Received 21 March 2023; revised 5 October 2023 and 23 March 2024; accepted 8 May 2024.
Date of publication 20 May 2024; date of current version 4 June 2024.

The associate editor coordinating the review of this article and approving it for publication was K. Zeng.

Digital Object Identifier 10.1109/TMLCN.2024.3403513

Deep Learning for Radio Resource Allocation Under DoS Attack

KE WANG, WANCHUN LIU (Member, IEEE), AND TENG JOON LIM (Fellow, IEEE)

School of Electrical and Computer Engineering, The University of Sydney, Camperdown, NSW 2050, Australia

CORRESPONDING AUTHOR: K. WANG (ke.wang@sydney.edu.au)

ABSTRACT In this paper, we focus on the problem of remote state estimation in wireless networked cyber-physical systems (CPS). 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.

INDEX TERMS Deep reinforcement learning, cyber-physical systems, sensor scheduling, power allocation, DoS attack.

I. INTRODUCTION

CYBER-PHYSICAL systems (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 as 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 [2]. 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 [3]. Work in [4] 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 [5].

Conversely, decision-making problems that incorporate dynamic states are often formulated as the Markov decision process (MDP). MDP considers Markovian state-transition dynamics and can be solved theoretically by the classic policy and value iteration methods [6], [7], [8]. However, due to the "curse of dimensionality" [9], such methods cannot be applied to problems with high-dimensional state and action spaces, and thus do not scale with the number of devices. Deep reinforcement learning (DRL) using deep

© 2024 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License.
For more information, see https://creativecommons.org/licenses/by/4.0/

neural networks (DNNs) as function approximators has emerged as a powerful alternative technique for solving large-scale MDP problems. In particular, the Deep Q network (DQN) is widely adopted for solving MDPs with discrete action spaces [10], while the deep deterministic policy gradient (DDPG) method is explicitly designed to operate over continuous action spaces [11].

There are many existing works applying DRL to optimize and enhance communications and networking capabilities in CPS environments [12], [13], [14], [15], [16]. Leong et al. [12] has proposed a DQN-based solution to the dynamic scheduling problem under a simple Gilbert–Elliott packet drop model. The follow-up work [13] 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 [14], [15]. Nguyen et al. [16] 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 non-trivial 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 [17]. Much research effort has gone into the detection, mitigation and analysis of cyber-attacks, including Denial-of-Service (DoS) [18], [19] 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 [18]. A dynamic binary channel selection scheme has been proposed for a multi-sensor remote estimation system defending against DoS attacks [20]. 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 [21], [22], [23], [24]. Classic change point detection algorithms, such as the Cumulative Sum Procedure (CUSUM) [25], 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 [21], [22], [23]. To address these concerns, data-driven approaches have been proposed for DoS attack detection, which does not require explicit statistical knowledge [24].

It is notable that most works on attack detection stop at discovering the onset of an attack [21], [22], [23], [24], 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 problems have not been thoroughly studied in the literature.

In this paper, 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 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 observation, 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 approach aimed at maximizing channel disruption. The change in attack policy is organized as a change detection problem, where DRL-based attacker that can adapt its attack policy in response to the defending actions are involved.
2) *Unified DRL framework.* We have developed an advanced DRL framework that bridges the gap between discrete action space from DQN and continuous action spaces from 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 structure training procedure.
3) *Specialized DNN-based detector.* We have introduced a tailored DNN detector for discerning changes in the attack policy. This detector is adept at processing two time-correlated and Markovian sequences,

TABLE 1. Notation summary.

Notation		Description
		Sensing Model
$\mathbf{x}_{i,k}$		State of process i at time k
$\widehat{\mathbf{x}}^{s}_{i,k}$		Local state estimate
$\mathbf{P}^{s}_{i,k}$		Local estimation error co-variance
		Remote Estimation
$\widehat{\mathbf{x}}_{i,k}$		Remote state estimate
$\mathbf{P}_{i,k}$		Remote estimation error co-variance
$\rho_{m,k}$		SINR of channel m at time k
$\gamma_{m,k}$		Packet transmission outcome
Sensor	Attacker	
$E^{s}_{m,k}$	$E^{a}_{m,k}$	Allocated transmission power
$G^{s}_{m,i}$	G^{a}_{m}	Channel gain to the gateway
		DRL
Defender	Attacker	
π	π^{a}	Policy
A^{c}_{k}	n.a.	Discrete scheduling action at time k
A^{p}_{k}	A^{a}_{k}	Continuous power allocation action
O_{k}	O^{a}_{k}	Observation state of gateway defender
r_{k}	r^{a}_{k}	Per-step reward
		Detector
$\kappa_{k',k}$		Binary label for attacker policy change
$\widehat{\kappa}_{k',k}$		Predicted output of detector
O^{d}_{k}		Defender's state-action pair
$\mathbf{L}_{k'}$		Anchored sequence
$\mathbf{L}_{k}$		Sliding sequence
$\mathbf{O}^{d}_{g,h}$		Time series of observation under policy pair (π_g, π^a_h)
$\mathbf{L}^{f}_{k'}$		Anchored sequence in offline training
$\mathbf{L}^{f}_{k}$		Sliding sequence in offline training
$\kappa^{f}_{k',k}$		Label for offline training data $(\mathbf{L}^f_k, \mathbf{L}^f_{k'})$

where traditional change detection methods typically consider an i.i.d. scenario. It employs a parallel offline training framework to ensure 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 paper is organized as follows: Section II describes the remote estimation system under DoS attack on data channel. Section III formulates a sensor scheduling and channel 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 IV extends the defender's scale toward heuristic feedback channel DoS attack. Section V proposes a novel DNN-based detection algorithm to detect the change in the attack policy. Section VI numerically evaluates the performance of our designed defender and detector, compared with different benchmarks. Lastly, Section VII gives a conclusion of our work. Some of the key notations of the paper are summarized in Table 1.

II. SYSTEM MODEL

We focus on remote state estimation under a physical-layer DoS attack, as illustrated in Fig. 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.

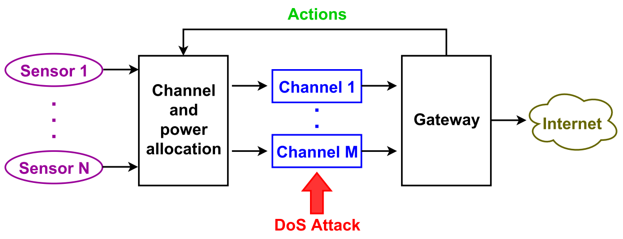

FIGURE 1. Remote state estimation with sensor scheduling jammed by DoS attack.

A. SENSING MODEL

We consider N independent, linear and discrete-time processes with the following state transition equation [26]

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

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 $\mathbf{W}_i$, i.i.d. 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

$$\mathbf{y}_{i,k} = \mathbf{C}_i \mathbf{x}_{i,k} + \nu_{i,k}, \quad i = 1, \dots, N, \tag{2}$$

where $\mathbf{C}_i$ is the measurement matrix, and $\nu_{i,k}$ is i.i.d. 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 [12], [19], and [26]. At each time step k, we use the classic Kalman filter update equations to compute the filtered state estimate $\widehat{\mathbf{x}}^s_{i,k}$,

$$\widehat{\mathbf{x}}^s_{i,k|k-1} = \mathbf{A}_i \widehat{\mathbf{x}}^s_{i,k-1|k-1}, \tag{3a}$$

$$\mathbf{P}^s_{i,k|k-1} = \mathbf{A}_i \mathbf{P}^s_{i,k-1|k-1} \mathbf{A}_i^\top + \mathbf{W}_i, \tag{3b}$$

$$\mathbf{K}_{i,k} = \mathbf{P}^s_{i,k|k-1} \mathbf{C}_i^\top (\mathbf{C}_i \mathbf{P}^s_{i,k|k-1} \mathbf{C}_i^\top + \mathbf{V}_i)^{-1}, \tag{3c}$$

$$\widehat{\mathbf{x}}^s_{i,k|k} = \widehat{\mathbf{x}}^s_{i,k|k-1} + \mathbf{K}_{i,k}(\mathbf{y}_{i,k} - \mathbf{C}_i \widehat{\mathbf{x}}^s_{i,k|k-1}), \tag{3d}$$

$$\mathbf{P}^s_{i,k|k} = (\mathbf{I}_i - \mathbf{K}_{i,k}\mathbf{C}_i)\mathbf{P}^s_{i,k|k-1}, \tag{3e}$$

where $\mathbf{I}_i$ denote the $n_i^y \times n_i^y$ identity matrix, $\widehat{\mathbf{x}}^s_{i,k|k-1}$ represent the prior state estimate with error covariance $\mathbf{P}^s_{i,k|k-1}$, $\widehat{\mathbf{x}}^s_{i,k|k}$ is the posterior state estimate with error covariance $\mathbf{P}^s_{i,k|k}$, 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, $\widehat{\mathbf{x}}^s_{i,k|k}$ will be abbreviated as $\widehat{\mathbf{x}}^s_{i,k}$, and $\mathbf{P}^s_{i,k|k}$ as $\mathbf{P}^s_{i,k} \triangleq \mathbb{E}[(\mathbf{x}_{i,k} - \widehat{\mathbf{x}}^s_{i,k})(\mathbf{x}_{i,k} - \widehat{\mathbf{x}}^s_{i,k})^\top]$. We assume that every $(\mathbf{A}_i, \mathbf{C}_i)$ is observable and every $(\mathbf{A}_i, \sqrt{\mathbf{W}_i})$ is controllable [26]. Under these assumptions, the local Kalman filter at every sensor is stable. Specifically, $\mathbf{P}^s_{i,k}$ converges to a finite constant matrix $\overline{\mathbf{P}_i}$ 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., $\mathbf{P}^s_{i,k} = \overline{\mathbf{P}_i}, \forall\ i = 1, \ldots, N, \forall\ k$, where $\overline{\mathbf{P}_i}$ is determined by $\mathbf{A}_i$, $\mathbf{C}_i$, $\mathbf{W}_i$ and $\mathbf{V}_i$ [12], [19], [26].

B. SCHEDULING AND DATA CHANNEL MODEL UNDER DoS ATTACK

Sensors need to transmit their local state estimates $\widehat{\mathbf{x}}^s_{i,k}$ 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. 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. We assume the transmission power of a sensor allocated on channel m at time k, denoted as $E^s_{m,k}$, is adjustable and subject to $0 \leq E^s_{m,k} \leq E^s_{\max}$. Lastly, we assume that the sum of power used on all the channels by the attacker is constrained as $\sum_{m=1}^M E^a_{m,k} \leq E^a_{\max}$. 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 $a_{m,k} \in \{0, 1, \ldots, N\}$, $m = 1, \ldots, M$, $k = 1, 2, \ldots$. 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

$$\rho_{m,k} = \frac{E^s_{m,k} G^s_{m,i,k}}{E^a_{m,k} G^a_{m,k} + \sigma^2}, \tag{4}$$

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

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 [26], [27]. Let $\mathbf{B}^s \triangleq \{b_1, \cdots, b_q\}$, $\mathbf{B}^a \triangleq \{b^a_1, \cdots, b^a_q\}$ denote the set of q-channel gain states from sensors and the attacker respectively. Then we have $G^s_{m,i,k} \in \mathbf{B}^s$, $G^a_{m,k} \in \mathbf{B}^a$. The channel transition probability of sensor i at channel m from state u to state v is

$$p^s_{u,v,m,i} \triangleq \mathbb{P}[G^s_{m,i,k+1} = b_v | G^s_{m,i,k} = b_u], \quad \forall u, v \in \mathbf{Q}, \tag{5}$$

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

$$p^a_{u,v,m} \triangleq \mathbb{P}[G^a_{m,k+1} = b^a_v | G^a_{m,k} = b^a_u], \quad \forall u, v \in \mathbf{Q}. \tag{6}$$

The corresponding channel transition probability matrices are $\mathbf{T}^s_{m,i}$ and $\mathbf{T}^a_m$ respectively.

We adopt finite-blocklength information theory [28] to approximate packet drop probabilities. The channel capacity $\mathsf{C}_{m,k}$ and the channel dispersion $\mathsf{v}_{m,k}$ of channel m at time k are defined as

$$\mathsf{C}_{m,k} = \log_2(1 + \rho_{m,k}), \tag{7}$$

and

$$\mathsf{v}_{m,k} = \rho_{m,k} \frac{2 + \rho_{m,k}}{(1 + \rho_{m,k})^2} (\log_2 e)^2, \tag{8}$$

where $e \approx 2.718$ is Euler's number. As in [29], each of the M channels is an AWGN channel with a finite-alphabet input. Let the transmission rate be R, and the Shannon capacity be $\mathsf{C}_{m,k}$, as defined in equation (7). The block length of a codeword is n, and $\widehat{x}^s_{i,k}$ is transmitted in a single codeword when sensor i is scheduled to transmit. Then the probability of successful packet transmission is approximated according to [29] as

$$\mu_{m,k} \approx 1 - \mathsf{Q}\left(\sqrt{\frac{\mathsf{n}}{\mathsf{v}_{m,k}}} (\mathsf{C}_{m,k} - \mathsf{R})\right), \tag{9}$$

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

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

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

C. REMOTE STATE ESTIMATION AT GATEWAY

As the gateway will not receive all sensors' local estimates in each time slot, it adopts an optimal [30] minimum mean-square error (MMSE) remote estimator for each process, given by

$$\widehat{\mathbf{x}}_{i,k} = \begin{cases} \widehat{\mathbf{x}}^s_{i,k} & \text{if } \exists m \text{ s.t. } a_{m,k} = i \text{ and } \gamma_{m,k} = 1 \\ \mathbf{A}_i \widehat{\mathbf{x}}_{i,k-1} & \text{otherwise.} \end{cases} \tag{10}$$

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

$$\begin{aligned} \mathbf{P}_{i,k} &\triangleq \mathbb{E}[(\mathbf{x}_{i,k} - \widehat{\mathbf{x}}_{i,k})(\mathbf{x}_{i,k} - \widehat{\mathbf{x}}_{i,k})^\top] \\ &= \begin{cases} \overline{\mathbf{P}_i} & \text{if } \exists m \text{ s.t. } a_{m,k} = i \text{ and } \gamma_{m,k} = 1 \\ h_i(\mathbf{P}_{i,k-1}) & \text{otherwise,} \end{cases} \end{aligned} \tag{11}$$

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

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

III. DEFENSE AND ATTACK POLICY DESIGN

A. GATEWAY DESIGN

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

$$O_k = (\mathrm{tr}P_{1,k-1}, \ldots, \mathrm{tr}P_{N,k-1}, G^s_{1,1,k}, \ldots, G^s_{M,N,k}), \quad (12)$$

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 [26], [31]. Let $A^c_k = (a_{1,k}, \ldots, a_{M,k})$ and $A^p_k = (E^s_{1,k}, \ldots, E^s_{M,k})$ denoting 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 $A_k \triangleq (A^c_k, A^p_k)$ 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:

$$\begin{aligned} &\max_{\pi(\cdot)} \mathbb{E}\left[\sum_{k=1}^{\infty} \delta^{k-1} r_k\right] \\ &\text{s.t. } A_k = \pi(O_k), \quad A_k \in \mathcal{A}^c \times \mathcal{A}^p, \ O_k \in \mathcal{O}, \end{aligned} \quad (13)$$

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

$$r_k \triangleq -\left(\sum_{i=1}^{N} \mathrm{tr}\mathbf{P}_{i,k} + \frac{1-\beta}{\beta}\sum_{m=1}^{M} E^s_{m,k}\right), \quad (14)$$

and $\beta \in (0, 1)$ is a design parameter reflecting the relative importance of energy consumption and estimation error.

B. ATTACKER DESIGN

The attacking power allocation action at time k is $A^a_k = (E^a_{1,k}, \ldots, E^a_{M,k})$, which satisfies the constraint $\sum_{m=1}^{M} E^a_{m,k} \leq E^a_{\max}$. 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

$$O^a_k = (\mathrm{tr}\mathbf{P}_{1,k-1}, \ldots, \mathrm{tr}\mathbf{P}_{N,k-1}, G^a_{1,k}, \ldots, G^a_{M,k}), \quad (15)$$

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

$$\begin{aligned} &\max_{\pi^a(\cdot)} \mathbb{E}\left[\sum_{k=1}^{\infty} \delta^{k-1} r^a_k\right] \\ &\text{s.t. } A^a_k = \pi^a(O^a_k), A^a_k \in \mathcal{A}^a, O^a_k \in \mathcal{O}^a, \end{aligned} \quad (16)$$

where r^a_k is the attacker's per-step reward function defined as

$$r^a_k = \sum_{i=1}^{N} \mathrm{tr}\mathbf{P}_{i,k}. \quad (17)$$

It can be proved that both problems (13) and (16) 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 [9], 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 (13) and (16), the state transition probability $\mathbb{P}[O_{k+1}|O_k, A_k]$ and $\mathbb{P}[O^a_{k+1}|O^a_k, A^a_k]$ 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.

C. 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 (13); then we develop an advanced DRL algorithm; finally, we present an algorithm for the DoS attacker to solve (16).

1) GATEWAY DEFENDER – BASIC ALGORITHMS

a: 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^c_k) and continuous (A^p_k) 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:

$$Q^*(O, A^c) = \mathbb{E}\left[r + \delta \max_{A^{c'} \in \mathcal{A}^c} Q^*(O', A^{c'})\right]. \quad (18)$$

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 $E^s_{\max}$ 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.

b: *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.

c: *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 (11), the estimation quality indicator $\mathrm{tr}\mathbf{P}_{i,k-1}$ 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 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.

2) GATEWAY DEFENDER – ADVANCED ALGORITHM

This motivates us to propose an advanced algorithm, hybrid deep deterministic policy gradient (HDDPG) as illustrated in Fig. 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 $A^c_{(i)}$ as the ith 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 $Q(O, A^c_{(1)}, A^p|\theta^Q), \ldots, Q(O, A^c_{(U)}, A^p|\theta^Q)$. 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

$$\left[r+\delta \max_{A^{c'}\in\mathcal{A}^c} Q(O', A^{c'}, \mu(O'|\theta^\mu)\big|\theta^Q) - Q(O, A^c, A^p|\theta^Q)\right]^2 \tag{19}$$

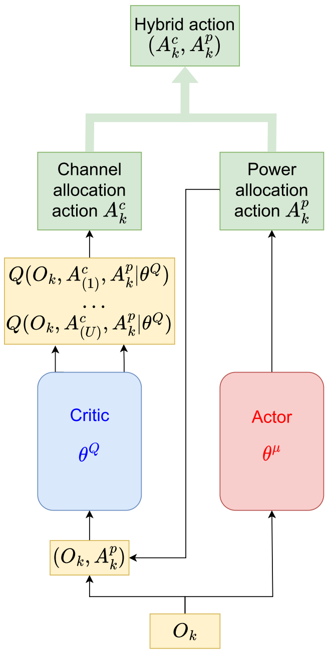
Hybrid action (A^c_k, A^p_k)
Channel allocation action A^c_k
Power allocation action A^p_k
$Q(O_k, A^c_{(1)}, A^p_k|\theta^Q)$
...
$Q(O_k, A^c_{(U)}, A^p_k|\theta^Q)$
Critic θ^Q
Actor θ^μ
(O_k, A^p_k)
O_k

FIGURE 2. The HDDPG architecture.

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

$$\max_{A^{c'}\in\mathcal{A}^c} Q(O, A^{c'}, \mu(O|\theta^\mu)\big|\theta^Q). \tag{20}$$

b: *Truncated state space*

We restrict the estimation quality state by a constant ϵ for HDDPG training, i.e. $\mathrm{tr}\mathbf{P}_{i,k} \leq \epsilon, i = 1, \ldots, N$, and the truncated state space is denoted as $\bar{\mathcal{O}}$. From the MDP theory [9], 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^p_k as $E^s_{m,k} = E^s_{\max}$ for all m; in the following l_2 episodes, we use the uniform random power allocation policy. In Section VI, 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 1.

Algorithm 1 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 $A_k^p = \begin{cases} (E_{\max}^s, \ldots, E_{\max}^s) & episode < l_1 \text{ (pre-training)} \\ \text{random } A_k^p & l_1 \le episode < l_2 \text{ (pre-training)} \\ \mu(O_k|\theta^\mu) + N_k^p & \text{others} \end{cases}$
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 (20):
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
```

D. DRL-BASED SOLUTION FOR ATTACK POLICY DESIGN

To solve (16) 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.

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

$$\lim_{K\to\infty} \frac{1}{K} \mathbb{E}\left[\sum_{k=1}^{K} A_k^b\right] \le \lambda, \tag{21}$$

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

$$\mathbb{P}[\zeta_{i,k} = x|A_k^b = 1] = \begin{cases} p_i^b & \text{if } x = 1 \\ 1 - p_i^b & \text{otherwise.} \end{cases} \tag{22}$$

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 power allocation.

V. 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 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. 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.

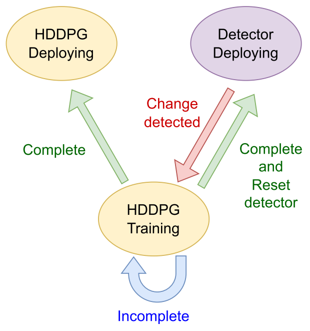

FIGURE 3. Integrated online attack policy change detection and defender policy re-training.

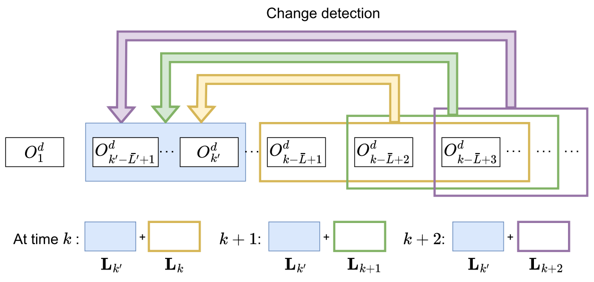

FIGURE 4. Online sequential change detection mechanism.

A. 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 $O_k^d \triangleq (O_k, A_k^p)$:

$$\begin{aligned} \mathbf{L}_{k'} &= (O^d_{k'-\bar{L}'+1}, \dots, O^d_{k'}), \\ \mathbf{L}_k &= (O^d_{k-\bar{L}+1}, \dots, O^d_k), \end{aligned} \tag{23}$$

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 $\widehat{\kappa}_{k',k} = d(\mathbf{L}_{k'}, \mathbf{L}_k) \in \{0, 1\}$ 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 III. In this sliding detection mechanism illustrated in Fig. 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 III, 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.

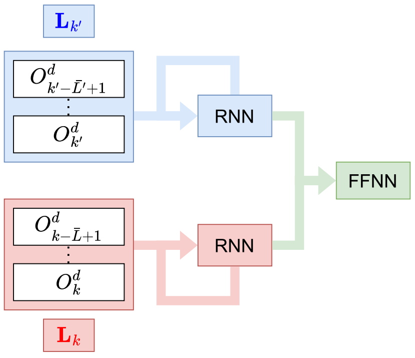

FIGURE 5. DNN-based attack policy change detector.

B. DNN-BASED DETECTOR DESIGN

1) 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 [32].

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. 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.

2) 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^a_{train} 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, $\mathbf{O}^d_{g,h}$ under specific defense-attack policy pair (π_g, π^a_h) over $\mathcal{K}$ time slots

$$\mathbf{O}^d_{g,h} = (O^d_1, O^d_2, \dots, O^d_{\mathcal{K}} | \pi = \pi_g, \pi^a = \pi^a_h), \tag{24}$$

where $g, h \in \{1, 2, \cdots, H\}$, $\pi_g \in \Pi_{train}$, and $\pi^a_h \in \Pi^a_{train}$. To sample the training data, we randomly select a sliding time series $\mathbf{O}^d_{g,h}$ and an anchored time series $\mathbf{O}^d_{g,h'}$ that share the same defense policy. We then extract two sequences of length

$\bar{L}$ and $\bar{L}'$ from $\mathbf{O}^d_{g,h}$ and $\mathbf{O}^d_{g,h'}$, respectively, with random time indices $k' < k$ as

$$\mathbf{L}^f_{k'} = (O^d_{k'-\bar{L}'+1}, \ldots, O^d_{k'} | \pi = \pi_g, \pi^a = \pi^a_{h'}).$$
$$\mathbf{L}^f_{k} = (O^d_{k-\bar{L}+1}, \ldots, O^d_{k} | \pi = \pi_g, \pi^a = \pi^a_{h}), \tag{25}$$

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

$$\kappa^f_{k',k} = \begin{cases} 1 & \text{if } h' \neq h \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

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

The process offline training algorithm is outlined in Algorithm 2. 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^a_{test} 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 $\widehat{\kappa}_{k',k}$. The median filter is a well-established nonlinear signal processing technique to eliminate spurious spike errors from the real-time detection output [33]. The filtered detection prediction, denoted as $\tilde{\kappa}_{k',k}$, is computed as the median of $\widehat{\kappa}_{k',k_f}$ 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 VI.

VI. EXPERIMENT RESULTS

A. DYNAMIC RESOURCE ALLOCATION UNDER DATA CHANNEL DoS ATTACK

1) 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: $E^a_{m,k} = \frac{E^a_{\max}}{M}$ for all m.
2) Greedy channel gain attack policy:
$$E^a_{m,k} = \begin{cases} E^a_{\max} & m = \operatorname{argmax}_{m'} G^a_{m',k} \\ 0 & \text{otherwise} \end{cases}$$
3) Random attack policy: $E^a_{m,k}$ is randomly allocated.
4) Round robin attack policy: allocate $E^a_{\max}$ 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 $\mathbf{A}_i, \mathbf{C}_i, \mathbf{W}_i, \mathbf{V}_i, i = 1, \ldots, N$ 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. The channel gain transition probability matrices $\mathbf{T}^s_{m,i}$ and $\mathbf{T}^a_m$ 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 $\mathsf{n} = 128$ and the data rate as $\mathsf{R} = 0.5$. We set $E^s_{\max} = E^a_{\max} = 1$, 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 2. 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 [12], 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.

According to our simulation, under all four different attack policies and all ten different environments, our HDDPG

Algorithm 2 Data Generation and Training Process of the Detector

```
1  Collect the series of observation $\mathbf{O}^d_{g,h}, \forall g, h \in \{1, \cdots, H\}$;
2  Randomly initialize detector $d(\cdot, \cdot|\theta^d)$ with parameter $\theta^d$, which has the structure illustrated in Fig. 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 $\{1, \cdots, H\}$;
6          Randomly sample $k'$ and $k$ subject to $k' < k \leq \mathcal{K}$;
7          Collect $\mathbf{L}^f_{k'}$ from $\mathbf{O}^d_{g,h'}$ and $\mathbf{L}^f_k$ from $\mathbf{O}^d_{g,h}$ based on (25);
8          Calculate the label of $(\mathbf{L}^f_{k'}, \mathbf{L}^f_k)$ as $y_i = \kappa^f_{k',k}$ based on (26);
9          Calculate the detector output $\hat{y}_i = d(\mathbf{L}^f_{k'}, \mathbf{L}^f_k|\theta^d)$;
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
```

TABLE 2. Simulation result of average reward (r_k) under fixed attack policy.

	Equal attack policy			Greedy channel gain attack policy			Random attack policy			Round robin attack policy		
	HDDPG	BM1	BM2	HDDPG	BM1	BM2	HDDPG	BM1	BM2	HDDPG	BM1	BM2
1	-31.51	-70.51	-52.06	-20.57	-90.03	-69.28	-30.90	-70.87	-46.03	-19.41	-72.03	-45.26
2	-34.71	-77.20	-55.46	-25.80	-93.75	-85.69	-33.29	-77.41	-51.05	-25.15	-77.62	-50.05
3	-29.13	-73.51	-69.91	-24.15	-117.01	-113.43	-30.26	-73.85	-52.85	-22.46	-76.05	-51.14
4	-38.30	-69.98	-59.15	-56.88	-85.77	-79.23	-43.57	-71.10	-51.67	-31.57	-73.05	-50.05
5	-30.40	-72.19	-58.90	-71.23	-98.74	-93.14	-32.82	-72.44	-48.58	-22.48	-73.70	-47.74
6	-32.33	-70.55	-50.59	-35.87	-94.65	-101.86	-33.17	-70.81	-45.36	-22.20	-71.37	-44.69
7	-42.30	-77.73	-69.53	-50.40	-108.23	-103.11	-44.24	-78.14	-55.89	-33.32	-79.21	-53.33
8	-44.80	-79.33	-63.59	-39.89	-85.54	-66.18	-49.29	-80.06	-58.18	-35.06	-81.07	-56.05
9	-37.83	-76.98	-55.69	-36.51	-82.68	-60.94	-38.62	-77.28	-52.41	-27.78	-77.87	-51.82
10	-34.75	-69.39	-54.35	-37.25	-98.84	-100.96	-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.

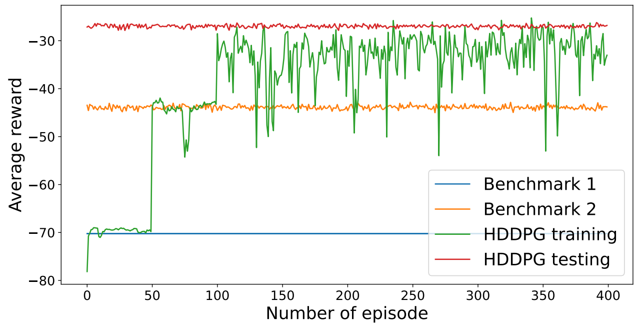

FIGURE 6. Training process and performance of HDDPG.

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.

An example of the HDDPG training and the performance comparison to the benchmark algorithms are shown in Fig. 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. 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 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.

2) 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

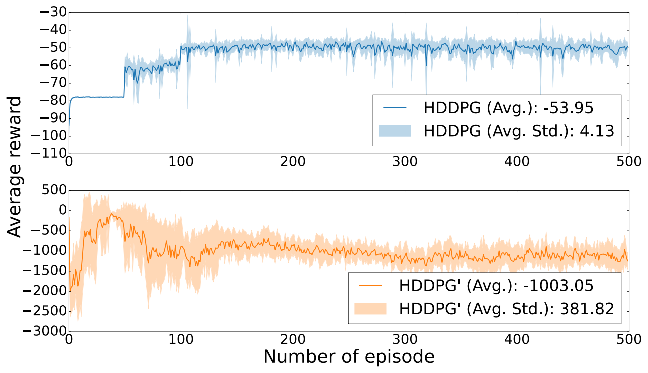

FIGURE 7. Training process and performance of HDDPG: effect of truncated state space and pre-training.

TABLE 3. Joint testing result after defender's and attacker's training.

	1st defender training		1st attacker training		2nd defender training		2nd attacker training	
	Avg. r_k	Avg. r_k^a	Avg. r_k	Avg. r_k^a	Avg. r_k	Avg. r_k^a	Avg. r_k	Avg. 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

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, retraining is triggered. The results of testing after two training phases are presented in Table 3. Our findings demonstrate that an intelligent attacker can significantly damage the network, while an intelligent DRL defender can adapt quickly to each new attack and restore the communication system, unlike a non-intelligent defender that relies on a fixed policy.

3) 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 $E^s_{\max}$ and $E^a_{\max}$). The results are presented in Figures 8 and 9, where the system parameters for the same sensors and channels in different setups are identical and the attacker's power constraint is fixed at $E^a_{\max} = 1$.

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 $E^s_{\max}$ relative to $E^a_{\max}$ 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

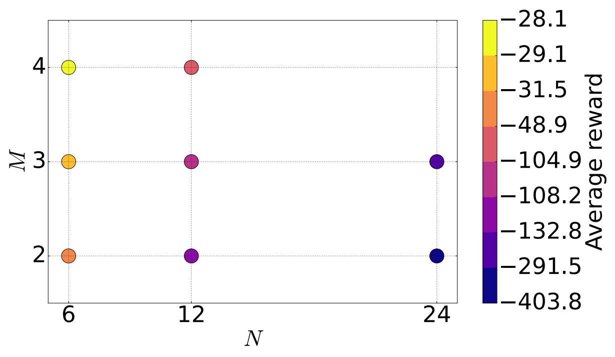

FIGURE 8. Testing result of scalability: *N* and *M*.

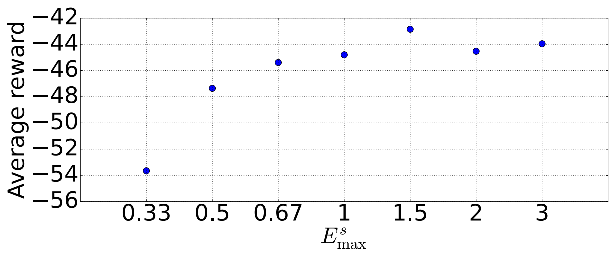

FIGURE 9. Testing result of scalability: $E^s_{\max}$ and $E^a_{\max}$.

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 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. 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 (13) effectively.

B. 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 IV, 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 4, sequential attacks are more likely to induce a

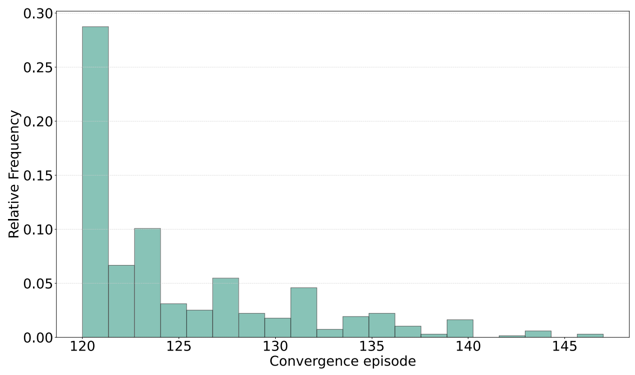

FIGURE 10. Convergence distribution.

TABLE 4. 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

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 4, 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.

C. DATA CHANNEL DoS ATTACK POLICY CHANGE DETECTOR

1) OFFLINE TRAINING AND TESTING

The proposed DNN-based detector is trained using Algorithm 2. 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^a_{train} consists of all the fixed policies in Section VI-A and the RL-based policies after the 1st and 2nd round of defender training in the example of Table 3, while the testing set Π^a_{test} contains RL-based attack policies after 3 and 4 rounds of defender training, extending to the example of Table 3. The end of frame $\mathcal{K}$ is set as 10000.

TABLE 5. 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

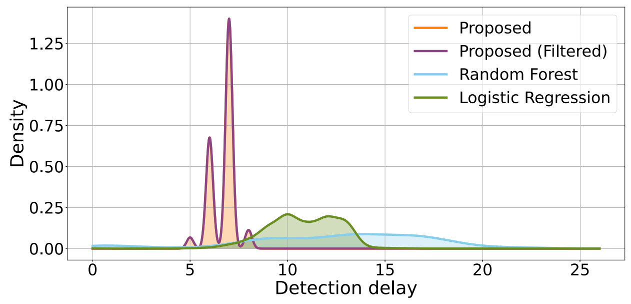

FIGURE 11. Estimated PDF of online policy change detection delay.

In Table 5, 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 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.

2) ONLINE TESTING

In Fig. 11, we present the estimated detection delay using kernel density estimation. The detection delay D is defined as

$$D \triangleq \min\{k - k^c \mid \widehat{\kappa}_k = 1, k \geq k^c\} \qquad (27)$$

The attack policy π^a transitions from π^a_3 to π^a_4 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^a_3 and π^a_4 refer to the DRL-based attack policy after the 3rd and the 4th defender training rounds, extended from the examples in Table 3. 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 5), 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.

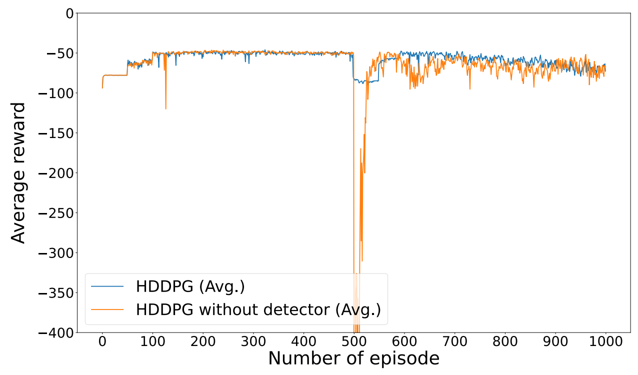

FIGURE 12. HDDPG training with and without the proposed policy change detector.

Fig. 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 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. 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 [34], the training time is about 1.5 minutes.

VII. CONCLUSION

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 attack to wireless networked control scenarios, with potential applications in autonomous driving, connected car platooning, and drone swarming.

ACKNOWLEDGMENT

An earlier version of this paper was presented in part at the 2022 IEEE International Conference on Communications (ICC) [DOI: 10.1109/ICC45855.2022.9838888].

REFERENCES

[1] K. Wang, W. Liu, and T. J. Lim, "Deep reinforcement learning for joint sensor scheduling and power allocation under DoS attack," in *Proc. IEEE Int. Conf. Commun.*, May 2022, pp. 1968–1973.
[2] A. Alipour-Fanid, M. Dabaghchian, and K. Zeng, "Self-unaware adversarial multi-armed bandits with switching costs," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 6, pp. 2908–2922, Jun. 2023.
[3] A. Alipour-Fanid, M. Dabaghchian, R. Arora, and K. Zeng, "Multiuser scheduling in centralized cognitive radio networks: A multi-armed bandit approach," *IEEE Trans. Cogn. Commun. Netw.*, vol. 8, no. 2, pp. 1074–1091, Jun. 2022.
[4] 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 Trans. Cogn. Commun. Netw.*, vol. 6, no. 1, pp. 394–407, Mar. 2020.
[5] X. Guo, R. Singh, P. R. Kumar, and Z. Niu, "Optimal energy-efficient regular delivery of packets in cyber-physical systems," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2015, pp. 3186–3191.
[6] K. Huang, W. Liu, M. Shirvanimoghaddam, Y. Li, and B. Vucetic, "Real-time remote estimation with hybrid ARQ in wireless networked control," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3490–3504, May 2020.
[7] 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 J.*, vol. 7, no. 3, pp. 1756–1772, Mar. 2020.
[8] K. Huang, W. Liu, Y. Li, A. Savkin, and B. Vucetic, "Wireless feedback control with variable packet length for industrial IoT," *IEEE Wireless Commun. Lett.*, vol. 9, no. 9, pp. 1586–1590, Sep. 2020.
[9] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[10] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
[11] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.
[12] 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, Mar. 2020, Art. no. 108759.
[13] W. Liu, K. Huang, D. E. Quevedo, B. Vucetic, and Y. Li, "Deep reinforcement learning for wireless scheduling in distributed networked control," 2021, *arXiv:2109.12562*.
[14] Y. S. Nasir and D. Guo, "Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2239–2250, Oct. 2019.
[15] Y. Cao, L. Zhang, and Y.-C. Liang, "Deep reinforcement learning for channel and power allocation in UAV-enabled IoT systems," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–6.
[16] 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. 164533–164543, 2019.
[17] H. Sandberg, S. Amin, and K. H. Johansson, "Cyberphysical security in networked control systems: An introduction to the issue," *IEEE Control Syst. Mag.*, vol. 35, no. 1, pp. 20–23, Feb. 2015.

[18] Y. Li, D. E. Quevedo, S. Dey, and L. Shi, "SINR-based DoS attack on remote state estimation: A game-theoretic approach," *IEEE Trans. Control Netw. Syst.*, vol. 4, no. 3, pp. 632–642, Sep. 2017.

[19] Y. Zhang, L. Du, and F. L. Lewis, "Stochastic DoS attack allocation against collaborative estimation in sensor networks," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 5, pp. 1225–1234, Sep. 2020.

[20] 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 Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 3212–3222, Oct. 2020.

[21] A. G. Tartakovsky, B. L. Rozovskii, R. B. 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 Trans. Signal Process.*, vol. 54, no. 9, pp. 3372–3382, Sep. 2006.

[22] H. Wang, D. Zhang, and K. G. Shin, "Change-point monitoring for the detection of DoS attacks," *IEEE Trans. Depend. Secure Comput.*, vol. 1, no. 4, pp. 193–208, Oct. 2004.

[23] H. Zhang, Y. Qi, H. Zhou, J. Zhang, and J. Sun, "Testing and defending methods against DOS attack in state estimation," *Asian J. Control*, vol. 19, no. 4, pp. 1295–1305, Jul. 2017, doi: 10.1002/asjc.1441.

[24] F. S. D. L. Filho, F. A. F. 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," *Secur. Commun. Netw.*, vol. 2019, pp. 1–15, Oct. 2019.

[25] L. Xie, S. Zou, Y. Xie, and V. V. Veeravalli, "Sequential (quickest) change detection: Classical results and new directions," *IEEE J. Sel. Areas Inf. Theory*, vol. 2, no. 2, pp. 494–514, Jun. 2021.

[26] W. Liu, D. E. Quevedo, Y. Li, K. H. Johansson, and B. Vucetic, "Remote state estimation with smart sensors over Markov fading channels," *IEEE Trans. Autom. Control*, vol. 67, no. 6, pp. 2743–2757, Jun. 2022.

[27] P. Sadeghi, R. Kennedy, P. Rapajic, and R. Shams, "Finite-state Markov modeling of fading channels—A survey of principles and applications," *IEEE Signal Process. Mag.*, vol. 25, no. 5, pp. 57–80, Sep. 2008.

[28] Y. Polyanskiy, H. V. Poor, and S. Verdu, "Channel coding rate in the finite blocklength regime," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2307–2359, May 2010.

[29] 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 Things J.*, vol. 8, no. 2, pp. 723–733, Jan. 2021.

[30] L. Schenato, "Optimal estimation in networked control systems subject to random delay and packet drop," *IEEE Trans. Autom. Control*, vol. 53, no. 5, pp. 1311–1317, Jun. 2008.

[31] S. Coleri, M. Ergen, A. Puri, and A. Bahai, "Channel estimation techniques based on pilot arrangement in OFDM systems," *IEEE Trans. Broadcast.*, vol. 48, no. 3, pp. 223–229, Sep. 2002.

[32] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, vol. 1. Cambridge, MA, USA: MIT Press, 1986, pp. 318–362.

[33] S. Wang, J. Cao, and P. S. Yu, "Deep learning for spatio-temporal data mining: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 8, pp. 3681–3700, Aug. 2022.

[34] M. Xu, L. Zhang, and S. Wang, "Position paper: Renovating edge servers with ARM SoCs," in *Proc. IEEE/ACM 7th Symp. Edge Comput. (SEC)*, Dec. 2022, pp. 216–223.

KE WANG received the Bachelor of Science degree in chemical technology from The Hong Kong Polytechnic University, Hong Kong, the Bachelor of Science degree in chemistry from Sun Yat-sen University, Guangzhou, China, in 2015, and the master's degree in data science from The University of Sydney, Sydney, Australia, in 2021, where he is currently pursuing the Ph.D. degree in engineering with the School of Electrical and Computer Engineering. His Ph.D. research topic focuses on resource allocation and active defense mechanisms development within networked cyber-physical systems by machine learning algorithm design and application.

WANCHUN LIU (Member, IEEE) received the B.S. and M.S.E. degrees in electronics and information engineering from Beihang University, Beijing, China, in 2010 and 2014, respectively, and the Ph.D. degree from The Australian National University, Canberra, Australia, in 2017. Following her graduation, she joined The University of Sydney as a Research Fellow. Her research interests include communications and networked control theory, wireless control for industrial Internet of Things (IIoT), and wireless human–machine collaborations for industry 5.0. She was the Co-Chair of Australian Communication Theory Workshop in 2020 and 2021. She has received several awards, including Australian Research Council's Discovery Early Career Researcher Award in 2023, the Dean's Award for Outstanding Research of an Early Career Researcher in 2022, and Chinese Government Award for Outstanding Students Abroad in 2017. Additionally, she was named one of N2Women: Rising Stars in Computer Networking and Communications in 2023, one of the ten global rising stars of the year.

TENG JOON LIM (Fellow, IEEE) received the B.Eng. degree (Hons.) in electrical engineering from the National University of Singapore (NUS) in 1992 and the Ph.D. degree from the University of Cambridge in 1996.

From September 1995 to November 2000, he was a Researcher with the Centre for Wireless Communications, Singapore, one of the predecessors of the Institute for Infocomm Research (I2R). From December 2000 to May 2011, he was an Assistant Professor, an Associate Professor, and then a Professor with the University of Toronto's Edward S. Rogers Sr. Department of Electrical and Computer Engineering. From June 2011 to January 2020, he was a Professor with the Electrical and Computer Engineering Department, NUS, where he was the Deputy Head from July 2014 to August 2015. From September 2015 to December 2019, he was the Vice-Dean (Graduate Programs) with the NUS Faculty of Engineering. Since January 2020, he has been the Deputy Dean and the Associate Dean (Education) with the Faculty of Engineering, The University of Sydney. His research interests include wireless communications and machine learning, including cyber-security in the Internet of Things, heterogeneous networks, cooperative transmission, energy-optimized communication networks, multi-carrier modulation, MIMO, and stochastic geometry for wireless networks. He has published widely in these areas.

Dr. Lim served as the TPC Co-Chair for IEEE Globecom 2017, chaired the Singapore Chapter of the IEEE Communications Society in 2017 and 2018, and was a Distinguished Lecturer of the IEEE Vehicular Technology Society in 2019 and 2020. He was a Co-Winner of the IEEE Communications Society's 2020 Heinrich Hertz Award for Best Communication Letter. From September 2013 to September 2018, he was an Area Editor of IEEE Transactions on Wireless Communications. Previously, he served as an Associate Editor for IEEE Transactions on Wireless Communications. He has also served as an Associate Editor for IEEE Wireless Communications Letters, *Transactions on Emerging Telecommunications Technologies* (ETT) (Wiley), IEEE Signal Processing Letters, and IEEE Transactions on Vehicular Technology.