DRAGON: Deep Reinforcement Learning for Autonomous Grid Operation and Attack Detection

Matthew Landen
Georgia Institute of Technology
United States of America

Keywhan Chung
Lawrence Livermore National Laboratory
United States of America

Moses Ike
Georgia Institute of Technology
United States of America

Sarah Mackey
Lawrence Livermore National Laboratory
United States of America

Jean-Paul Watson
Lawrence Livermore National Laboratory
United States of America

Wenke Lee
Georgia Institute of Technology
United States of America

ABSTRACT

As power grids have evolved, IT has become integral to maintaining reliable power. While providing operators improved situational awareness and the ability to rapidly respond to dynamic situations, IT concurrently increases the cyberattack threat surface – as recent grid attacks such as Blackenergy and Crashoverride illustrate. To defend against such attacks, modern power grids require a system that can maintain reliable power during attacks and detect when these attacks occur to allow for a timely response. To help address limitations of prior work, we propose DRAGON– deep reinforcement learning for autonomous grid operation and attack detection, which (i) autonomously learns how to maintain reliable power operations while (ii) simultaneously detecting cyberattacks. We implement DRAGON and evaluate its effectiveness by simulating different attack scenarios on the IEEE 14 bus power transmission system model. Our experimental results show that DRAGON can maintain safe grid operations 225.5% longer than a state-of-the-art autonomous grid operator. Furthermore, on average, our detection method reports a true positive rate of 92.9% and a false positive rate of 11.4%, while also reducing the false negative rate by 63.1% compared to a recent attack detection method.

CCS CONCEPTS

• **Security and privacy** → **Intrusion detection systems**; **Domain-specific security and privacy architectures**.

KEYWORDS

power grid reliability, cyberattack detection, deep reinforcement leearning

ACM Reference Format:
Matthew Landen, Keywhan Chung, Moses Ike, Sarah Mackey, Jean-Paul Watson, and Wenke Lee. 2022. DRAGON: Deep Reinforcement Learning for Autonomous Grid Operation and Attack Detection. In *Annual Computer Security Applications Conference (ACSAC '22), December 5–9, 2022, Austin, TX, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3564625.3567969

ACSAC '22, December 5–9, 2022, Austin, TX, USA
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9759-9/22/12.
https://doi.org/10.1145/3564625.3567969

1 INTRODUCTION

Smart grids, which integrate IT networks for enhanced monitoring and control, are becoming increasingly targeted by cyber attackers. Cyberattacks on the power grid can cause large-scale blackouts, which may lead to critical safety and economic impacts, as illustrated by recent attacks such as Blackenergy and Crashoverride [40, 45, 75]. During such attacks, attackers executed malicious commands to reduce grid reliability and poisoned grid measurements to delay effective response and remain undetected. Although many security solutions exist for monitoring IT networks (the most common initial point of compromise used during power grid attacks) [12, 20, 21, 63], previous, successful grid attacks demonstrate that adversaries intent on disrupting power grids can bypass these measures and inflict damage. Thus, we propose DRAGON: an advisory tool that monitors the grid to protect the grid by detecting power grid attacks and proposing response actions to operators.

To minimize the damage caused by cyberattacks on the power grid, DRAGON must maximize the number of attacks it detects (true positive rate (TPR)) while also minimizing the number of false alarms it generates (false positive rate (FPR)). The former minimizes the damage to the grid while the latter minimizes alert fatigue. Also, DRAGON should minimize the amount of time that an attack remains undetected in a grid (attack dwell time). This also reduces the damage caused by cyberattacks. Detecting attacks is critical, but an operator must also continue to operate the grid reliably during and after an attack. Unfortunately, most industrial control systems (ICS) detection research has not considered this crucial aspect [1, 8, 38, 64, 78, 87]. To aid human operators, DRAGON explicitly considers reliability as its second objective. To accomplish this objective, DRAGON contains a reliability component that acts as an advisor. This tool helps operators by proposing control actions that keep the grid in a reliable state for as long as possible. To assess this component's effectiveness, we measure the amount of time that the grid serves all of its loads. By achieving these two core objectives, DRAGON can help meet customers' power demands in an evolving grid, targeted by an increasing number of cyberattacks.

Despite numerous attack detection systems proposed by grid researchers, there remains significant gaps in the detection of realistic grid attacks. Previous work that detects ICS attacks using statistical [23, 28, 30, 36, 50, 68, 78] or physical models [2, 82] cannot detect sophisticated attackers who leverage well-known physical models to blend into normal operations more accurately than IT systems [2]. In fact, in our evaluation, we show that a state-of-the-art model-based detector [26] reports a 96.8% false negative rate when detecting attackers who intentionally hide their actions. In contrast, DRAGON is extremely accurate with a false negative rate of 7.1%. Also, prior specification-based detection [22, 28, 56], which detects deviations from normal state machine transitions, suffers from state explosion in real-world power grids. For example, the small grid [34] used in our evaluation has at least $1.59 * 10^{24}$ states. To build a detector that can **generalize** to this state space, DRAGON leverages machine learning (ML). Unfortunately, to train supervised classifiers to distinguish between normal and attack events, we would need a properly labeled training dataset, which is challenging to build in adversarial environments where attackers can distort information [61]. In contrast, DRAGON leverages deep reinforcement learning (DRL) methods to produce agents that act optimally with limited knowledge of the environment [61, 81] and adapt to unseen events, which is particularly useful when detecting sophisticated, dynamic cyberattacks. Background on power operations and reinforcement learning is given in Appendices A and B respectively.

To maintain reliable power, past works have proposed manual, planning and, expert systems based approaches. However, these approaches either cannot scale to realistic sized grids, take too much time to make decisions, or do not consider realistic threat models. Historically, power grids were relatively static, enabling operators to rely on domain knowledge to address problems as they arose [41]. However, this is no longer the case, because modern power grids are extremely large and highly dynamic. Even with the size of our small grid, it is infeasible for operators to anticipate all possible scenarios where a response is needed in advance. Additionally, current planning tools are insufficient as they cannot reason about cascading component failures, which cyberattacks can initiate [35]. To design novel grid response tools that can reason about cyberattacks, prior work investigated expert systems. While potentially able to address an operator's **time constraints**, it cannot define expert heuristics that are effective in all diverse scenarios encountered by the modern smart grid. For example, Marot et el. [55]'s expert system only achieved a 75% success rate when trying to find actions that remove all line overflows, clearly struggling to **generalize**. Without a generalized solution, the system would likely require reconfiguration or retraining, creating a large burden for operators. To satisfy both the time and generalizability requirements, we investigate how ML algorithms can complement expert systems to learn effective grid operation actions from limited training data and generalize their learned knowledge to unseen scenarios. Supervised learning would not be effective here because it is infeasible to create a training dataset given the limitations of current grid operators and expert systems.

We found that reinforcement learning (RL) can enhance prior expert system-based approaches to achieve more reliable power. Applying RL consists of two main steps – problem setup and training. To set up a RL problem, a practitioner defines a set of features of an environment, different actions that affect the environment, and a reward function that measures the quality of different actions in the context of the overall goal of the problem. Then, an algorithm is applied to train an agent to select actions that achieve the goal. Stemming from the success of this research area, prominent energy research and development organizations, Réseau de Transport d'Électricité (RTE) [70] and Electric Power Research Institute (EPRI) [18], initiated an open competition series called *Learning To Run a Power Network* (L2RPN) that applies DRL to automate operator response in power grid operations [53, 54]. Although this competition produced operators that could maintain reliable power against command injection attacks, they did not consider false data injection attacks (FDIAs) that were seen in real-world attacks [76].

In this work, we propose DRAGON: deep reinforcement learning for autonomous grid operation and attack detection, an assistant to grid operators that detects cyberattacks and proposes control actions to maintain reliable grid operation. Our system defends against adversaries that execute FDIAs and command injection attacks simultaneously, observed in real-world cyberattacks [45]. To achieve both DRAGON's detection and reliability objectives, we design two customized DRL problems and apply DRL algorithms to train an operator and a detector agent. The ***Operator Agent*** proposes control actions that modify the grid's topology to keep the grid reliable for as long as possible, which is critical for deployment as demand for electricity must be met even during attack/remediation. The ***Detector Agent*** notifies the operator of suspected cyberattacks while also minimizing false positives. These alerts are necessary for removing attackers from power grids and reducing the damage that attackers cause. As part of our problem designs, we create feature spaces, actions, and rewards tailored to reliable grid operation and timely attack detection.

Experimental results, using scenarios provided by RTE, show that our operator agent can maintain reliable power for 225.5% longer than a previous, top-performing, autonomous operator. Additionally, our system detects attacks with an average TPR of 92.9% and FPR of 11.4% when exposed to attackers with varying levels of sophistication. We analyzed the false positive alerts raised by our system to understand these situations and ultimately guide operators in triage of potential false positives when DRAGON is deployed in practice.

Our primary contributions in this paper are:

- We address the challenge of reliable power grid operation and cyberattack detection by designing two DRL problems that algorithms can solve. Our design involved careful selection of grid features and development of specialized action sets. The key to DRAGON's success is the design of unique reward models that were informed by real cyber threats to the grid. These novel problem definitions enable DRAGON to maintain grid reliability while simultaneously detecting cyberattacks.
- Through an extensive evaluation with a diverse set of grid scenarios, we show that DRAGON can detect cyberattacks 15.1 minutes earlier than prior detection methods while reducing the FNR by 63.1%. Further, when deployed against sophisticated attackers, DRAGON can maintain reliable power for 225.5% longer than a state-of-the-art, autonomous operator.

2 MOTIVATING EXAMPLE AND THREAT MODEL

In this section, we discuss a real-world power grid attack to demonstrate DRAGON's capabilities and then introduce our threat model.

2.1 Motivating Example: Metcalf Attack

To showcase DRAGON's capabilities, we illustrate how the system would function if deployed in the control center of the grid targeted during the Metcalf attack. During this event, a transformer's cooling fins were physically damaged, shutting down the transformer [65, 86]. Such outages can cause significant damage to critical infrastructure and endanger citizens. After receiving an alarm, the operator responded to the situation by rerouting power around the site. We adapt this incident into a cyberattack case study in §8.4 using known malware capabilities (e.g., HAVEX [17] and HARVEY [25]) and the MITRE's ATT&CK knowledge base [57].

If DRAGON was deployed in practice as part of the control room's monitoring software, it would assist the operator in detecting and responding to this attack. First, if the envisioned cyber attackers injected false data (to disguise its presence), DRAGON's detector agent would detect that there is an attack and alert the human operator that a response action may be required. Concurrently, DRAGON's operator agent would propose optimal response actions to reconfigure the grid's topology to maintain the grid's stability while the disabled transformers are repaired.

2.2 Threat Model

We consider an attacker who compromises a programmable logic controller (PLC) in a substation that controls a line's connection status. Once the attacker establishes access to the PLC, the attacker can block any operator command affecting this line. We assume that the attacker maintains control of the line for the entirety of the attack, which can last for multiple hours. Each attack is launched at a randomly chosen time within 24 hours. We decided to limit the attacker to compromise one PLC as we aim to detect stealthy attacks and as an attacker compromises more devices, they leave behind more traces that can reveal their presence. Instead of increasing the attack impact by compromising multiple devices, our attacker leverages cascading failures, which have been shown to result in large-scale blackouts, requiring an effective response [6, 11, 42]. This poses a challenge for both the operator and detector agents. Our attacker can also modify the grid measurements seen by the operator. Such FDIAs are well studied in research [1, 47, 84] and observed in real-world events [76]. With this attack vector, attackers can hide evidence of their attacks and delay an effective response. We assume an extremely strong FDIA threat model where the attacker (1) has full information about the current grid measurements, including the topology, (2) can modify any measurement in the operator's observation of the grid, and (3) has access to the same monitoring and planning software used by the grid operator to forecast the outcome of a control action. Even though it is unlikely for attackers to have access to all three types of information, we show in our evaluation that even in the extreme case, DRAGON can still detect attacks. Finally, we assume that the attacker cannot compromise the supervisory control and data acquisition (SCADA) human machine interface (HMI) workstation where the operator and detector agents run as these systems run Windows and use IT security solutions that could flag the attacker. Although these security solutions could protect the HMI, the PLCs in the grid would be left vulnerable, leading to the need for DRAGON.

Table 1: Power Grid Attack Detection Taxonomy. Describes prior work's threat models, detection features, whether they evaluated on real grid data or IEEE synthetic grid models, and which attack scenarios they investigated.

Threat model	Ref.	Features	Power grids evaluated		Attack scenarios		
			Real data	IEEE grids	# Targets	FDIA	Network
SCADA	[73]	Radio frequency emissions	X		1	X	
	[47]			9, 14, 118, 300		X	
	[91]	Expert rules					X
	[84]	SCADA		24, 118		X	
	[1]	SCADA		9, 14, 30		X	
μPMU	[38]	μPMU, crafted rules	X	34	1		
Grid load manipulation	[77]			WSCC 9	4 - 300 IoT devices		
PLC	[25]	–		9	1		
	DRAGON	SCADA		14	1	X	

2.3 Limitations of Related Work

Table 1 compares ICS attack/defense work, focusing on the threat models adopted, the features used for detection, and their evaluation setup. For a complete discussion of related work, refer to §10. Compared to these works, DRAGON considers an adversary with a number of unique features. First, unlike other papers that analyze isolated attacks (where an attacker deploys an attack and the detector responds), DRAGON considers attack strategies that launch a sequence of multiple attacks. This is more realistic as, in reality, attackers can deploy a sequence of malicious actions to maximize impact. Furthermore, our threat model challenges DRAGON to respond to attacks earlier and prepare for imminent, future attacks. Second, DRAGON considers attackers who deploy combined attacks that damage grid components and poison sensor data. This poses additional challenges that are not addressed by prior tools such as An et al. [1], which consider sequential FDIAs. When the attacker poison the operator's grid view, the operator cannot directly see the attack. This can delay response and lead to more damage. Next, in contrast with prior work, DRAGON takes the two critical objectives into account, attack detection and reliable power. The papers in Table 1 only consider detection, which could lead to a destabilized grid during attack remediation. Lastly, while the majority of papers analyze attackers who compromise the SCADA HMI workstations, we consider attackers that infect a PLC device in substations. Given that both attacks have been shown to be feasible and that PLC based attacks have received less attention, we start to fill this gap with our work.

3 OVERVIEW OF DRAGON

DRAGON's core contribution is the design of customized RL problems that capture the objectives of reliable power and effective cyberattack detection respectively. DRAGON applies DRL training algorithms to these problem to produce high-performing operator and detector agents. In this section, we introduce how these agents

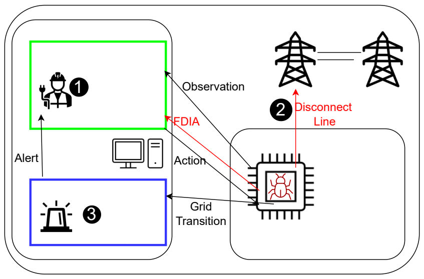

Figure 1: Overview of DRAGON's operation and detection problem definitions.

interact within a real power grid configuration and then discuss technical challenges that DRAGON has to address.

3.1 Overview

DRAGON has two main phases: *training* and *deployment*, where training occurs within a simulated grid environment and then the trained agents can be deployed in a control room HMI's digital twin of the grid [66]. Figure 1 depicts the sequence of interactions that the operator and detector agents have with the grid. ❶ First, the operator agent observes the current grid. Next, it proposes a control command to the human operator that modifies the grid's topology to ensure that (i) all loads receive sufficient power and (ii) lines' thermal limits are not violated. After the human operator accepts the proposed command, it is sent over the OT network to the PLC that controls the targeted grid component. ❷ After this, any cyberattacker that has compromised this PLC can block the incoming command and disconnect the compromised power line. If the operator's control command is not intercepted by an attacker, the PLC will adjust its actuators to apply the requested command to the physical grid. Then, the grid evolves until the HMI requests sensor data from the grid. Note, these measurements could be falsified by FDIAs. ❸ Lastly, the detector agent determines whether it believes an attack occurred. During training, both agents receive a reward that represents its action's quality. The observations, actions, and rewards are saved to be used during the training process by DRL algorithms, discussed in further detail in §6.

System Deployment in Practice. In this work, the training and deployment phases use state-of-the-art simulation software. Although using simulations allows us to evaluate our tool with different configurations, this evaluation setup has the risk of not being realistic, which we strive to mitigate. First, we use Grid2Op [15], developed by researchers at RTE, a top transmission service operation R&D division [16], which uses a well-known KLU solver [14] to run simulations. Next, we evaluate DRAGON on the IEEE 14 bus model, which represents a portion of the real American Electric Power System [34]. Lastly, the grid's load and generation schedules are populated with RTE's historical data [16]. This setup parallels our deployment scenario that uses a digital twin to mirror a physical grid. The only difference is our SCADA measurements are produced by simulations that replicate the physical reality whereas, with the digital twin, measurements come from the real grid. We believe that this setup represents reality as it is based on realistic grid topologies, load and generation schedules, and well-known power flow solvers. Note, this setup was used to evaluate research from prior top cybersecurity conferences [33, 77].

3.2 Challenges Addressed

There are technical challenges that arise when designing and implementing DRAGON. The first challenge is the state – action space combinatorics. There are at least $1.59 * 10^{24}$ states and 179 control commands in the small grid we evaluated. To make training tractable, DRAGON must represent what it learns more compactly than standard RL, which stores values for every state-action pair. DRAGON addresses this challenge by using DRL to learn behaviors that can generalize to multiple grid states, eliminating the need to learn about each state (action) independently.

The second challenge is handling the combined physical/FDIAs we consider. When an attacker poisons the operator's view of the grid after altering the grid's topology, the operator's response may be delayed because they do not know that the grid is in a dangerous state. To overcome this challenge, our operator uses DRL algorithms where the rewards, based on the true grid state, guide the agent (during training) towards learning to differentiate when the grid view is corrupted to change their strategy accordingly.

4 MAINTAINING RELIABLE POWER GRID OPERATION

When a cyberattack on a power grid occurs, it is critical that reliable power is maintained during the attack and remediation. In this section, we describe the RL problem we designed that captions this goal and the trained agent's action selection procedure.

4.1 Operation Problem Design

Our operation problem is a partially observable Markov decision process (POMDP) and includes threat-informed feature selection, action space definition, and a threat-informed reward function.

Power Grid Observation Features. During each timestep, the operator agent observes the current grid state through a feature vector. As some aspects of the grid state may be hidden from the operator (e.g., due to cyberattacks), the agent receives partial observations, which includes features related to generators and loads, power flow, and line capacities. The complete list of features can be found in Table 5 in Appendix D. Through experimentation, we found that the agent struggled to maintain reliable power when FDIAs were deployed because the agent must learn to propose control commands when evidence of an attack is removed. To improve this agent, we leverage a common capability of grid operations that predicts the outcome of control commands [88] to compute the difference between the expected and actual outcome and include this difference in the operator agent's observation.

Possible Control Actions. Our operator agent can propose two types of control actions to improve the grid's reliability: line reconnections and substation reconfigurations. Each line in the grid can be reconnected if it is de-energized. This action is useful when recovering from an attack by re-energizing lines after the attack ends. An operator agent can also perform "bus bar" switching actions on any of the substations [72]. Bus bar switching actions dictate which, of the two buses within a substation each line carries power through and can relieve line overflows. As the total number of potential actions grows exponentially with grid size, we apply the action reduction technique used by the second place team in the 2020 L2RPN competition [4] to make training more tractable. This technique samples a large set of scenarios and collects the actions that result in the minimum sum of the percent utilization of all the power lines. These actions are collected into the final action set. For the IEEE 14 grid, we found 79 substation actions.

Grid Transitions. Transitioning from one state to another involves a series of steps. First, the control command is applied and the next load and generation setpoints are set. Next, any attackers can begin or continue an attack. Finally, the power flow solver is used to advance the grid to the next timestep (5 simulated minutes). Similar to L2RPN, lines with flows greater than twice their thermal limits are disconnected, potentially resulting in cascading outages. The simulation terminates when either (i) a load or generator is disconnected from the rest of the grid, or (ii) the grid breaks into multiple, isolated sub-grids. Grid2Op [15] decided to stop the simulation at these points, a behavior that we adopt.

Operator Agent Rewards. To train our operator to maintain grid reliability, we define a custom reward function that communicates the value of taking each action. If our operator reaches a terminal state, it receives the minimum reward to encourage our operator to maintain reliable power as long as possible. Otherwise, there are three factors that contribute to the reward: **grid connectivity (GC), line capacities (PL), and generator re-dispatch (GD)**. The GC reward is equal to the number of paths from a generator to a load, averaged over all of the loads in the grid. Higher GC reward means the grid is more likely to withstand line disconnections because the loads can receive power from other generators when one path is unavailable. Second, the PL reward encourages the operator to minimize power line flows to avoid cascading failures. It is defined as the ratio of `power` to a line's `capacity`, summed over the set of lines, $\mathcal{L}$, (i.e., $\sum_{\forall\ l \in \mathcal{L}} \frac{power_l}{thermal\_limit_l}$). Lastly, the GD reward captures the cost of changing the generation setpoints.

The three rewards are combined in a linear combination: $r = c_{GC} r_{GC} + c_{PL} r_{PL} + c_{GD} r_{GD}$. The coefficients were assigned empirically by analyzing the relative magnitudes of rewards and weighting each type by importance. We weight r_{GC} the highest because lower grid connectivity leads directly to load loss and terminal states. We weight r_{PL} the second highest because, while overflowing lines can cause cascades and eventual blackouts, these effects are delayed and can be addressed after all loads are served. Finally, r_{GD} is weighted lowest because it is not critical for a high performing agent.

4.2 Operator Control Action Selection

The operator agent decides what action to propose using the function discussed in this subsection. This function (or policy) has three steps: *Reconnection policy* to keep as many power line connected as possible, *Boltzmann policy* to prioritize the most promising actions, and *simulation policy* to choose the best action. First, if there are lines that are disconnected and not under attack, the operator agent reconnects one of these lines. We use the reconnection policy because many of the high-ranking teams in the L2RPN competition adopted some version of automatically reconnecting lines. Although our evaluation results using this policy are encouraging, studies have found that in some cases, disconnecting a line can lead to a more stable state [24]. Although we did not identify such a circumstance in our evaluation, we will reevaluate this policy as we apply DRAGON to different scenarios.

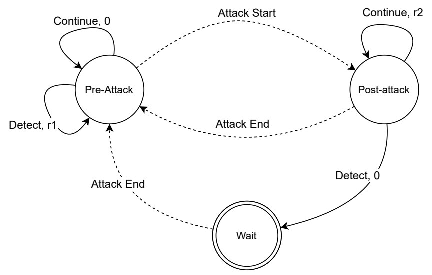

Figure 2: Attack Detection Transition and Reward Model (solid arrows represent detection agent actions and rewards while the dotted arrows are transitions caused by attacks).

If there are no lines that can be reconnected, the operator proposes a bus switching action. First, the agent samples a set of actions using the Boltzmann function [81] and the current action values. This function has a temperature parameter that controls how much the agent should explore unfamiliar actions. By annealing the temperature over the course of training, this policy allows the agent to improve, while avoiding low-valued actions, which is critical because some actions immediately destabilize the grid. Then, for each sampled action, the agent simulates its effects on the grid[1] and scores it. Each simulation returns the simulated next observation and a reward. If the simulated next observation is terminal, the action is eliminated. If not, the action is scored based on how many lines are above their capabilities and how many were disconnected due to prolonged overflow or cascading failures. These counts are combined with the function, $f(\rho_t) = -\#(\rho_t > 1) - 2\#(\rho_t > 1.5) - 3\#(cascading\_failures)$, where ρ_t is the ratio of the current power flowing through each line divided by the line's thermal limit at time t, $\#(\cdot)$ is a function that returns how many power lines satisfy the condition, and the coefficients were chosen to assign a higher penalty to actions that result in more line overflows or more lines that are disconnected by cascading failures. $f(\cdot)$ is combined with the immediate, simulated reward in the following: $f(\rho_t) + c_1 r_t$, where c_1 is a constant to allow prioritization of the overflow or the immediate reward, r_t. The policy ranks actions according to this quantity and returns the action with the maximum score.

5 DETECTING CYBERATTACKS

DRAGON's second objective is to detect cyberattacks. This section describes the RL problem we design to accomplish this objective.

5.1 Detection Problem Design

We design our attack detection problem as a POMDP, which consists of a hidden state space, observations, and actions. Furthermore, we design a custom reward function for our detector agent to detect attacks while minimizing the FPR.

Hidden State Space and Agent Action Set. For this problem, we design a custom state space that tracks when the grid is under attack using three states: `pre-attack`, `post-attack`, and `wait`. In a state, this agent has two possible actions: `detect` an attack or `continue` normal operation. To model the interactions between the detector agent and the hidden state, we present a state transition diagram

[1]As the operator does not have access to the true next loads and generation, the simulation uses forecasted values that are imprecise.

in Figure 2 where solid arrows represent transitions caused by the detector agent with their associated rewards and dotted arrows represent transitions caused by attackers. If the agent detects an attack, the system transitions to the `wait` state. This state is terminal because the agent detected the current attack. Revisiting the Metcalf attack, after DRAGON detects the attack on the transformer and the system transitions to the `wait` state, the operator would investigate the situation and ideally identify the cyber intrusion that caused the event and remove any attacker tools from the OT network, preventing further damage to the grid.

Detection Observation Features. We design the observations for this problem such that the agent receives sufficient information to accurately detect attacks. The observation has three components: the operator agent's previous and current observation and the control command deployed. All of these components are needed to detect power line disconnection and FDIAs. When an attacker disconnects a line and poisons the operator's observation, the current operator observation is likely to appear different from what was expected because the true grid measurements are replaced with fake values. By examining the three observation components, the agent can detect this inconsistency. We found that the agent had difficulty detecting attacks with this raw information. To highlight the important features, we process these components by computing the difference between the two observations and only reporting measurement differences related to power line statuses and the grid's topography. This feature selection process helps the agent focus on differences that line disconnection attacks cause.

Detector Agent Rewards. To train our detector agent, we design a reward function, based on domain knowledge. Ideally, our agent would choose the `detect` action at the first timestep of an attack for a FNR of 0 and `continue` otherwise for a FPR of 0. Our reward function guides the agent towards this policy. If the agent detects an attack when no attack is happening, this results in a false positive (FP) and the agent receives a penalty proportional to how different the poisoned observation is from the expected observation, scaled by a constant c_2: when attacks cause more grid disturbance, our agent should detect the attack with higher confidence. When a false negative (FN) occurs, a penalty is returned that increases in magnitude as the attack (that started at timestep τ) remains undetected, scaled by a constant c_3. This encourages the agent to minimize the attacker's dwell time and the damage caused by the attack. Otherwise, the agent receives a reward of 0. Given the detector action, a_t, real and simulated observations, o_{t+1}, and $\hat{o}_{t+1}$, and the mean differences between features of the simulated and real observations, $\bar{o}$, computed prior to training the detection agent, the rewards are assigned as follows:

$$r(a_t, o_{t+1}, \hat{o}_{t+1}, \tau) = \begin{cases} -c_2 \frac{||o_{t+1} - \hat{o}_{t+1}||}{\bar{o}} & \text{FP, } a_t = \texttt{detect} \\ -c_3 |t - \tau| & \text{FN, } a_t = \texttt{continue} \\ 0 & \text{otherwise.} \end{cases}$$

5.2 Detection Decision Selection

To decide whether to detect an attack or continue normal operation, our agent leverages a well-known RL policy, called the epsilon greedy policy [81]. This policy trades off exploration and exploitation by choosing the action with the maximum action value (i.e., exploitation) a certain percent of the time and choosing a random action (i.e., exploration) the rest of the time. The exploration percentage starts high to allow the agent to explore. As training progresses, this percentage decreases, allowing the agent to exploit what it has learned. We found this to be effective at detecting attacks.

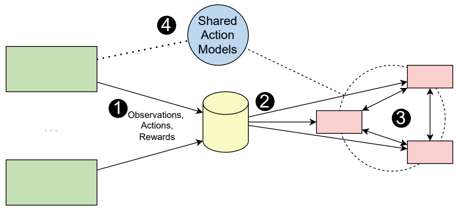

Figure 3: DRAGON's training process, which used distributed resources to accelerate training.

6 TRAINING PROCEDURE

After defining our operation and detection problems, we now discuss how DRAGON uses DRL algorithms to train agents to solve each problem. DRAGON's training framework leverages proposed distributed training algorithms [10, 58, 60, 79] to accelerate training.

Figure 3 shows DRAGON's training process. First, DRAGON spawns multiple `Actor` and `Trainer` processes. Each Actor contains an independent problem setup, shown in Figure 1. As the operator and detector agents interact with the grid, they generate observations, actions, and rewards, collectively termed experiences. Periodically, ❶ the agents send these experiences to replay memories. Simultaneously, each Trainer ❷ samples batches of experiences from memory and performs forward and backward passes over the neural network-based policies with the experiences to calculate gradients. Next, ❸ all Trainers perform synchronous `all_reduce` operations to compute averaged gradients. Finally, ❹ the Trainers update the networks' parameters using the averaged gradients and share the updated policies with the Actors.

7 EVALUATION SETUP

In this section, we describe DRAGON's implementation and introduce the metrics that we use to measure its performance. Finally, we briefly explain our training parameters.

Implementation. DRAGON leverages Grid2Op [15]: an open-source grid simulation and RL python package, which uses a steady-state, C++ KLU power solver with 5 minute timesteps. Grid2Op provides a RL framework that enables artificial agents to take actions on the grid and observe the consequences. Additionally, Grid2Op automates several common controls present in real power grids, including automatic generation control, voltage control, and over-current protection. By automating these controls, Grid2Op allows the operator agent to focus on recovering from attacks. Each agent's policy is represented by a Deep Recurrent Q Network (DRQN) [32], written using Pytorch [67]. This model uses a Long Short-Term Memory (LSTM) network to compute action value. We chose this model because it has been shown to achieve higher performance in partially observable environments compared to the Double DQN [85] used by past winners of the L2RPN competition [48].

Experimental Design. We evaluated DRAGON using the IEEE 14 bus transmission system, which contains 20 power lines, 14 substations, 11 loads, and 4 generators. This grid is commonly used in recent research papers [59, 71] and during L2RPN competitions [70] as a representative grid topology. We enhance their threat model by including FDIAs. We used 1,002 load and generation profiles, prepared by RTE, each one is a month long, which span different months and years. These profiles provide a diverse dataset to evaluate the generalizability of DRAGON.

For the grid operation experiments, we report the *operator reward* and *number of steps survived*, averaged over the profiles. A higher step count translates to customers receiving reliable power for longer. When evaluating DRAGON's detection performance, we measure the true positive rate (recall), precision, F1 score, dwell time, and false positive and false negative rates. The first three metrics are common for classification problems while the last two rates are important for security because they quantify the number of false alarms an analyst will have to investigate and the number of undetected attacks respectively. Finally, we report the dwell time of the attack as it is critical to detect attacks as soon as possible to reduce damage and maximize availability.

During evaluation, we employ cross-validation with three folds over the set of training profiles per best practices [93]. Before running cross-validation, we performed hyperparameter optimization, with a held out validation set to select the model architectures, learning rate schedules, gamma, and batch size. We used the ASHA [46] algorithm to find high performing network structures and then population based training [37] to optimize the other hyperparameters (See [44] for specific hyperparameter settings). Training runs for a pre-defined number of learning steps (i.e., synchronized updates to the weights of the agents' policy networks), set to balance sufficient evaluation performance and reasonable training times: 50,000 for the operator agent and 25,000 for the detector agent. In our evaluation, results are averaged over all folds.

Attack Strategies. We study DRAGON's performance against two different types of attack strategies proposed by RTE. First, the ***Weighed Random Attacker*** [62] selects a line to disconnect probabilistically, weighted by the line's current percent capacities. The attacker disconnects its targeted line and blocks any attempt by the operator to act on the line for 48 timesteps (4 hours). Then, the attacker releases control of the line and randomly selects a time within the next 24 hours to attack next. The second attacker type is the ***Geometric Attacker***, which uses the same settings as the weighted random, except for the attack length. This attacker deploys attacks of variable lengths, sampled from a geometric distribution. Finally, we restrict both attackers to target lines in a defined subset of the grid to replicate the likely reconnaissance efforts an attacker would use to identify and target the most critical lines. We explain the details of this reduction in Appendix C. When deploying FDIAs, the attackers generate false measurements in the same manner as the Harvey malware [25], by simulating the intended operator command and injecting measurements from the simulated outcome. An important concern when leveraging ML for security is the potential for mimicry attacks where an attacker intentionally appear benign to the detector. By adopting Harvey's FDIA technique, we expose DRAGON to such mimicry attacks so that our agent learns how to detect these sophisticated attacks as demonstrated in our experimental evaluation.

Table 2: The mean of the total episode steps and rewards with different attacker types. DRAGON outperforms the baseline, Oroas [4], in all scenarios.

Attacker		Operator	Steps	Reward
	None	Oroas	1632.17	586.28
		DRAGON	5032.44	3694.28
Weighted Random	No FDIA	Oroas	770.36	447.22
		DRAGON	4284.34	2973.51
	FDIA	Oroas	807.53	472.17
		DRAGON	1033.38	635.33
Geometric	No FDIA	Oroas	1153.62	726.56
		DRAGON	3045.50	2045.42
	FDIA	Oroas	1169.50	737.44
		DRAGON	1678.18	1059.54

8 EVALUATION

Our evaluation seeks to answer the following research questions:

RQ1: How effectively can DRAGON maintain reliable power when the grid is under cyberattack?
RQ2: Can DRAGON detect more cyberattacks with a smaller attack dwell time than state-of-the-art?
RQ3: How robust is DRAGON to different types of attackers?
RQ4: To what degree does the distributed training procedure improve the efficiency of training?

8.1 Maintaining Reliable Grid Operation

To answer **RQ1**, we trained and evaluated DRAGON's operator agent against no attacker and the two attack strategies introduced in Section 7, with and without FDIAs. As DRAGON's first core objective is to maintain a reliable grid for as long as possible, we report the number of steps survived and the total reward obtained during a scenario. Our baseline operator is the 2020 L2RPN second place solution, Oroas [4], which also used DRL and expert heuristics. We chose the second place as a baseline because the first place's training code was not released, preventing comparison. Instead, we trained the second place against all our attackers. Their DRL policy is warm-started with a supervised network trained to propose the control command that reduces the lines' percent capacity the most. Then, they employ Proximal Policy Optimization (PPO) to train the policy. We run PPO training until we observed the average reward plato (500 epochs).

We present the experimental results in Table 2. For every attacker type, DRAGON outperforms Oroas, maintaining reliable power for longer. On average, our system maintains a reliable grid for 1908.1 more steps (a 225.5% improvement), which translates to 159 hours of reliable power more than the baseline. DRAGON achieves these results because of how we incorporate domain knowledge differently than the baseline. After DRAGON's policy network outputs potential commands, each command is simulated to identify and avoid potential negative effects to the grid. In contrast, the Oroas team uses domain knowledge in a pretraining phase to learn initial weights for a policy network, instead of verifying the safety of each action before sending it to the grid. Although reasonably effective, this approach does not prevent a harmful action from being chosen

Table 3: Attack detection metrics against the attackers. DRAGON reports a significantly smaller false negative rate than the cumsum [26].

Attacker Type	Detector	Dwell Time (min)	Precision (%)	Recall (%)	F1 Score (%)	False Pos. Rate (%)	False Neg. Rate (%)
Weighted Random	CUMSUM	12.4	7.36	3.21	4.47	0.18	96.79
	DRAGON	10.3	6.81	95.38	12.71	5.68	4.62
Geometric	CUMSUM	32.29	20.1	56.3	29.63	0.45	43.7
	DRAGON	4.28	1.07	90.35	2.12	17.01	9.65

by the operator agent and damaging the grid, leading to terminal states and a lower, average step count than DRAGON.

The second finding is that DRAGON's method of operating the grid in the face of FDIAs is more effective compared to prior work. Our operator survives 305 more steps against FDIAs (a 37.4% improvement) compared to the baseline. A key difference between these two agents is that DRAGON's operator has additional features that measure the difference between the observation and the operator's simulation of the intended control action. These features indicate that the agent should respond differently compared to normal conditions. Also, DRAGON bases its action selection on *a sequence of* past observations, rather than the most recent observation. This provides more information about recent events on the grid, such as attacks, compared to the Oroas system.

8.2 Attack Detection

To answer **RQ2**, we train DRAGON's detection component against each attacker with FDIAs. If the attacker does not use FDIAs, the detector could use the expected outcome of a command to detect attacks. As a baseline, we use a cumulative sum, stateful ICS detector [26] (cumsum) that showed superior detection results in recent power grid research papers [3, 49] as well as top security conferences [9, 82]. This method learns normal behavior in different states and raises an alert when the sum of residuals exceeds a threshold. We define our state as the control command deployed during the last transition because the same command has similar effects on the grid. When tuning the detection threshold, we tested values ranging from 0.01 to 0.5 and selected the value that reported the lowest FNR, as shown in Figure 6 in Appendix D.

In Table 3, we would like to highlight that DRAGON has a significantly lower FNR compared to cumsum, with a 92.2% reduction and a 34.1% reduction against the weighted random and geometric attackers respectively. Although cumsum has been used successfully to detect cyberattacks in ICS and smart grids in particular, this experiment shows that it is ineffective at detecting our sophisticated attacks. By using grid simulation software, our stealthy attackers can inject false measurements that are close enough to the expected outcome of the control command such that the stateful residual is almost always less than the cumsum threshold. In particular, attackers can use the grid's state estimation tool to forecast the effects of the intended control command on the current grid state and inject the expected values, resulting in a small residual. Hence, cumsum misses many attacks, resulting in a high FNR. In contrast, DRAGON detects more attacks by considering a sequence of previous observations and employing non-linear activations in its policy model that can learn more nuanced indicators of attack as well as non-attack transitions. This allows DRAGON to detect more attacks than the cumsum approach, resulting in a significantly lower FNR and reduced damage to the grid. DRAGON reports an average FPR of 11.4%. When detecting the weighted random attacker, DRAGON reports the best performance with a 95.4% TPR and 5.7% FPR. This is a 92.2% increase in TPR with only a 5.5% increase in FPR. Although DRAGON's performance decreases when detecting the geometric attacker, it still achieves a 34.1% reduction in the FNR, compared to the cumsum detector. As a low FNR is the most critical in security, a higher FPR is an acceptable trade-off to benefit from DRAGON's enhanced detection capabilities. In future work, we plan to investigate additional FP reduction mechanisms such as incorporating features from the control network.

Despite DRAGON's TPR and FNR improvements, cumsum achieves lower FPRs and higher precision. When setting the cumsum threshold, we simulated the power flow with no attacks and selected a threshold such that the FPR was close to zero, as the authors recommend [26]. With this setting, the cumulative sum of residuals rarely exceeds the threshold during normal operations, leading to a lower FPR and higher precision, compared to DRAGON.

Additionally, DRAGON detects attacks 15.1 minutes faster than cumsum, on average. For context, in the Metcalf motivating example, the operator responded to the attack within an hour, restoring the system to a reliable state [86]. To further quantify our detector's timeliness, we compute the percent of the attack that is completed before being detected by comparing the dwell time with the 4 hour maximum attack length. On average, the attacker only completes 6.3% of its planned attack prior to being detected, vs. cumsum, which detects attacks after 46.6% of the attack is completed. As DRAGON detects attacks much earlier than cumsum, it is well-suited to detect attacks on power grids early, triggering an incident response investigation, and significantly reducing the damage these attacks inflict while also preventing threat fatigue with its low FPRs.

To understand challenging situations for our detector agent, we analyzed false detection decisions in-depth. We can classify FN cases into two situations: the attacker completes its entire attack or the operator agent fails to maintain grid stability at which point the scenario ends. In the latter case, which accounts for 27.3% of the FNs, there are fewer opportunities for the detector to flag the attack compared to other attack instances that last the entire 4 hours. By improving the operator agent, the number of cases where an attack goes undetected because the scenario terminates would decrease.

For FPs, there were two prevalent cases. First, a FP occurred when the operator took actions that modified a particular substation (i.e., substation 5 of the IEEE 14 model in Figure 5 in Appendix D), but did not cause any components to change their bus assignment. This could happen if some of the targeted components are already connected to the intended bus in the substation, resulting in little change to the measurements for the components around substation 5, similar to what the operator would expect to see if an attacker returned simulated measurements rather than the true observation, leading to the FP. This case accounts for 72.2% of the FPs for the DRAGON detector. Second, a small amount of FPs (19.4%) occurred when the control command acted on one of the lines targeted by the attacker. Such actions can cause similar grid transitions as an attack on the line would, causing the detector to suspect an attack.

8.3 Operator Robustness

When deployed in practice, DRAGON will likely encounter different types of attackers than the types it was trained against. As DRAGON's operator agent must still maintain reliable power in such situations, we evaluated all possible pairs of training/testing against our attackers to assess this capability. We report the operator's performance in Table 4.

When evaluated against no attacker, the operator, trained against the weighted random attacker, performs the best. This operator agent maintained grid reliability for 1822.8 more steps (a 26.6% improvement) on average, compared to the operator trained against no attacker. By including an attacker in the training process, the grid is put into more diverse states as lines are disconnected, allowing the operator to collect more interesting experiences related to responding to hazardous events. The trainers use these experiences to train a better model, leading to the performance improvement.

Similarly, this experiment shows that it is always preferable to include an attacker during training. In fact, the operator, trained against the weighted random attacker, outperforms the operator trained against no attacker by 2278.98 steps (189.92 hours), on average. This suggests that this attacker is the best to use while training an operator agent. Compared to the geometric attacker, the weighted random attacker always attacks a line for the full 4 hour attack period, but the geometric attacker sometimes performs shorter attacks. By maximizing the length of attacks, the weighted random attacker challenges the operator most during training, generating more experiences related to attack response. Therefore, the trainers sample the attack response experiences more and produce a higher performing attack response policy.

Finally, we found that training an operator agent with FDIAs enabled does not produce a superior operator when evaluated against FDIAs. On average, an operator trained against FDIAs survives 1021.3 fewer steps compared to the operator trained without FDIAs. The core challenge of training against FDIAs is that the operator agent is not given as much of an opportunity to learn attack response skills. Specifically, during FDIAs, the operator is given poisoned observations, which are incorporated into the DRL policy during training, corrupting the model. In contrast, when there are no FDIAs, the operator's policy is trained with correct measurements, which improves the operator's attack response skills, This indicates that it is better to train an operator agent without FDIAs to prepare it for attackers with and without FDIAs.

8.4 Case Study: Metcalf Substation Incident

In this case study, we provide insights on how DRAGON would respond to our motivating example of the Metcalf attack. We adapt the physical attack into a cyberattack by providing MITRE Techniques, Tactics, and Procedures (TTPs) [57], previously leveraged against power grids [45], which could be used to cause the Metcalf incident. The overall attack plan is to infect one of the PLCs in the Metcalf substation to disconnect a power line and cause cascading outages. First, attackers would gain a foothold in the utility company using a wide range of techniques, such as spearfishing attachments (TTP T0865). Next, attackers would likely conduct reconnaissance using the standard OPC protocol to map the OT network as seen in the HAVEX malware [17] (Automated Collection, T0802). At this point, the attacker has knowledge of what devices are used to operate the grid and is prepared to craft their attack. As they do not want to be flagged by IT security measures (AV, firewall, etc.), they decide to target the PLC that controls the status of the targeted line. After crafting malicious firmware updates, the attacker can use strategies adopted by the HARVEY malware to infect the PLC [25] (System Firmware, T0857). Once the attacker-developed firmware is running on the PLC, it blocks commands from the control center that try to modify the line (Denial of Control, T0813). Also, when sending measurement data to the operator, the malicious firmware modifies values to hide evidence of their attack from the operator (Denial of View, T0815). The weighted random attacker deploys this attack and thus we complete this case study using the operator and detector agents trained against this attacker.

Detection of Metcalf Inspired Attack. We first investigate how this cyberattack would be detected by DRAGON. Our detector agent benefits from the operator agent by detecting when operator commands cause unexpected changes to the next observation of the grid, due to interference from cyberattacks. In contrast, many prior ICS detection work compares a single observation to a model of expected behavior, losing the additional context of how the particular command affects the grid. To showcase this novelty, we examined the detector agent's observations where all of the discrete information (i.e. bus bar topology, line connectivity) matched when under attack vs. not under attack. This information is straightforward for the attacker to predict and poison the measurements accordingly.

We notice a difference in the real power measurements when comparing transitions under attack vs. not. The real power value differences averaged 19.7 with an attacker, but 27.5 without an attacker. The power values depend on the generator and load setpoints and how the power distributes itself throughout the grid. Even though the generator schedules are fairly predictable, loads are stochastic as they change depending on usage. As such, when an attacker injects loads that match the operator's simulated next observation closely, it raises suspicion because the values are unexpectedly accurate. These values cascade to the real power values injected by the attacker and the detector's policy can identify this cascading effect to detect the Metcalf inspired attack

Operator Response to Metcalf Inspired Attack. Next, we analyze how the operator agent would respond to this attack. In one attack, the line between substations 3 and 4 is disconnected. This removes a path for power from generator 0 to flow to the top right of the grid. To improve power flow, our operator reconfigures substation 1 such that the generator and one line are transferred to their own bus. This focuses the power from generator 0 to parts of the grid that were partially cut off from sufficient power due to the attack. During a different attack, the line between substations 2 and 3 is disconnected, removing 1 of 4 lines between a significant portion of the generation and the loads in the network. Then, the power is constrained to the other 3 main lines, overloading the line between substations 1 and 4. To relieve this overflow, DRAGON proposes to switch a line in substation 4 to the same bus as the overloaded line to draw power away from this line.

8.5 Training Runtime

As the size of the grid increases, DRAGON will require more training data to learn effectively. One approach to address that problem

Table 4: The mean steps and rewards with different combinations of attacker types.

Training Attacker		None Steps	None Reward	No FDIA Weighted Random Steps	No FDIA Weighted Random Reward	No FDIA Geometric Steps	No FDIA Geometric Reward	FDIA Weighted Random Steps	FDIA Weighted Random Reward	FDIA Geometric Steps	FDIA Geometric Reward
No FDIA	None	5032.44	3694.28	825.57	473.59	1525.17	997.41	644.16	349.55	1394.77	905.754
	Weighted Random	6855.25	4975.09	4284.34	2973.51	5110.82	3640.03	1456.75	937.13	3109.83	2142
	Geometric	6765.21	4658.99	2064.73	1340.6	3045.5	2045.42	1084.36	650.69	2154.28	1410.1
FDIA	Weighted Random	4556.41	3257.46	2110.88	1416.41	2659.6	1813.37	1033.38	635.33	1860.01	1228.12
	Geometric	4769.58	3267.24	1914.97	1194.15	2861.48	1878.02	703.02	380.92	1678.18	1059.54

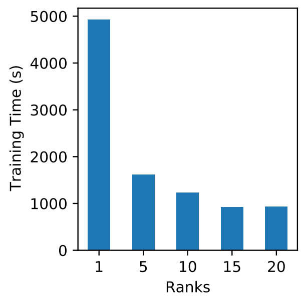

Figure 4: The Training times for running 1,000 policy updates with varying number of learners.

without extended training time is parallel training. To demonstrate that our training framework can scale to a larger number of trainers, we report how long DRAGON took to train an operator agent against no attacker for 1,000 learning steps with different numbers of distributed trainers in Figure 4. By using ten trainers instead of one, we reduced the training time by 25%. Increasing the number of trainers to 20, the training time reduced to 19% of serial training runtime. This speed up can be harnessed to perform more training iterations in the same amount of time to accelerate learning.

9 DISCUSSION

DRAGON is a system that learns how to detect cyberattacks on the grid while simultaneously maintaining reliable power by proposing control commands to human grid operators. Our extensive evaluation shows that DRAGON can maintain reliable power for 159.1 more hours, compared to a state-of-the-art, RL-based grid operator. Also, DRAGON's detection agent significantly outperforms a recent ICS attack detection system by reducing the FNR by 34.1% - 92.2% and detecting attacks 15.1 minutes sooner.

Future Work. As future work, we are interested in exploring how to integrate the incident response process into our autonomous operator framework to remove attackers from the network. Additionally, we are interested in defending against attackers that intentionally poison DRL policies, proposed by Wu et al. [89]. In robotics, Mandlekar et al. [52], proposed augmenting the training procedure of robots with adversarial examples, which could potentially be applicable to power grids.

Additionally, there are other ML approaches, besides RL, that could accelerate DRAGON's training process. The subfield of weak supervision reduces the need for extensive, high-quality labeled training data by using heuristics to generate noisy labels and then using ML models to more accurately label a large set of unlabeled data [69]. Since there are heuristics related to power grid control, weak supervision could be an effective way to generalize these heuristics and propose control commands in unexpected situations. As it might be challenging to specify sufficient heuristics up front, interactive weak supervision can incrementally refine them [5].

Limitations. We conclude our discussion with a few limitations of DRAGON and suggest mitigations. First, the operator agent relies on the ability to execute commands correctly during attacks to maintain grid reliability. If the attacker compromises this channel, the operator may not be able to withstand the attack. One potential mitigation is to add a layer of encryption to the commands, as suggested by Dan and Sandberg [13], to detect command modifications. Another limitation is the size of the grid used in our evaluation (a grid with 14 substations). In reality, power grids can have thousands of substations, which would increase the size of the observations, the number of actions, and would likely increase the training time. By adding more computing resources during training, the computation that one Trainer has to complete would likely not increase to an unreasonable amount, keeping training times from exploding.

10 RELATED WORK

ICS Attack Detection. Past work uses statistical analysis [23, 28, 36, 68], physical models [2, 82], or state-based modeling [7, 26] to detect ICS attacks. Although these methods can detect noisy attacks, sophisticated attackers can blend into normal system behavior [2]. Other methods use network activity [28, 31, 74, 83, 92] or timing [36, 68] aspects to detect attacks. Flow-based approaches [36, 74] analyze abnormal function codes, but are evaded by modern attacks that use legitimate ICS commands. Techniques that detect attacks on PLCs using formal verification [80] or control flow monitoring [8, 22, 90] either do not protect PLCs directly in the field or raise FPs when the PLC's control programs are updated. By leveraging the predictability of some ICS systems, specification-based detection methods [22, 28, 56] use state machines to ensure that the physical processes obey given transitions and detect attacks that deviate. Although effective, these methods suffer from state space explosion when the number of state is very large such as our application of the power grid [80]. One strategy to overcome this state space explosion, as we leverage in this work, is ML. Supervised learning has been used to detect attacks [50, 51, 78, 87]. However, in adversarial environments where operators have partial visibility, labeled datasets are challenging to accurately create. RL algorithms have been successful applied in these instances such that the algorithm learns to detect attacks with limited knowledge [1, 43].

Lastly, Shekari et al. [73] propose an IDS based on radio frequencies that compares the magnetic field measurements of substations with global lightning data to detect discrepancies and possible attacks.

11 CONCLUSION

DRAGON is a system that operates a power grid and detects cyberattacks by learning through experience collected by interacting with a grid simulator. Our system actively explores potential attacks that maximize the damage to the grid to improve and prepare the operator agent. Our evaluation shows that DRAGON can maintain reliable power delivery by learning how to respond to anomalies in the grid. Our system detected 92.9% of attacks 15.05 minutes before the baseline. These early detection signals allows grid operators to intervene and facilitate effective incident response operations.

ACKNOWLEDGMENTS

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes. This work was performed in part under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

Disclaimer: This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

Co-author Matthew Landen was supported by an NSF Scholarship-for-service (SFS) award number 1565523 and by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-2039655. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Dou An, Qingyu Yang, Wenmao Liu, and Yang Zhang. 2019. Defending against Data Integrity Attacks in Smart Grid: A Deep Reinforcement Learning-based Approach. *IEEE Access* 7 (2019), 110835–110845.

[2] Wissam Aoudi, Mikel Iturbe, and Magnus Almgren. 2018. Truth Will Out: Departure-based Process-level Detection of Stealthy Attacks on Control Systems. In *Proceedings of the Conference on Computer and Communications Security*. 817–831.

[3] Kavya Ashok, Dan Li, Nagi Gebraeel, and Deepak Divan. 2021. Online Detection of Inter-turn Winding Faults in Single-phase Distribution Transformers using Smart Meter Data. *IEEE Transactions on Smart Grid* 12, 6 (2021), 5073–5083.

[4] Chen Binbin. 2021. AsprinChina/L2RPN_NIPS_2020_a_PPO_ Solution. https://github.com/AsprinChina/L2RPN_NIPS_2020_a_PPO_ Solution original-date: 2020-11-12T03:53:04Z.

[5] Benedikt Boecking, Willie Neiswanger, Eric Xing, and Artur Dubrawski. 2021. Interactive Weak Supervision: Learning Useful Heuristics for Data Labeling. In *International Conference on Learning Representations*. https://openreview.net/forum?id=IDFQI9OY6K

[6] Francesco Cadini, Gian Luca Agliardi, and Enrico Zio. 2017. A Modeling and Simulation Framework for the Reliability/Availability Assessment of a Power Transmission Grid Subject to Cascading Failures under Extreme Weather Conditions. *Applied energy* 185 (2017), 267–279.

[7] Andrea Carcano, Alessio Coletta, Michele Guglielmi, Marcelo Masera, I Nai Fovino, and Alberto Trombetta. 2011. A Multidimensional Critical State Analysis for Detecting Intrusions in SCADA Systems. *IEEE Transactions on Industrial Informatics* 7, 2 (2011), 179–186.

[8] Long Cheng, Ke Tian, and Danfeng Yao. 2017. Orpheus: Enforcing Cyber-physical Execution Semantics to Defend Aagainst Data-oriented Attacks. In *Proceedings of the 33rd Annual Computer Security Applications Conference*. 315–326.

[9] Kyong-Tak Cho and Kang G Shin. 2016. Fingerprinting Electronic Control Units for Vehicle Intrusion Detection. In *Proceedings of the USENIX Security Symposium*. 911–927.

[10] Alfredo V Clemente, Humberto N Castejón, and Arjun Chandra. 2017. Efficient Parallel Methods for Deep Reinforcement Learning. *arXiv preprint arXiv:1705.04862* (2017).

[11] Jose Cordova-Garcia, Xin Wang, Dongliang Xie, Yue Zhao, and Lei Zuo. 2019. Control of Communications-Dependent Cascading Failures in Power Grids. *IEEE Transactions on Smart Grid* 10, 5 (2019), 5021–5031.

[12] CrowdStrike. 2022. Falcon Complete: Managed Detection and Response. CrowdStrike Website. https://www.crowdstrike.com/endpoint-security-products/falcon-complete/.

[13] György Dán and Henrik Sandberg. 2010. Stealth Attacks and Protection Schemes for State Estimators in Power Systems. In *Proceedings of the International Conference on Smart grid Communications*. IEEE, 214–219.

[14] Timothy A. Davis and Ekanathan Palamadai Natarajan. 2010. Algorithm 907: KLU, A Direct Sparse Solver for Circuit Simulation Problems. *ACM Trans. Math. Softw.* 37, 3, Article 36 (sep 2010), 17 pages.

[15] Benjamin Donnot. 2020. Grid2op- A testbed platform to model sequential decision making in power systems. . https://GitHub.com/rte-france/grid2op.

[16] Benjamin Donnot, Isabelle Guyon, Marc Schoenauer, Patrick Panciatici, and Antoine Marot. 2017. Introducing Machine Learning for Power System Operation Support. In *Proceedings of the IREP Symposium*.

[17] Dragos Inc. 2019. The Evolution of Cyber Attacks on Electric Operations. https://www.dragos.com/blog/industry-news/the-evolution-of-cyber-attacks-on-electric-operations/.

[18] Electric Power Research Institute. 2022. Welcome to the AI.EPRI L2RPN challenge portal. https://www.epri.com/l2rpn.

[19] Mircea Eremia and Mohammad Shahidehpour. 2013. *Handbook of Electrical Power System Dynamics: Modeling, Stability, and Control*. Vol. 92. John Wiley & Sons.

[20] FireEye. 2021. Email Security Solution. FireEye website. https://www.fireeye.com/products/email-security.html.

[21] FireEye. 2021. Endpoint Security Software and Solutions. FireEye Website. https://www.fireeye.com/products/endpoint-security.html.

[22] David Formby and Raheem Beyah. 2019. Temporal Execution Behavior for Host Anomaly Detection in Programmable Logic Controllers. *IEEE Transactions on Information Forensics and Security* 15 (2019), 1455–1469.

[23] David Formby, Preethi Srinivasan, Andrew M Leonard, Jonathan D Rogers, and Raheem A Beyah. 2016. Who's in Control of Your Control System? Device Fingerprinting for Cyber-Physical Systems.. In *Proceedings of the Network and Distributed System Security Symposium*. The Internet Society.
[24] Marguerite Frank. 1981. The Braess Paradox. *Mathematical Programming* 20, 1 (1981), 283–302.
[25] Luis Garcia, Ferdinand Brasser, Mehmet Hazar Cintuglu, Ahmad-Reza Sadeghi, Osama A Mohammed, and Saman A Zonouz. 2017. Hey, My Malware Knows Physics! Attacking PLCs with Physical Model Aware Rootkit.. In *Proceedings of the Network and Distributed System Security Symposium*. The Internet Society.
[26] Hamid Reza Ghaeini, Daniele Antonioli, Ferdinand Brasser, Ahmad-Reza Sadeghi, and Nils Ole Tippenhauer. 2018. State-aware Anomaly Detection for Industrial Control Systems. In *Proceedings of the Annual ACM Symposium on Applied Computing*. 1620–1628.
[27] J Duncan Glover, Mulukutla S Sarma, and Thomas Overbye. 2012. *Power System Analysis & Design, SI version*. Cengage Learning.
[28] Niv Goldenberg and Avishai Wool. 2013. Accurate Modeling of Modbus/TCP for Intrusion Detection in SCADA Systems. *International Journal of Critical Infrastructure Protection* 6, 2 (2013), 63–75.
[29] András György and Levente Kocsis. 2011. Efficient Multi-start Strategies for Local Search Algorithms. *Journal of Artificial Intelligence Research* 41 (2011), 407–444.
[30] Dina Hadžiosmanović, Robin Sommer, Emmanuele Zambon, and Pieter H Hartel. 2014. Through the Eye of the PLC: Semantic Security Monitoring for Industrial Processes. In *Proceedings of the Annual Computer Security Applications Conference*. 126–135.
[31] James Halvorsen and Julian L Rrushi. 2017. Target Discovery Differentials for 0-Knowledge Detection of ICS Malware. In *Proceedings of the Intl Conf on Dependable, Autonomic and Secure Computing*. IEEE, 542–549.
[32] Matthew Hausknecht and Peter Stone. 2015. Deep Recurrent Q-learning for Partially Observable MDPs. In *Proceedings of AAAI Fall Symposium Series*.
[33] Bing Huang, Alvaro A Cardenas, and Ross Baldick. 2019. Not Everything is Dark and Gloomy: Power Grid Protections Against IoT Demand Attacks. In *Proceedings of the USENIX Security Symposium* . 1115–1132.
[34] Information Trust Institution. 2013. IEEE 14-Bus System. ITI Website. https://icseg.iti.illinois.edu/ieee-14-bus-system.
[35] Grid Modernization Initiative. 2018. *2018 Grid Modernization Peer Review Report: Foundational Projects and Technical Area Portfolio Review.* Technical Report. Department of Energy. https://www.energy.gov/sites/default/files/2019/12/f69/GMI-Peer-Review-Report-2018_FINAL.pdf.
[36] Celine Irvene, Tohid Shekari, David Formby, and Raheem Beyah. 2019. If I Knew Then What I Know Now: On Reevaluating DNP3 Security using Power Substation Traffic. In *Proceedings of the Annual Industrial Control System Security (ICSS) Workshop*. 48–59.
[37] Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, et al. 2017. Population based Training of Neural Networks. *arXiv preprint arXiv:1711.09846* (2017).
[38] Mahdi Jamei, Anna Scaglione, Ciaran Roberts, Emma Stewart, Sean Peisert, Chuck McParland, and Alex McEachern. 2017. Anomaly Detection Using Optimally Placed μPMU Sensors in Distribution Grids. *IEEE Transactions on Power Systems* 33, 4 (2017), 3611–3623.
[39] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101, 1 (1998), 99–134.
[40] Kaspersky. 2021. BlackEnergy APT Attacks in Ukraine. https://www.kaspersky.com/resource-center/threats/blackenergy.
[41] Adrian Kelly, Aidan O'Sullivan, Patrick de Mars, and Antoine Marot. 2020. Reinforcement Learning for Electricity Network Operation. *arXiv preprint arXiv:2003.07339* (2020).
[42] R James Ranjith Kumar and Biplab Sikdar. 2021. Detection of Stealthy Cyber-physical Line Disconnection Attacks in Smart Grid. *IEEE Transactions on Smart Grid* 12, 5 (2021), 4484–4493.
[43] Mehmet Necip Kurt, Oyetunji Ogundijo, Chong Li, and Xiaodong Wang. 2018. Online Cyber-attack Detection in Smart Grid: A Reinforcement Learning Approach. *IEEE Transactions on Smart Grid* 10, 5 (2018), 5174–5185.
[44] Matthew Landen. 2022. Dragon Hyperparameters. https://github.com/mlanden/Dragon-Hyperparameters.
[45] Robert M Lee, MJ Assante, and T Conway. 2017. Crashoverride: Analysis of the threat to electric grid operations. *Dragos Inc., March* (2017).
[46] Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Moritz Hardt, Ben Recht, and Ameet Talwalkar. 2018. A System for Massively Parallel Hyperparameter Tuning. *arXiv preprint* (2018).
[47] Yao Liu, Peng Ning, and Michael K Reiter. 2011. False Data Injection Attacks against State Estimation in Electric Power Grids. *Transactions on Information and System Security (TISSEC)* 14, 1 (2011), 1–33.
[48] Jason Lu. 2020. Lujasone/ neurips_2020_l2rpn_comp_an_approach: The implementation of neurips_2020_l2rpn_track1 (robustness) and Track2 (adaptability) competition. https://github.com/lujasone/NeurIPS_2020_L2RPN_Comp_An_Approach.
[49] Kang-Di Lu and Zheng-Guang Wu. 2021. Constrained-Differential-Evolution-Based Stealthy Sparse Cyber-Attack and Countermeasure in an AC Smart Grid. *IEEE Transactions on Industrial Informatics* 18, 8 (2021), 5275–5285.
[50] Leandros A Maglaras and Jianmin Jiang. 2014. Intrusion Detection in SCADA Systems using Machine Learning Techniques. In *Proceedings of the Science and Information Conference*. IEEE, 626–631.
[51] Leandros A Maglaras, Jianmin Jiang, and Tiago Cruz. 2014. Integrated OCSVM Mechanism for Intrusion Detection in SCADA systems. *Electronics Letters* 50, 25 (2014), 1935–1936.
[52] Ajay Mandlekar, Yuke Zhu, Animesh Garg, Li Fei-Fei, and Silvio Savarese. 2017. Adversarially Robust Policy Learning: Active Construction of Physically-plausible Perturbations. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 3932–3939.
[53] Antoine Marot, Benjamin Donnot, Gabriel Dulac-Arnold, Adrian Kelly, Aïdan O'Sullivan, Jan Viebahn, Mariette Awad, Isabelle Guyon, Patrick Panciatici, and Camilo Romero. 2021. Learning to run a Power Network Challenge: a Retrospective Analysis. *arXiv preprint arXiv:2103.03104* (2021).
[54] Antoine Marot, Benjamin Donnot, Camilo Romero, Balthazar Donon, Marvin Lerousseau, Luca Veyrin-Forrer, and Isabelle Guyon. 2020. Learning to Run a Power Network Challenge for Training Topology controllers. *Electric Power Systems Research* 189 (2020), 106635.
[55] A Marot, B Donnot, S Tazi, and P Panciatici. 2018. Expert System for Topological Remedial Action Discovery in Smart Grids. (2018).
[56] Stephen E McLaughlin, Saman A Zonouz, Devin J Pohly, and Patrick D McDaniel. 2014. A Trusted Safety Verifier for Process Controller Code.. In *Proceedings of the Network and Distributed System Security Symposium*, Vol. 14. The Internet Society.
[57] MITRE. 2021. ATT&CK® for Industrial Control Systems. https://collaborate.mitre.org/attackics.
[58] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous Methods for Deep Reinforcement Learning. In *Proceedings of the International Conference on Machine Learning*. PMLR, 1928–1937.

[59] Sean Monemi, Travon Dent, and Antonio Nunez. 2022. A Model of System Protection in IEEE 14-bus Power Grid. In *2022 IEEE International Conference in Power Engineering Application (ICPEA)*. 1–5. https://doi.org/10.1109/ICPEA53519.2022.9744697

[60] Arun Nair, Praveen Srinivasan, Sam Blackwell, Cagdas Alcicek, Rory Fearon, Alessandro De Maria, Vedavyas Panneershelvam, Mustafa Suleyman, Charles Beattie, Stig Petersen, et al. 2015. Massively Parallel Methods for Deep Reinforcement Learning. *arXiv preprint arXiv:1507.04296* (2015).

[61] Thanh Thi Nguyen and Vijay Janapa Reddi. 2019. Deep reinforcement learning for cyber security. *arXiv preprint arXiv:1906.05799* (2019).

[62] Loïc Omnes, Antoine Marot, and Benjamin Donnot. 2021. Adversarial Training for a Continuous Robustness Control Problem in Power Systems. In *Proceedings of the IEEE Madrid PowerTech*. IEEE, 1–6.

[63] Palo Alto Networks. 2021. Next-Generation Firewalls. Palo Alto Networks website. https://www.paloaltonetworks.com/network-security/next-generation-firewall.

[64] Shengyi Pan, Thomas Morris, and Uttam Adhikari. 2015. Developing a Hybrid Intrusion Detection System using Data Mining for Power Systems. *IEEE Transactions on Smart Grid* 6, 6 (2015), 3104–3113.

[65] Paul W Parfomak. 2014. Physical Security of the US Power Grid: High-voltage Transformer Substations.

[66] Colin Parris. [n. d.]. U.S. Department of Energy Recognizes GE's Work with Digital Twins. https://www.ge.com/digital/blog/us-department-energy-recognizes-ges-work-digital-twins.

[67] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[68] Stanislav Ponomarev. 2015. *Intrusion Detection System of Industrial Control Networks using Network Telemetry.* Louisiana Tech University.

[69] Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016. Data Programming: Creating Large Training Sets, Quickly. *Advances in Neural Information Processing Systems* 29 (2016).

[70] Réseau de Transport d'Électricité. 2022. L2RPN Challenge. https://l2rpn.chalearn.org/.

[71] Diaa Salman, Mehmet Kusaf, Yonis Khalif Elmi, and Ammar Almasri. 2022. Optimal Power Systems Planning for IEEE-14 Bus Test System Application. In *2022 10th International Conference on Smart Grid (icSmartGrid)*. 290–295. https://doi.org/10.1109/icSmartGrid55722.2022.9848574

[72] Wei Shao and Vijay Vittal. 2005. Corrective Switching Algorithm for Relieving Overloads and Voltage Violations. *IEEE Transactions on Power Systems* 20, 4 (2005), 1877–1885.

[73] Tohid Shekari, Christian Bayens, Morris Cohen, Lukas Graber, and Raheem Beyah. 2019. RFDIDS: Radio Frequency-based Distributed Intrusion Detection System for the Power Grid.. In *Proceedings of the Network and Distributed System Security Symposium*. The Internet Society.

[74] Chetna Singh, Ashwin Nivangune, and Mrinal Patwardhan. 2016. Function Code based Vulnerability Analysis of DNP3. In *Proceedings of the International Conference on Advanced Networks and Telecommunications Systems (ANTS)*. IEEE, 1–6.

[75] Joe Slowik. 2019. Crashoverride: Reassessing the 2016 ukraine electric power event as a protection-focused attack. *Dragos, Inc* (2019).

[76] Joe Slowik. 2020. Stuxnet to Crashoverride to Trisis: Evaluating the History and Future of Integrity-Based Attacks on Industrial Environments. https://www.dragos.com/resource/stuxnet-to-crashoverride-to-trisis-evaluating-the-history-and-future-of-integrity-based-attacks-on-industrial-environments/.

[77] Saleh Soltan, Prateek Mittal, and H Vincent Poor. 2018. BlackIoT: IoT Botnet of High Wattage Devices can Disrupt the Power Grid. In *Proceedings of the USENIX Security Symposium*. 15–32.

[78] Barnaby Stewart, Luis Rosa, Leandros A Maglaras, Tiago J Cruz, Mohamed Amine Ferrag, Paulo Simoes, and Helge Janicke. 2017. A Novel Intrusion Detection Mechanism for SCADA Systems which Automatically Adapts to Network Topology Changes. *EAI Endorsed Transactions on Industrial Networks and Intelligent Systems* 4, 10 (2017).

[79] Adam Stooke and Pieter Abbeel. 2018. Accelerated Methods for Deep Reinforcement Learning. *arXiv preprint arXiv:1803.02811* (2018).

[80] Ruimin Sun, Alejandro Mera, Long Lu, and David Choffnes. 2021. SoK: Attacks on Industrial Control Logic and Formal Verification-based Defenses. In *Proceedings of the IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 385–402.

[81] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction.* MIT press.

[82] David I Urbina, Jairo A Giraldo, Alvaro A Cardenas, Nils Ole Tippenhauer, Junia Valente, Mustafa Faisal, Justin Ruths, Richard Candell, and Henrik Sandberg. 2016. Limiting the Impact of Stealthy Attacks on Industrial Control Systems. In *Proceedings of the Conference on Computer and Communications Security*. ACM, 1092–1105.

[83] Alfonso Valdes and Steven Cheung. 2009. Communication Pattern Anomaly Detection in Process Control Systems. In *Proceedings of the Conference on Technologies for Homeland Security*. IEEE, 22–29.

[84] Jorge Valenzuela, Jianhui Wang, and Nancy Bissinger. 2012. Real-time Intrusion Detection in Power System Operations. *IEEE Transactions on Power Systems* 28, 2 (2012), 1052–1062.

[85] Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep Reinforcement Learning with Double Q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 30.

[86] Heidi Vella. 2014. Under threat: protecting substations and power lines from attack. https://www.power-technology.com/features/featureunder-threat-protecting-substations-and-power-lines-from-attack-4192867/.

[87] David Wilson, Yufei Tang, Jun Yan, and Zhuo Lu. 2018. Deep Learning-aided Cyber-attack Detection in Power Transmission Systems. In *Proceedings of the Power & Energy Society General Meeting (PESGM)*. IEEE, 1–5.

[88] Allen J Wood, Bruce F Wollenberg, and Gerald B Sheblé. 2013. *Power generation, operation, and control.* John Wiley & Sons.

[89] Xian Wu, Wenbo Guo, Hua Wei, and Xinyu Xing. 2021. Adversarial Policy Training against Deep Reinforcement Learning. In *Proceedings of the USENIX Security Symposium*.

[90] Huan Yang, Liang Cheng, and Mooi Choo Chuah. 2018. Detecting Payload Attacks on Programmable Logic Controllers (PLCs). In *Proceedings of the Conference on Communications and Network Security*. IEEE, 1–9.

[91] Yi Yang, Keiran McLaughlin, Sakir Sezer, Tim Littler, Eul Gyu Im, Bernardi Pranggono, and HF Wang. 2014. Multiattribute SCADA-specific Intrusion Detection System for Power Networks. *IEEE Transactions on Power Delivery* 29, 3 (2014), 1092–1102.

[92] Jeong-Han Yun, Sung-Ho Jeon, Kyoung-Ho Kim, and Woo-Nyon Kim. 2013. Burst-based Anomaly Detection on the DNP3 Protocol. *International Journal of Control and Automation* 6, 2 (2013), 313–324.

[93] Chiyuan Zhang, Oriol Vinyals, Remi Munos, and Samy Bengio. 2018. A Study on Overfitting in Deep Reinforcement Learning. *arXiv preprint arXiv:1804.06893* (2018).

A POWER SYSTEMS BACKGROUND

Electric power grids consists of three components: generation, transmission, and distribution [19, 27, 88]. Generators (e.g. nuclear or gas power plants, solar installations, and wind turbines) inject power into transmission networks, which transport electricity through a network of transmission lines to substations. From substations, power is distributed to customers (loads) via a distribution grid. A small example of a transmission grid is shown in Figure 5 in Appendix D.

Transmission systems must be operated in a stable and secure state to ensure reliable power delivery. A key quantity for evaluating grid performance is how close each power line is to its *thermal limit* – the maximum amount of power that can flow through a line before it becomes overloaded. If a line is overloaded for a prolonged amount of time, it will generally be de-energized by automated protective relays to prevent physical damage. As a result of line outages, power must be re-routed through other portions of the transmission system and generation may have to be re-dispatched to serve load. Such re-configuration may in turn lead to overflows on other lines, with cascading impacts. Ultimately, transmission and generation should be operated such that transmission line thermal limits are satisfied and all demand is met.

B DEEP REINFORCEMENT LEARNING

Reinforcement learning is the process of learning what actions to take in different situations to maximize a reward signal over time by interacting with the environment and collecting experiences [81]. This sequential decision-making problem is often formalized as a Markov decision process (MDP), which is defined as a tuple $\langle \mathcal{S}, \mathcal{A}, T, \mathcal{R} \rangle$. Here, $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the transition function that assigns a probability distribution over potential next states, given the current state and selected action, and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ is the reward function that returns a real-valued reward for each transition. Here the goal is to learn a policy, $\pi : \mathcal{S} \rightarrow \mathcal{A}$, such that the *expected total discounted reward* or *return* denoted R, obtained over time is maximized. After fixing a *discount factor* $\gamma \in (0, 1]$, the return is defined as $R = \sum_{t=0}^{N} \gamma^t r_t$, where r_t is the reward obtained at timestep t when policy π is followed[2]. The return is used to trade off immediate and long-term reward. A common RL algorithm is Q-Learning [81], which, after each action, updates the estimated value of a state, action pair using the following formula: $Q^{t+1}(s_t, a_t) = (1 - \alpha)Q^t(s_t, a_t) + \alpha[r_t + \gamma \max_{a \in \mathcal{A}} Q^t(s_{t+1}, a)]$ where $Q^t(\cdot, \cdot)$ is the action value function that takes a state s_t and action a_t and returns the agent's action value estimate at timestep t and α is the *learning rate*. Then, the agent's policy is defined as $\pi(s_t) = \text{argmax}_{a \in \mathcal{A}} Q(s_t, a)$.

Applying this framework to real-world problems, such as cybersecurity, requires two extensions: partial observability and function approximation. With partial observability, the agent does not see certain aspects of the current state. This resembles reality where perfectly measuring each component of the power grid is infeasible. For example, when an attacker de-energizes a power line, the operator does not know whether the event was caused by an attack or a natural hazard. To represent such hidden information, a partial-observable Markov decision process (POMDP) is needed. A POMDP is defined as a tuple $\langle \mathcal{S}, \mathcal{A}, T, R, \Omega, O \rangle$ where the first four terms define a MDP [39]. Instead of receiving the current state of the environment, the agent receives an observation drawn from the set of observations Ω according to the function $O : \mathcal{S} \times \mathcal{A} \rightarrow \Pi(\Omega)$, which returns a probability distribution over possible observations, given a state and an action. The goal is still to learn a policy that maximizes the return, however this policy now maps observations to action instead of states giving $\pi : O \rightarrow \mathcal{A}$. In reality, agents must operate in large state-action spaces. In realistic domains, the number of possible observations and actions can grow very large. In such instances, the method of estimating a value for each state/observation is infeasible. Instead, function approximators replace traditional, tabular data structures [81]. In DRL, neural networks are used to approximate value functions (named policy networks).

[2]Each timestep consists of taking an action in the current state, and receiving a reward and next state.

C INTELLIGENT ATTACK SPACE DESIGN

The attackers we consider in this work can disable powerlines for a period of time. We assume that our attackers can obtain prior knowledge of the grid configuration and use this knowledge to target a subset of power lines that would be most challenging for the operator to respond to. To identify this subset, we leverage research on heuristic based, multi-start optimization, following methods described by György and Kocsis [29]. These methods sample a pool of initial starting points and perform a sequence of local improvement steps, guided by a fitness function, to produce final points that satisfy a desired fitness level. We apply this algorithm to our task of attack space creation by defining a sample, an improvement step and a fitness function to guide the optimization.

First, a sample is defined as a subset of powerlines. Next, we define two samples to be neighbors if one can be obtained from the other by removing and adding a line. To evaluate a sample's fitness, we measure the ability of the operator agent to respond to attacks that are drawn from the sample by comparing how long our operator, trained without an attacker, maintains the grid to an operator that does not take actions against a weighted random attacker with the given set of targeted lines. After running the two agents, we compute the differences between the number of steps reached by each agent. This difference is used as a heuristic for the local search as it captures the intuition that an agent trained with no attacker learns to respond to hazards such as overloaded powerline being automatically disconnected. If our agent performs worse compared to an operator who does not take any actions, the attacks drawn from the current subset are challenging for the operator and the subset is a good candidate for our attacker. We start by drawing 50 random attack sets to begin searching from. Then, we run a local search process starting from each set until the attacker exceeds a target threshold used to define a sufficiently challenging attacker. We collect the samples in which our agent maintained the grid for fewer steps than the agent that takes no actions in 30% of the test scenarios. The final set is constructed by selecting the lines that were seen with the highest frequency within the identified sample sets. We enforced a fixed attack space size of 17% of the powerlines to target, as this is the size used during the L2RPN 2020 competition.

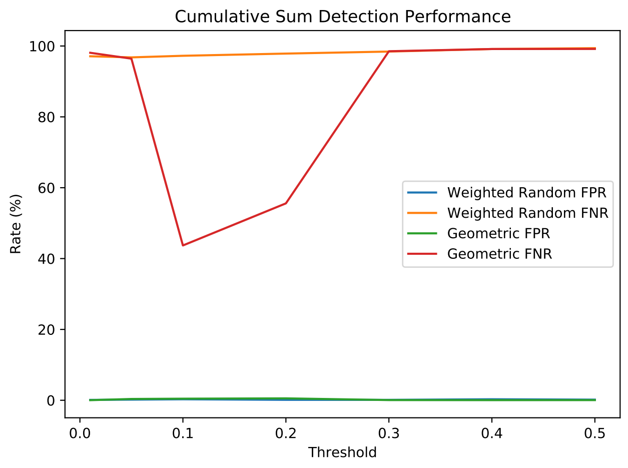

Figure 6: Results for the cumulative sum detector against the geometric and weighted random attackers.

Table 5: Power grid observation features.

Features	**Description**
Active and reactive power, voltages	For each generator and line, these features describe the power flow in the grid
Generator dispatch levels and loads	These value capture the requirements of the operator related to how much power exists and how much demand exists
Grid connectivity	These features tell the operator what bus each line is connected to and the available paths that power can be routed through
Percent line capacities	For each line, these features help the agent identity possible overflows and take action to minimize their impacts

D POWER GRIDS

Figure 5 shows a visual representation of the power grid used for the evaluation. The numbered circles represent substations that are connected with powerlines. The green outlined pentagons are generators and the yellow triangles are loads.

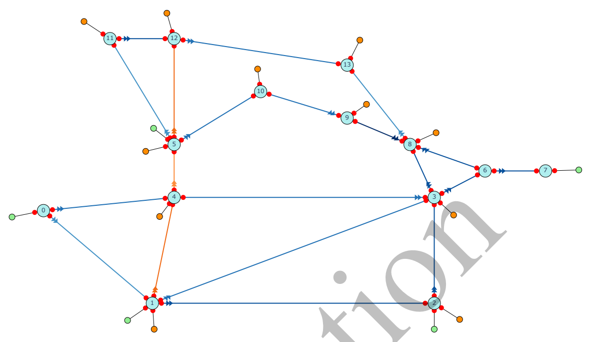

Figure 5: Pictures of the IEEE 14 power grid used in our evaluation.

Also, Table 5 shows the different features used by the agents for each grid observation and also defines each type.

E HYPERPARAMETERS

The exploration parameters were set manually by the authors to ensure that the respective agents sufficiently explore the problem space and slowly begin to decrease exploration as learner continues. For the Boltzmann starting temperatures, the value was set such that the probability distribution started close to uniform and all the actions were tried. The ending temperatures were set such that there was at least one action with probability noticeably greater than the uniform probability. Finally, the decay rate and frequencies for the Boltzmann politics were configured so that the agent's performance increased throughout the process, indicating sufficient exploration to learn useful information. Similar heuristics were used to set set the epsilon hyperparameters for the epsilon greedy, detection policy.