Deep Reinforcement Learning for Electric Transmission Voltage Control

Brandon L. Thayer, *Member, IEEE,* Thomas J. Overbye, *Fellow, IEEE,*

***Abstract*—Today, human operators primarily perform voltage control of the electric transmission system. As the complexity of the grid increases, so does its operation, suggesting additional automation could be beneficial. A subset of machine learning known as deep reinforcement learning (DRL) has recently shown promise in performing tasks typically performed by humans. This paper applies DRL to the transmission voltage control problem, presents open-source DRL environments for voltage control, proposes a novel modification to the "deep Q network" (DQN) algorithm, and performs experiments at scale with systems up to 500 buses. The promise of applying DRL to voltage control is demonstrated, though more research is needed to enable DRL-based techniques to consistently outperform conventional methods.**

***Index Terms*—Machine learning, artificial intelligence, automatic voltage control, power system simulation**

I. Introduction

THE electric power grid is critical to the functioning of our modern society, and is undergoing a period of major change. Large portions of the U.S. electric power system have undergone deregulation since the 1990's, distributing the responsibilities of electric power generation, transmission, and distribution to separate entities and opening up power markets [1]. Additionally, electricity generation from intermittent renewable resources such as wind and solar is rising while traditional generation sources such as coal and nuclear are declining [2]. Due to numerous factors including increasing extreme weather events, the reliability of the electric grid has been declining in recent years [3]. Further complicating these challenges are current and upcoming labor shortages in the electric power industry [4], [5].

Fortunately, the digital computing revolution has been a boon to the electric power system. Phasor measurement units (PMUs) have been increasing visibility into the electric power system, "smart" meter installations are on the rise, and energy management systems (EMS) are continually improving. Despite recent advances, many grid control actions are still taken by humans [6], [7], [8], [9]. As grid operation becomes increasingly complex, the automation of grid control is more important than ever.

To ensure secure grid operation, all buses in the transmission system are kept within a prescribed voltage band. Bus voltages are maintained by a variety of devices through a mixture of human and automatic control. The most prevalent voltage control devices are generators, switched shunts (*e.g.*, capacitors), and on-load tap changing (OLTC) transformers [10]. Generators typically follow a pre-determined voltage schedule created by the transmission or system operator, and voltage control is achieved by modulating reactive power output. Capacitors and voltage regulators may operate automatically based on local measurements, be directly controlled by human operators, and/or be controlled by a centralized optimization program [6], [10], [11], [12], [13]. Recent advances in system-wide transmission voltage control leverage hierarchical control schemes and the use of "pilot" (bellwether) buses in pre-determined voltage control areas [12], [13], [14]. While effective, these voltage control schemes make major modeling and control simplifications (*e.g.*, linearization) due to their reliance on conventional optimization techniques. The large scale and complexity of the transmission system makes this optimization time consuming, which can render these techniques impractical when the need for rapid control decisions arises.

This paper presents the application of deep reinforcement learning (DRL) for automating transmission voltage control without requiring power system modeling simplifications. Open-source DRL environments were created, a novel modification to a popular DRL algorithm was made, and extensive experimentation was performed with 14, 200, and 500 bus test systems. The remainder of the paper is organized as follows: Section II provides background on DRL and its application to power system problems; Section III describes the DRL environments created for this work, the DQN algorithm used, and a novel algorithm modification that led to improved performance; Section IV describes random and graph-based algorithms for comparison with DRL; Section V presents extensive experimentation with the IEEE 14 bus system; Section VI tests DRL scalability to larger power systems; and Section VII concludes the work.

II. Background and Prior Work

This section provides background on reinforcement learning and its application to power system control.

A. Reinforcement Learning (RL) and "Deep" RL (DRL)

Reinforcement learning (RL) is a form of learning in which an agent receives rewards or penalties for performing actions which affect its environment. Based on these rewards or penalties, an agent can potentially learn how to maximize its rewards according to the Bellman equation [15]. Q-learning is a model-free RL algorithm where agents attempt to learn the action-utility function (Q-function) that provides the expected

This research was supported by funding from the Texas A&M Engineering Experiment Station (TEES) Smart Grid Center (SGC). B. L. Thayer is with Pacific Northwest National Laboratory located in Richland, WA, USA. T. J. Overbye is with Texas A&M University located in College Station, TX, USA.

utility (Q-value) of performing a particular action given the state of the environment. By learning the Q-function, the agent can then make decisions so as to optimize its cumulative future rewards. The learning of the Q-function often takes the form of a table where each entry represents the value of an action given the state of the environment [15]. This lookup table approach doesn't scale well to environments with large state and/or observation spaces, so recent advances use neural networks as an estimator of the Q-function. States are passed to the input layer of a neural network and an estimate of the expected utility of each possible action is emitted from the network's output layer. RL algorithms that leverage neural networks as Q-function estimators are collectively known as "deep reinforcement learning" (DRL) algorithms and have been proven successful in domains with large state and action spaces such as Atari video games and the board game Go [16], [17], [18].

RL training and testing are broken up into "episodes" and "time steps," wherein an episode consists of a discrete number of time steps. At each step, the agent receives an observation from the environment and a reward pertaining to their last action, and subsequently takes their next action. The action impacts the state of the environment, and the process continues. Episode initialization represents a "reset" of the environment, and episodes are typically terminated when the agent succeeds, fails, or exceeds a preset number of time steps.

B. Reinforcement Learning for Power System Control

The use of RL in the power system domain had been stymied by the large scale of states and control (actions) in the power system [19]. As DRL techniques have evolved and proven capable of operating in environments with larger state and action spaces, interest in RL's potential for power system control has increased. Reference [20] provides a literature survey of RL applied to electric power system control. Researchers have begun investigating RL for a variety of power system control problems including transient generator angle stability, congestion management, economic dispatch, and voltage control [20]. In [20], it was suggested that the recent success of DRL may warrant a revisiting of previous grid control work where RL was applied before recent breakthroughs in the DRL domain.

One example of successfully revisiting grid control problems with new DRL algorithms is [21]. The work in [21] leveraged the open-source reinforcement learning environment framework known as Gym [22] as well as recently published open-source DRL algorithms [23], both created by OpenAI, in order to perform grid emergency control. OpenAI's Gym and DRL algorithms are discussed in more detail later on. In [21], two grid control problems were investigated: dynamic generator braking and under-voltage load shedding. It was shown that reinforcement learning agents could be successfully trained both to apply a resistive generator brake in order to prevent the loss of generator synchronism and to shed the minimal amount of load required while maintaining a particular voltage recovery envelope. While the work presented in [21] is quite promising, the studies were performed using small test systems (10 and 14 buses).

The problem of power system reactive power/voltage control involves finding a solution to the power flow equations wherein all bus voltages stay within a prescribed band while reactive power reserves remain available for emergencies. Additional details pertaining the power flow equations can be found in [24]. In [25], traditional, tabular Q-learning was applied to reactive power/voltage control. The system states were simplified by using a binary representation: if a component was within its operating limits (generator reactive power limits, transformer power flows, and bus voltages), the corresponding component of the state vector was 0. If a component was outside its operating limits, the corresponding state vector component was -1. The discrete action space consisted of transformer tap positions, shunt switch positions, and generator voltage set points. It was shown that Q-learning could be used to successfully determine control settings to reduce violations in 14 and 136 bus test cases.

C. The "GridMind" Voltage Control Experiment

Similarly to [25], [26] presented the application of reinforcement learning to reactive power/voltage control. However, newer DRL algorithms were used instead of tabular Q-learning. The reinforcement learning environment and agent in [26] were collectively referred to as "GridMind." As the work in this paper significantly expands upon the findings of [26], an extended description of [26] is presented in this section.

In [26], per unit (p.u.) bus voltage magnitudes were used as observations for the deep reinforcement learning agent, and each individual action represented a set of voltage set points for all available generators (*i.e.*, each action takes the form $\{v_{g_1}, v_{g_2} \dots v_{g_n}\}$ where v_{g_1} is the voltage set point for generator 1 and so on). Generator voltage set points were discretized into the set $\{0.95, 0.975, 1.0, 1.025, 1.5\}$. Note that this action definition leads to an action space with $n_v^{n_g}$ available actions, where n_v is the number of discrete voltage set points and n_g is the number of generators. The presented reward scheme gave a reward of $+100$ if all bus voltages were within $[0.95, 1.05]$ p.u., a penalty of -50 if any single bus voltage fell in either $[0.8, 0.95)$ or $(1.05, 1.25]$ p.u., or a penalty of -100 if any single bus voltage was < 0.8 or > 1.25 p.u. At the end of each episode, the agent received an additional reward (or penalty) equal to the mean of all rewards for the episode in question.

The experiments performed in [26] used the IEEE 14 bus test system, which can be found at [27]. The generators at buses 1 and 2 were considered to be available for active power dispatch, and the remaining three generators were available for voltage support only. Training episodes were created by randomly varying individual load levels between 80% and 120% of nominal without changing power factor. Tests were carried out with and without single line contingencies. The generators at buses 1 and 2 used participation factor control to adjust their active power set points. Episodes were terminated if all bus voltages fell within $[0.95, 1.05]$ p.u., the power flow failed to converge, or the agent reached a pre-determined per-episode action cap.

It's worth noting that in the absence of contingencies (all lines/transformers in service), loading levels of 80%-120%

never result in low voltage conditions (< 0.95 p.u.) for the 14 bus system. In fact, at maximum loading (all loads at 120% of nominal), the lowest voltage in the system is approximately 1.01 p.u. Conversely, the IEEE 14 bus base case has three generators set above the maximum acceptable voltage of 1.05 p.u. Thus, every episode begins with bus over-voltages.

The primary results presented in [26] were episode rewards as training progresses: at first, the agent did not earn very high rewards, but by the end of training, the agent could consistently earn high rewards and take very few actions.

The work presented in this paper builds on the methodology of [26], and the experiments from [26] were reproduced as diligently as possible. Experiments were performed both with and without single line contingencies. All of the following discussion in this paragraph pertains to the reproduction of [26], and code can be found at [28], [29]. Agents were trained for 500,000 simulation steps, and testing was performed on 5000 testing episodes that were not present in training. Without contingencies, the DRL agent could successfully bring all voltages into the acceptable band in 100% of testing episodes. However, upon examining the actions taken by the agent in testing, it was found that only two of the available 3125 actions ($n_v = 5$, $n_g = 5$, $5^5 = 3125$) were utilized. In other words, the agent learned two sets of voltage set points that worked for every scenario. With single line contingencies included in each episode's initialization, the DRL agent was able to successfully bring all voltages in band in 83.8% of testing episodes. Upon investigating the specific sequences of actions taken during testing, it was found that the agent can exhibit cyclic or repetitive behavior if it does not succeed in its first action. In other words, at times the agent took the same single action repeatedly until an episode terminated, and other times the agent would repeatedly cycle through identical sequences of actions until episode termination. Section III-C presents a novel algorithm modification that addresses this cyclic and repetitive behavior.

III. DRL Algorithms and Environments for Voltage Control

This section contains details pertaining to the DRL algorithm used, enhanced environments created, and proposed algorithm modification for applying DRL to voltage control.

A. Overview and DRL Algorithm Details

Fig. 1 presents a conceptual flowchart depicting DRL training for voltage control. Episode initialization (Fig. 1-a) can be adjusted to make an environment more or less "challenging." Note that Fig. 1 does not depict the inner workings of the DRL algorithm itself (Figs. 1-d, 1-e), and the reader can find these details in [16], [23], [29], [30], [31], [32], [33].

A software package was developed in the Python programming language for interfacing with PowerWorld Simulator, which was used to solve the power flow (Figs. 1-b, 1-f). This Python package, named EasySimAuto (ESA), significantly simplifies interfacing with Simulator's application programming interface (API) [34].

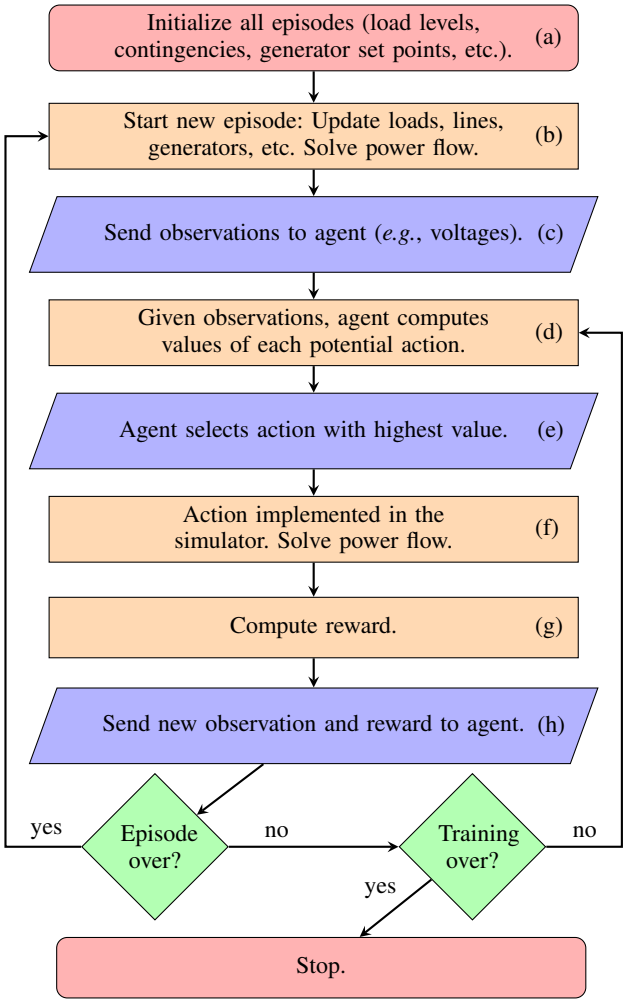


Fig. 1: Conceptual Flowchart: DRL for Voltage Control

A Python package called "Gym," created by OpenAI, is an open-source toolkit for developing and comparing RL algorithms, and is commonly used in RL research as a way to standardize RL environment development [22]. For the work presented here, a set of several environments were constructed, and the software repository can be found at [28]. The environments use ESA to send commands to and retrieve data from the power flow simulator (Figs. 1-b, 1-c, 1-f, 1-h). The environment itself is additionally responsible for initialization (Fig. 1-a) and reward computation (1-g).

A collection of high-quality DRL algorithm implementations is provided by [23] (also from OpenAI). An improved and documented version of the algorithms in [23] is provided by [30], and was leveraged in this work. Specifically, an advanced DRL algorithm with so-called "deep Q networks" (DQN) originally presented in [16] was used. Several modifications/improvements to the algorithm have been published over the years including "dueling DQN" [31], "double-Q learning" [32], and prioritized experience replay [33]. The work here used all the aforementioned DQN algorithm improvements. DQN algorithms require discrete action spaces, so continuous control elements such as generator voltage set points must be discretized. The DQN algorithm with all

improvements has many different hyper parameters which can be tuned. In this work, the majority of hyper parameters were left at their default values, and are explicitly defined in [29].

B. DRL Environment Details

As discussed in Section II-C, the GridMind experiment varied loads between 80% and 120% of nominal, varied generator active power output linearly from the base case settings, and used constant generator voltage set points for all episodes. By contrast, this section presents details on a significantly more challenging environment which can be used to test the limits of DRL applied to transmission voltage control. The environment described here was used along with the DQN algorithm and all improvements (see Section III-A) to generate the results presented in Sections V and VI.

1) Episode Initialization Overview: The design of how episodes (scenarios) are created can have a major impact both on a reinforcement learning agent's ability to learn and on the ultimate usefulness of a trained agent. For instance, training under a narrow range of load and generator conditions may enable an agent to learn the best control actions quickly, but the agent's decision making may not generalize well to other conditions not encountered in training.

2) Episode Initialization - System Loading: For each scenario, total system active power loading is drawn from the uniform distribution between 60% and 140% of the base case's total active power loading. Next, a value from the uniform distribution on the interval $[0, 1)$ is independently drawn for each load in the system. These values are subsequently summed and then linearly scaled such that after scaling the values sum to one. After this scaling, the new values for each load represent the load's fraction of total system active power loading (P). In this way, each load can theoretically take on a value between 0% and 100% of the given episode's total system loading (though the extremes are incredibly unlikely to occur). Next, each load has a power factor (pf) drawn from the uniform distribution on the interval $[0.8, 1.0)$, and reactive power levels (Q) are computed via the relationship $Q = P \cdot \tan(\arccos(pf))$. Finally, each load has a 10% chance for its power factor to be changed from lagging to leading (flipping the sign of Q).

3) Episode Initialization - Generator Active Power: After computing system loading for each scenario, the generator commitment can be determined and active power levels can be dispatched to meet demand. Since generators have active power output minimum and maximum allowable values, the procedure for determining generation levels differs somewhat from the procedure for determining individual loads. The following description is functionally equivalent to what is done in the environment code [28], but is explained in a simplified manner. The actual code is completely vectorized.

For each scenario, a random ordering of all generators in the case is drawn. Then, the generators are looped over in the given random order, and an active power output is drawn from the uniform distribution between the particular generator's P_{min} and P_{max}. This process continues until the total active power output of the generators meets or exceeds the total loading for the given scenario, plus assumed losses of 3%. In this way, different generators may be active for each scenario, effectively building in generator contingencies and creating different unit commitments for each scenario. Assuming some losses are present ensures the slack generator must not cover all active power losses, which can be significant depending on the number of lines and their resistance.

4) Episode Initialization - Generator Voltage Set Points: Random voltage regulation set points are drawn uniformly for each episode and each generator from the set $\{0.95, 0.975, 1.00, 1.025, 1.05\}$ p.u.

5) Episode Initialization - Lines and Shunts: For cases which contain shunts, initial shunt states (open/closed) are simply randomly drawn from the uniform distribution for all shunts and all episodes. A single branch (line or transformer) is randomly removed from service for each episode.

6) Observation Design: Engineering the observations given to the DRL agent is critical to successful learning, and requires significant experimentation. It's important to ensure the appropriate amount of information is given to the agent - too little information and the agent may fail to learn due to a lack of observability, while too much information can significantly scale up the size of the neural network required for DRL and cause a failure to learn due to increased difficulty in finding relationships between features.

There are many different options available for configuring the environments described in this work [28]. The following observation combinations are used in Sections V and VI:

- Bus voltage magnitudes only
- Bus voltage magnitudes and gen. states (on/off)
- Bus voltage magnitudes and branch states (open/closed)
- Bus voltage magnitudes, gen. states, and branch states
- Bus voltage magnitudes, gen. states, branch states, and shunt states (open/closed)

All of the observation combinations above can also be used with transformed bus voltage magnitudes. In these cases, voltages are transformed via "min-max" scaling on the interval $[0, 1]$. The environments consider a power flow to be "failed" if any single bus voltage is < 0.7 p.u. or > 1.2 p.u., even if the simulator is able to solve the power flow. In this way, the absolute lower and upper voltage bounds are known ahead of time for all scenarios/episodes, enabling min-max scaling.

7) Action Space: As mentioned in Section II-C, the action space presented in [26] scales as $n_v^{n_g}$. If voltage set points are discretized into five settings and five generators are present (as in the IEEE 14 bus system), the action space has dimension 3125. This clearly does not scale well to larger systems. By contrast, the environments in this work map each voltage setting for each generator to a single action, resulting in an action space that scales as $n_v \cdot n_g$. For systems with switchable shunts (namely capacitors), a single action is available per shunt that toggles the shunt state (open/closed). In all cases, a "no-op" action is included which does not lead to any change in the system.

8) Reward Design Overview: In some of the recent RL for power systems literature, agents were primarily rewarded based on the post-action state of the system, rather than the *change* that the given action *induced* in the system [21], [26].

If the objective is to bring all bus voltages within the range $[0.95, 1.05]$ per unit, an action that moves a bus voltage from 0.93 to 0.945 p.u. should be rewarded despite the voltage not moving into the acceptable band, while an action that moves a bus voltage from 0.96 to 0.94 p.u. should be penalized. Two movement-based control schemes are proposed in this work. Readers seeking details beyond what is presented in the following two sections can consult [28].

9) Reward Scheme 1: This reward scheme provides rewards for moving voltages in the right direction (toward the acceptable band), while providing penalties for voltages that move in the wrong direction (away from the acceptable band). The movement rewards and penalties are scaled by the movement magnitude so that a larger voltage movement obtains a larger reward or penalty. Additionally, a penalty is given for taking an action in order to help incentivize the agent to minimize the number of actions taken.

10) Reward Scheme 2: This scheme keeps all rewards within the range $[-1, +1]$, inspired by the reward "clipping" done in [16], [17]. In [16], [17] it was noted that keeping rewards in a fixed band "limits the scale of the error derivatives" and thus makes the reward function useful across multiple games. In the case of the work presented in this paper, the same notion applies, but instead helps the reward scheme generalize across power system cases. The clipped scheme presented here has the following discrete reward possibilities: $\{-1.00, -0.75, -0.50, -0.25, -0.10, 0.00, +0.25, +0.50, +0.75, +1.00\}$. A reward of -1 is given if the power flow diverges, and a reward of $+1$ is given if all bus voltages are in band. Other rewards (penalties) are given based on the net number of buses that move in the right (wrong) direction.

C. DQN Algorithm Modification

As mentioned in Section II-C, the unmodified DQN algorithm can select the same action multiple times in a given episode. One novel contribution of this work is the modification of the DQN algorithm such that each action is only permitted to be taken once per episode. If, during the course of training or testing, the agent determines an action which has already been taken in the current episode to have the highest value, the action with the second highest value is chosen instead. This process continues until an action which has not been taken yet is selected.

The aforementioned algorithm modification is useful in training as the agent is forced to perform additional exploration. Additionally, this modification makes sense in the context of power system voltage control: repeatedly commanding a capacitor closed or sending a generator voltage set point will not improve system voltage conditions. The experiments presented in Section V show that this modification leads to significantly higher success rates in the environments described in this section. The code for the modification can be found at [29].

IV. Random and Graph-Based Agents For Comparison

In order to provide a basis for comparison with DRL results, both a random agent and a graph-based agent were developed. Both agents interact with the environments described in Section III and attempt to solve the voltage control problem. All testing is performed against the exact same testing episodes that the DRL agents were tested against (Sections V and VI). As per usual, each episode proceeds either until all voltage issues have been fixed, the power flow diverges, any bus voltage goes below 0.7 p.u. or above 1.2 p.u., or the agent hits the per-episode action cap.

The random agent behaves exactly as one might expect: during each time step it randomly chooses and takes an action from the environment's action space. The random agent is run both with and without the unique-actions-per-episode requirement. For RL generally, it is considered best practice to compare results with a random agent as a baseline to ensure that the RL algorithm is functioning as intended.

The graph-based agent has been created to provide a reasonable benchmark for comparison with the DRL agents. In contrast to the DRL agent, the graph-based agent uses a model of the power system and is heuristically driven, leveraging the notion that voltage issues are typically "local" and are often remedied by dispatching reactive power resources near to the buses with voltage issues. In short, the agent constructs a weighted, undirected graph of the power system network and changes the voltage set point at the generator which is "nearest" to the bus with the minimum/maximum voltage violation. Graph weights are defined as branch reactances, and "nearest" is defined as the shortest path length considering reactances as distances. For cases that contain capacitors, capacitors at neighboring buses to the bus with the largest violation are actuated appropriately (opened for high voltage violations, closed for low voltage violations) before generator set points are considered. The code for the graph-based agent can be found in [29].

V. Experiments with IEEE 14 Bus System

This section presents results for DRL agents attempting to control voltage in the IEEE 14 bus system. All neural networks are fully connected and contain two hidden layers with 64 neurons each. The environment and DQN algorithm with all improvements are as described in Section III. Tests were performed both with and without the DQN algorithm modification discussed in Section III-C. In contrast to what was done in [26] (Section II-C), all five generators are considered available for active power dispatch (Section III-B3). All episodes contain a single line contingency: the lines considered for contingencies were (from bus - to bus) 1-5, 2-3, 4-5, and 7-9. All training sessions were allowed to proceed for 500,000 simulation steps (total actions taken by agent), and all episodes were capped at 10 actions (twice the number of generators). Note the 14 bus system does not contain any capacitors. Testing is performed on 5000 testing episodes not seen in training.

Table I depicts results grouped by observations provided to the agents. Note that the abbreviated column headers are explicitly defined at the bottom of Table I. All success percentages and rewards represent the average across three independent training/testing runs with different random number generator seeds. The column labeled "PSOOBV" presents the

success rate for testing episodes which began with out-of-band (OOB) voltages (outside $[0.95, 1.05]$). For comparison, the graph-based and random agents obtained OOB success rates of 40.8% and 16.5%, respectively. All OOB success rates lower than the random agent are *italicized* for emphasis. Across all four observation groupings in Table I, the combination of the unique actions per episode (UAE) algorithm modification, min-max scaled voltages (MMV), and reward scheme (RS) 1 generally led to the best results (values in these rows are **bold**). Success rates for this combination were very similar across observation groupings. While this aforementioned combination was best, the graph-based agent still performed better with its OOB success rate of 40.8%.

TABLE I: Results for 14 bus experiments

Observations	UAE	MMV	RS	PS	PSOOBV
Voltage Only	No	No	1	13.4	*6.0*
	Yes	No	1	38.5	31.8
	No	Yes	1	19.4	*11.6*
	Yes	**Yes**	**1**	**42.8**	**36.2**
	Yes	Yes	2	14.8	*9.6*
Voltage and Gen. State	No	No	1	19.4	*12.8*
	Yes	No	1	31.7	24.9
	No	Yes	1	21.0	*14.5*
	Yes	**Yes**	**1**	**41.6**	**35.7**
	Yes	Yes	2	16.4	*11.2*
Voltage and Branch State	No	No	1	16.9	*8.9*
	Yes	No	1	37.4	30.5
	No	Yes	1	17.3	*9.7*
	Yes	**Yes**	**1**	**41.3**	**34.6**
	Yes	Yes	2	14.6	*8.7*
Voltage, Gen. State, and Branch State	No	No	1	19.5	*12.8*
	Yes	No	1	40.8	35.4
	No	Yes	1	20.1	*13.9*
	Yes	**Yes**	**1**	**40.9**	**35.1**
	Yes	Yes	2	16.0	*11.1*

UAE: Unique Actions per Episode, **MMV**: Min-Max Voltages, **RS**: Reward Scheme, **PS**: Percent Success, **PSOOBV**: Percent Success, Out-of-Band Voltages

It can also be seen in Table I that the use min-max scaled voltages generally led to better results than p.u. voltages ("No" in the MMV column). This is due to the fact that min-max scaling magnifies differences in voltages and makes it easier for the agent to learn. Table I also shows that reward scheme 2 consistently led to worse results than the random agent.

Due to the simplicity of the 14 bus system, it is hypothesized that the graph-based agent's success rate of 40.8% approaches the actual percentage of episodes in which it was physically possible to bring all voltages in band (limited by generator reactive power limits). In this light, it can be seen that in several cases the DRL agents performed relatively well.

VI. Experiments with 200 and 500 Bus Systems

In this section, the findings from Section V are exploited to test the scalability of DRL for voltage control to significantly larger and more realistic systems. Synthetic grid test cases with 200 and 500 buses presented in [35] and available at [27] are leveraged. These synthetic grids have been constructed to be representative of the real electric grid, but are intentionally different in order to avoid data sensitivity issues.

The 200 bus system contains 49 generators, four switched shunts, and 180 transmission lines, while the 500 bus system contains 90 generators, 17 shunts, and 468 lines.

Training and testing was performed as in Section V, but training was allowed to proceed for 2,000,000 and 3,000,000 time steps for the 200 and 500 bus systems, respectively. Experiments used fully connected neural networks with two hidden layers. Hidden layers contained 1024 neurons each for 200 bus experiments, and 2054 neurons for 500 bus experiments. Since out-of-band (OOB) success rates for the various observation groups listed in Table I were similar and these systems are more complex than the 14 bus system, agents were given min-max scaled voltages, generator states (on/off), and shunt states (open/closed) as observations. Each system was tested both with and without single line contingencies. If contingencies were included, the agent is additionally given all line states (open/closed) as observations.

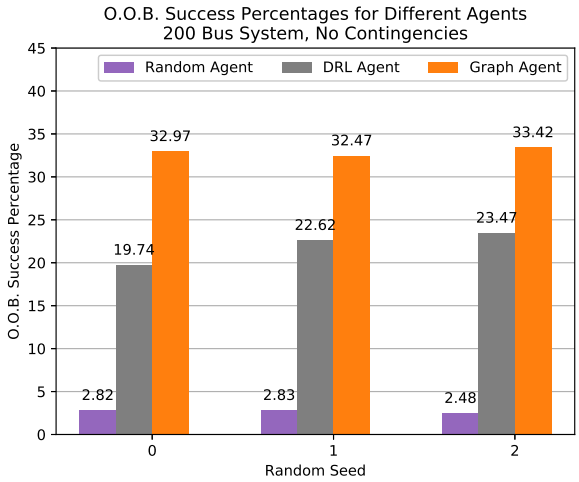
Fig. 2 presents OOB success rates for random, DRL, and graph-agents across three independent runs with different random seeds for the 200 bus system. Fig. 2a shows success rates when no contingencies are applied in the system, and Fig. 2b shows success rates when single line contingencies are applied for each episode. Similarly to the results in Section V, the graph-based agent consistently outperforms the DRL agents, and the DRL agents consistently outperform the random agents. However, it can be seen in Fig. 2b that the DRL agents approached the performance of the graph-based agent for two out of three experimental runs. One key takeaway from Fig. 2 is that the DRL agent's performance on the whole was actually better when single line contingencies were applied. It is hypothesized that this is a result of an oversized neural network in the case of no contingencies: when contingencies are included the observation space grows from 253 measurements to 433.

Fig. 3 presents average training rewards over time for two different random number generator seeds and the 200 bus system. Note that Fig. 3a exhibits large reward fluctuations while Fig. 3b shows steady improvements. This may be indicative of an oversized neural network, sub-optimal neural network architecture, and/or a need for hyper parameter tuning.

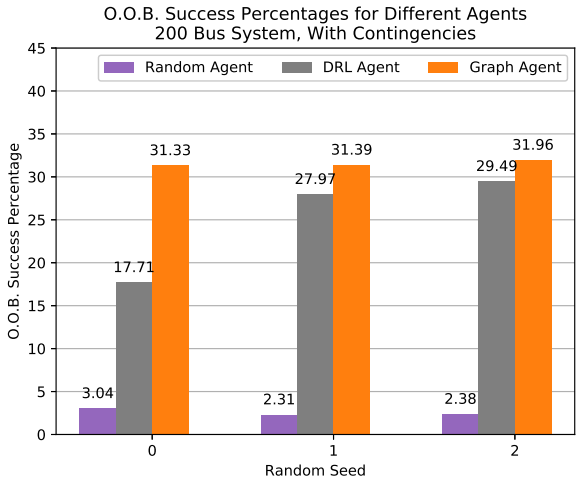
Fig. 4 presents OOB success rates in the same fashion as Fig. 2, but for the 500 bus system experiments. Interestingly, the random and graph-based agents exhibit significantly higher success rates than for the 200 bus system. This fact is likely related to structural differences between the two systems. Training instabilities can be seen in both Fig. 4a and Fig. 4b as each depicts a single instance of the DRL agent performing significantly worse than the corresponding random agent. Similar to the results for the 200 bus system, the presence of contingencies generally improved DRL agent performance, likely due to shortcomings in neural network architecture.

VII. Conclusions and Future Work

This paper provided an in-depth exploration of the application of deep reinforcement learning to the electric power transmission system voltage control problem. It was found that a novel deep-Q network algorithm modification wherein the

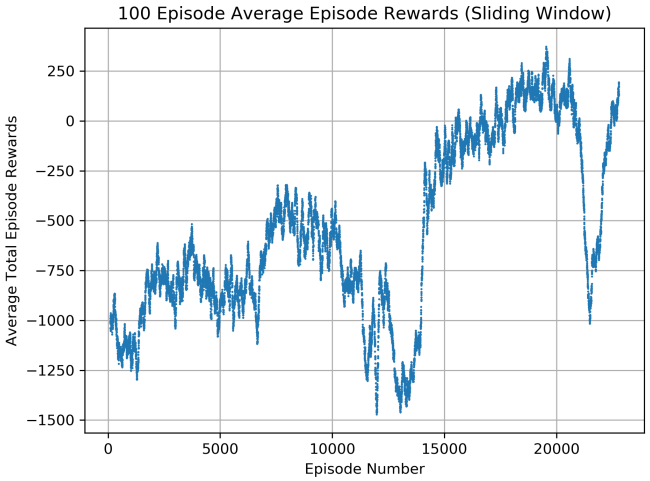


(a) No contingencies

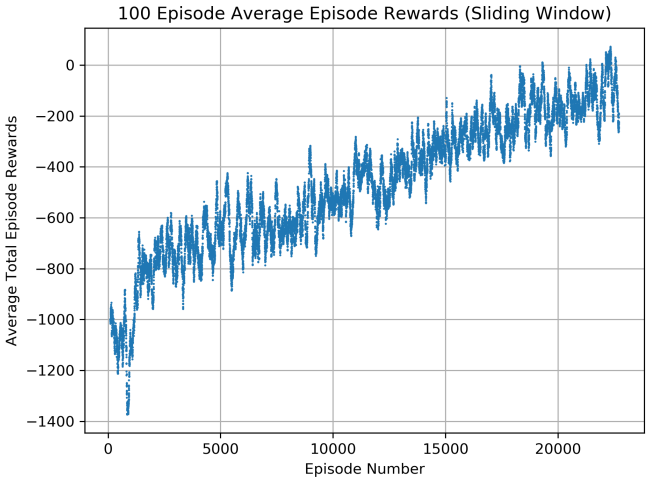


(b) With contingencies

Fig. 2: OOB success rates for random, DRL, and graph-based agent across different random seeds, 200 bus system, with and without contingencies



(a) Seed: 0



(b) Seed: 1

Fig. 3: 100 episode average episode rewards (sliding window), 200 bus system, no contingencies, different random seeds

agent is not allowed to take the same action multiple times in any given training or testing episode leads to significant DRL performance improvements. Additionally, it was shown that the min-max scaling of bus voltage observations could lead to performance improvements as opposed to simply using per unit voltages. DRL agents were trained to control 200 and 500 bus power systems in order to prove the scalability of DRL for voltage control. While no DRL agents were able to exceed the performance of the graph-based agents which were developed for comparison with DRL agents, there were cases with both the 14 and 200 bus systems where DRL agent performance approached graph-based agent performance. Training instabilities were observed for the larger test systems. The research presented in this paper shows clear potential for using DRL to solve the voltage control problem, but more work is needed to ensure DRL techniques can consistently outperform conventional techniques.

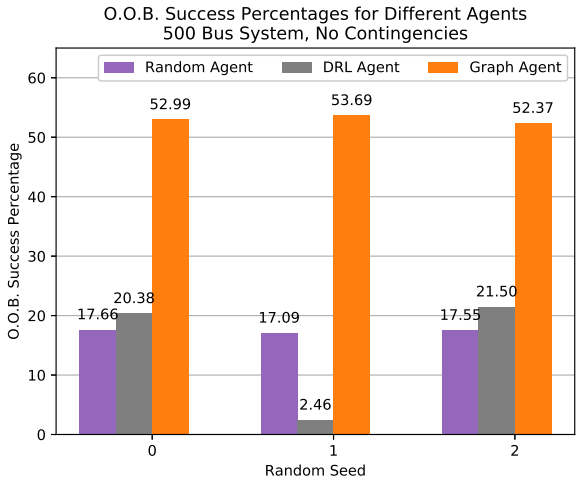
Opportunities for future work are numerous. An in-depth hyper parameter tuning study is likely necessary to get the most out of the advanced DQN algorithms used in this work. One intriguing prospect is the use of geographically arranged observations and convolutional neural networks (CNNs) instead of fully connected networks in the DQN algorithm. Alternatively, custom neural network structures that better exploit the local nature of power system voltage issues could improve performance. Finally, it would be valuable to compare DRL performance with state-of-the-art optimization-based voltage control schemes.

References

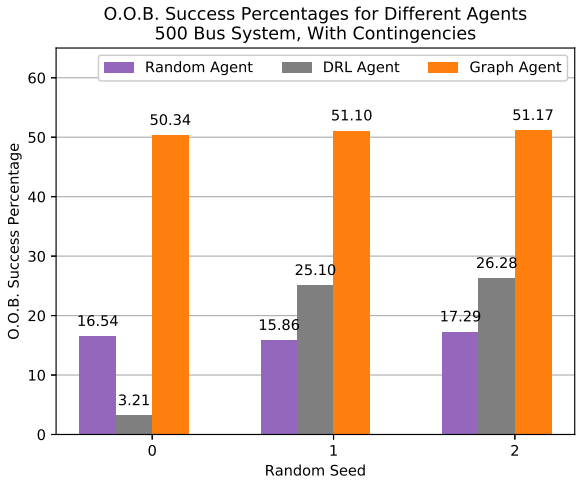
[1] W. M. Warwick, "A primer on electric utilities, deregulation, and restructuring of u.s. electricity markets," https://www.pnnl.gov, Pacific Northwest National Laboratory, Tech. Rep., 2002.

[2] U.S. Energy Information Administration, "Eia forecasts renewables will be the fastest growing source of electricity generation," https://www.eia.gov/todayinenergy/detail.php?id=38053#, accessed: 2019-12-23.

[3] P. H. Larsen, K. H. LaCommare, J. H. Eto, and J. L. Sweeney, "Recent trends in power system reliability and implications for evaluating future investments in resiliency," *Energy*, vol. 117, pp. 29 – 46, 2016.



(a) No contingencies



(b) With contingencies

Fig. 4: OOB success rates for random, DRL, and graph-based agent across different random seeds, 500 bus system, with and without contingencies

[4] T&D World, "Trends report: Power companies facing labor shortage and skills gap," https://www.tdworld.com, accessed: 2019-12-23.

[5] Incremental Systems Corporation, "What is a system operator?" http://www.incsys.com, accessed: 2019-12-23.

[6] C. Taylor, M. Venkatasubramanian, and C. Yonghong, "Wide-area stability and voltage control," *Symposium of Specialists in Electric Operational and Expansion Planning*, vol. 1, 01 2000.

[7] Electric Reliability Council of Texas, "Transmission and security desk operating procedure," http://www.ercot.com, accessed: 2019-12-23.

[8] ——, "ERCOT control room video," https://youtu.be/TlsfItQc05o, accessed: 2019-12-23.

[9] PJM, "A day in the life of dispatch," https://youtu.be/hDNouhlJWa4, accessed: 2019-12-23.

[10] North American Electric Reliability Corporation, "Reliability guideline - reactive power planning," North American Electric Reliability Corporation, Tech. Rep., Dec. 2016, available: https://www.nerc.com.

[11] R. Wilson and C. Taylor, "Using dynamic simulations to design the wide-area stability and voltage control system (wacs)," in *IEEE PES Power Systems Conference and Exposition, 2004.*, Oct 2004, pp. 100–107 vol.1.

[12] S. Corsi, M. Pozzi, C. Sabelli, and A. Serrani, "The coordinated automatic voltage control of the italian transmission grid-part i: reasons of the choice and overview of the consolidated hierarchical system," *IEEE Transactions on Power Systems*, vol. 19, no. 4, pp. 1723–1732, Nov 2004.

[13] J. Tong, D. W. Souder, C. Pilong, M. Zhang, Q. Guo, H. Sun, and B. Zhang, "Voltage control practices and tools used for system voltage control of pjm," in *2011 IEEE Power and Energy Society General Meeting*, July 2011, pp. 1–5.

[14] H. Sun, Q. Guo, B. Zhang, W. Wu, and J. Tong, "Development and applications of system-wide automatic voltage control system in china," in *2009 IEEE Power Energy Society General Meeting*, July 2009, pp. 1–5.

[15] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Pearson, Dec. 2009.

[16] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, "Playing atari with deep reinforcement learning," *CoRR*, vol. abs/1312.5602, 2013. [Online]. Available: http://arxiv.org/abs/1312.5602

[17] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[18] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of Go without human knowledge," *Nature*, vol. 550, pp. 354–359, Oct. 2017.

[19] T. Ahamed, E. Jasmin, and E. Al-Ammar, "Reinforcement learning in power system scheduling and control: A unified perspective," in *2011 IEEE Symposium on Computers Informatics*, March 2011, pp. 650–655.

[20] M. Glavic, R. Fonteneau, and D. Ernst, "Reinforcement learning for electric power system decision and control: Past considerations and perspectives," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 6918 – 6927, 2017, 20th IFAC World Congress.

[21] 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*, vol. 11, no. 2, pp. 1171–1182, 2020.

[22] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *CoRR*, vol. abs/1606.01540, 2016. [Online]. Available: http://arxiv.org/abs/1606.01540

[23] P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, Y. Wu, and P. Zhokhov, "Openai baselines," https://github.com/openai/baselines, 2017.

[24] A. Wood, B. Wollenberg, and G. Shebl, *Power Generation, Operation, and Control*, 3rd ed. Wiley, 2014.

[25] J. G. Vlachogiannis and N. D. Hatziargyriou, "Reinforcement learning for reactive power control," *IEEE Transactions on Power Systems*, vol. 19, no. 3, pp. 1317–1325, Aug 2004.

[26] R. Diao, Z. Wang, D. Shi, Q. Chang, J. Duan, and X. Zhang, "Autonomous voltage control for grid operation using deep reinforcement learning," in *2019 IEEE Power Energy Society General Meeting (PESGM)*, Aug 2019, pp. 1–5.

[27] Texas A&M University, "Electric grid test cases," https://electricgrids.engr.tamu.edu/electric-grid-test-cases/, 2019, accessed: 2019-12-27.

[28] B. Thayer, "gym-powerworld," https://github.com/blthayer/gym-powerworld, 2020.

[29] ——, "drl-powerworld," https://github.com/blthayer/drl-powerworld, 2020.

[30] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, and Y. Wu, "Stable baselines," https://github.com/hill-a/stable-baselines, 2018.

[31] Z. Wang, N. de Freitas, and M. Lanctot, "Dueling network architectures for deep reinforcement learning," *CoRR*, vol. abs/1511.06581, 2015. [Online]. Available: http://arxiv.org/abs/1511.06581

[32] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," *CoRR*, vol. abs/1509.06461, 2015. [Online]. Available: http://arxiv.org/abs/1509.06461

[33] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *CoRR*, vol. abs/1511.05952, 2016.

[34] B. L. Thayer, Z. Mao, Y. Liu, K. Davis, and T. J. Overbye, "Easy simauto (esa): A python package that simplifies interacting with powerworld simulator," *Journal of Open Source Software*, vol. 5, no. 50, p. 2289, 2020. [Online]. Available: https://doi.org/10.21105/joss.02289

[35] A. B. Birchfield, T. Xu, K. M. Gegner, K. S. Shetye, and T. J. Overbye, "Grid structural characteristics as validation criteria for synthetic networks," *IEEE Transactions on Power Systems*, vol. 32, no. 4, pp. 3258–3265, July 2017.