Title

Deep Reinforcement Learning for DER Cyber-Attack Mitigation

Authors
Roberts, Ciaran
Ngo, Sy-Toan
Milesi, Alexandre
et al.

Publication Date
2020-11-13

DOI
10.1109/smartgridcomm47815.2020.9302997

Peer reviewed

Deep Reinforcement Learning for DER Cyber-Attack Mitigation

Ciaran Roberts
Sy-Toan Ngo, Alexandre Milesi
Sean Peisert, Daniel Arnold
Lawrence Berkeley National Laboratory
{cmroberts,sytoanngo,amilesi,
sppeisert,dbarnold}@lbl.gov

Shammya Saha
Anna Scaglione
Nathan Johnson
Arizona State University
{sssaha,ascaglio,
nathanjohnson}@asu.edu

Anton Kocheturov
Dmitriy Fradkin
Siemens Corporation
Corporate Technology
{anton.kocheturov,
dmitriy.fradkin}@siemens.com

***Abstract*—The increasing penetration of DER with smart-inverter functionality is set to transform the electrical distribution network from a passive system, with fixed injection/consumption, to an active network with hundreds of distributed controllers dynamically modulating their operating setpoints as a function of system conditions. This transition is being achieved through standardization of functionality through grid codes and/or international standards. DER, however, are unique in that they are typically neither owned nor operated by distribution utilities and, therefore, represent a new emerging attack vector for cyber-physical attacks. Within this work we consider deep reinforcement learning as a tool to learn the optimal parameters for the control logic of a set of uncompromised DER units to actively mitigate the effects of a cyber-attack on a subset of network DER.**

I. Introduction

The increasing penetration of distributed energy resources (DER) in electrical distribution systems is causing a paradigm shift in how these networks are managed. While these systems were historically passive, distributed power generation is forcing distribution grids to become more dynamic as DER are expected to provide grid services, *e.g.* voltage control. This transition presents several challenges, particularly in the area of cyber-physical security [1], [2].

DER are especially unique when it comes to cyber-physical security. These devices are typically neither utility owned nor directly controlled and, therefore, present a new attack vector for adversaries seeking to disrupt normal grid operating conditions. Additionally, many manufacturers and/or aggregators remotely control large populations of these devices via cellular networks, customers' WiFi routers, or wired internet connections [3]. This makes ensuring the integrity of commands significantly more difficult. While recent standards (e.g. IEEE 1547 standard) seek to specify minimal control requirements for these devices, they do not explicitly address the associated cyber-physical security challenges [4]. Inverter manufacturers and aggregators have the ability to remotely monitor and control the settings for inverters/DER deployed in the field. Once access to the central system is gained, that system can be used to push malicious control logic back to all DER. Thus, utilities have already expressed concerns about how the impact of a single cyber intrusion into an aggregators'/manufacturers' internal network could be exploited to compromise an aggregator's/manufacturer's entire DER fleet [3]. In regions with high penetration of these devices, this could have devastating effects.

In this work we adopt a purely physics-based approach for the mitigation of cyber-physical attacks on DER (specifically, solar photovoltaic inverters). We assume that the adversary has already gained access to a subset of DER on a given network and seeks to maliciously re-configure the control settings of smart inverters to disrupt distribution grid operations. Our approach does not focus on detecting the cyber-intrusion but rather mitigating the resulting physical manifestation of the attack on the grid. To develop optimal control policies that mitigate the impact of these attacks, we train a deep reinforcement learning (DRL) policy that re-configure the control settings of uncompromised DER. This trained policy is then deployed locally on controllable DER and determines smart inverter parameter updates based on locally observed information.

DRL has been gaining increasing attention in recent years, including in power systems, for determining control policies for highly complex non-linear systems. In [5], the authors use Deep Q-Network (DQN) learning, a reinforcement learning (RL) algorithm that combines Q-Learning with deep neural networks, to control both generator dynamic braking and load shedding in the event of a contingency to ensure post-fault recovery. In [6], the authors consider the problem of coordinated voltage regulation using capacitors and smart inverters. Exploiting the timescale separation of these devices, they solve a convex optimization problem to determine the control policies of the smart inverters while using a DQN network to learn an optimal policy for capacitor bank switching. In [7], a deep deterministic policy gradient (DDPG) RL agent is used to co-ordinate across DER and directly modulate active and reactive power to regulate the grid voltage during normal operations.

This work differs from those described above in that we focus on developing a supervisory control policy that continuously monitors system conditions and takes action during

This research was supported in part by the Director, Cybersecurity, Energy Security, and Emergency Response, Cybersecurity for Energy Delivery Systems program, of the U.S. Department of Energy, under contract DE-AC02-05CH11231. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsors of this work.

sustained abnormal behavior. This controller, therefore, should not impact an inverters response to normal disturbances, e.g. line-to-ground faults. While the proposed controller design is motivated by the need to respond to cyber-physical attacks, it is agnostic to the cause of the abnormal conditions. Consequently, it can also serve to autonomously re-configure controller settings in the event that an intended action has resulted in abnormalities, for instance, when connecting different microgrids with independently optimized controllers. This paper presents a framework for DRL for smart grid applications and explores the use case of a cyber-physical attack intended to induce oscillatory behavior in the grid voltage.

The remainder of the paper is organized as follows. Section II gives a brief introduction to DRL and the terms that will be used throughout the paper. Section III gives an overview of the power system models and networks used in the study. Finally, Section IV presents the results and Section V summarizes some of the key conclusions.

II. Deep Reinforcement Learning

A. Reinforcement Learning

RL is a branch of machine learning focused on optimal decision making in stochastic environments. The goal of RL techniques is to train an agent (*i.e.* the decision-maker) to interact with an environment in such a way as to maximize a cumulative reward. The environment is usually cast as a Markov Decision Process (MDP), which consists of the following elements:

- A state space, $\mathcal{S}$, containing states observable by the agent;
- An action space, $\mathcal{A}$, containing all the possible actions the agent can execute;
- A state transition function, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$, specifying the probability distribution over the next state s' when an action a is taken at state s;
- A reward function, $\mathcal{R} : \mathcal{A} \times \mathcal{S} \times \mathcal{S} \to \mathbb{R}$, specifying the reward received by the agent when the environment transitions from state s to state s' with action a;
- A discount factor, $\gamma \in [0,1]$, representing the trade-off between immediate and future rewards.

A RL agent learns optimal actions by repeatedly interacting with the environment and assessing the value of resulting rewards, $R_t \in \mathbb{R}$, dependent on the actions taken, $a_t \in \mathcal{A}$, and the states of the environment, $s_t \in \mathcal{S}$. The agent-environment interaction is visualized in Fig. 1. As shown in the figure, the agent takes action a_t following policy π causing a state transition in the environment. The new state, s_t, and subsequent reward, R_t, are observed by the agent and can then be used to update the policy π. The objective of the agent is to maximize the discounted reward $J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{T} \gamma^t R_t\right]$, where T is the terminal time step, by following a policy π which can be deterministic or stochastic in nature.

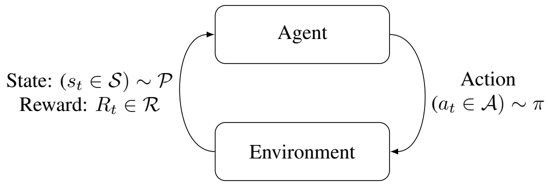
Agent
State: $(s_t \in \mathcal{S}) \sim \mathcal{P}$
Reward: $R_t \in \mathcal{R}$
Action
$(a_t \in \mathcal{A}) \sim \pi$
Environment

Fig. 1: Reinforcement learning loop.

B. Deep Reinforcement Learning

Classical RL relies on feature engineering and is difficult to apply to environments with high dimensional, continuous action and/or state spaces [8]. Such spaces, typically, must be discretized first, leading to a combinatorial explosion in complexity and unreasonable training time (the so-called *curse of dimensionality*). In addition, classical RL has trouble capturing patterns in the presence of noisy or incomplete data. DRL solves these issues by leveraging neural networks with multiple hidden layers that take the agent observations as input and output a policy that determines what action to take in a given state.

With DRL, the inputs to the neural network can be structured data (tabular data), unstructured data (images, text, video), or both. The weights of these neural networks are efficiently learned end-to-end via gradient-based optimization to find the best intermediate features and an optimal output policy. The need for precise feature engineering is then greatly reduced, thanks to the automatic high-dimensional feature extraction of the hidden layers. In DRL, one can use the networks to explicitly approximate an optimal policy distribution, π, over possible actions. This distribution is then sampled by the agent to determine the next action, as in policy gradient methods. They may also be used to approximate either a value function, $V^\pi(s)$, or an action-value function, $Q^\pi(s,a)$, from gathered data, leading to an action decision based on inferred values for all possible future states, as in DQN. The value function, $V^\pi(s)$, is the expected discounted reward when starting in state s and following the policy π, whereas the action-value function $Q^\pi(s,a)$ is defined as the expected discounted reward when starting in state s, taking action a, and then following the policy π thereafter.

Thanks to its flexibility, DRL has been successfully applied to robotic control [9], video games [10], [11] and board game playing [12], [13].

C. Policy Gradient and PPO

Policy gradient methods employ a policy modeled by a neural network which is trained directly by gradient ascent on the expected return. The most basic method (vanilla policy gradient) is simple to implement but has the drawback of having a high gradient variance. In response, Actor-Critic (AC) methods were proposed[14], where another, possibly shared, neural network approximates the value function.

Let $\pi_\theta(a|s)$ be a stochastic policy, parameterized by θ, modeling the probability distribution of action $a \in \mathcal{A}$ given the state $s \in \mathcal{S}$. Let $V_\phi^\pi(s)$ be a value function parameterized by ϕ, estimating the cumulative discounted reward from the current state to the terminal state. The gradient of $J(\theta)$ is:

$$\nabla_\theta J(\theta) = \underset{\tau \sim \pi_\theta}{\mathbb{E}} \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) A_\phi^\pi(s_t, a_t) \right], \qquad (1)$$

where τ is the trajectory generated by policy π_θ and $A_\phi^\pi(s_t, a_t) = R_t + \gamma V_\phi^\pi(s_{t+1}) - V_\phi^\pi(s_t)$ is the advantage estimation, representing how much better taking action a_t is, as opposed to following the policy π when in state s_t. The policy and value function are updated by gradient ascent/descent:

$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta J(\theta), \qquad (2)$$

$$\phi_{k+1} = \phi_k - \beta \nabla_\phi (R_t + V_\phi^\pi(s_{t+1}) - V_\phi^\pi(s_t))^2. \qquad (3)$$

As the training of AC methods can be unstable when the data distribution changes due to a large policy update, the Trust Region Policy Optimization (TRPO) was introduced [15]. TRPO limits the updates in the policy space by enforcing a Kullback–Leibler divergence constraint on the size of each update. A Proximal Policy Optimization (PPO) [16] using a clipped surrogate objective simplifies the aforementioned method and yields similar performance:

$$L^{\text{CLIP}}(\theta) = \hat{\mathbb{E}} \left[\min \Big(r_t(\theta)\hat{A}_t,\ \text{clip}\big(r_t(\theta), 1-\epsilon, 1+\epsilon\big)\hat{A}_t \Big) \right],$$

$$\text{where } r_t(\theta) \triangleq \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \text{ and } \hat{A}_t \triangleq A_\phi^\pi(s_t, a_t)$$

This clip operation encourages a more gradual updates to the policy rather than large changes, and the minimum between the unclipped and the clipped objective is used so that the final objective is a lower bound on the unclipped objective [16]. The hat over the expectation means that we compute a Monte Carlo estimate of it.

PPO is a state-of-the-art method that was successfully used in video games [11] and robotics in simulation [17]. We consider here its application to the control of smart inverters. Before we map the specific problem onto the RL formalism, the following remark is in order:

Remark 1: In many applications, the state of the entire system, s_t, is not directly observed. In this case, the problem falls in the class of Partially Observable MDPs (POMDP). In a POMDP, the additional element in the model is:

- An observation transition function (also called perceptual distribution or emission probability) $\mathcal{V} : \mathcal{S} \times \mathcal{O} \to [0, 1]$ that specifies the probability distribution of the observation o_t given the state s_t.

The policy function in this case takes as input the observation rather than the state, i.e. the goal is to find the optimum $\pi(a_t|o_t)$. As mentioned later, the formulation in this paper falls in the class of POMDP. Also, we note that neither the state transition function nor the perceptual distribution are explicitly given; hence the policy neural network is trained through a Monte Carlo method.

III. Methodology

A. Modeling the DER action space

In response to evolving standards and requirements, DER are increasingly being deployed with the ability to modulate their real and reactive power injection/consumption in response to locally measured grid conditions. In this work we focus specifically on smart inverter Volt-VAR (VV) and Volt-Watt (VW) control functionality as these operating modes are designed to help regulate distribution system voltages in the presence of large amounts of renewable generation. Under VV/VW control schemes, each inverter seeks to modulate active and reactive power injections in response to measured system voltage. The amount by which reactive and active power injections are modulated is governed according to piece-wise linear functions of voltage, often referred to as "droop" curves. Different parameterizations of VV and VW curves exist, however, existing guidelines often depict shapes shown in Figs. 2 - 3, which are parameterized by the five parameters that define the piece-wise linear curves shown, which will be referred to as the components of the setpoint-vector $\eta = [\eta_1, \ldots, \eta_5]$. In this work, the action is a 5×1 vector $a = \Delta\eta \triangleq \eta - \eta^o$, where η^o is the default set of parameters. Note that, even though in principle the action is continuous, we quantize the possible range for the action and search directly for the categorical vector a.

The VV curve injects reactive power when voltages in the system are low and transitions to VAR consumption as voltages increase. The VW curve provides maximum real power injection under most voltage levels, but curtails PV output as voltage levels increase. The additional capacity resulting from active power curtailment can then be used for additional reactive power consumption.

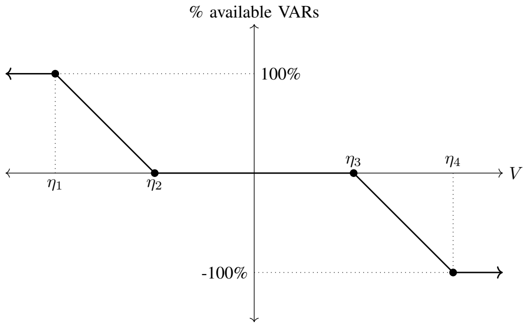

Fig. 2: Inverter Volt-VAR curve. Positive percent of VAR injection.

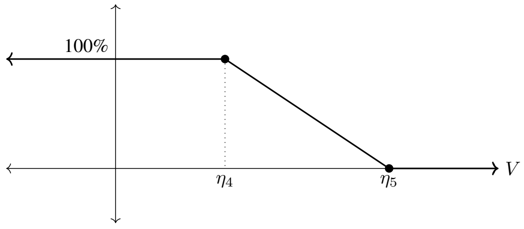

Fig. 3: Inverter Volt-Watt curve. Positive percent of watt injection.

Without loss of generality, we assume all inverters in the subsequent analysis possess both VV and VW functionality. Let p^{max} be the maximum output of the PV unit under presently available solar insolation, and q^{avail} the limit for reactive power in absolute value. In some instances, the amount of reactive power available for injection/consumption may be fixed (in the case of an oversized inverter relative to the capacity of the PV panels) while in others, q^{avail} may depend on the amount of real power being generated from the PV system:

$$q^{\text{avail}} \leq \sqrt{s^2 - f^p(\bar{v})^2}, \tag{4}$$

where s is the inverter capacity. Let $u_{p,i}$ and $u_{q,i}$ denote the active and reactive power control signal of inverter i. They are function of the *averaged* measured voltage magnitude at the bus (c.f. (7a)). Rather than considering completely arbitrary VV and VW mappings $u_{p,i}$ and $u_{q,i}$ that respect the limits $p^{\max}$ and q^{avail}, we seek policies that are expressed as:

$$u_i^p = f_i^p(\bar{v}) \triangleq \begin{cases} p^{\max} & \bar{v} \in [0, \eta_4] \\ \left(\frac{\eta_5 - \bar{v}}{\eta_5 - \eta_4}\right) p^{\max} & \bar{v} \in (\eta_4, \eta_5] \\ 0 & \bar{v} \in (\eta_5, \infty) \end{cases}. \tag{5}$$

$$u_i^q = f_i^q(\bar{v}) \triangleq \begin{cases} q^{\text{avail}} & \bar{v} \in [0, \eta_1] \\ \left(\frac{\eta_2 - \bar{v}}{\eta_2 - \eta_1}\right) q^{\text{avail}} & \bar{v} \in (\eta_1, \eta_2] \\ 0 & \bar{v} \in (\eta_2, \eta_3) \\ -\left(\frac{\eta_3 - \bar{v}}{\eta_4 - \eta_3}\right) q^{\text{avail}} & \bar{v} \in [\eta_3, \eta_4] \\ -q^{\text{avail}} & \bar{v} \in (\eta_4, \infty) \end{cases}, \tag{6}$$

The scheme of (5) - (4) illustrates the combined use of VV and VW control with VW precedence [18]. Under VW precedence, priority is given to the VW controller to determine any needed curtailment before determining the VARs available (q^{avail}). After q^{avail} is fixed, u_i^q is computed from (6).

In the event of a cyber-physical attack we assume that an adversary has the capability to re-dispatch a set of voltage breakpoints $\eta = [\eta_1, \ldots \eta_5]$ that parametrize the droop curves in Figs. 2 - 3 for a subset of DER on the network. Within the context of this work, the remaining set of non-compromised DER can then be updated with new parameters vector $\eta' = a + \eta^o$ to re-shape their own local droop curves to transition the system voltages to a *safe* region, devoid of oscillatory behavior.

Finally, the structure of the DER VV and VW control dynamic response, similarly to [18]–[20], includes the following first order low pass filters that average the input voltage and determine the active and reactive power injections:

$$\bar{v}_{i,t} = \bar{v}_{i,t-1} + \tau_i^m \big(v_{i,t} - \bar{v}_{i,t-1}\big), \tag{7a}$$

$$p_{i,t} = p_{i,t-1} + \tau_i^o \big(f_i^p(\bar{v}_{i,t}) - p_{i,t-1}\big), \tag{7b}$$

$$q_{i,t} = q_{i,t-1} + \tau_i^o \big(f_i^q(\bar{v}_{i,t}) - q_{i,t-1}\big), \tag{7c}$$

where $\bar{v}_i$ denotes a low-pass filtered measured of the voltage magnitude, v_i, at node i, τ_i^m is its associated measurement time constant, τ_i^o is the output filter time constant and $f_i^p(\bar{v}_{i,t})$ and $f_i^q(\bar{v}_{i,t})$ are the piecewise linear functions of the measured nodal voltages for node i given by (5) and (6) respectively. Note the equilibrium of (7b) - (7c) is given by (5) - (6).

The stability of (7a) - (7c) has been studied in [19] and [21], where it has been observed that instabilities manifest as oscillations in inverter power injections and nodal voltages.

As said before, the RL agent indirectly manipulates the outputs of the inverter by modifying the vector of parameters $\eta_t = a_t + \eta^o$. Next we define a component of the observation o_t used as an input to the DRL controller in our POMDP formulation. The quantity is a local measure of the presence and severity of voltage magnitude aforementioned oscillations.

B. A Measure of Unstable Oscillations

We propose the use of a simple filter to determine the "energy" associated with voltage oscillations in the distribution grid. The filter consists of the series of a highpass filter, and an energy detector, consisting of a square-law, followed by a lowpass filter. A discrete time block diagram of the process is shown in Fig. 4

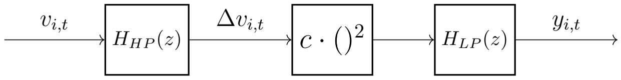

Fig. 4: Block diagram of instability detector using a transfer function representation of high and low pass filters.

where H_{HP} and H_{LP} are high-pass and low-pass filters respectively, realized using a bilinear transform equivalent of a first-order high/low-pass filter, and c is a positive gain. The high-pass filter removes DC content from $v_{i,t}$, yielding $\Delta v_{i,t}$. This signal is then squared to produce a DC term which is then extracted via low pass filtering. The output signal, y_i, t is a measure of the intensity of the instability. The filter parameters should be chosen such that the filter does not attenuate oscillations due to inverter instabilities.

C. DER Cyber-Attack Mitigation as a RL problem

The primary goal of the DRL controller is to mitigate instabilities introduced by DER smart inverter VV/VW controllers due to maliciously chosen set-points. Let the graph $G = (\mathcal{N}, \mathcal{L})$ represent the topology of the distribution feeder considered, where $\mathcal{N}$ is the set of nodes of the feeder (with 0 indexing the feeder head) and $\mathcal{L}$ is the set of lines. For simplicity of presentation, we assume the presence of a VV/VW capable smart-inverter at every node in the system, so that the total number of inverters in the system is $|\mathcal{N}|$. We suppose the set $\mathcal{N}$ is partitioned into two sets, $\mathcal{H}$, and $\mathcal{U}$, where $\mathcal{H} \bigcup \mathcal{U} = \mathcal{N}$ which represent the "compromised" and '""uncompromised" inverters respectively. Furthermore we assume that $\mathcal{U} \neq \emptyset$, i.e. we have some controllable resources to mitigate the effects of the cyber-physical attack. Given $\mathcal{U} \subsetneq \mathcal{N}$ and the temporal dependency of load and solar irradiance, as mentioned in Remark 1, the model is a POMDP where we wish to determine the optimum stochastic policy, $\pi_\theta(a|o)$, parameterized by the neural network parameters θ, modeling the probability distribution of action $a \in \mathcal{A}$ given the observation $o \in \mathcal{O}$.

Training: Rather than training multiple agents simultaneously, we adopt the following heuristics to aid convergence:

1) For agent training, we define a single agent whose input observation vector is the mean of the input observation vectors of all controllable inverters $\in \mathcal{U}$ and whose action, a_t, is a deviation/offset, $\Delta\eta$, from default VV/VW control curves that apply across inverters.

2) Once a single agent has been trained, this agent optimal policy is deployed locally on each individual inverter and only acts on local observations.
3) Rather than optimize over arbitrarily shaped VV/VW curves ($f^q(\bar{v})$ and $f^p(\bar{v})$), we optimize over the deviation, i.e. $a = \Delta\eta$, from the default parameters defining the curves in Figs. 2 - 3. An example of this is shown in Fig. 5. The translation is in range from -0.05 pu to 0.05 pu around an inverters default VV/VW curve, with the action space being discretized into k bins.
4) New parameterizations of VV/VW functions will be chosen so that measurement and power injection dynamics evolve on a faster timescale. This choice will preserve the Markov property between actions taken by the RL controller.

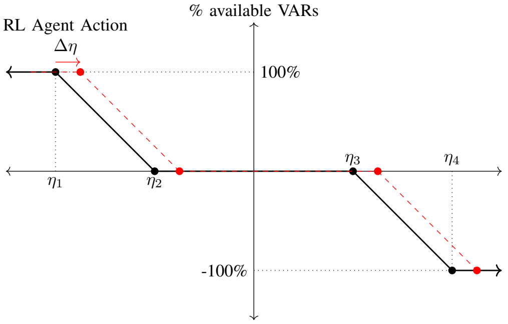

Fig. 5: Action example.

Observation: The observation vectors $o_{i,t}, i \in \mathcal{U}$ at each RL agent (i.e. the input to the neural network that learns the optimum policy $\pi(a|o)$), consist of:

1) $y_{i,t}$: the mean of the estimation of voltage oscillation energy at node i since the last agent environment interaction.
2) $y^{\max}_{i,t}$: the maximum of $y_{i,t}$ over the previous n environment interactions. This is a tunable parameter that stores information of the recent oscillation energy.
3) $q^{\text{avail, nom}}_{i,t}$: the available reactive power capacity without active power curtailment.
4) $a^{\text{one-hot}}_{i,t-1}$: one-hot encoding of the previous action taken by the agent.

Reward: At a timestep t, the reward function, $R_t(a_t, o_t)$ is:

$$R_t = -\Bigg(\frac{1}{|\mathcal{U}|}\sum_{i=1}^{|\mathcal{U}|}\sigma_y y_{i,t} + \sigma_a \mathbb{1}_{a_t \neq a_{t-1}} + \sigma_0 \|a_t\|_2 + \frac{1}{|\mathcal{U}|}\sum_{i=1}^{|\mathcal{U}|}\sigma_p\left(1 - \frac{p_{i,t}}{p^{\max}_{i,t}}\right)^2\Bigg). \quad (8)$$

The first component seeks to minimize the voltage oscillation y; the second one penalizes configuration changes on inverters; the third component encourages the agent to use the default inverter configurations in the absence of voltage oscillations and the final component penalizes any active power curtailment.

D. The PyCIGAR DRL Environment

Any learning method requires sufficient training over a variety of scenarios. As is done in other application of deep learning in the context of critical infrastructure systems, such training can be performed through realistic Monte Carlo simulations that cover a variety of operating conditions and cyber-physical attacks. We named PyCIGAR the modular software architecture we designed to train the DRL agent described in the previous section.

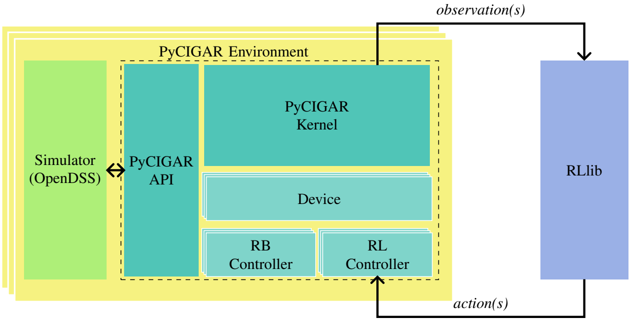

Fig. 6: PyCIGAR Architecture.

PyCIGAR is a Python library for distributed reinforcement learning for electric power distribution grids on quasi-static time scales. The library provides a link between power system simulators and a reinforcement learning library - RLlib [22]. PyCIGAR is a unified API that can interface different power system simulators (e.g. OpenDSS), while on the RL side PyCIGAR uses RLlib in order to deploy large scale experiments on a server, machine cluster or on a cloud.

A diagram of the PyCIGAR architecture is shown in Fig. 6. In addition to RL-based controllers, PyCIGAR also includes rule-based (RB) control devices (e.g. tap-changing transformers) and can easily be extended to support the integration of other more complicated DER (e.g. electric vehicle charging and battery storage systems). PyCIGAR provides a foundation for the rapid development of learning-based control algorithms for heterogeneous classes of DER in electric power distribution grids.

IV. Results

We conduct experiments on the IEEE 37-bus feeder with all load buses having an active power generation of 50% of the nominal load with an additional 10% inverter over-sizing for reactive power headroom. The agent training environment consists of 700 one-second timesteps per simulation. At the end of each experiment, the training environment is reset with randomized load and solar generation profiles and percentage of compromised inverters. This diversity creates a rich environment that exposes the RL agent to attacks that could occur anytime throughout the day under a variety of loading, solar conditions and proportions of compromised inverters. For each case, all inverters start with their default VV/VW settings and at a particular time in the simulation the attacker gains controls of 15% to 50% of the installed inverter capacity at each node to create a voltage instability. It does so by translating the VV/VW curves and steepening the slopes to induce an

The name stands for Python based Cybersecurity via Inverter-Grid Automatic Reconfiguration.

oscillation. This attack vector represents a subset of possible attack vectors. The agent is allowed to reconfigure the VV /VW curves of non-compromised DER to mitigate oscillations that result from the cyber-attack. We consider two types of action, 1) translating the entire VV/VW piecewise functions from its default configuration (offset action) and 2) adjusting the slope of the piece wise function in the region $\in (\eta_1, \eta_2]$ and $\in (\eta_3, \eta_4]$ (slope action). Within the simulation, the agent receives observations and updates inverters' functions f_i^q and f_i^p every 35 seconds. The training is conducted on an Intel® Xeon® E5-2623 v3 processor, 64GB RAM server and takes 1 hour of training time to converge.

Fig. 7 shows the baseline case caused by a 45% percentage attack around noon with no action taken to mitigate the result of the attack. The attack creates oscillations in system voltages that are detected by the oscillation detector (see Section III-B). The malicious re-dispatch of settings are shown in the action subplot and the components of the reward function, (8), are shown at the bottom. In the absence of an control the reward is solely composed of the penalty for the oscillation.

Fig. 8 - 9 show the behavior from the trained RL agent at a random node in the network in mitigating instabilities from compromised DER at two different times of day and different percentage of compromised DER. At simulation time $t = 200\,\mathrm{s}$ the attack is introduced in a portion of DER. This can be seen in the action subplot, which shows the breakpoints of the piecewise linear curves of 2 - 3 being suddenly moved to a new configuration. This triggers an oscillation in grid voltages. The output of the *oscillation observer* is both fed into the RL agent as an observation and included as a negative penalty in the reward function. The agent, therefore, should control non-compromised assets to minimize the oscillation. This is what occurs, as the agent changes the breakpoints of non-compromised units just after $t = 250\,\mathrm{s}$ by translating the default VV/VW curves. This action almost immediately stops the oscillation in the system voltages, resulting in a defeat of the original cyber-attack. Fig. 8 features an attack in the morning, around 9am, where there is significant excess capacity for reactive power compensation available for the agent to mitigate the cyber-attack. The agent, therefore, does not need to curtail active power generation to successfully mitigate the attack. This, however, is not the case in Fig. 9 where the attack occurs around midday and the agent is forced to curtail active power in order to have enough controllability to defeat the attack. Across numerous training configurations it was observed that RL offset was the preferred action of the agent.

Also worth highlighting is the behavior of the agent after the compromised DER have been identified and returned to their original settings, at $t = 450\,\mathrm{s}$. Shortly after, we observe the agent also returning to its original configuration. In demonstrating this behavior, it can be seen that the RL agent will take steps to ameliorate the effects of the cyber attack, but will return a state of inactivity once the threat of the attack has passed.

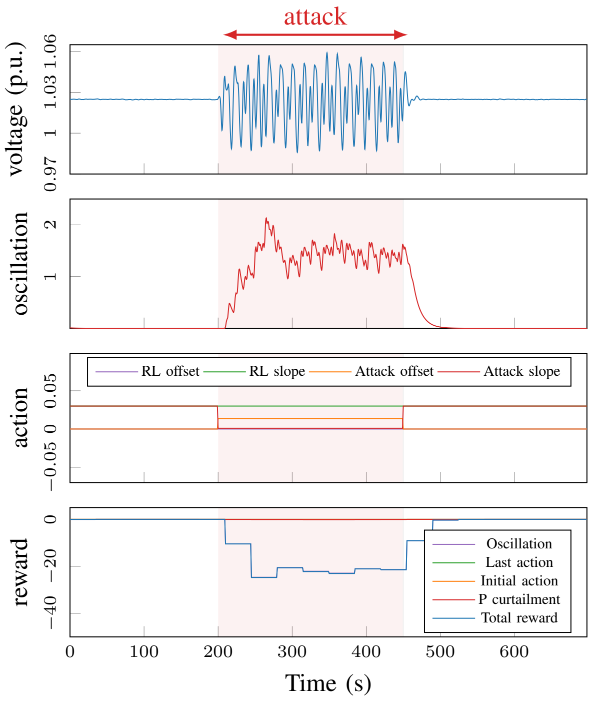

Fig. 7: Result of an evaluation episode at 45% attack without agent defense

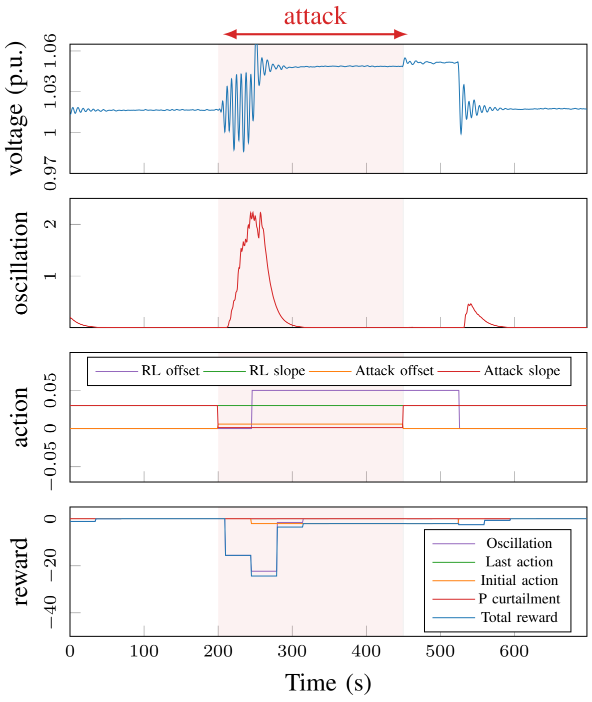

Fig. 8: Result of an evaluation episode at 20% attack around 9AM

V. Conclusions

This paper has proposed a reinforcement learning approach for mitigating the oscillation due to unstable smart inverter settings by training an agent to translate the VV/VW curves. The resultant policy successfully mitigated adversary induced

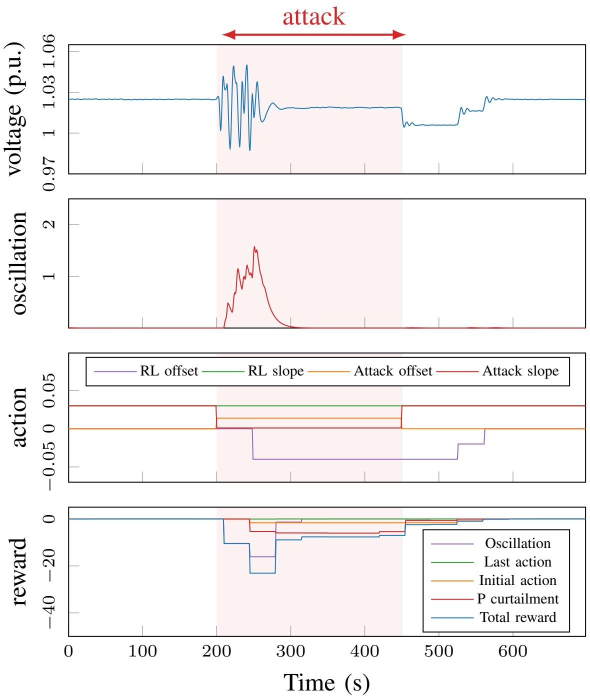

Fig. 9: Result of an evaluation episode at 45% attack around noon

voltage oscillatory behavior for the cases considered.

Future work will investigate the value of this approach for larger networks and the sensitivity of the trained agents to specific network topologies/configuration. Additionally, we will explore different neural network architectures, including Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM), which have proven to be the state of the art in solar and load forecasting and may improve the performance of the agent. Additional types of attacks will also be considered, including, but not limited to, voltage imbalance attacks. An adversary may seek to exploit DER interaction with utility voltage regulation systems to create system voltage imbalances, leading to device trips and possible system collapse.

References

[1] J. Qi, A. Hahn, X. Lu, J. Wang, and C.-C. Liu, "Cybersecurity for distributed energy resources and smart inverters," *IET Cyber-Physical Systems: Theory & Applications*, vol. 1, no. 1, pp. 28–39, 2016.

[2] S. Sahoo, T. Dragičević, and F. Blaabjerg, "Cyber security in control of grid-tied power electronic converters–challenges and vulnerabilities," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, 2019.

[3] "Modernizing Hawai'i's Grid For Our Customers," Tech. Rep., 2017.

[4] "IEEE Standard 154^{TM} — Communications and Interoperability: New Requirements Mandate Open Communications Interface and Interoperability for Distributed Energy Resources," Tech. Rep., 2017.

[5] Q. Huang, R. Huang, W. Hao, J. Tan, R. Fan, and Z. Huang, "Adaptive power system emergency control using deep reinforcement learning," *IEEE Transactions on Smart Grid*, 2019.

[6] Q. Yang, G. Wang, A. Sadeghi, G. B. Giannakis, and J. Sun, "Two-timescale voltage control in distribution grids using deep reinforcement learning," *IEEE Transactions on Smart Grid*, 2019.

[7] C. Li, C. Jin, and R. K. Sharma, "Coordination of pv smart inverters using deep reinforcement learning for grid voltage regulation," *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pp. 1930–1937, 2019.

[8] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[9] O. Andrychowicz, B. Baker, M. Chociej, R. Józefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 11 2019.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning (2013)," *arXiv preprint arXiv:1312.5602*, vol. 99, 2013.

[11] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse *et al.*, "Dota 2 with large scale deep reinforcement learning," *arXiv preprint arXiv:1912.06680*, 2019.

[12] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, p. 484, 2016.

[13] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.

[14] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International Conference on Machine Learning*, 2016, pp. 1928–1937.

[15] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, 2015, pp. 1889–1897.

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[17] N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. Eslami *et al.*, "Emergence of locomotion behaviours in rich environments," *arXiv preprint arXiv:1707.02286*, 2017.

[18] B. Seal, "Common Functions for Smart Inverters, 4th Ed." Electric Power Research Institute, Tech. Rep. 3002008217, 2017.

[19] M. Farivar, L. Chen, and S. Low, "Equilibrium and dynamics of local voltage control in distribution systems," in *52nd IEEE Conference on Decision and Control*, Dec 2013, pp. 4329–4334.

[20] J. H. Braslavsky, L. D. Collins, and J. K. Ward, "Voltage stability in a grid-connected inverter with automatic volt-watt and volt-var functions," *IEEE Transactions on Smart Grid*, vol. PP, no. 99, pp. 1–1, 2017.

[21] K. Baker, A. Bernstein, E. Dall'Anese, and C. Zhao, "Network-cognizant voltage droop control for distribution grids," *IEEE Transactions on Power Systems*, vol. 33, no. 2, pp. 2098–2108, 2018.

[22] E. Liang, R. Liaw, R. Nishihara, P. Moritz, R. Fox, K. Goldberg, J. Gonzalez, M. Jordan, and I. Stoica, "RLlib: Abstractions for distributed reinforcement learning," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. Stockholmsmässan, Stockholm Sweden: PMLR, 10–15 Jul 2018, pp. 3053–3062.

Appendix

Hyperparameter	**Value**
α (learning rate)	1×10^{-3}
γ (reward discount factor)	0.5
λ (GAE parameter)	0.95
ϵ (PPO clip param)	0.1
batch size	420
activation function	tanh
network hidden layers	dense (64, 64, 32)
σ_y (oscillation penalty)	15
σ_a (action penalty)	0.05
σ_0 (penalty for deviation from default VV/VW curve)	18
σ_p (penalty for curtailing active power)	80
action range	-0.05 pu to 0.05 pu
k (action range discretization)	0.01 p.u.

TABLE I: Hyperparameters of the network, training and reward