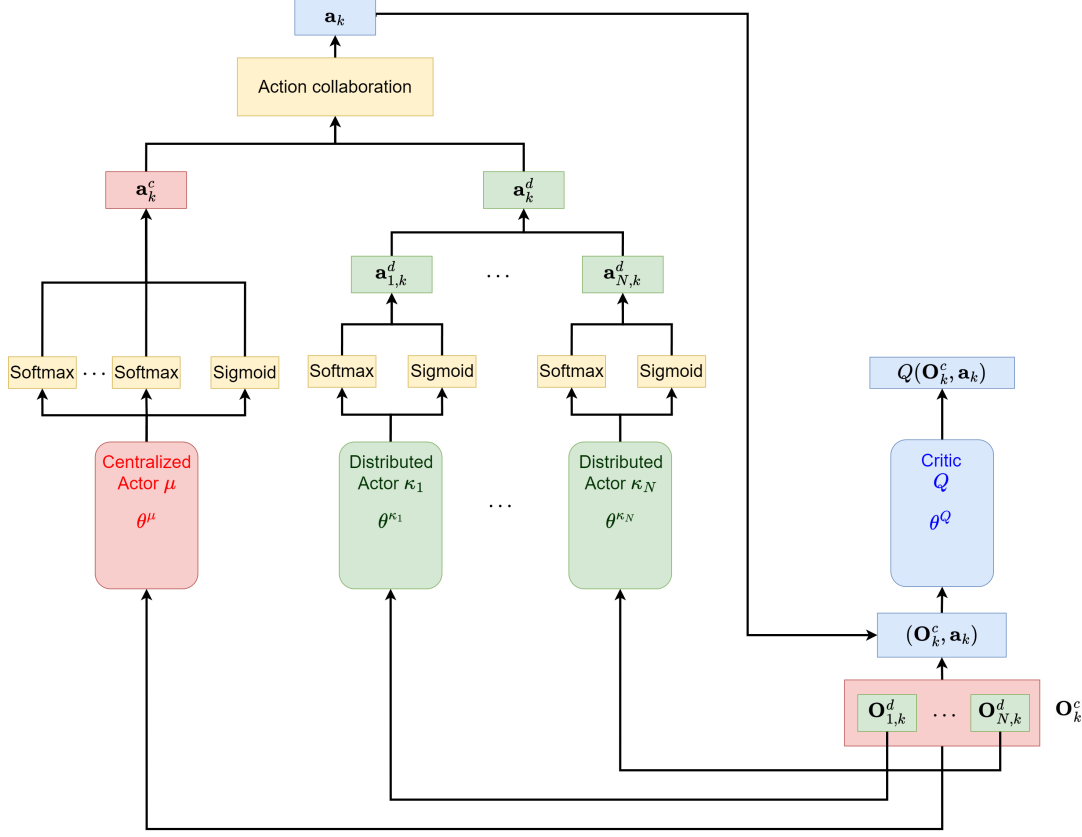


Figure 4.2: The network architecture of CDC-DRL

of these outputs provides the transmission power satisfying the power constraint. The centralized actor is configured in a similar manner. The network architecture is summarized in Figure 4.2.

4.3.2 Two-stage Training of CDC-DRL

The joint training of CDC policies is challenging, particularly due to the complex interactions between distributed and centralized decision-making under dynamic conditions. The presence of the DoS attacker with stochastic behavior further complicates these interactions, impacting training speed and stability. To address these challenges, our CDC-DRL framework adopts a two-stage training approach. Initially, separate pre-training for centralized and distributed policies optimizes the initial DNN parameters, providing a stable starting point and isolating training from adversarial disruptions. This preparation enables effective joint training, where these policies are fine-tuned

to synergistically generate power allocation commands over the dynamic DoS attacks. This strategy not only quickens convergence but also improves the resilience and efficacy of the CDC-DRL algorithm.

Separate Centralized and Distributed Pre-training

In this stage, we first train the centralized critic and actor using a classic centralized DRL approach, where $\mathbf{a}_k = \mathbf{a}_k^c$. The training follows an off-policy scheme based on transitions consisting of the global state, collaborative action, cost, and next global state, denoted as $\langle \mathbf{O}^c, \mathbf{a}, c, \tilde{\mathbf{O}}^c \rangle$. The optimal parameters θ^Q and θ^μ should satisfy the Bellman equation [77] as

$$Q(\mathbf{O}_k^c, \mu(\mathbf{O}_k^c | \theta^\mu) | \theta^Q) = \mathbb{E} [c_k + \delta Q(\mathbf{O}_{k+1}^c, \mu(\mathbf{O}_{k+1}^c | \theta^\mu) | \theta^Q)]. \quad (4.3.4)$$

θ^Q and θ^μ are updated iteratively. In each iteration, θ^Q is first updated via gradient descent to minimize the expected temporal difference (TD) error, using a mini-batch of $\mathcal{N}$ transitions defined as

$$TD^{\text{Sep}} = \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} \left[c_n + \delta Q(\tilde{\mathbf{O}}_n^c, \mu(\tilde{\mathbf{O}}_n^c | \theta^\mu) | \theta^Q) - Q(\mathbf{O}_n^c, \mathbf{a}_n | \theta^Q) \right]^2. \quad (4.3.5)$$

The rationale behind TD error minimization is to narrow the gap between the left-hand side and right-hand side of the Bellman equation (4.3.4). Next, θ^μ is updated by gradient descent to minimize the loss J^c , which is the expected cost estimated by the centralized critic:

$$J^c = \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} Q(\mathbf{O}_n^c, \mu(\mathbf{O}_n^c | \theta^\mu) | \theta^Q). \quad (4.3.6)$$

After training θ^Q and θ^μ , we proceed to train the distributed actors. By the guidance of centralized critic, the distributed actor DNN parameters $\theta^{\kappa_1}, \dots, \theta^{\kappa_N}$ are updated by minimizing the expected cost as

$$J^d = \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} Q(\mathbf{O}_n^c, \mathbf{a}_n^d | \theta^Q, \mathbf{a}_{i,n}^d = \kappa_i(\mathbf{O}_{i,n}^d | \theta^{\kappa_i}), \forall i). \quad (4.3.7)$$

Joint Centralized and Distributed Training

In contrast to the separate centralized and distributed pre-training, joint training under DoS attacks employs a collaborative power allocation action defined in (4.2.16). Our goal is to train the centralized critic to estimate future costs from the global state, $\mathbf{O}_k^c$, and the executed (collaborative) action, $\mathbf{a}_k$, and use this to guide actor training.

Unlike pre-training, where the subsequent action is determined solely by the next state and the policy of either the centralized or distributed actor, in joint training, the next action becomes stochastic. This stochasticity arises from the collaborative actions of the centralized and distributed actors under the influence of the DoS attack. To enhance the training process, we introduce a new data transition format, consisting of the global state, both centralized and distributed actions, the action collaboration indicator, cost, next global state, and next collaborative action, denoted as $\langle \mathbf{O}^c, \mathbf{a}^c, \mathbf{a}^d, \beta, c, \tilde{\mathbf{O}}^c, \tilde{\mathbf{a}} \rangle$.

Following this, we define the critic's loss function for the joint training stage similar to that in (4.3.5):

$$TD^{\text{Joint}} = \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} \left[c + \delta Q(\tilde{\mathbf{O}}_n^c, \tilde{\mathbf{a}}_n | \theta^Q) - Q(\mathbf{O}_n^c, \mathbf{a}_n | \theta^Q) \right]^2. \quad (4.3.8)$$

Subsequently, the loss function for jointly optimizing both the centralized and distributed actors is defined as:

$$J^{\text{Joint}} = \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} Q(\mathbf{O}_n^c, \mathbf{a}_n | \theta^\mu, \theta^{\kappa_1}, \dots, \theta^{\kappa_N}, \theta^Q). \quad (4.3.9)$$

In (4.3.8) and (4.3.9), $\mathbf{a}_n = \mathbf{a}_n^c \beta_n + \mathbf{a}_n^d (\mathbf{I} - \beta_n)$ as introduced in (4.2.16). The CDC-DRL training framework is detailed in Algorithm 5.

Algorithm 5: CDC-DRL

```

1 Initialize network parameters  $\theta^Q, \theta^\mu, \theta^{\kappa_1}, \dots, \theta^{\kappa_N}$ , replay buffer  $\mathcal{R}$ .
2 1) Separate Pre-training:
3 for  $episode = 1$  till converge do
4   Initialize Ornstein-Uhlenbeck(OU) process  $\{N_k^c\}$  [77].
5   for  $k = 1$  to  $K$  do
6     Obtain the centralized power allocation action  $\mathbf{a}_k^c = \mu(\mathbf{O}_k^c | \theta^\mu) + N_k^c$ ;
7     Execute  $\mathbf{a}_k^c$ , observe cost  $c_k$  and observe new global state  $\mathbf{O}_{k+1}^c$ ;
8     Store transition  $\langle \mathbf{O}_k^c, \mathbf{a}_k^c, r_k, \mathbf{O}_{k+1}^c \rangle$  in  $\mathcal{R}$ ;
9     Sample a random mini-batch of  $\mathcal{N}$  transitions from  $\mathcal{R}$ . Use gradient descent (GD) to
        update  $\theta^Q$  and  $\theta^\mu$  by minimizing the loss functions  $TD^{\text{Sep}}$  in (4.3.5) and  $J^c$ 
        in (4.3.6), respectively;
10    end
11 end
12 for  $episode = 1$  till converge do
13   for  $k = 1$  to  $K$  do
14     Sample a random mini-batch of  $\mathcal{N}$  transitions from  $\mathcal{R}$ . Use GD to update
         $\theta^{\kappa_1}, \dots, \theta^{\kappa_N}$  by minimizing  $J^d$  in (4.3.7);
15   end
16 end
17 2) Joint Centralized and Distributed Training:
18 Reinitialize replay buffer  $\mathcal{R}$ .
19 for  $episode = 1$  till converge do
20   Initialize OU processes  $\{N_k^c\}$  and  $\{N_{n,k}^d\}$ .
21   for  $k = 1$  to  $K$  do
22     Obtain  $\mathbf{a}_k^c = \mu(\mathbf{O}_k^c | \theta^\mu) + N_k^c$ ;
23     Obtain  $\mathbf{a}_k^d$  with  $\mathbf{a}_{i,k}^d = \kappa_i(\mathbf{O}_{i,k}^d | \theta^{\kappa_i}) + N_{i,k}^d, \forall i$ ;
24     Observe  $\beta_k$  and obtain  $\mathbf{a}_k$ ;
25     Execute  $\mathbf{a}_k$ , observe  $c_k$  and  $\mathbf{O}_{k+1}^c$ ;
26     If  $k > 1$ : store transition  $\langle \mathbf{O}_{k-1}^c, \mathbf{a}_{k-1}^c, \mathbf{a}_{k-1}^d, \beta_{k-1}, c_{k-1}, \mathbf{O}_k^c, \mathbf{a}_k \rangle$  in  $\mathcal{R}$ ;
27     Sample a random mini-batch of  $\mathcal{N}$  transitions from  $\mathcal{R}$ . Use GD to update  $\theta^Q$  by
        minimizing  $TD^{\text{Joint}}$  in (4.3.8). Use GD to update  $\theta^\mu, \theta^{\kappa_1}, \dots, \theta^{\kappa_N}$  by minimizing
         $J^{\text{Joint}}$  in (4.3.9).
28   end
29 end

```

4.4 Numerical Results

4.4.1 Simulation setting

We present numerical results of the CDC-DRL algorithm for transmission power allocation of the remote estimation system, based on the simulation parameters summarized in Table 4.1.

Table 4.1: Summary of simulation parameters for CDC-DRL

Process state transition matrix, A_n	Randomly generated*
Channel states, $\mathbb{G}$	$\{10^{-4}, 10^{-3.5}, \dots, 10^{-0.5}\}$
Channel state transition matrix, $\mathbf{M}_n$	Randomly generated
Codeword length, L	200
Transmission rate, R	2
Background white noise, σ^2	0.01
Sensors' power budget, E^s	20
Server's power budget, E^d	200
Attacker's power budget, E^a	1000

*Spectral radius is randomly drawn in $(1, 1.3)$.

We consider three DoS attackers characterized by their Markovian action state transition matrices T^a , as presented in Table 4.2. Although all attackers share the same expected attack frequency, they exhibit different behavioral patterns: the dynamic attacker frequently alters its actions, the balanced attacker shows moderate variability, and the persistent attacker maintains a more consistent attacking/non-attacking behavior. The centralized critic has two hidden layers with 1024 neurons.

Table 4.2: Control channel attacker dynamics

Attacker	Balanced	Dynamic	Persistent
T^a	$\begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}$	$\begin{bmatrix} 0.1 & 0.9 \\ 0.9 & 0.1 \end{bmatrix}$	$\begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}$

Each centralized or distributed actor has two hidden layers with 256 neurons. The activation function of all hidden layers is ReLU. Our simulation compares the performance of CDC-DRL against the following benchmarks:

1. Centralized DDPG agent (DDPG): A purely centralized DRL approach where $\mathbf{a}_k^c$ is generated by DDPG [77]. There is no sensor's data channel transmission once its centralized control packet is lost, i.e., $\mathbf{a}_{n,k}^d = \mathbf{0}, \forall n$.
2. Distributed DDPG agent (MADDPG): A purely distributed DRL strategy with $\mathbf{a}_{n,k}^d$ generated by Multi-Agent DDPG[114], $\forall n$.

3. Centralized DDPG combined with distributed DDPG (DDPG+MADDPG): A straightforward combination of centralized actions from DDPG with distributed actions from MADDPG.

4.4.2 Testing Error Comparison

First, we compared the performance of CDC-DRL against benchmarks in small-scale systems ($N = 6, M = 3$) using six different sets of randomly generated parameters and three types of attackers. The results are shown in Fig. 4.3. Next, we evaluated the performance in large-scale systems

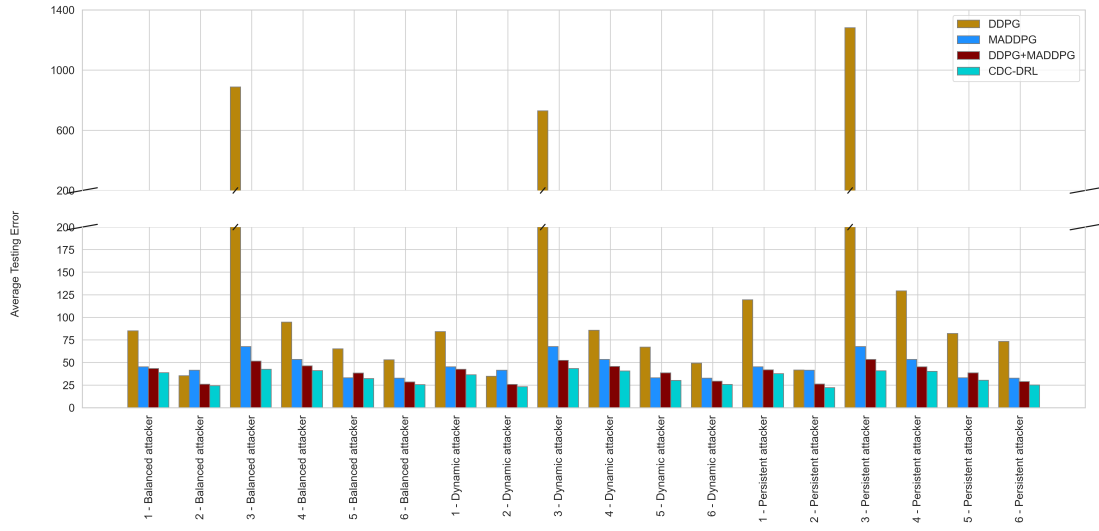


Figure 4.3: Comparison of average testing costs in small-scale systems.

($N = 20, M = 10$), as illustrated in Fig. 4.4. A summary of the average performance across all systems is provided in Fig. 4.5. The DoS attacker significantly degrades the performance of centralized approaches, with the balanced attacker being the most challenging to counter due to its unpredictable pattern. Notably, CDC-DRL consistently outperforms all combinations of DDPG and MADDPG, achieving a significantly lower average cost. In small-scale systems, CDC-DRL surpasses the best benchmark by 14.89%. As system scale increases, the complexity of the problem also rises, leading to greater performance variance in benchmark algorithms. In contrast, CDC-DRL shows a substantial performance improvement, reducing the average error by 88.32%.

Among the benchmarks, the centralized DDPG approach performs the worst, as it fails to counter

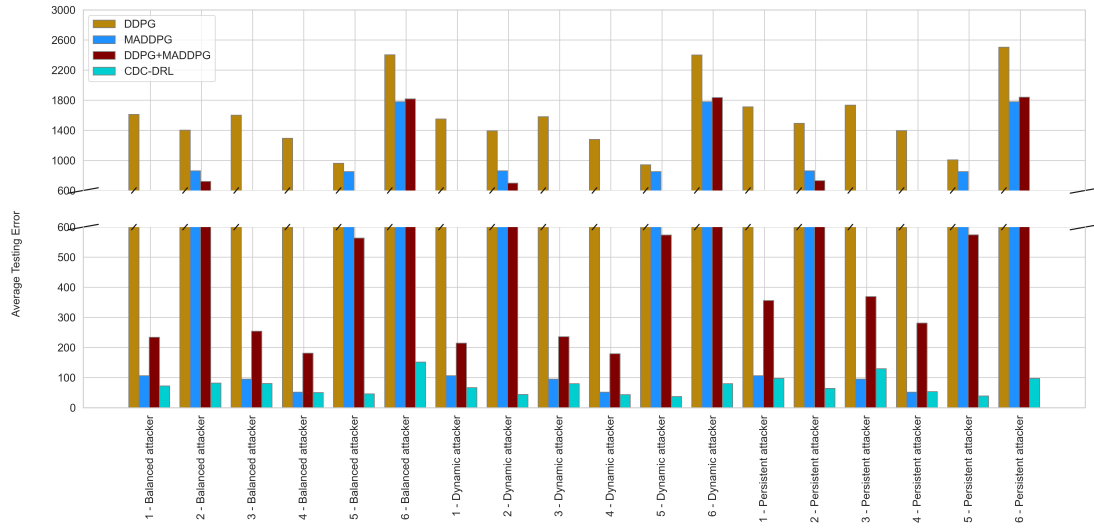


Figure 4.4: Comparison of average testing costs in large-scale systems.

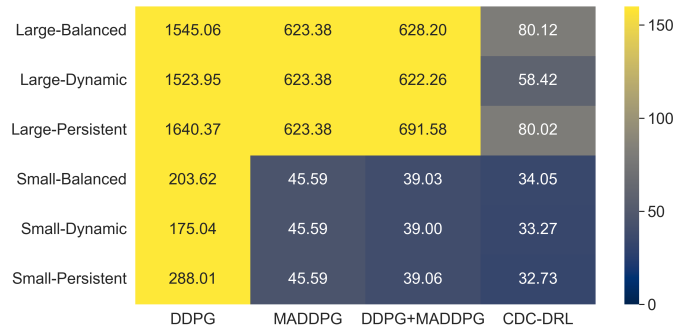


Figure 4.5: Comparison of average testing costs across the systems.

the attacker and leaves sensors inactive when not receiving centralized commands. While the distributed MADDPG approach remains unaffected by attackers, it results in sub-optimal performance due to the lack of comprehensive information available to the sensors. The direct combination of centralized and distributed algorithms (DDPG + MADDPG) is found unstable. This indicates that the CDC algorithm not only effectively addresses power allocation issues in the presence of various control channel attackers but also enhances training stability as system scale increases.

4.4.3 Convergence Analysis

We conducted an empirical convergence analysis of the proposed CDC-DRL algorithm. Training was halted when no performance improvement was observed over the last 100 episodes. The number of episodes required for convergence is shown in Fig. 4.6 and Fig. 4.7 for small and large-scale systems, respectively. As expected, the large-scale system required more time to converge compared to the small-scale system. Among the training stages, centralized pre-training took the longest, as it needed to learn the environment dynamics from scratch. The overall training duration for CDC-DRL was approximately 534 episodes for the small system and 641 episodes for the large system. Our experiments were conducted using an RTX 3070 GPU and an 11th Gen Intel i7-11700F CPU. Training CDC-DRL for 600 episodes took about 40 minutes. Using a standard edge server with 8 NVIDIA A40 GPUs [107], the training time would be reduced to approximately 3 minutes.

4.5 Chapter Conclusions

We have developed a CDC-DRL framework to address the power allocation challenges in remote estimation systems under control channel DoS attacks. Our proposed solution significantly outperforms benchmarks based on either centralized or distributed DRL approaches, particularly in large-scale systems. In future work, we will explore more comprehensive wireless resource allocation strategies for CPSs facing both data and control channel attacks.

The following chapter will study a wireless resource allocation problem in a two-hop relay system, which is vulnerable to an intelligent data channel DoS attacker.

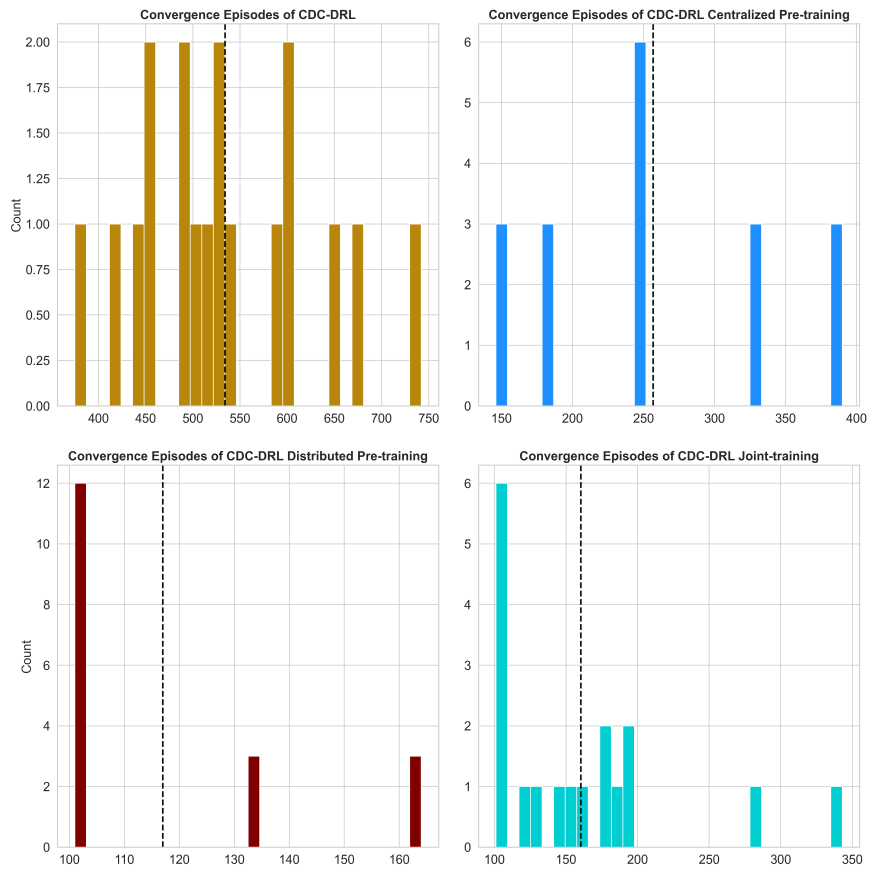


Figure 4.6: Convergence analysis of CDC-DRL in small scale systems.

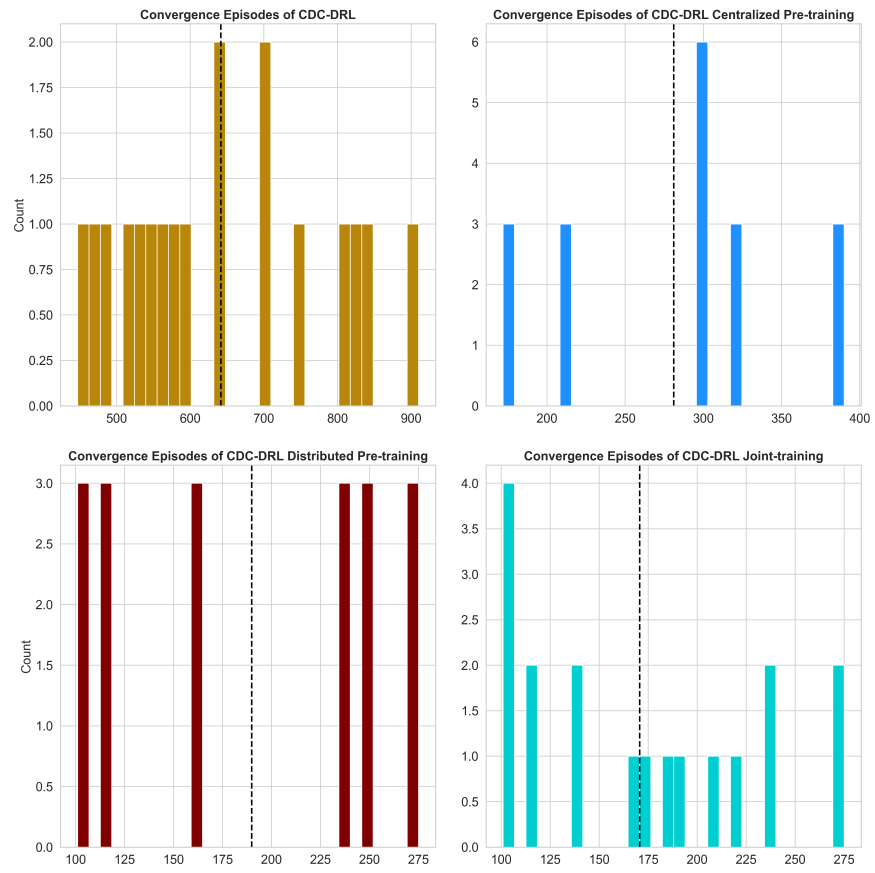


Figure 4.7: Convergence analysis of CDC-DRL in large scale systems.

Chapter 5

Wireless Resource Allocation of a two-hop relay network under DoS attack

This chapter explores a wireless resource allocation problem of remote state estimation in a two-hop relay system, which is vulnerable to intelligent data channel DoS attackers. A knowledge-based DRL framework is proposed to mitigate this type of attacker.

5.1 Chapter Introduction

The rapid advancement of wireless communication technologies has led to the widespread adoption of CPSs, which integrate physical processes with computational and networking capabilities [2]. These systems are increasingly used in critical applications such as intelligent transportation, advanced industries, smart grids, and intelligent agriculture [115–118]. The complexity and dynamic nature of CPSs make them vulnerable to various security threats, such as malicious DoS attacks. Ensuring the security and stability of CPSs is a crucial research area focused on developing defense mechanisms against cyber attacks.

In many CPS applications, sensors must transmit observations to remote estimators over imperfect wireless channels to monitor dynamic plant status in real-time [119]. A significant challenge is that sensors often have limited communication ranges, making direct information exchange with remote estimators impractical. Deploying relays between sensors and remote estimators is a viable solution. Han et al. studied a sensor and relay power scheduling problem for remote state estimation, considering a single sensor and multiple relays [120]. They proposed an optimal algorithm for scheduling the sending sequences for the sensor and relays. Gan et al. investigated a remote state estimation problem in a two-hop relay wireless network subject to a DoS attacker, formulating the attacker's optimization problem under the constraint that only one channel can be attacked [121]. Their follow-up work showed that attacking an unequal number of channels is more effective than attacking an equal number [122].

Motivated by these studies, this work focuses on a multi-sensor remote estimation problem in a two-hop relay network under physical layer DoS attacks. To mitigate the attacker's impact, we explore a novel knowledge-based DRL framework that enhances learning efficiency and optimizes joint resource allocation for both the sensor-relay and relay-server segments. The main contributions are summarized below.

1. *Novel problem formulation.* We formulate a comprehensive problem under a DoS attack that addresses sensor-relay scheduling and their corresponding power allocations. The scheduling and power allocation components are formulated as MDPs with partial observation, where sensor-relay scheduling operates in a discrete action space and power allocations are continuous. This multifaceted problem is further complicated by the attacker's strategy to maximize channel disruption.
2. *Knowledge-based DRL framework.* We develop an advanced DRL framework that operates on hybrid discrete and continuous action spaces and leverages domain knowledge to enhance learning efficiency and stability. This DRL architecture, employing a single policy, effectively reduces the overall complexity of the resource allocation challenges.

The remainder of this work is organized as follows. Section 5.2 introduces the system model of remote state estimation over a two-hop relay network. Section 5.3 presents the defender and

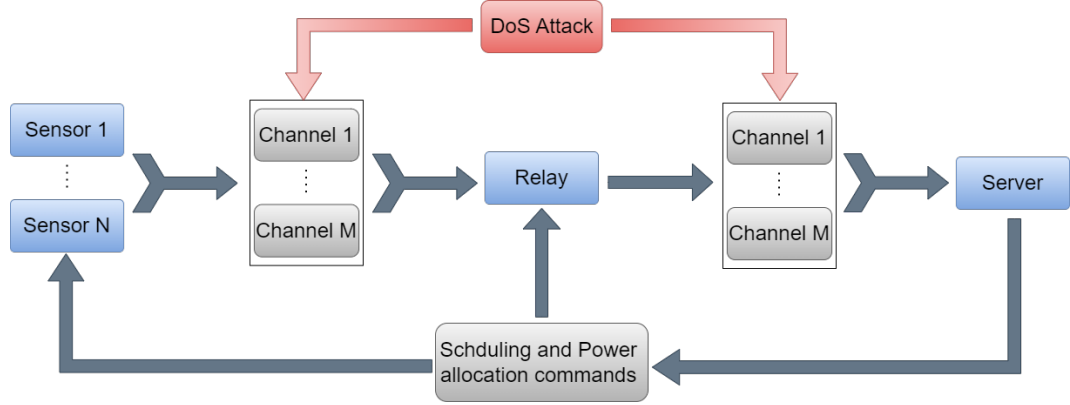


Figure 5.1: Multi-sensor remote state estimation in a two-hop relay network under DoS attack.

attacker's problem setup. The design of the DRL framework is detailed in Section 5.4. Numerical simulations are carried out in Section 5.5. Finally, Section 5.6 concludes this article.

5.2 System Model

We consider a remote estimation system comprising of N sensors, a relay and a remote estimator, as illustrated in Fig. 5.1. The system utilizes M channels for transmitting sensor information to the relay and for the relay's transmission to the remote estimator.

5.2.1 Local State Estimation with Smart Sensors

The physical process measured by sensor n is modeled as a discrete-time linear system as defined in section 3.2.1:

$$\begin{aligned} \mathbf{x}_{i,k+1} &= \mathbf{A}_i \mathbf{x}_{i,k} + \omega_{i,k}, \\ \mathbf{y}_{i,k} &= \mathbf{C}_i \mathbf{x}_{i,k} + \nu_{i,k}, \end{aligned} \tag{5.2.1}$$

where non-negative integer $k \in \mathbb{Z}^+$ is the time index. $\mathbf{x}_{i,k}$ represents the system state and $\mathbf{A}_i$ is the system state transition matrix. The measurement of sensor i is denoted by $\mathbf{y}_{i,k}$, with $\mathbf{C}_i$ as the measurement matrix. The terms $\omega_{i,k}$ and $\nu_{i,k}$ represent i.i.d. Gaussian measurement noise with covariance matrix $\mathbf{W}_i$ and $\mathbf{V}_i$, respectively. Instead of transmitting raw measurements $y_{i,k}$ directly, each sensor is assumed to be equipped with computational capability and pre-processes its measurement data using a local Kalman filter. The sensors then send their local state estimates $\hat{\mathbf{x}}_{i,k}^s$.

In this remote estimation scenario, we assume that each local Kalman filter is stable and operates at its steady state with the estimation error covariance matrix defined as $\overline{\mathbf{P}}_i \triangleq \mathbb{E}[(\mathbf{x}_{i,k} - \hat{\mathbf{x}}_{i,k}^s)(\mathbf{x}_{i,k} - \hat{\mathbf{x}}_{i,k}^s)^\top], \forall k \in \mathbb{Z}^+$.

5.2.2 Cooperative Transmission of Two-hop Relay Network

In practical scenarios, sensors may be located far from the remote estimator and have limited communication ranges, leading to potentially significant packet errors in direct transmissions. To enhance communication quality, we employ a half-duplex relay between the sensors and the server, forming a cooperative system. Due to limited wireless communication resources and potential packet loss, communication between the sensor and relay or between the relay and server may not always succeed in every time slot. The relay maintains a local memory buffer to store the most recently received packets from the sensors and transmits them based on the resource allocation policy described in the following section.

The scheduling decision at time k is represented by the binary variable $a_k \in \{0, 1\}$, where $a_k = 1$ indicates that sensors are scheduled to transmit their packets to the relay, and $a_k = 0$ indicates that the relay is scheduled to communicate with the remote estimator. The outcome of the transmission from sensor i to the relay is denoted by the binary variable $\phi_{i,k}$, where $\phi_{i,k} = 1$ indicates a successful transmission and $\phi_{i,k} = 0$ indicates a failure. Similarly, the outcome of the transmission from the relay to the remote estimator for sensor i 's packet is denoted by the binary variable $\gamma_{i,k}$, where $\gamma_{i,k} = 1$ indicates success and $\gamma_{i,k} = 0$ indicates failure. Consequently, when $a_k = 1$, $\gamma_{i,k} = 0$ for all i , and when $a_k = 0$, $\phi_{i,k} = 0$ for all i .

The relay's age of information (AoI) for sensor i at time k is defined as:

$$\lambda_{i,k} \triangleq \min\{t \geq 0 : \phi_{i,k-t} = 1\}, \quad (5.2.2)$$

and the aggregated AoI at the relay is denoted by $\Lambda_k \triangleq [\lambda_{1,k}, \dots, \lambda_{N,k}]$. Thus, the up-to-date packet at the relay for sensor i used in the estimation process is given by:

$$\tilde{\mathbf{x}}_{i,k} = \mathbf{A}_i^{\lambda_{i,k}} \hat{\mathbf{x}}_{i,k-\lambda_{i,k}}^s \quad (5.2.3)$$

Similarly, we define the AoI at the server at time k as:

$$\begin{aligned}\tau_{i,k} &\triangleq \min\{\lambda_{i,t} + t \geq 0 : \gamma_{i,k-t} = 1\} \\ &= \begin{cases} \lambda_{i,k} & \text{if } \gamma_{i,k} = 1 \\ \tau_{i,k-1} + 1 & \text{otherwise,} \end{cases},\end{aligned}\tag{5.2.4}$$

and $T_k \triangleq [\tau_{1,k}, \dots, \tau_{N,k}]$ denotes the aggregated AoI at the server.

5.2.3 Remote State Estimation

Due to limited wireless communication resources, the scheduling decision a_k , and potential packet errors, the server may not receive all local sensor estimates in each time slot. Therefore, it employs a Minimum Mean-Square Error (MMSE) estimator for each process [103]. The remote state estimate and the error covariance matrix are given by:

$$\begin{aligned}\hat{\mathbf{x}}_{i,k} &= \mathbf{A}_i^{\tau_{i,k}} \hat{\mathbf{x}}_{i,k-\tau_{i,k}}^s, \\ \mathbf{P}_{i,k} &= h_i^{\tau_{i,k}}(\overline{\mathbf{P}}_i),\end{aligned}\tag{5.2.5}$$

where $h_i(X) \triangleq \mathbf{A}_i X \mathbf{A}_i^\top + W_i$ is the Lyapunov operator and $h_i^{j+1}(X) = h_i(h_i^j(X))$. The error covariance matrix $\mathbf{P}_{i,k}$ in (5.2.5) can be rewritten as:

$$\mathbf{P}_{i,k} = h_i^{\tau_{i,k}}(\overline{\mathbf{P}}_i).\tag{5.2.6}$$

5.2.4 NOMA-based Data Transmission Channel

Sensor-Relay transmission

We consider M orthogonal channels (e.g., subcarriers) for sensor data transmissions. To model the time-varying and correlated nature of wireless fading channels, we use a time-homogeneous, ergodic Markov block-fading channel model with ψ quantization levels [100]. Each channel's power gain remains constant within a packet transmission time slot but may vary between packets [99].

The channel power gain from sensor i to the relay across M channels is defined as

$$\mathbf{G}_{i,k} \triangleq [G_{1,k}^i, \dots, G_{M,k}^i]^\top,\tag{5.2.7}$$

where $G_{m,k}^i \in \mathbb{G}$ represents the discrete state of channel power gain on channel m . The set of channel gains $\mathbb{G}$ is defined as

$$\mathbb{G} \triangleq \{h_1, \dots, h_\psi\}.\tag{5.2.8}$$

The sensor's channel state vector $\mathbf{G}_{i,k}$ is modeled as a multi-state Markov chain with $\bar{\psi}$ states:

$$\mathbf{G}_{i,k} \in \{\mathbf{h}_1, \dots, \mathbf{h}_{\bar{\psi}}\}. \quad (5.2.9)$$

Assuming independent Markov channel states for different sensors due to their unique locations and varying radio propagation conditions, the channel state transition matrix for sensor i , $\mathbf{M}_i$, is defined as

$$\mathbf{M}_i \triangleq \begin{bmatrix} \bar{m}_{1,1} & \dots & \bar{m}_{1,\bar{\psi}} \\ \vdots & \ddots & \vdots \\ \bar{m}_{\bar{\psi},1} & \dots & \bar{m}_{\bar{\psi},\bar{\psi}} \end{bmatrix} \in \mathbb{R}^{\bar{\psi} \times \bar{\psi}}, \quad (5.2.10)$$

where $\bar{m}_{\bar{i},\bar{j}}$ represents the transition probability from state $\mathbf{h}_{\bar{i}}$ to $\mathbf{h}_{\bar{j}}$.

We adopt an advanced NOMA scheme based on [57]. Each channel can accommodate multiple concurrent sensors, and each sensor can send its packet across one or more of the M orthogonal channels given a total power budget E^s . The transmission power of sensor i in all M channels at time slot k is captured in the vector

$$\mathbf{E}_{i,k} \triangleq [E_{1,k}^i, \dots, E_{M,k}^i]^\top, \quad (5.2.11)$$

where $\sum_{m=1}^M E_{m,k}^i \leq E^s$. If $\sum_{m=1}^M E_{m,k}^i = 0$, then sensor i is not scheduled for transmission at time k . The aggregate transmission power of all sensors is represented by:

$$\mathbf{E}_k \triangleq [\mathbf{E}_{1,k}, \dots, \mathbf{E}_{N,k}]. \quad (5.2.12)$$

Relay-Server transmission

The relay-server makes use of the same M orthogonal channels for communication. The channel gain from the relay to the remote estimator is defined as

$$\tilde{\mathbf{G}}_k \triangleq [\tilde{G}_{1,k}, \dots, \tilde{G}_{M,k}]^\top, \quad (5.2.13)$$

where $\tilde{G}_{m,k} \in \mathbb{G}, \forall m \in \{1, \dots, M\}$. In parallel to the sensor-relay communication, the channel state vector from the relay to the remote estimator is also modeled as a multi-state Markov chain with $\bar{\psi}$ states as

$$\tilde{\mathbf{G}}_k \in \{\mathbf{h}_1, \dots, \mathbf{h}_{\bar{\psi}}\}. \quad (5.2.14)$$

The channel state transition matrix for the relay is defined as $\widetilde{\mathbf{M}} \in \mathbb{R}^{\bar{\psi} \times \bar{\psi}}$, similar to equation (5.2.10).

The relay transmit all sensors' most recent packet $\widetilde{\mathbf{x}}$ to the server, where $\widetilde{\mathbf{x}}$ is defined as

$$\widetilde{\mathbf{x}} \triangleq [\widetilde{\mathbf{x}}_{1,k}, \dots, \widetilde{\mathbf{x}}_{N,k}] \quad (5.2.15)$$

The transmission power of sensor i 's most recent packet in all M channels at time slot k is captured in the vector

$$\widetilde{\mathbf{E}}_{i,k} \triangleq [\widetilde{E}_{1,k}^i, \dots, \widetilde{E}_{M,k}^i]^\top, \quad (5.2.16)$$

with the energy constraint

$$\sum_{i=1}^N \sum_{m=1}^M \widetilde{E}_{m,k}^i \leq E^r, \quad (5.2.17)$$

where E^r is the energy budget of the relay. If $\sum_{m=1}^M \widetilde{E}_{m,k}^i = 0$, then relay are not transmitting sensor i 's packet at time k . The aggregate transmission power at the relay of all sensors is represented by

$$\widetilde{\mathbf{E}}_k \triangleq [\widetilde{\mathbf{E}}_{1,k}, \dots, \widetilde{\mathbf{E}}_{N,k}]. \quad (5.2.18)$$

DoS Attacker at Data Channel

We assume a DoS attacker targets the M channels by transmitting a wireless jamming signal. Given the attacker's energy budget E^a and the need to remain undetected, we assume the attacker can target at most one channel per time slot. The attacker's decision at time k is defined as:

$$\mathbf{a}_k^a = [a_{1,k}^a, \dots, a_{M,k}^a], \quad (5.2.19)$$

where the constraints on a_k^a are given by

$$\begin{aligned} a_{m,k}^a &\in \{0, E^a\}, \\ \sum_{m=1}^M \mathbb{1}(a_{m,k}^a = E^a) &\leq 1. \end{aligned} \quad (5.2.20)$$

Similar to (5.2.7) and (5.2.13), the channel gain from the attacker to the relay is defined as

$$\mathbf{G}_k^a \triangleq [G_{1,k}^a, \dots, G_{M,k}^a]^\top, \quad (5.2.21)$$

and the channel gain from the attacker to the remote estimator is defined as

$$\widetilde{\mathbf{G}}_k^a \triangleq [\widetilde{G}_{1,k}^a, \dots, \widetilde{G}_{M,k}^a]^\top. \quad (5.2.22)$$

Here, both $\mathbf{G}_k^a$ and $\widetilde{\mathbf{G}}_k^a$ are modeled as a multi-state Markov chain with $\bar{\phi}$ states.

Packet detection

We follow an N -round IRC and SIC algorithm to recover each packet from the superimposed signals across all M channels. When the sensors are scheduled to transmit their packets, i.e., $a_k = 1$, sensor i 's signal for the relay to detect is rewritten from equation (4.2.7) as

$$\begin{aligned} \mathbf{u}_k &= \sum_{i=1}^N (\sqrt{\mathbf{E}_{i,k}} \odot \mathbf{G}_{i,k}) u_{i,k} + \mathbf{q}_{i,k}, \\ \mathbf{q}_{i,k} &= \sum_{n=1, n \neq i}^N (\sqrt{\mathbf{E}_{n,k}} \odot \mathbf{G}_{n,k}) u_{n,k} + \mathbf{z}. \end{aligned} \quad (5.2.23)$$

Here, $\mathbf{q}_{i,k}$ is treated as the colored interference to $u_{i,k}$ since the interference at different channels are correlated. Due to the existence of the attacker, the covariance of $\mathbf{q}_{i,k}$ is

$$\begin{aligned} \mathbf{R}_{i,k} &\triangleq \mathbb{E} [\mathbf{q}_{i,k} \mathbf{q}_{i,k}^H] \\ &= \sigma^2 \mathbf{I}_M + \sum_{n=1, n \neq i}^N (\sqrt{\mathbf{E}_{n,k}} \odot \mathbf{G}_{n,k}) (\sqrt{\mathbf{E}_{n,k}} \odot \mathbf{G}_{n,k})^H \\ &\quad + (\sqrt{\mathbf{a}_k^a} \odot \mathbf{G}_k^a) (\sqrt{\mathbf{a}_k^a} \odot \mathbf{G}_k^a)^H \end{aligned} \quad (5.2.24)$$

where $(\cdot)^H$ is the conjugate transpose operator.

Similarly, when the relay transmits the sensors' packets stored in its local memory, sensor i 's signal for the remote estimator to detect is written as

$$\begin{aligned} \tilde{\mathbf{u}}_k &= \sum_{i=1}^N (\sqrt{\tilde{\mathbf{E}}_{i,k}} \odot \tilde{\mathbf{G}}_{i,k}) \tilde{u}_{i,k} + \tilde{\mathbf{q}}_{i,k}, \\ \tilde{\mathbf{q}}_{i,k} &= \sum_{n=1, n \neq i}^N (\sqrt{\tilde{\mathbf{E}}_{n,k}} \odot \tilde{\mathbf{G}}_{n,k}) \tilde{u}_{n,k} + \tilde{\mathbf{z}}. \end{aligned} \quad (5.2.25)$$

The covariance of $\tilde{\mathbf{q}}_{i,k}$ is

$$\begin{aligned} \tilde{\mathbf{R}}_{i,k} &\triangleq \mathbb{E} [\tilde{\mathbf{q}}_{i,k} \tilde{\mathbf{q}}_{i,k}^H] \\ &= \sigma^2 \mathbf{I}_M + \sum_{n=1, n \neq i}^N (\sqrt{\tilde{\mathbf{E}}_{n,k}} \odot \tilde{\mathbf{G}}_{n,k}) (\sqrt{\tilde{\mathbf{E}}_{n,k}} \odot \tilde{\mathbf{G}}_{n,k})^H \\ &\quad + (\sqrt{\mathbf{a}_k^a} \odot \tilde{\mathbf{G}}_k^a) (\sqrt{\mathbf{a}_k^a} \odot \tilde{\mathbf{G}}_k^a)^H \end{aligned} \quad (5.2.26)$$

Based on the scheduling decision a_k , in each iteration, the server or the relay computes the SINR for the undecoded sensors using IRC and executes SIC to detect the packet with the highest SINR. The process terminates when either a packet detection error occurs or all packets are detected

successfully. Assuming each sensor packet has a block length of L and data rate R , the packet drop probabilities for each sensor can be accurately approximated based on the detailed SINR calculation and finite-blocklength information theory [101], as shown in [57]. Further specifics are included in section 4.2.2.

5.3 Defending and Attacking Policies

5.3.1 Defender design

The remote estimator dynamically determines the scheduling action a_k , the power allocation action for the sensors $\mathbf{E}_k$ and power allocation for the relay $\tilde{\mathbf{E}}_k$ based on its policy π at each time k , adhering to the constraints. The action spaces for scheduling and the two power allocations are defined as $\mathcal{A}$, $\mathcal{E}$, and $\tilde{\mathcal{E}}$, respectively. Here $\mathcal{A}$ is finite and discrete. $\mathcal{E}$ and $\tilde{\mathcal{E}}$ are two different infinite and continuous action spaces.

The defender policy $\pi(\cdot)$ is a mapping from its observation state $\mathbf{O}_k$ to the actions $\mathbf{A}_k \triangleq (a_k, \mathbf{E}_k, \tilde{\mathbf{E}}_k)$. The state $\mathbf{O}_k$ is defined as

$$\mathbf{O}_k = [\Lambda_{k-1}, T_{k-1}, \mathbf{G}_{1,k}^\top, \dots, \mathbf{G}_{N,k}^\top, \tilde{\mathbf{G}}_k^\top] \quad (5.3.1)$$

which takes into account the AoI information of all sensors at the relay from the previous time step, the AoI at the remote estimator from the previous time step, the current channel states from the sensors to the relay, and the current channel states from the relay to the remote estimator. In particular, the attacker's action $\mathbf{a}_k^a$ and the attacker's channel states $(\mathbf{G}_k, \tilde{\mathbf{G}}_k)$ are assumed unknown to the defender. This takes consideration that the malicious attacker is usually difficult to detect. The state space is defined as $\mathcal{O}$.

5.3.2 Attacker design

The attacker's decision at time k is $\mathbf{a}_k^a$, which satisfies the constraint specified in (5.2.20). The discrete attacker action space is denoted as $\mathcal{A}^a$. We assume that the attacker has access to the observed states at the defender but not the gateway actions. Realistically, such knowledge is not available, and hence attacks will not be as sophisticated as this "genie-aided" scenario. The attacker

state is defined in a similar way to the defender's as

$$\mathbf{O}_k^a = [\Lambda_{k-1}, T_{k-1}, \mathbf{G}_k^{a^\top}, \tilde{\mathbf{G}}_k^{a^\top}] \quad (5.3.2)$$

and the corresponding state space is defined as $\mathcal{O}^a$.

5.3.3 Problem setup

We define the cost function as the expected sum of estimation errors for all processes to measure the remote estimation performance:

$$c_k \triangleq \sum_{i=1}^N \text{tr} \mathbf{P}_{i,k}, \quad (5.3.3)$$

where tr denotes the trace operator. The cost c_k is a function of the sensors' AoI, T_k . Let $\pi(\cdot)$ and $\pi^a(\cdot)$ denote the policy of the remote estimator and the attacker, respectively. The policy is a mapping from its observation to the action. The remote estimator and the attacker have the following two-player minmax game with a long-term discounted cost function $J(\pi(\cdot), \pi^a(\cdot))$:

$$\begin{aligned} \min_{\pi(\cdot)} \max_{\pi^a(\cdot)} J(\pi(\cdot), \pi^a(\cdot)) &= \mathbb{E} \left[\sum_{k=1}^{\infty} \delta^{k-1} c_k \right], \\ \text{s.t. } a_k, \mathbf{E}_k, \tilde{\mathbf{E}}_k &= \pi(\cdot), \\ a_k &\in \{0, 1\}, \\ \sum_{m=1}^M E_{m,k}^i &\leq E^s, \forall i \in \{1, \dots, N\}, \\ \sum_{i=1}^N \sum_{m=1}^M \tilde{E}_{m,k}^i &\leq E^r, \\ \mathbf{a}_k^a &= \pi^a(\cdot), \\ a_{m,k}^a &\in \{0, E^a\}, \forall m \in \{1, \dots, M\}, \\ \sum_{m=1}^M \mathbb{1}(a_{m,k}^a = E^a) &\leq 1, \end{aligned} \quad (5.3.4)$$

where c_k is the per-step cost defined in (5.3.3) and $\delta \in (0, 1)$ is the discount factor. For the remote estimator, the optimization problem is to minimize $J(\cdot)$ while the attacker aims to maximize it.

This is a decision-making problem, as the current actions depend on the current states (i.e., the observations) and will influence the next states due to (5.2.4). We can prove that this decision making problem (5.3.4) has the Markov property, i.e., given the current state and action, the current reward

and the next state are independent of all previous states and actions [88], and thus can formulate the problems into Markov Decision Processes (MDPs). Classic MDPs are commonly solved by the value or policy iteration method, which requires explicit model information, i.e., the state transition probabilities. However, it is easy to show that in problem (5.3.4), the state transition probability $\mathbb{P}[\mathbf{O}_{k+1}|\mathbf{O}_k, a_k, \mathbf{E}_k, \tilde{\mathbf{E}}_k]$ or $\mathbb{P}[\mathbf{O}_{k+1}^a|\mathbf{O}_k^a, \mathbf{a}_k^a]$ is also determined by the adversary's policy, which is completely unknown. Thus, we resort to model-free solutions.

5.4 Deep Reinforcement Learning Algorithms

DRL is a widely applied model-free method for solving decision-making problems. Unlike classical MDPs, DRL does not require explicit state transition probabilities. Instead, it leverages sampled state transitions, in our case represented as $\langle O, A, c, O' \rangle$, where O is the current state, A is the action, c is the cost, and O' is the next state. These transitions are used to train neural networks to approximate the optimal policy that minimize the discounted long-term average cost.

5.4.1 Defender's Algorithm

Given that the defender's action space includes both discrete and continuous components, we employ the Hybrid Deep Deterministic Policy Gradient (HDDPG) method, as detailed in Section 3.3.3.

Network architecture

Our HDDPG algorithm comprises three main components: a critic network $Q(\mathbf{O}_k, a_k, \mathbf{E}_k, \tilde{\mathbf{E}}_k|\theta^Q)$ and two actor networks $\mu(\mathbf{O}_k|\theta^\mu)$ and $\tilde{\mu}(\mathbf{O}_k|\theta^{\tilde{\mu}})$.

The critic network approximates the Q-function, which estimates the average discounted sum of future costs given the current state and actions $(\mathbf{O}_k, a_k, \mathbf{E}_k, \tilde{\mathbf{E}}_k)$ at time k :

$$Q^\pi(\mathbf{O}, a, \mathbf{E}, \tilde{\mathbf{E}}) = \mathbb{E} \left[\sum_{t=k}^{\infty} -\delta^{t-k} c_t \middle| \mathbf{O}_k = \mathbf{O}, a_k = a, \mathbf{E} = \mathbf{E}_k, \tilde{\mathbf{E}} = \tilde{\mathbf{E}}_k, (a_t, \mathbf{E}_t, \tilde{\mathbf{E}}_t) = \pi(\mathbf{O}_t), \forall t > k \right] \quad (5.4.1)$$

The optimal Q-function with the optimal policy, i.e., $Q^*(\cdot)$ should have the minimum expected future discounted cost given the states and actions. Define U as the cardinality of the discrete scheduling action space, i.e., $U \triangleq |\mathcal{A}|$, and $a_{(n)}$ as the n th element of $\mathcal{A}$. The critic with parameter set θ takes

$(\mathbf{O}, \mathbf{E}, \tilde{\mathbf{E}}) \in \mathcal{O} \times \mathcal{E} \times \tilde{\mathcal{E}}$ as the input and has U outputs $Q(\mathbf{O}, a_{(1)}, \mathbf{E}, \tilde{\mathbf{E}}|\theta), \dots, Q(\mathbf{O}, a_{(U)}, \mathbf{E}, \tilde{\mathbf{E}}|\theta)$ as the approximations of the optimal Q-values $Q^*(\mathbf{O}, a_{(1)}, \mathbf{E}, \tilde{\mathbf{E}}), \dots, Q^*(\mathbf{O}, a_{(U)}, \mathbf{E}, \tilde{\mathbf{E}})$.

The actor networks function as policy executors. The first actor network, parameterized by θ^μ , maps the input state $\mathbf{O} \in \mathcal{O}$ to the power allocation action for the sensors $\mathbf{E} \in \mathcal{E}$, i.e., $\mathbf{E} = \mu(\mathbf{O}|\theta^\mu)$. To satisfy the power allocation constraints in (5.3.4), the actor μ is designed with $N + 1$ heads, denoted by $h_{(1)}, \dots, h_{(N)}, h'$, to determine the sensor's transmission power. For the n th sensor, $h_{(n)}$ outputs $\mathbf{L}_{(n)} \in \mathbb{R}^{M \times 1}$, which is processed through a softmax activation function to determine the energy distribution across M channels. The concatenated outputs of $h_{(1)}, \dots, h_{(N)}$ are defined as

$$\mathbf{L} \triangleq [\mathbf{L}_{(1)}, \dots, \mathbf{L}_{(N)}] \in \mathbb{R}^{M \times N}. \quad (5.4.2)$$

The final head, h' , outputs $\mathbf{S} \in \mathbb{R}^M$ using a sigmoid activation function to determine the fractions of energy for the sensors. The final output of μ is given by

$$\mathbf{E} = E^* \mathbf{L} \mathbf{S}^d, \quad (5.4.3)$$

where the diagonal matrix $\mathbf{S}^d \triangleq \text{diag}\{\mathbf{S}\}$. The product of these outputs provides the transmission power that satisfies the power constraint.

The second actor network, parameterized by $\theta^{\tilde{\mu}}$, translates the input state $\mathbf{O} \in \mathcal{O}$ into the power allocation action for the relay, denoted as $\tilde{\mathbf{E}} \in \tilde{\mathcal{E}}$. $\tilde{\mu}$ is similarly structured with two heads, labeled $\tilde{h}$ and $\tilde{h}'$. The head $\tilde{h}$ generates the energy distribution for all sensors, represented as $\tilde{\mathbf{L}} \in \mathbb{R}^{M \times N}$, by applying a softmax activation function. The second head $\tilde{h}'$ produces $\tilde{\mathbf{S}} \in \mathbb{R}^M$ using a sigmoid activation function to determine the energy fractions for the sensors. The final output of $\tilde{\mu}$ is expressed as

$$\tilde{\mathbf{E}} = E^r \tilde{\mathbf{L}} \tilde{\mathbf{S}}^d, \quad (5.4.4)$$

where the diagonal matrix $\tilde{\mathbf{S}}^d \triangleq \text{diag}\{\tilde{\mathbf{S}}\}$.

The architecture of these neural networks is summarized in the figure below.

Challenges

Implementing HDDPG directly presents several challenges:

- 1). *Ineffective actions due to problem constraints.* Beyond the energy constraints specified in (5.3.4), our problem has intrinsic characteristics. The scheduling action directly determines the

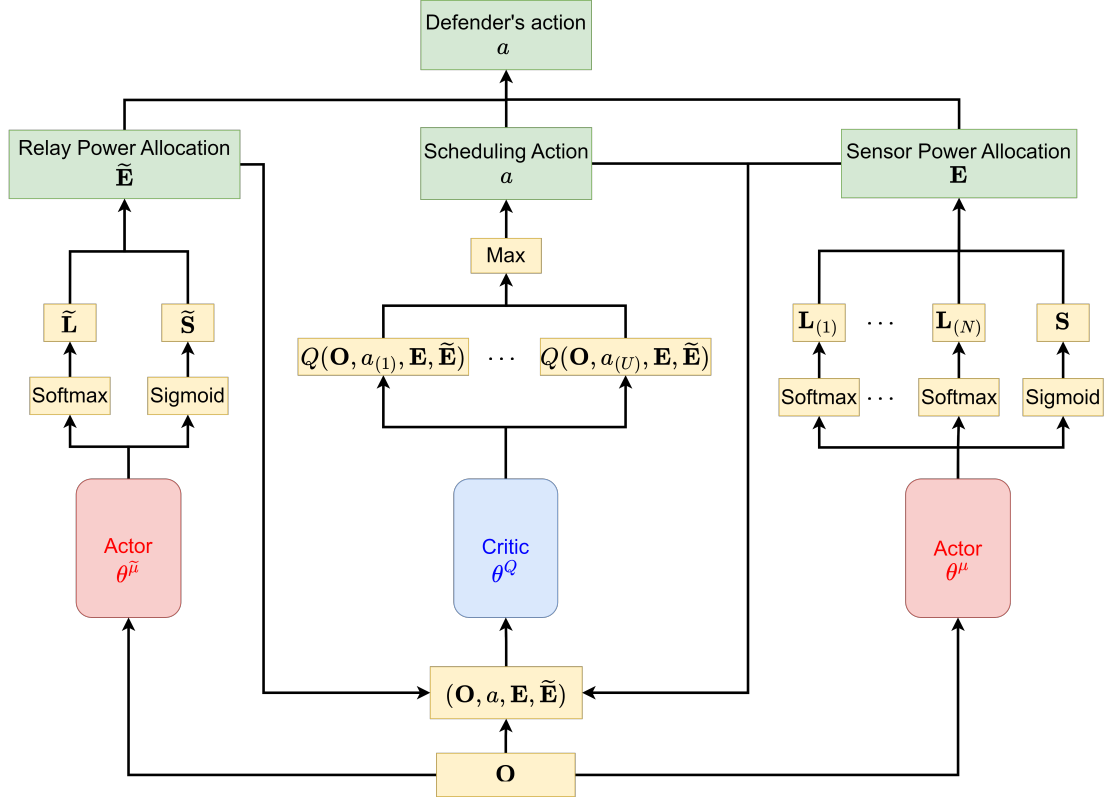


Figure 5.2: HDDPG architecture for resource allocation in a two-hop relay network.

transmission outcome of the two communication components. Specifically, if sensors are scheduled to communicate with the relay, the relay-server communication will always fail. Consequently, the power allocation action for the relay-server part no longer affects the state transition probability. Let the ineffective power allocation be denoted as $\mathbf{E}'_k$. This prior information can be summarized as:

$$\mathbf{E}'_k = \begin{cases} \tilde{\mathbf{E}}_k & \text{if } a_k = 1 \\ \mathbf{E}_k & \text{otherwise} \end{cases} \quad (5.4.5)$$

2). *Lack of prior knowledge application.* The sensor-relay and relay-server communications can at most have one successful transmission, i.e., $\mathbb{P}(\gamma_{i,k} = 1 \cap \phi_{i,k} = 1) = 0, \forall i, \forall k$. Thus, when $\tau_{i,k-1} = \lambda_{i,k-1}$, the Age of Information (AoI) at the server's update equation 5.2.4 can be rewritten as:

$$\tau_{i,k} = \begin{cases} \lambda_{i,k} = \lambda_{i,k-1} + 1 = \tau_{i,k-1} + 1 & \text{if } \gamma_{i,k} = 1 \\ \tau_{i,k-1} + 1 & \text{otherwise} \end{cases} \quad (5.4.6)$$

Since the successful detection of sensor i 's packet no longer affects its AoI at the server, the power allocation $\tilde{\mathbf{E}}_{i,k}$ should be an all-zero vector to increase the detection probabilities of other sensors in this case.

3). *Large action space exploration.* The algorithm's large hybrid discrete and continuous action space $\mathcal{A} \times \mathcal{E} \times \tilde{\mathcal{E}}$ can lead to unstable training. This motivates us to propose an advanced knowledge-based algorithm to address these issues.

This motivates us to propose an advanced knowledge based HDDPG to tackle the issues above.

Training Procedure

The Critic Network

The training process follows an off-policy scheme using transitions that include the state, action, cost, and next state, denoted as $\langle \mathbf{O}, (a, \mathbf{E}, \tilde{\mathbf{E}}), c, \mathbf{O}' \rangle$. The Q-values of the optimal policy π^* satisfy the Bellman equation:

$$Q^*(\mathbf{O}, a, \mathbf{E}, \tilde{\mathbf{E}}) = \mathbb{E} \left[r + \delta \max_{a' \in \mathcal{A}} Q^*(\mathbf{O}', a', \mathbf{E}', \tilde{\mathbf{E}}') \right]. \quad (5.4.7)$$

Thus, the optimal scheduling action is $\arg \max_{a \in \mathcal{A}} Q^*(\mathbf{O}, a, \mathbf{E}, \tilde{\mathbf{E}})$. We approximate the Q-function using the critic network $Q(\mathbf{O}_k, a_k, \mathbf{E}_k, \tilde{\mathbf{E}}_k | \theta^Q)$ and train the critic to approach the optimal Q by minimizing the squared temporal difference (TD) error, which measures the difference between the predicted Q-value and the target Q-value. The loss function for updating the critic is the TD error over a mini-batch of $\mathcal{N}$ transitions, defined as:

$$TD = \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} \left[c_n + \delta \max_{a' \in \mathcal{A}} Q(\mathbf{O}'_n, a', \mu(\mathbf{O}'_n | \theta^\mu), \tilde{\mu}(\mathbf{O}'_n | \theta^{\tilde{\mu}})) - Q(\mathbf{O}_n, a_n, \mathbf{E}_n, \tilde{\mathbf{E}}_n | \theta^Q) \right]^2. \quad (5.4.8)$$

Minimizing the TD error narrows the gap between the left-hand side and right-hand side of the Bellman equation (5.4.7), ensuring the critic network's predictions become more accurate over time.

The Actor Network

The actor networks are trained to minimize the expected cost derived from the critic. The loss function for updating the actors is the average expected cost over a mini-batch, defined as:

$$L = \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} Q(\mathbf{O}_n, a_n, \mu(\mathbf{O}_n | \theta^\mu), \tilde{\mu}(\mathbf{O}_n | \theta^{\tilde{\mu}}) | \theta^Q). \quad (5.4.9)$$

Knowledge-Based Training Adjustments

Given the challenges outlined in Section 5.4.1, we propose the following knowledge-based adjustments to enhance the training quality of HDDPG:

1). *Data Augmentation for Model Robustness*

To address ineffective power allocation $\mathbf{E}'_k$ resulting from various scheduling actions, we introduce additional generated transitions, denoted as GT :

$$GT = \begin{cases} \langle \mathbf{O}, (a, \mathbf{E}, \tilde{g}(\tilde{\mathbf{E}} + \epsilon)), c, \mathbf{O}' \rangle & \text{if } a_k = 1 \\ \langle \mathbf{O}, (a, g(\mathbf{E} + \epsilon), \tilde{\mathbf{E}}), c, \mathbf{O}' \rangle & \text{otherwise,} \end{cases} \quad (5.4.10)$$

where ϵ is zero-mean Gaussian noise. Functions $g(\cdot)$ and $\tilde{g}(\cdot)$ normalize power allocation to meet energy constraints in 5.3.4. The purpose of GT is to introduce noise into ineffective power allocations, thereby enhancing the robustness and stability of the training process.

2). *Relay Power Allocation Adjustment Based on Domain Knowledge*

If the condition $\tau_{i,k-1} = \lambda_{i,k-1}$ is met, we adjust the power allocation $\tilde{\mathbf{E}}_{i,k}$ to an all-zero vector to further improve training efficiency.

3). *Heuristic Power Allocation Exploration*

To ensure sufficient exploration of the action space, we implement a two-stage pre-training process. In the first stage, we use a one-sensor-to-one-channel strategy, where each channel is occupied by a random distinct sensor's packet. This teaches the policy to successfully transmit some sensor packets. In the second stage, we apply a random power allocation strategy to further learn the channel statistics.

Overall Training Algorithm

The overall training framework of knowledge-based HDDPG is described in Algorithm 6.

5.4.2 DoS Attacker's Algorithm

To solve (5.2.4) with the discrete action space $\mathcal{A}^a$, we use the Deep Q Learning (DQL) algorithm [76], which includes the Deep Q Network (DQN) $Q^a(\mathbf{O}^a, \mathbf{a}^a | \theta^a)$. The defender's policy is assumed to be unknown to the attacker. Therefore, the attacker is trained based on a stochastic defending policy where $\mathbf{A}_k$ is randomly generated.

Algorithm 6: Knowledge-based HDDPG training

```

1 Initialize network parameters  $\theta^Q$ ,  $\theta^\mu$ ,  $\theta^{\tilde{\mu}}$ , replay buffer  $\mathcal{R}$ .
2 for  $episode = 1$  to  $E_{train}$  do
3   Initialize Ornstein-Uhlenbeck(OU) process  $\{N_k\}$  and  $\{\tilde{N}_k\}$  [77].
4   for  $k = 1$  to  $K$  do
5     Obtain the sensor-relay power allocation action
6     
$$\mathbf{E}_k = \begin{cases} \text{One-sensor-to-one-channel} & \text{if in the first pre-training stage} \\ \text{Random power allocation} & \text{elif in the second pre-training stage} \\ \mu(\mathbf{O}_k|\theta^\mu) + N_k & \text{otherwise;} \end{cases}$$

7     Obtain the relay-server power allocation action
8     
$$\tilde{\mathbf{E}}_k = \begin{cases} \text{One-sensor-to-one-channel} & \text{if in the first pre-training stage} \\ \text{Random power allocation} & \text{elif in the second pre-training stage} \\ \tilde{\mu}(\mathbf{O}_k|\theta^{\tilde{\mu}}) + \tilde{N}_k & \text{otherwise;} \end{cases}$$

9     Obtain the scheduling action  $a_k = \arg \max_{a \in \mathcal{A}} Q^*(\mathbf{O}, a, \mathbf{E}, \tilde{\mathbf{E}})$ , following a behavior
        distribution for action exploration [76];
10    if  $\tau_{i,k-1} = \lambda_{i,k-1}$  then
11       $\tilde{\mathbf{E}}_{i,k}$  is adjusted to  $\mathbf{0}$ 
12    end
13    Execute  $a_k, \mathbf{E}_k, \tilde{\mathbf{E}}_k$ , observe cost  $c_k$  and observe new state  $\mathbf{O}_{k+1}$ ;
14    Store transition  $\langle \mathbf{O}_k, (a_k, \mathbf{E}_k, \tilde{\mathbf{E}}_k), c_k, \mathbf{O}_{k+1} \rangle$  in  $\mathcal{R}$ ;
15    Generate additional transition  $GT$  based on (5.4.10), store  $GT$  in  $\mathcal{R}$ ;
16    Sample a random mini-batch of  $\mathcal{N}$  transitions from  $\mathcal{R}$ . Use gradient descent to
        update  $\theta^Q$  by minimizing the loss function  $TD$  in (5.4.8);
17    Use gradient descent to update  $\theta^\mu$  and  $\theta^{\tilde{\mu}}$  by minimizing the loss function  $L$ 
        in (5.4.9).
18  end
19 end

```

5.5 Numerical Results

We present numerical results of the knowledge-based HDDPG for transmission power allocation of the remote estimation system under a two hop relay network, based on the simulation parameters summarized in Table 5.1.

The key hyper-parameters of the neural networks' training are summarized in Table 5.2:

Our simulation compares the performance of knowledge-based HDDPG against the following benchmarks:

1. Heuristic benchmark with random power allocation and random scheduling: Rd + Rd.
2. Heuristic benchmark with random power allocation and round robin scheduling: Rd + Rb.

Table 5.1: Summary of simulation parameters for knowledge-based HDDPG

Number of sensors, N	6
Number of channels, M	3
Process state transition matrix, $\mathbf{A}_1, \dots, \mathbf{A}_N$	Randomly generated*
Channel states, $\mathbb{G}$	$\{10^{-4}, 10^{-3.5}, \dots, 10^{-0.5}\}$
Channel state transition matrix, $\mathbf{M}_1, \dots, \mathbf{M}_N$ and $\widetilde{\mathbf{M}}$	Randomly generated
Codeword length, L	200
Transmission rate, R	2
Background white noise, σ^2	0.01
Sensors' power budget, E^s	50
Relay's power budget, E^r	300
Attacker's power budget, E^a	50

*Spectral radius is randomly drawn in $(1, 1.3)$.

3. Heuristic benchmark with random power allocation and DQN scheduling: Rd + DQN.
4. DRL benchmark by Deep Deterministic Policy Gradient: DDPG.
5. Advanced DRL benchmark by Hybrid Deep Deterministic Policy Gradient: HDDPG.

Figure 5.3 presents the performance of the knowledge-based HDDPG algorithm compared to various benchmarks in 10 different random system setting, including the influence of an intelligent DQN-based DoS attacker. The results demonstrate that the knowledge-based HDDPG significantly outperforms all other benchmarks, consistently achieving the lowest average testing error. The problem involves a complex hybrid action determination process that requires both discrete and continuous actions. This complexity renders the performance of DDPG and DQN suboptimal. Among the benchmarks, the other DRL methods shows generally better performance compared with the heuristic benchmarks, but a significantly worse performance stability than the knowledge-based HDDPG. In average, know-based HDDPG improves the average estimation error by 84 percent compared with the best benchmark. This improvement highlights the effectiveness of the knowledge-based algorithm in addressing the power allocation problem, even in the presence of attackers, and enhances the scalability of the solution.

Table 5.2: Summary of neural network parameters of knowledge-based HDDPG

Number of training episodes	1000
Number of stage one pre-training episodes	100
Number of stage two pre-training episodes	100
Number of hidden layers of all neural networks	2
Activation function of all neural networks	Relu
Hidden size of $h_{(1)}, \dots, h_{(N)}, \tilde{h}$	1024
Hidden size of $h', \tilde{h}'$	256
Hidden size of Q	1024

5.6 Chapter Conclusions

In this study, we addressed the wireless resource allocation problem in a two-hop relay system, which is susceptible to intelligent DoS attacks. We proposed a DRL agent that incorporates domain knowledge to enhance learning efficiency and optimize the joint resource allocation for both the sensor-relay and relay-server segments. Our proposed architecture demonstrated superior performance compared to state-of-the-art DRL benchmarks. Specifically, the knowledge-based HDDPG algorithm achieved the lowest average testing error, significantly outperforming other benchmarks. The results showed that the knowledge-based HDDPG improved performance by 60 percent over the standard HDDPG, highlighting its effectiveness in handling the power allocation problem in the presence of attackers. The findings underscore the potential of DRL-based approaches in solving complex resource allocation problems in CPSs. By enhancing system robustness and overall performance, our approach paves the way for the development of more resilient and efficient CPSs. Future work will focus on further refining the algorithm and exploring its applicability to other challenging scenarios in wireless communication.

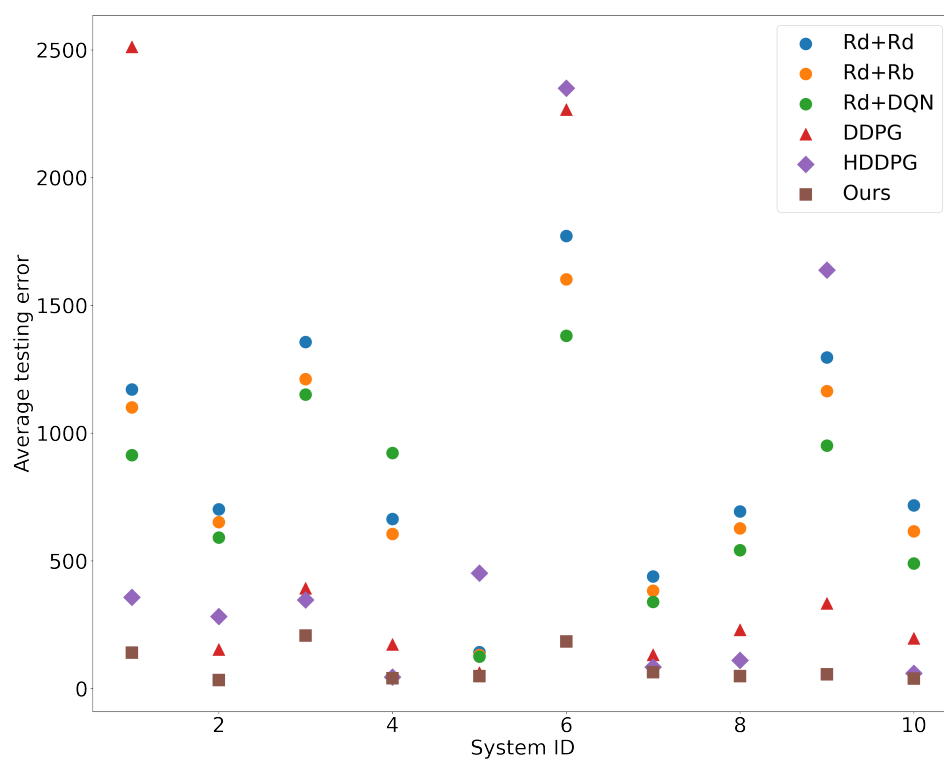


Figure 5.3: Performance comparison of the algorithms

Chapter 6

Conclusions and Future Work

6.1 Chapter Introduction

This chapter concludes the thesis and propose some directions for future research.

6.2 Summary of Results

This thesis has addressed critical challenges in wireless resource allocation for remote state estimation in CPSs under the threat of DoS attacks. We have proposed innovative solutions leveraging DRL to enhance system resilience and performance.

Firstly, we addressed the problem of joint channel and power allocation in a wireless networked CPS under the presence of data channel DoS attacks. Our specialized DRL framework for hybrid action space, HDDPG, demonstrated significant improvements in minimizing state estimation error and maximizing energy efficiency. Additionally, we designed a deep learning-based attack policy change detector to further enhance the system's defense mechanisms. This detector allows the system to adapt dynamically and promptly to changes in attack policies, ensuring robust performance while reducing the computational burden associated with continuous training.

Secondly, we introduced a CDC resource allocation approach to mitigate the impact of DoS attacks on control channels. In scenarios where sensors may not always receive the server's control commands due to control channel DoS attacks, local decision-making becomes crucial. The collaboration between distributed and centralized DRL requires a stable training design. Our proposed CDC-DRL algorithm outperformed existing DRL benchmarks, demonstrating its effectiveness in defending against control channel DoS attacks.

Lastly, we explored transmission power allocation in a two-hop relay system for remote state estimation under data channel DoS attacks. We proposed a DRL agent that incorporates domain knowledge to optimize resource allocation for both sensor-relay and relay-server segments and mitigate the attacker's effect. This approach significantly enhanced system robustness and overall performance, showcasing the potential of DRL-based solutions in both resource allocation and defending complex CPS environments.

6.3 Future Work

Future research can build on the findings of this thesis in several directions:

- **Scalability and Adaptivity:** Investigate the scalability and adaptivity of the proposed DRL algorithms in larger and more complex CPS environments. Transfer learning or meta learning methods could be combined with the DRL techniques and provide solutions to this type of the problems.
- **Security and Privacy:** Extend the research to address security and privacy concerns in CPSs. This may explore a centralized learning method without data sharing, and implementing federated learning framework.
- **Robustness to More Advanced Attacks:** Investigate the proposed or develop new DRL frameworks against novel and sophisticated attack strategies. This involves continuously updating the threat models and enhancing the defensive mechanisms accordingly.

Bibliography

- [1] A. Chattopadhyay, A. Prakash, and M. Shafique, “Secure cyber-physical systems: Current trends, tools and open research problems,” in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, 2017, pp. 1104–1109.
- [2] J. Shi, J. Wan, H. Yan, and H. Suo, “A survey of cyber-physical systems,” in *2011 international conference on wireless communications and signal processing (WCSP)*. IEEE, 2011, pp. 1–6.
- [3] E. A. Lee, “Cyber physical systems: Design challenges,” in *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, 2008, pp. 363–369.
- [4] P. Aarnio, V. Vyatkin, and D. Hästbacka, “Context modeling with situation rules for industrial maintenance,” in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2016, pp. 1–9.
- [5] O. Harcuba and P. Vrba, “Ontologies for flexible production systems,” in *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*. IEEE, 2015, pp. 1–8.
- [6] N. Petersen, L. Halilaj, I. Grangel-González, S. Lohmann, C. Lange, and S. Auer, “Realizing an rdf-based information model for a manufacturing company—a case study,” in *The Semantic Web–ISWC 2017: 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II 16*. Springer, 2017, pp. 350–366.
- [7] C. Hildebrandt, A. Köcher, C. Küstner, C.-M. López-Enríquez, A. W. Müller, B. Caesar, C. S. Gundlach, and A. Fay, “Ontology building for cyber–physical systems: Application in the manufacturing domain,” *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 3, pp. 1266–1282, 2020.
- [8] N. Dey, A. S. Ashour, F. Shi, S. J. Fong, and J. M. R. Tavares, “Medical cyber-physical systems: A survey,” *Journal of medical systems*, vol. 42, pp. 1–13, 2018.
- [9] R. Bishop, “Intelligent vehicle r&d: A review and contrast of programs worldwide and emerging trends,” *Annals of Telecommunications*, vol. 3, no. 60, pp. 228–263, 2005.
- [10] T. Zhang, Y. Zou, X. Zhang, N. Guo, and W. Wang, “Data-driven based cruise control of connected and automated vehicles under cyber-physical system framework,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 10, pp. 6307–6319, 2021.
- [11] R. Du, P. Santi, M. Xiao, A. V. Vasilakos, and C. Fischione, “The sensible city: A survey on the deployment and management for smart city monitoring,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1533–1560, 2019.
- [12] H. Sandberg, S. Amin, and K. H. Johansson, “Cyberphysical security in networked control systems: An introduction to the issue,” *IEEE Control Systems Magazine*, vol. 35, no. 1, pp. 20–23, 2015.

-
- [13] Y. Mo, T. H.-J. Kim, K. Brancik, D. Dickinson, H. Lee, A. Perrig, and B. Sinopoli, "Cyber-physical security of a smart grid infrastructure," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 195–209, 2012.
 - [14] Y. Won, B. Yu, J. Park, I.-H. Park, H. Jeong, J. Baik, K. Kang, I. Lee, S. H. Son, K.-J. Park, and Y. Eun, "An attack-resilient cps architecture for hierarchical control: A case study on train control systems," *Computer*, vol. 51, no. 11, pp. 46–55, 2018.
 - [15] A. Teixeira, D. Pérez, H. Sandberg, and K. H. Johansson, "Attack models and scenarios for networked control systems," in *Proceedings of the 1st International Conference on High Confidence Networked Systems*, ser. HiCoNS '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 55–64. [Online]. Available: <https://doi.org/10.1145/2185505.2185515>
 - [16] Y. Mo and B. Sinopoli, "Secure control against replay attacks," in *2009 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2009, pp. 911–918.
 - [17] G. Park, C. Lee, H. Shim, Y. Eun, and K. H. Johansson, "Stealthy adversaries against uncertain cyber-physical systems: Threat of robust zero-dynamics attack," *IEEE Transactions on Automatic Control*, vol. 64, no. 12, pp. 4907–4919, 2019.
 - [18] Y. Guan and X. Ge, "Distributed attack detection and secure estimation of networked cyber-physical systems against false data injection attacks and jamming attacks," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 1, pp. 48–59, 2017.
 - [19] T. Li, B. Chen, L. Yu, and W.-A. Zhang, "Active security control approach against dos attacks in cyber-physical systems," *IEEE Transactions on Automatic Control*, vol. 66, no. 9, pp. 4303–4310, 2020.
 - [20] S. Kim, K.-J. Park, and C. Lu, "A survey on network security for cyber-physical systems: From threats to resilient design," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 3, pp. 1534–1573, 2022.
 - [21] Z. Guo, J. Wang, and L. Shi, "Optimal denial-of-service attack on feedback channel against acknowledgment-based sensor power schedule for remote estimation," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 5997–6002.
 - [22] J. Zhou, J. Shang, and T. Chen, "On information fusion in optimal linear fdi attacks against remote state estimation," *IEEE Transactions on Control of Network Systems*, vol. 10, no. 4, pp. 2085–2096, 2023.
 - [23] H. Guo, J. Sun, Z.-H. Pang, and G.-P. Liu, "Event-based optimal stealthy false data-injection attacks against remote state estimation systems," *IEEE Transactions on Cybernetics*, vol. 53, no. 10, pp. 6714–6724, 2023.
 - [24] W. Xu, Z. Wang, L. Hu, and J. Kurths, "State estimation under joint false data injection attacks: Dealing with constraints and insecurity," *IEEE Transactions on Automatic Control*, vol. 67, no. 12, pp. 6745–6753, 2022.
 - [25] H. Zhang, P. Cheng, L. Shi, and J. Chen, "Optimal dos attack policy against remote state estimation," in *52nd IEEE Conference on Decision and Control*, 2013, pp. 5444–5449.
 - [26] H. Zhang, Y. Qi, J. Wu, L. Fu, and L. He, "Dos attack energy management against remote state estimation," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 1, pp. 383–394, 2018.
 - [27] D. Ye, M. Mei, and J. Wei, "Dos attack allocation against remote state estimation over packet-dropping networks," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 3, pp. 1507–1511, 2022.
 - [28] K. Ding, X. Ren, D. E. Quevedo, S. Dey, and L. Shi, "Dos attacks on remote state estimation with asymmetric information," *IEEE Transactions on Control of Network Systems*, vol. 6, no. 2, pp. 653–666, 2019.

- [29] L. Gao, B. Chen, and L. Yu, "Fusion-based fdi attack detection in cyber-physical systems," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 8, pp. 1487–1491, 2020.
- [30] Z. Abdollahi Biron, S. Dey, and P. Pisu, "Real-time detection and estimation of denial of service attack in connected vehicle systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 12, pp. 3893–3902, 2018.
- [31] Y. Guan and X. Ge, "Distributed attack detection and secure estimation of networked cyber-physical systems against false data injection attacks and jamming attacks," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 1, pp. 48–59, 2018.
- [32] Y. Raghuvamsi and K. Teeparthi, "Detection and reconstruction of measurements against false data injection and dos attacks in distribution system state estimation: A deep learning approach," *Measurement*, vol. 210, p. 112565, 2023.
- [33] M. Hosseinzadeh, B. Sinopoli, and E. Garone, "Feasibility and detection of replay attack in networked constrained cyber-physical systems," in *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2019, pp. 712–717.
- [34] D. Zhao, B. Yang, Y. Li, and H. Zhang, "Replay attack detection for cyber-physical control systems: A dynamical delay estimation method," *IEEE Transactions on Industrial Electronics*, pp. 1–9, 2024.
- [35] A. Naha, A. Teixeira, A. Ahlén, and S. Dey, "Sequential detection of replay attacks," *IEEE Transactions on Automatic Control*, vol. 68, no. 3, pp. 1941–1948, 2023.
- [36] L. Cao, X. Jiang, Y. Zhao, S. Wang, D. You, and X. Xu, "A survey of network attacks on cyber-physical systems," *IEEE Access*, vol. 8, pp. 44 219–44 227, 2020.
- [37] M. Babaei, J. Shi, and S. Abdelwahed, "A survey on fault detection, isolation, and reconfiguration methods in electric ship power systems," *IEEE Access*, vol. 6, pp. 9430–9441, 2018.
- [38] M. Wang, H. Liu, K. Huang, Y. Mao, and H. Yu, "Remote robust state estimation for nonlinear cyber-physical systems under denial-of-service attacks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 71, no. 9, pp. 4189–4198, 2024.
- [39] M. Fauser and P. Zhang, "Resilience of cyber-physical systems to covert attacks by exploiting an improved encryption scheme," in *2020 59th IEEE Conference on Decision and Control (CDC)*, 2020, pp. 5489–5494.
- [40] J. Zhou, W. Ding, and W. Yang, "A secure encoding mechanism against deception attacks on multisensor remote state estimation," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 1959–1969, 2022.
- [41] M. Pajic, J. Weimer, N. Bezzo, P. Tabuada, O. Sokolsky, I. Lee, and G. J. Pappas, "Robustness of attack-resilient state estimators," in *2014 ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*, 2014, pp. 163–174.
- [42] M. Pajic, I. Lee, and G. J. Pappas, "Attack-resilient state estimation for noisy dynamical systems," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 1, pp. 82–92, 2017.
- [43] A. Ahmad, S. Ahmad, M. H. Rehmani, and N. U. Hassan, "A survey on radio resource allocation in cognitive radio sensor networks," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 888–917, 2015.
- [44] X. Gao, E. Akyol, and T. Başar, "Optimal sensor scheduling and remote estimation over an additive noise channel," in *2015 American Control Conference (ACC)*, 2015, pp. 2723–2728.

-
- [45] M. M. Vasconcelos and U. Mitra, "Data-driven sensor scheduling for remote estimation in wireless networks," *IEEE Transactions on Control of Network Systems*, vol. 8, no. 2, pp. 725–737, 2021.
 - [46] A. Forootani, R. Iervolino, M. Tipaldi, and S. Dey, "Transmission scheduling for multi-process multi-sensor remote estimation via approximate dynamic programming," *Automatica*, vol. 136, p. 110061, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0005109821005896>
 - [47] A. S. Leong, S. Dey, and D. E. Quevedo, "Transmission scheduling for remote state estimation and control with an energy harvesting sensor," *Automatica*, vol. 91, pp. 54–60, 2018. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0005109818300360>
 - [48] A. S. Leong, A. Ramaswamy, D. E. Quevedo, H. Karl, and L. Shi, "Deep reinforcement learning for wireless sensor scheduling in cyber-physical systems," *Automatica*, vol. 113, p. 108759, 2020.
 - [49] W. Liu, K. Huang, D. E. Quevedo, B. Vucetic, and Y. Li, "Deep reinforcement learning for wireless scheduling in distributed networked control," *submitted to Automatica*, 2021. [Online]. Available: <https://arxiv.org/pdf/2109.12562.pdf>
 - [50] J. Wei and D. Ye, "Multisensor scheduling for remote state estimation over a temporally correlated channel," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 1, pp. 800–808, 2023.
 - [51] D. Han, Y. Mo, J. Wu, S. Weerakkody, B. Sinopoli, and L. Shi, "Stochastic event-triggered sensor schedule for remote state estimation," *IEEE Transactions on Automatic Control*, vol. 60, no. 10, pp. 2661–2675, 2015.
 - [52] Y. Li, C. S. Chen, and W. S. Wong, "Power control for multi-sensor remote state estimation over interference channel," *Systems & Control Letters*, vol. 126, pp. 1–7, 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167691119300155>
 - [53] T. Farjam, F. Fardno, and T. Charalambous, "Power allocation of sensor transmission for remote estimation over an unknown gilbert-elliott channel," in *2020 European Control Conference (ECC)*, 2020, pp. 1461–1467.
 - [54] Y. Li, D. E. Quevedo, V. Lau, and L. Shi, "Multi-sensor transmission power scheduling for remote state estimation under sinr model," in *53rd IEEE Conference on Decision and Control*, 2014, pp. 1055–1060.
 - [55] M. Mei, D. Ye, and J. Wei, "Multisensor power allocation for remote estimation over semi-markov fading channels," *IEEE Transactions on Control of Network Systems*, vol. 11, no. 3, pp. 1300–1309, 2024.
 - [56] L. Peng, X. Cao, and C. Sun, "Optimal transmit power allocation for an energy-harvesting sensor in wireless cyber-physical systems," *IEEE Transactions on Cybernetics*, vol. 51, no. 2, pp. 779–788, 2021.
 - [57] G. Pang, W. Liu, Y. Li, and B. Vucetic, "Drl-based resource allocation in remote state estimation," *IEEE Transactions on Wireless Communications*, vol. 22, no. 7, pp. 4434–4448, 2023.
 - [58] X. Gao, E. Akyol, and T. Başar, "On remote estimation with communication scheduling and power allocation," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 5900–5905.
 - [59] —, "Joint optimization of communication scheduling and online power allocation in remote estimation," in *2016 50th Asilomar Conference on Signals, Systems and Computers*, 2016, pp. 714–718.
 - [60] L. Yang, H. Rao, M. Lin, Y. Xu, and P. Shi, "Optimal sensor scheduling for remote state estimation with limited bandwidth: a deep reinforcement learning approach," *Information Sciences*, vol. 588, pp. 279–292, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0020025521012652>

- [61] L. Yang, Y. Xu, Z. Huang, H. Rao, and D. E. Quevedo, "Learning optimal stochastic sensor scheduling for remote estimation with channel capacity constraint," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 3, pp. 2565–2573, 2023.
- [62] Z. Zhao, W. Liu, D. E. Quevedo, Y. Li, and B. Vucetic, "Deep learning for wireless-networked systems: A joint estimation-control-scheduling approach," *IEEE Internet of Things Journal*, vol. 11, no. 3, pp. 4535–4550, 2024.
- [63] Y. Cao, L. Zhang, and Y.-C. Liang, "Deep reinforcement learning for channel and power allocation in UAV-enabled IoT systems," *2019 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, 2019.
- [64] S. Wang, S. Bi, and Y.-J. A. Zhang, "Locational detection of the false data injection attack in a smart grid: A multilabel classification approach," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8218–8227, 2020.
- [65] P. Kukadiya, T. Jain, and N. Hubballi, "Reducing the impact of dos attack on static and dynamic se using a deep learning-based model," *IEEE Transactions on Industrial Informatics*, vol. 20, no. 10, pp. 11 644–11 654, 2024.
- [66] N. Costilla-Enriquez and Y. Weng, "Attack power system state estimation by implicitly learning the underlying models," *IEEE Transactions on Smart Grid*, vol. 14, no. 1, pp. 649–662, 2023.
- [67] E. Uysal, O. Kaya, A. Ephremides, J. Gross, M. Codreanu, P. Popovski, M. Assaad, G. Liva, A. Munari, B. Soret, T. Soleymani, and K. H. Johansson, "Semantic communications in networked systems: A data significance perspective," *IEEE Network*, vol. 36, no. 4, pp. 233–240, 2022.
- [68] A. Zappone, M. Di Renzo, and M. Debbah, "Wireless networks design in the era of deep learning: Model-based, ai-based, or both?" *IEEE Transactions on Communications*, vol. 67, no. 10, pp. 7331–7376, 2019.
- [69] L. Xie, S. Zou, Y. Xie, and V. V. Veeravalli, "Sequential (quickest) change detection: Classical results and new directions," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 2, pp. 494–514, 2021.
- [70] T. M. Mitchell and T. M. Mitchell, *Machine learning*. McGraw-hill New York, 1997, vol. 1, no. 9.
- [71] J. Jagannath, N. Polosky, A. Jagannath, F. Restuccia, and T. Melodia, "Machine learning for wireless communications in the internet of things: A comprehensive survey," *Ad Hoc Networks*, vol. 93, p. 101913, 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1570870519300812>
- [72] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, "A survey on distributed machine learning," *ACM Comput. Surv.*, vol. 53, no. 2, Mar. 2020. [Online]. Available: <https://doi.org/10.1145/3377454>
- [73] M. R. Baker and R. B. Patil, "Universal approximation theorem for interval neural networks," *Reliable Computing*, vol. 4, pp. 235–239, 1998.
- [74] M. Islam, G. Chen, and S. Jin, "An overview of neural network," *American Journal of Neural Networks and Applications*, vol. 5, no. 1, pp. 7–11, 2019.
- [75] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.
- [76] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv e-Print*, 2013. [Online]. Available: <https://arxiv.org/abs/1312.5602>
- [77] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv e-Print*, 2019.

-
- [78] Z. Ding, Y. Huang, H. Yuan, and H. Dong, *Introduction to Reinforcement Learning*. Singapore: Springer Singapore, 2020, pp. 47–123.
 - [79] D. T. Gillespie, “Exact numerical simulation of the ornstein-uhlenbeck process and its integral,” *Physical review E*, vol. 54, no. 2, p. 2084, 1996.
 - [80] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, “Deterministic policy gradient algorithms,” in *International conference on machine learning*. Pmlr, 2014, pp. 387–395.
 - [81] A. Alipour-Fanid, M. Dabaghchian, and K. Zeng, “Self-unaware adversarial multi-armed bandits with switching costs,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 6, pp. 2908–2922, 2023.
 - [82] A. Alipour-Fanid, M. Dabaghchian, R. Arora, and K. Zeng, “Multiuser scheduling in centralized cognitive radio networks: A multi-armed bandit approach,” *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 2, pp. 1074–1091, 2022.
 - [83] J. Xu, S. Gong, Y. Zou, W. Liu, K. Zeng, and D. Niyato, “Redundant sniffer deployment for multi-channel wireless network forensics with unreliable conditions,” *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 1, pp. 394–407, 2020.
 - [84] X. Guo, R. Singh, P. R. Kumar, and Z. Niu, “Optimal energy-efficient regular delivery of packets in cyber-physical systems,” in *2015 IEEE International Conference on Communications (ICC)*, 2015, pp. 3186–3191.
 - [85] K. Huang, W. Liu, M. Shirvanimoghaddam, Y. Li, and B. Vucetic, “Real-time remote estimation with hybrid ARQ in wireless networked control,” *IEEE Transactions on Wireless Communications*, vol. 19, no. 5, pp. 3490–3504, 2020.
 - [86] K. Huang, W. Liu, Y. Li, B. Vucetic, and A. Savkin, “Optimal downlink-uplink scheduling of wireless networked control for Industrial IoT,” *IEEE Internet Things Journal*, vol. 7, no. 3, pp. 1756–1772, Mar. 2020.
 - [87] K. Huang, W. Liu, Y. Li, A. Savkin, and B. Vucetic, “Wireless feedback control with variable packet length for industrial IoT,” *IEEE Wireless Communications Letters*, vol. 9, no. 9, pp. 1586–1590, 2020.
 - [88] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
 - [89] V. Mnih, K. Kavukcuoglu, *et al.*, and D. Hassabis, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, Feb 2015.
 - [90] Y. S. Nasir and D. Guo, “Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks,” *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, p. 2239–2250, Oct 2019.
 - [91] K. K. Nguyen, T. Q. Duong, N. A. Vien, N.-A. Le-Khac, and L. D. Nguyen, “Distributed deep deterministic policy gradient for power allocation control in d2d-based v2v communications,” *IEEE Access*, vol. 7, pp. 164 533–164 543, 2019.
 - [92] Y. Li, D. E. Quevedo, S. Dey, and L. Shi, “SINR-based DoS attack on remote state estimation: A game-theoretic approach,” *IEEE Transactions on Control of Network Systems*, vol. 4, no. 3, pp. 632–642, 2017.
 - [93] Y. Zhang, L. Du, and F. L. Lewis, “Stochastic DoS attack allocation against collaborative estimation in sensor networks,” *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 5, pp. 1225–1234, 2020.
 - [94] P. Dai, W. Yu, H. Wang, G. Wen, and Y. Lv, “Distributed reinforcement learning for cyber-physical system with multiple remote state estimation under DoS attacker,” *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 3212–3222, 2020.

- [95] A. Tartakovsky, B. Rozovskii, R. Blazek, and H. Kim, "A novel approach to detection of intrusions in computer networks via adaptive sequential and batch-sequential change-point detection methods," *IEEE Transactions on Signal Processing*, vol. 54, no. 9, pp. 3372–3382, 2006.
- [96] H. Wang, D. Zhang, and K. Shin, "Change-point monitoring for the detection of dos attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 4, pp. 193–208, 2004.
- [97] H. Zhang, Y. Qi, H. Zhou, J. Zhang, and J. Sun, "Testing and defending methods against dos attack in state estimation," *Asian Journal of Control*, vol. 19, no. 4, pp. 1295–1305, 2017. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/asjc.1441>
- [98] F. S. d. Lima Filho, F. A. Silveira, A. de Medeiros Brito Junior, G. Vargas-Solar, and L. F. Silveira, "Smart detection: an online approach for DoS/DDoS attack detection using machine learning," *Security and Communication Networks*, vol. 2019, 2019.
- [99] W. Liu, D. E. Quevedo, Y. Li, K. H. Johansson, and B. Vucetic, "Remote state estimation with smart sensors over Markov fading channels," *to appear in IEEE Trans. Autom. Control*, 2021.
- [100] P. Sadeghi, R. A. Kennedy, P. B. Rapajic, and R. Shams, "Finite-state markov modeling of fading channels - a survey of principles and applications," *IEEE Signal Processing Magazine*, vol. 25, no. 5, pp. 57–80, 2008.
- [101] Y. Polyanskiy, H. Poor, and S. Verdú, "Channel coding rate in the finite blocklength regime," *IRE Professional Group on Information Theory*, vol. 56, no. 5, pp. 2307–2359, May 2010.
- [102] W. Liu, G. Nair, Y. Li, D. Nesic, B. Vucetic, and H. V. Poor, "On the latency, rate, and reliability tradeoff in wireless networked control systems for IIoT," *IEEE Internet of Things Journal*, vol. 8, no. 2, pp. 723–733, 2021.
- [103] L. Schenato, "Optimal estimation in networked control systems subject to random delay and packet drop," *IEEE Transactions on Automatic Control*, vol. 53, no. 5, pp. 1311–1317, 2008.
- [104] S. Coleri, M. Ergen, A. Puri, and A. Bahai, "Channel estimation techniques based on pilot arrangement in ofdm systems," *IEEE Transactions on Broadcasting*, vol. 48, no. 3, pp. 223–229, 2002.
- [105] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, pp. 318–362, 1986.
- [106] K. Wang, W. Liu, and T. J. Lim, "Deep reinforcement learning for joint sensor scheduling and power allocation under dos attack," in *ICC 2022 - IEEE International Conference on Communications*, 2022, pp. 1968–1973.
- [107] M. Xu, L. Zhang, and S. Wang, "Position paper: Renovating edge servers with arm socs," *2022 IEEE/ACM 7th Symposium on Edge Computing (SEC)*, pp. 216–223, 2022.
- [108] D. Ding, Q.-L. Han, X. Ge, and J. Wang, "Secure state estimation and control of cyber-physical systems: A survey," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 176–190, 2021.
- [109] K. Wang, W. Liu, and T. Joon Lim, "Deep learning for radio resource allocation under dos attack," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 2, pp. 703–716, 2024.
- [110] A. Chan, X. Liu, G. Noubir, and B. Thapa, "Broadcast control channel jamming: Resilience and identification of traitors," in *2007 IEEE International Symposium on Information Theory*, 2007, pp. 2496–2500.
- [111] P. Tague, M. Li, and R. Poovendran, "Mitigation of control channel jamming under node capture attacks," *IEEE Transactions on Mobile Computing*, vol. 8, no. 9, pp. 1221–1234, 2009.

- [112] B. F. Lo and I. F. Akyildiz, "Multiagent jamming-resilient control channel game for cognitive radio ad hoc networks," in *2012 IEEE International Conference on Communications (ICC)*, 2012, pp. 1821–1826.
- [113] A. N. Barreto and R. D. Vieira, "A critical analysis of receiver diversity with multiple antennas with spatially coloured noise," in *2008 IEEE 9th Workshop on Signal Processing Advances in Wireless Communications*, 2008, pp. 471–475.
- [114] R. Lowe, Y. WU, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.
- [115] K.-D. Kim and P. R. Kumar, "Cyber-physical systems: A perspective at the centennial," *Proceedings of the IEEE*, vol. 100, no. Special Centennial Issue, pp. 1287–1308, 2012.
- [116] S. Karnouskos, "Cyber-physical systems in the smartgrid," in *2011 9th IEEE international conference on industrial informatics*. IEEE, 2011, pp. 20–23.
- [117] X. Guan, B. Yang, C. Chen, W. Dai, and Y. Wang, "A comprehensive overview of cyber-physical systems: from perspective of feedback system," *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 1, pp. 1–14, 2016.
- [118] Y. Liu, Y. Peng, B. Wang, S. Yao, and Z. Liu, "Review on cyber-physical systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 1, pp. 27–40, 2017.
- [119] D. Son, B. Krishnamachari, and J. Heidemann, "Experimental study of the effects of transmission power control and blacklisting in wireless sensor networks," in *Proceedings of the First IEEE Conference on Sensor and Adhoc Communication and Networks*. Santa Clara, California, USA: IEEE, Oct. 2004, pp. 289–298. [Online]. Available: <https://ant.isi.edu/%7Ejohnh/PAPERS/Son04a.html>
- [120] Y. Han, M. Cui, and S. Liu, "Optimal sensor and relay nodes power scheduling for remote state estimation with energy constraint," *Sensors*, vol. 20, no. 4, 2020. [Online]. Available: <https://www.mdpi.com/1424-8220/20/4/1073>
- [121] R. Gan, J. Shao, Y. Xiao, H. Zhang, and W. X. Zheng, "Optimizing attack schedules based on energy dispatch over two-hop relay networks," *IEEE Transactions on Automatic Control*, vol. 65, no. 9, pp. 3832–3846, 2020.
- [122] R. Gan, Y. Xiao, J. Shao, and J. Qin, "An analysis on optimal attack schedule based on channel hopping scheme in cyber-physical systems," *IEEE Transactions on Cybernetics*, vol. 51, no. 2, pp. 994–1003, 2021.