A Data-driven Method for Fast AC Optimal Power Flow Solutions via Deep Reinforcement Learning

Yuhao Zhou, *Student Member*, *IEEE*, Bei Zhang, *Member*, *IEEE*, Chunlei Xu, Tu Lan, Ruisheng Diao, *Senior Member*, *IEEE*, Di Shi, *Senior Member*, *IEEE*, Zhiwei Wang, *Senior Member*, *IEEE*, and Wei-Jen Lee, *Fellow*, *IEEE*

***Abstract*—With the increasing penetration of renewable energy, power grid operators are observing both fast and large fluctuations in power and voltage profiles on a daily basis. Fast and accurate control actions derived in real time are vital to ensure system security and economics. To this end, solving alternating current (AC) optimal power flow (OPF) with operational constraints remains an important yet challenging optimization problem for secure and economic operation of the power grid. This paper adopts a novel method to derive fast OPF solutions using state-of-the-art deep reinforcement learning (DRL) algorithm, which can greatly assist power grid operators in making rapid and effective decisions. The presented method adopts imitation learning to generate initial weights for the neural network (NN), and a proximal policy optimization algorithm to train and test stable and robust artificial intelligence (AI) agents. Training and testing procedures are conducted on the IEEE 14-bus and the Illinois 200-bus systems. The results show the effectiveness of the method with significant potential for assisting power grid operators in real-time operations.**

***Index Terms*—Alternating current (AC) optimal power flow (OPF), deep reinforcement learning (DRL), imitation learning, proximal policy optimization.**

I. Introduction

In modern power grid, the high penetration and integration of fluctuating energy resources, including renewables, have led to bigger challenges in terms of ensuring system security and economics, which requires a faster solution to the non-convex alternating current (AC) optimal power flow (OPF) problem for real-time applications [1].

In recent decades, a number of dedicated approaches have been considered to solve the AC OPF problem. Reference [2] applies the primal-dual interior-point method. In [3], the convex relaxation method is proposed for tackling the AC OPF problem for radial networks through the application of second-order cone programming (SOCP). In [4]-[7], semidefinite programming (SDP) for meshed networks is proposed, and the condition of achieving the global optimum solutions is also discussed. All of these methods, however, are time-intensive for achieving convergence, especially in large-scale systems, and thus their applications are limited in real-time operation.

Sub-optimal solvers have been used in industry to compromise the computation time requirements. Nearly all vendors' tools adopt this assumption as well as a simplification to reach feasible and suboptimal solutions for real-world implementation, e.g., use of DC-based power flow (PF) equations with relaxed security constraints, which is only an approximation of the power grid status.

To address this issue and further expedite the solving speed aiming at real-time applications, a single-iteration quasi-Newton method is proposed to expedite real-time AC OPF solutions with the prerequisite of deriving an accurate estimation of the second-order information [1]. In [8], linearized AC PF equations are implemented to achieve real-time OPF in distribution systems. In addition, there are reports of several supervised-learning-based methods that approximate OPF solutions but increase the solving speed significantly. In [9], a graph neural network (NN) has been applied to approximate the solutions of PF solvers. Deep NNs (DNN) are similarly utilized to solve the DC OPF [10], [11] and AC OPF problems [12], [13]. A commonality among all these methods is that they primarily depend on supervised learning techniques to train the NNs while using massive simulations that must be obtained ahead of time for approximating optimal solutions. However, their small loss value results after the training process cannot guarantee the feasibility for the AC OPF problem with full security constraints.

The recent success of deep reinforcement learning (DRL) algorithms in many control problems such as games, robotics, autonomous driving, and finance provide a promising al-

Manuscript received: July 26, 2020; accepted: October 21, 2020. Date of CrossCheck: October 21, 2020. Date of online publication: November 26, 2020.

This work was supported by State Grid Science and Technology Program "Research on Real-time Autonomous Control Strategies for Power Grid Based on AI Technologies" (No. 5700-201958523A-0-0-00).

Y. Zhou and W. Lee are with the Electrical Engineering Department, University of Texas at Arlington, Arlington, TX 76019, USA (e-mail: yuhao.zhou@mavs.uta.edu; wlee@uta.edu).

B. Zhang, T. Lan, R. Diao, D. Shi (corresponding author), and Z. Wang are with GEIRI North America, San Jose, CA 95134, USA (e-mail: bei.zhang@geirina.net; tu.lan@geirina.net; ruisheng.diao@geirina.net; di.shi@geirina.net; zhiwei.wang@geirina.net).

C. Xu is with State Grid Jiangsu Electric Power Company, Nanjing, China (email: chunleixu@js.sgcc.com.cn).

DOI: 10.35833/MPCE.2020.000522

ternative for power system decision making [14]. Since a well-trained DRL agent does not need to rely on full system models when making control decisions, it can respond instantaneously to a range of conditions, thereby enabling many real-time applications.

Recently, there have been reports of DRL-based applications as part of smart grid. In [15], for example, the multi-agent Q-learning method is applied to optimal reactive power dispatch. In [16], multi-agent $Q(\lambda)$ learning is adopted to perform OPF tasks under discretized action spaces with partitioned power grid. In [17], the deep Q network (DQN) is used to perform optimal control for smart buildings. In [18] and [19], a novel platform "Grid Mind" is proposed, and the DQN and the deep deterministic policy gradient (DDPG) algorithms are implemented to perform autonomous voltage control. The multi-agent DRL based approach is adopted for voltage regulation of distribution system with high penetration of photovoltaic in [20]. In [21], the DDPG algorithm is applied for the load frequency control. Adaptive power system emergency control using the DRL is proposed in [22]. References [23] and [24] adopt the DRL in microgrid control. In [25], the DRL is adopted to solve the energy management problem in the field of Energy Internet (EI). In [26], the DRL is utilized to handle voltage control and dynamic load shedding problem in a real power grid model. In [27], the DRL is used to solve the EV charging scheduling problem for real-time control, which can adaptively learn the transition probability and does not require any system model information. In [28], a DRL agent is trained by the asynchronous advantage actor-critic (A3C) algorithm to provide the optimal parameters for the power system stabilizer.

Inspired by the above efforts, this paper presents a novel DRL-based method to derive a near-optimal solution for the AC OPF problem, with great potential for real-time applications. The DRL trains the DNN to significantly improve the solution success rate, especially when the feasible region is small. Different from most of the aforementioned work, in this effort, continuous control action space is considered, corresponding to the actual needs of power grid control, which alleviates the compromise as well as the "curse of dimension" caused by the discretization of control action. In this effort, the grid does not have to be partitioned to introduce further approximation or inaccuracy as in [16]. Additionally, the state-of-the-art proximal policy optimization (PPO) algorithm [29] is employed to train the agent in order to solve the AC OPF problem. Unlike the DDPG, the PPO algorithm tries to ensure safe updates within certain trust regions, which greatly improves its robustness, performance, and extensibility to the systems of larger sizes. To further guarantee the success in training as well as to expedite the entire training process, the imitation learning (IL) technique is used to obtain an initial policy weight for better training in the DRL process.

Numerical experiments are conducted on two power grid models to validate the approach. Compared with the results from interior-point solvers, the results from a well-trained DRL agent show that not only the high-resolution optimality is achieved, but also there are significant improvements in computation time, thus increasing the potential for employing AI techniques in future power system control and operations.

The key contributions of this paper are summarized as follows: ① by formulating the original AC OPF problem as Markov decision process with appropriate power grid simulation environment and a reasonable reward function, the DRL agent can be trained to learn the optimal generator set-points without violating the operation constraints under various operation conditions; ② by applying the imitation learning technique as initial weights of the DRL agent and adopting the state-of-the-art PPO algorithm, the well-trained DRL agent could achieve near-optimal solutions but with significant improvements in computation time compared with the results from the interior-point solver; ③ it is shown that the DRL agent could effectively solve the AC OPF problem when the $N-1$ contingencies on topology change are considered, which makes it more practical and further indicates its computational advantage. Therefore, this paper presents the potential of employing AI in future power system operations and control.

The remainder of the paper is organized as follows. Section II provides the principles of DRL, IL, and PPO algorithms. Section III details the procedures of the methodology. In Section IV, numerical experiments are conducted using both the IEEE 14-bus and Illinois 200-bus systems to demonstrate DRL performance and effectiveness of the proposed method. Section V provides a conclusion and the future work.

II. Problem Formulation, Principles of DRL, IL, and PPO Algorithm

A. AC OPF Problem Formulation

Assume that a system consists of a set of buses N with a subset G as the generator buses, and a set of transmission lines L. The mathematical model of the AC OPF problem is formulated as follows [4]-[7]. The objective of the AC OPF in this paper is to find the optimal generator set-points such that the total quadratic generator costs are minimized while the security constraints are satisfied simultaneously.

$$\min \sum_{i \in G} (c_{2i} P_{gi}^2 + c_{1i} P_{gi} + c_{0i}) \tag{1}$$

s.t.

$$P_{gi}^{\min} \le P_{gi} \le P_{gi}^{\max} \quad \forall i \in G \tag{2}$$

$$Q_{gi}^{\min} \le Q_{gi} \le Q_{gi}^{\max} \quad \forall i \in G \tag{3}$$

$$V_i^{\min} \le |V_i| \le V_i^{\max} \quad \forall i \in N \tag{4}$$

$$|S_{flow,il}| \le S_{flow,il}^{\max} \quad \forall (i,l) \in L \tag{5}$$

$$P_j = |V_j| \sum_{i \in N} |V_i| (g_{ji} \cos(\alpha_j - \alpha_i) + b_{ji} \sin(\alpha_j - \alpha_i)) \quad \forall j \in N \tag{6}$$

$$Q_j = |V_j| \sum_{i \in N} |V_i| (g_{ji} \sin(\alpha_j - \alpha_i) + b_{ji} \cos(\alpha_j - \alpha_i)) \quad \forall j \in N \tag{7}$$

where P, Q, V, and S are the active power, reactive power, bus voltage, and apparent power, respectively; c_0, c_1, and c_2 in (1) are the non-negative coefficients accounting for the cost of active power generation; the subscripts g and *flow* denote the generator and transmission line, respectively; the

subscripts min and max denote the minimum and maximum values of the variable, respectively; P_j and Q_j in (6) and (7) characterize the power injections at bus j, which are the differences between the power of the generator and the load; α in (6) and (7) is the angle of the bus voltage; and g_{ji} and b_{ji} are the conductance and susceptance between bus j and bus i, respectively.

B. DRL Basis

The goal of reinforcement learning (RL) is to maximize a long-term expected reward by continuously interacting with the environment as depicted in Fig. 1 [14], where s_t is the observed state by the agent at time step t; a_t is the agent's action according to the current policy π at time step t; and r_t is the immediate reward from the environment at time step t. After executing action a_t, the environment transits to the next state s_{t+1}. By unceasingly interacting with the environment, the agent's policy rollouts and forms a trajectory. During such process, the set of states S, the set of actions A, the transition probability $T(s_{t+1}|s_t, a_t)$, which maps the distribution of the next state with the current (state, action) pair, the set of reward $R(s, a, r)$, and a discount factor $\gamma \in [0, 1]$ on the future reward together form a Markov decision process denoted as a tuple $\langle S, A, T, R, \gamma \rangle$.

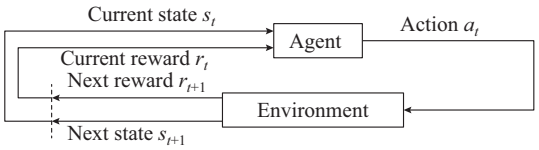

Fig. 1. Description of RL agent interacting with environment process.

Every trajectory of a policy returns a total accumulated reward R_t from time step t shown in (8) [30], and the agent aims at learning an optimal policy π^* that maximizes the expected return.

$$R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{8}$$

Two other important concepts of RL are state-value and action-value functions as defined in (9) and (10), respectively.

$$V^{\pi}(s) = E\left(R_t | s_t = s; \pi\right) \tag{9}$$

$$Q^{\pi}(s, a) = E\left(R_t | s_t = s, a_t = a; \pi\right) \tag{10}$$

where $E(\cdot)$ is the function for expectation calculation [30]; $V^{\pi}(s)$ is the state-value function representing how good a state is by calculating the expected reward starting from such state following a certain policy; and $Q^{\pi}(s, a)$ is the action-value function assessing how good an action is at a certain state by evaluating the expected reward starting from the state s with the action a following a policy π.

Combining deep learning (DL) algorithms [31] with RL defines the fields of DRL. In the DRL, the deterministic or the stochastic policy $\pi_\theta(a_t|s_t)$ is usually approximated by an NN with parameters θ, which represents the weights and biases in the NN. The agent learns to update θ such that the optimal policy is achieved.

C. IL

IL, or learning by demonstration, focuses on imitating experts' demonstrations [32]. The goal of IL is to learn a policy that maps the states to actions, which is similar to the DRL. Nonetheless, the training technique is slightly different, which usually adopts the dataset aggregation algorithm (DAgger). Starting from running the supervised IL, by following the previous trained policy to collect new trajectories but recording the expert's behavior, new tuple (state, expert's action) pairs are formulated and aggregated as an updated training dataset to further train the policy until it converges [32]. Due to the close connection between the DRL and the IL, the training result of the IL could be adopted as the initialization of the policy NN for the DRL, which would: ① alleviate the computational feasibility problems in DRL (the problem with sparse reward may confuse the agent and result in failure in finding the policy); ② solve the sample-inefficiency caused by numerous trial and error without expert's demonstration; ③ make the training safer by avoiding catastrophic consequence due to expert's knowledge [33].

D. PPO with Clipped Surrogate Loss

The policy gradient (PG) algorithm [30] optimizes the policies directly by adopting the stochastic gradient ascent algorithm. Nevertheless, the conventional PG algorithm does not have a good convergence, especially with high dimensional and/or continuous action spaces [29]. Originating from the PG algorithm, the proximal policy optimization (PPO) algorithm is proposed to ensure a better convergence, which is achieved by: ① the actor-critic structure: an actor NN to learn the stochastic optimal policy and a critic NN to estimate the value function; ② two other enhancements illustrated as follows.

First, the generalized advantage estimation (GAE) function $A_t^{GAE(\gamma,\lambda)}$ is utilized to replace (10) during the update in order to reduce the variance of the estimation, which is shown in (11) [34].

$$\begin{cases} A_t^{GAE(\gamma,\lambda)} = (1-\lambda)(\hat{A}_t^{(1)} + \lambda \hat{A}_t^{(2)} + \lambda^2 \hat{A}_t^{(3)} + ...) \\ \hat{A}_t^{(k)} = \sum_{l=0}^{k-1} \gamma^l \delta_{t+l}^V = -V^{\pi}(s_t) + r_t + \gamma r_{t+1} + ... + \\ \qquad \gamma^{k-1} r_{t+k-1} + \gamma^k V^{\pi}(s_{t+k}) \\ \delta_t^V = r_t + \gamma V^{\pi}(s_{t+1}) - V^{\pi}(s_t) \end{cases} \tag{11}$$

where $V^{\pi}(s_t)$ is the state value shown in (9) starting from time step t, which comes from the critic NN; λ controls the average degree of n-step advantage values; and δ_t^V is the temporal difference (TD) residual of V^{π} [34].

Second, the PPO updates the parameters within an appropriate trust-region [35], and this helps avoid falling off the "cliff" from the hyper-surfaces of the complex reward functions which may be hard to escape from. Such a safe update is achieved by modifying the objective function L shown in (12), where θ indicates parameters of the actor NN $\pi_\theta(a_t|s_t)$; the clipping range ε (a hyper-parameter in PPO) determines the range of the trust-region for the update, and the advantage value A_t is calculated from (11) $A_t^{GAE(\gamma,\lambda)}$; $\hat{E}$ means the expectation calculation; after applying the clip function, the

output of the function guarantees that the input $r_t(\theta)$ falls in the range between $(1-\varepsilon)$ and $(1+\varepsilon)$. The minimization operator makes sure that the new policy does not benefit from going too far away from the old policy, which, therefore, regulates the update of the parameter.

$$L(\theta)=\hat{E}_t\left\{\min\left(r_t(\theta)A_t^{\pi_{\theta,old}}, clip(r_t(\theta),1-\varepsilon,1+\varepsilon)A_t^{\pi_{\theta,old}}\right)\right\} \quad (12)$$

where $r_t(\theta)=\pi_\theta(a_t|s_t)/\pi_{\theta_{old}}(a_t|s_t)$. The policy π_θ is stochastic in PPO, which is parameterized as a conditional Gaussian policy $\pi_\theta \sim N(\boldsymbol{\mu}_\theta(s), \boldsymbol{\Sigma}_{\pi\theta})$. The mean vector $\boldsymbol{\mu}_\theta(s)$ is the output of the actor NN, and the covariance $\boldsymbol{\Sigma}_{\pi\theta}$ is a diagonal matrix initially assigned manually but updated during the backpropagation. Additionally, θ_{old} is the policy parameter before the update of the actor NN.

The objective function of the critic NN is shown as:

$$\min_{\varphi} \sum_{(s_t,R_t)\in D_{batch}} \frac{1}{N_{batch}} \left\| R_t - V_\varphi^\pi(s_t) \right\|_2^2 \quad (13)$$

where R_t is the discounted accumulated reward from (8); D_{batch} is the trajectories with batch size N_{batch}; φ is the parameter of the critic NN; $V_\varphi^\pi(s_t)$ is the output of the critic NN; and $\|\cdot\|_2^2$ is the square of $(l-2)$ norm calculation.

III. DRL-BASED AC OPF WITH IL

The goal of the DRL agent is to minimize total generation costs without violating operation constraints or shedding the load. This is done by controlling the voltage set-points and power outputs of generators under various loading conditions. The agent can be trained offline through historical data and/or massive simulations, and then applied online in the real power grid. During the online application, a well-trained agent is capable of providing control actions momentarily regardless of the system size. This section mainly focuses on the efficient offline training process of such agent, which includes the design of several important components during DRL training: offline power grid environment, state and action spaces, NN structures, IL implementation, and the detailed training process.

A. Offline Grid Environment

In this paper, the offline environment is developed by mimicking the OpenAI Gym environments (benchmark systems for the DRL studies [36]), which consists of an AC PF (PF) solver and several important functions as shown below:

1) Initialization: function *reset*(·) initializes a case by running the PF and outputs the initial state s_t.

2) Interaction: function *step*(·) applies the agent's action, runs the PF, and then provides the agent with the resulting state, "done" signal, and the corresponding reward.

3) Reward calculation: the detailed design for the reward function r_{reward} is presented in (14):

$$r_{reward}=\begin{cases}-5000 & \text{PF solver is diverged}\\ R_{pg,v}+R_{v,v}+R_{br,v} & \text{there are constraints violations}\\ wC_{generators}+z & \text{system is normal}\end{cases} \quad (14)$$

where $R_{pg,v}$, $R_{v,v}$, and $R_{br,v}$ correspond to the total amount of violations as the negative reward if any inequality violation is detected on: ① the active power limits of generators; ② the voltage magnitude limits of buses; and ③ the branch thermal flow limits (in both directions) of transmission lines, respectively. For example, assuming that $P_g^{\max}$ in (2) for all generators is 100 MW, if the PF result suggests that two generators are violating the constraint (2) and the corresponding calculated values are 120 MW and 105 MW, $R_{pg,v}=-25$; $R_{v,v}$, and $R_{br,v}$ are calculated in a similar way. Besides, if the agent's actions lead to the divergence of PF solver, a large negative reward is adopted as a big penalty. The DRL training aims at avoiding these "worst actions". Coefficients w (w is a negative number) and z are applied to transform the convex cost function linearly into a concave reward function for the DRL training. $C_{generators}$ is the generator cost.

From (14), the negative rewards correspond to the operation constraints of original system shown in (2)-(7), and thus the positive reward is guaranteed when there are no violations of operation constraints, which indicates that the larger positive reward contributes to the cheaper generation costs. As the goal of the DRL training is to achieve the maximum long-term reward return, the designed reward function is reasonable for the DRL agent to learn the optimal generator set-points while minimizing the generation costs and concurrently satisfying the feasibility.

Figure 2 illustrates the interaction between the DRL agent and the offline power grid environment during one episode, which starts from the initialization on the case through *reset*(·) until the "end" in the figure with "done" set to "True" by *step*(·).

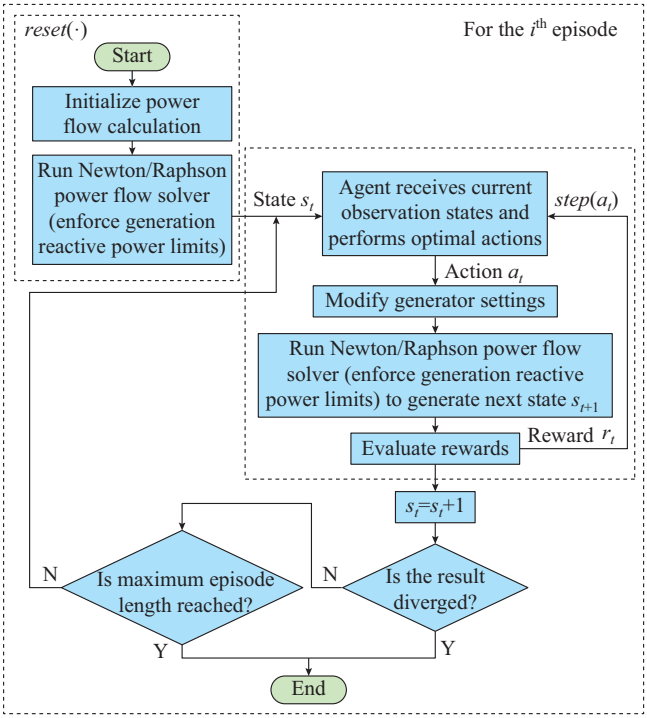

Fig. 2. Flowchart of grid environment interacting with an agent.

Furthermore, the sampled action in *step*(·) function comes from the conditional Gaussian policy $\pi_\theta \sim N(\mu_\theta(s), \Sigma_{\pi\theta})$, which is discussed in Section II-D. Also, in the OpenAI Gym environments, the "done" signal usually indicates whether the de-

sired state is reached when the agent is interacting with its environment during the training process. However, the "done" signal is hard to determine in solving the AC OPF problem, since it is difficult to determine whether the optimal cost has been reached. Therefore, in our environment, the "done" signal becomes "True" when the PF result diverges, which indicates the "game over" status, or when the maximum number of steps is reached.

B. State and Action Spaces

The state $\boldsymbol{s}$, which is the input for the DRL agent, includes the active and reactive power of the load, P_{di} and Q_{di}, at each bus i ($i = 1, 2, \ldots, n$), and the initial active power setting P_{gj} and voltage setting V_{gj} for every generator j ($j = 1, 2, \ldots, t$), as denoted in (15), where n is the number of loads; and t is the number of generators in the power grid. The *MinMax* scaling preprocessing [37] needs to be conducted on this vector before passing it to the agent to handle different scalings of various parameters.

$$\boldsymbol{s}=[P_{d1},\ldots,P_{dn},Q_{d1},\ldots,Q_{dn},P_{g1},\ldots,P_{gm},V_{g1},\ldots,V_{gt}] \quad (15)$$

The action space consists of adjustments of generator settings as presented in (16). Such selection on the action $\boldsymbol{a}$ can further correlate the states and actions.

$$\boldsymbol{a}=[\Delta P_{g1},\ldots,\Delta P_{gm},\Delta V_{g1},\ldots,\Delta V_{gt}] \quad (16)$$

C. NN Structure in PPO Training

The actor and critic structures in PPO are shown in Fig. 3 and Fig. 4, respectively, the inputs for which are denoted in (15).

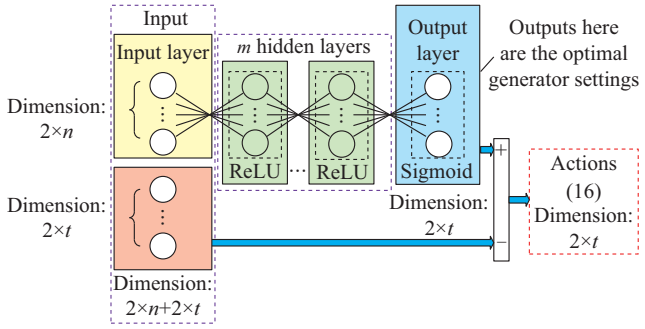

Fig. 3. Actor NN structure in PPO training.

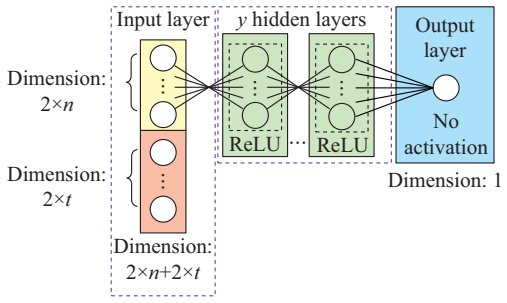

Fig. 4. Critic NN structure in PPO training.

In Fig. 3, the inputs of the yellow and orange blocks are $[P_{d1},\ldots,P_{dn},Q_{d1},\ldots,Q_{dn}]$ and $[P_{g1},\ldots,P_{gm},V_{g1},\ldots,V_{gt}]$, respectively; the input for the NN is the yellow block; and the output are the actions shown in (16). In Fig. 4, the input for the NN are the yellow and orange blocks; and the output is the state value $V^{\pi}(s)$ shown in (9). The rectified linear unit (ReLU) activation function in the hidden layers can effectively prevent the vanishing gradient, and the sigmoid activation in the actor output layer makes sure that the output can have bounded negative or positive values. The one neuron in the critic output layer has no activation, which outputs the value function of a given state.

One special design on the actor NN, as shown in Fig. 3, greatly improves the convergence of the network: only active and reactive loads are fed as the input of the NN in the actor, and the difference between the output of the NN and the voltage settings together with the active power outputs are then finally set as the actions, as denoted in (16) and illustrated in Fig. 3.

There are three main considerations behind such design to achieve the desired performance: ① the OPF results are mainly determined by load variations and operation constraints; ② to achieve better performance of NN, it is helpful to simplify its mapping logic: mapping the optimal settings alone would be much easier than directly mapping the actions, because the action can be the difference between the optimal and the initial generator settings; ③ such design does not require very accurate initial generator settings, and even if the initial PF diverges, the structure would still be able to provide the optimal actions.

D. IL Implementation

IL can help initialize the weights for the actor NN and therefore speed up the DRL agent training process. For control problems with large state and action spaces, learning from scratch may even fail during the training process. In this work, we adopt DL to conduct the IL for the actor NN, since: ① we target at obtaining the AC OPF solutions using fewer steps, preferably in one step; ② the mapping between the load and the optimal generator settings does not involve the trajectory of multi-step interactions with the environment within one episode. The IL training method adopted in this paper is similar to that in [12].

The "expert" actions for optimal generator settings, which are denoted by $\hat{a}_t$ shown in (17), are obtained by running AC OPF solver offline for training dataset D_{train} of size N_{IL}. Afterward, these (state, "expert" action) pairs are utilized in imitation learning and are formulated as a regression problem, where the inputs are the states and the "labels" are the "expert" actions. By adopting (17) as the loss function via the first-order optimizer such as stochastic gradient descent, the initial mean value $\mu_{\theta}(s)$ of the stochastic policy π_{θ} in the PPO agent is trained to clone the optimal generator settings from OPF solver results. Subsequently, the IL results are served as the initial weights of the actor NN.

$$\min_{\theta}\sum_{(s_t,\hat{a}_t)\in D_{train}}\frac{1}{N_{IL}}\left\|\hat{a}_t-\mu_{\theta}(a_t|s_t)\right\|_2^2 \quad (17)$$

In the meantime, the IL training result could serve as a validation for the structure of the actor NN. Figure 5 exemplifies a comparison of the loss during the training process between two different designs on the actor NNs using one scenario of the IEEE 14-bus system. These two NNs have the same structure of hidden layers but different input layers: ① the structure of the NN at the top left in Fig. 5 is the

same as that illustrated in Fig. 3; ② the NN at the bottom left in Fig. 5 directly maps all inputs including initial settings of load and generators to the action space. The inputs are given in (15) with 38 dimensions and the actions are given in (16). From Fig. 5, it is clearly seen that the training loss of the NN with the proposed structure in Fig. 3 is much smaller than that of the NN which includes all the information of (15) as input (specifically, the loss shows that the training does not converge in such a design), which validates the significant advantages of the presented design for the actor NN.

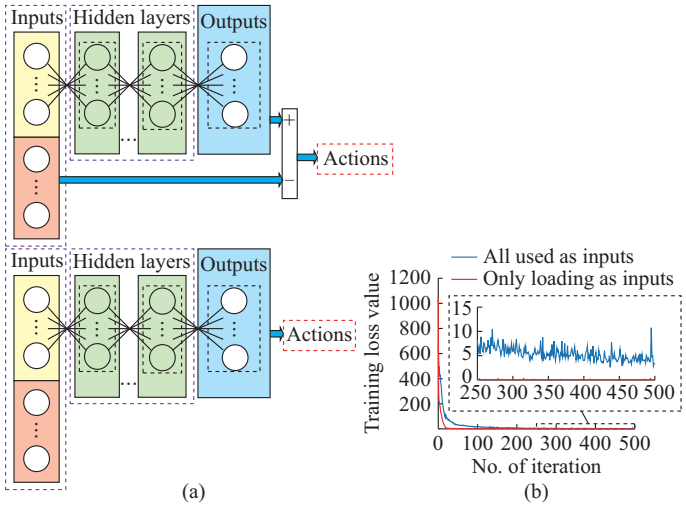

Fig. 5. IL comparisons based on 2 different NN structures. (a) NN structure. (b) Training loss.

However, IL alone here is not enough (further illustrated in Section IV) since it is very difficult to generate enough good samples that are representative of all different operation conditions [30] through DL alone. The DRL process thus needs to be adopted to further train the control policy for better control performance.

E. Detailed PPO Training Algorithm in Solving AC OPF Problem

The training process of the DRL agent can generate data that cover representative situations during the agent's interaction with the environment. By combining IL and DRL, their corresponding advantages are maximized. The detailed PPO training algorithm is shown in Algorithm 1. In Algorithm 1, one *epoch* means that all the training data have been trained once in the DRL; "*range*(·)" is a function in Python. In addition, Kullback-Leibler (KL)-divergence D_{KL} measures the difference between two distributions, as defined in (18), where P and Q are two probability distributions with individual probability density function $p(x)$ and $q(x)$, respectively [38]. Most formulas involving the KL-divergence hold regardless of the base of the logarithm. In this paper, we adopt base 10.

$$D_{KL}(P\|Q)=\int_{-\infty}^{\infty}p(x)(\lg(p(x))-\lg(q(x)))dx \tag{18}$$

Besides, in Algorithm 1, *Step 1* is the initialization process, which initializes the hyper-parameters of the PPO algorithm, initializes the weights of the critic NN, and loads the IL training results as the initial weights of actor NN; *Step 2* directs to read the training dataset for the DRL training; *Steps 3-20* are the DRL training process; *Step 4* shuffles the training dataset for each epoch of training; *Step 6* gets a batch size of data from the training dataset; *Steps 7-9* refer to the interactions between the DRL agent and the power grid environment such that the trajectories are collected; *Steps 10-13* imply the training of the actor NN, and (8), (10) (the outputs of the critic NN), and (11) are calculated according to the previously collected trajectories, then Adam optimizer [39] is applied to maximize (12); and *Steps 14-16* indicate the training of the critic NN, and (8) is calculated from the previously collected trajectories, then Adam optimizer is applied to minimize the objective function (13).

Algorithm 1: PPO training for solving AC OPF problem

1: **initialize:** the number of training data $E_{p,\max}$, episode length T, discounting factor γ, λ in (11), KL_{tar}, batch size N_{batch} in (13), policy log covariance $\Sigma_{\pi\theta}$ used in actor NN, training epoch number *epo*, actor NN *policy*, critic NN *valuefn* shown in (13), updating NN number N_{NN}, clipping range ε shown in (12)
2: parse in the training dataset D_{train} containing the information of load and generator settings
3: **for** each *epoch* in range(*epo*):
4: shuffle the training data and set $index=0$
5: **while** $index<E_{p,\max}$:
6: get a new batch of data with batch size N_{batch}
7: **for** each episode e in $range(N_{batch})$:
8: collect the trajectories' information for every step including (s_t, a_t, r_t, s_{t+1}) from Fig. 2
9: **end for**
10: **for** in $range(N_{NN})$:
11: optimize (12) w.r.t. θ via Adam optimizer
12: **break if** $D_{KL}>KL_{tar}$
13: **end for**
14: **for** in $range(N_{NN})$:
15: optimize (13) w.r.t. φ via Adam optimizer
16: **end for**
17: $index=index+N_{batch}$
18: **end while**
19: **end for**
20: **return:** *policy*

Moreover, the hyper-parameter KL_{tar} controls the dynamic training updates for the actor NN, which additionally provides oversight to the balance between explorations and exploitations in PPO. The detailed explanation is discussed as follows. From *Step 12* in Algorithm 1, *KL-divergence* calculated in (18) describes the distribution difference between the old actor NN and the most recent trained actor NN after *Step 11*. If the calculated *KL-divergence* is bigger than KL_{tar}, the training actor NN is stopped in the "for loop" shown in *Step 10*, which regulates the number of updating the weights of actor NN μ_θ. Therefore, the number of updating the weights of actor NN may be different from a different batch of training data, which makes it more stable during the DRL training. Besides, the sampled actions during the trajectories collection process come from the conditional Gaussian policy $\pi_\theta \sim N(\mu_\theta(s), \Sigma_{\pi\theta})$, which is discussed in Section II-D.

When optimizing (12) in *Step 11*, the covariance matrix $\Sigma_{\pi\theta}$ is also updated, which makes the trained stochastic policy π_θ more deterministic and leads to the sampled space ranges smaller. Therefore, KL_{tar} determines the exploration and exploitation of the DRL training. The smaller the sampled action space ranges are, the less exploitation and more exploration there will be for the agent.

IV. Case Study

The proposed approach to solving the AC OPF problem is tested on both the IEEE 14-bus system (with 14 buses, 5 generators, and 20 lines) [40] and the Illinois 200-bus system (with 200 buses, 38 generators, and 245 lines) [41]. The simulation platform is developed using Python 3.7, and the offline power grid environment is established on the PF solver in PYPOWER [42] (Python version of MATPOWER [43]), which provides the Newton-Raphson PF solver and interior-point AC OPF solver (IPS). The maximum number of iteration and termination tolerance for the PF solvers are 10 and 10^{-8}, respectively. The main objective is to minimize the total generation cost without violating security constraints, including voltage secure zones and thermal limits of transmission lines for the base operation scenarios.

1) *Data generation*: each load is randomly perturbed between [0.6, 1.4] p.u. with uniform distribution, where the original load data in the case files is applied as the base value. Each generator setting is also randomly perturbed between $[P_{\text{gmin}}, P_{\text{gmax}}]$ for active power control and $[V_{\text{gmin}}, V_{\text{gmax}}]$ for reactive power control.

2) *Label creation*: the IPS is adopted to generate the optimal action labels for IL and to indicate whether the OPF problem is feasible or not.

3) *Data arrangement*: all the data with feasible OPF solutions are collected and divided into three datasets for different purposes, as shown in Fig. 6: ① training dataset used for IL and PPO training, containing the cases that can be solved by the PF solver; ② testing dataset I used for testing the trained agent, verifying its performance, and containing the cases that can be solved by the PF solver; ③ testing dataset II used for testing the trained DRL agent in some "dangerous" situations, which initially diverge during the PF calculation due to the unreasonable generator settings.

By following the aforementioned procedures for data generation, the dataset information of the test systems in this paper is shown in Table I.

The performance evaluation of the trained DNN is based on whether the solution is feasible and how close the solution is compared with the mathematical solver instead of showing the loss values. To validate the performance of the trained agent, the cost comparison in percentage (κ), success rate that reflects the feasibility rate, and the total running time are chosen as the evaluation indices. Comparison in percentage (κ) is calculated as in (19) [12]. The success rate represents the percentage among the testing data that the agent successfully solves the OPF problem without violating any constraints. The running time comparison is made using a laptop with Intel i7-7700HQ CPU at 2.8 GHz and 12 GB RAM.

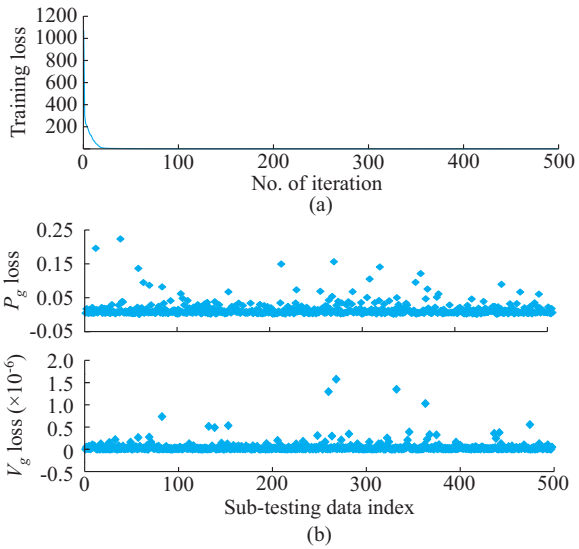

Fig. 6. IL results. (a) Training loss curve. (b) Losses during testing process.

TABLE I
Dataset Information of Test Systems

Test system	No. of training dataset	No. of testing dataset I	No. of testing dataset II
IEEE 14-bus system	55000	17364	2000
Illinois 200-bus system	60000	20000	5000

$$\kappa = \frac{cost_{ips} - cost_{agent}}{cost_{ips}} \tag{19}$$

where $cost_{agent}$ and $cost_{ips}$ are the system cost obtained through the PPO agent and IPS solver, respectively.

A. IEEE 14-bus System

The dimensions of the state and action space are 38 and 10, respectively, in the IEEE 14-bus system. There are no line flow limits in the original system model, and thus, a limit of 32 MVA on line 4-5 is used to test the agent performance. Under these circumstances, the AC OPF solver based on semidefinite programming (*runsdpopf*(·) from MATPOWER [5]) fails to recover the $rank-1$ matrix under the original loading condition, which means that the OPF solution is more difficult to obtain. In this case, the w and z values in (14) are set as –0.05 and 1000, respectively. The maximum episode length T is set to be 5.

Three hidden layers with (380, 195, 100) neurons are applied in the actor NN, and three hidden layers with (380, 44, 5) are applied in the critic NN in PPO. In addition, the IL training process could be regarded as the verification of the NN structure. If the loss value is big, the NN needs to be deeper. The training process for IL is shown in Fig. 6, where 99% of the data in the training dataset is used as a sub-training dataset and the remaining 1% data is regarded as a sub-testing dataset for the IL process.

The losses from (17) are very small as indicated in Fig. 7, which validates the structure of actor NN. Then, we apply

this actor NN initialized by IL to perform the OPF task in testing dataset I, and the results are shown in Table II.

TABLE II
COMPARISON OF PPO AGENTS' PERFORMANCE ON TESTING DATASET I FOR MODIFIED IEEE 14-BUS SYSTEM

Training method	No. of success data	No. of failure data	Success rate (%)	Maximum $\|\kappa\|$ among success data (%)	Minimum $\|\kappa\|$ among success data (%)	Average $\|\kappa\|$ among success data (%)
IL	7138	10226	41.11	0.33	1.93×10^{-5}	1.29×10^{-2}
PPO	15944	1420	91.82	17.55	0.56	3.27
PPO with IL	17364	0	100.00	1.75	0.21	0.597

From Table II, it is observed that although the training and testing losses in IL become very small, there are still 10226 out of 17364 cases in testing dataset I that violate the system constraints, and most of the violations are on the modified line flow due to the smaller feasible regions of the modified system. This indicates that applying the DL training technique alone is not enough in cases with a relatively small feasible region.

To further verify the effectiveness of PPO, the initialized policy is generated where only 9900 out of 55000 training cases are employed to perform IL. Thereafter, the agent with this initialization is trained for 2 epochs with PPO, where KL_{tar} is set to be 0.03. Moreover, to verify the significance of the IL for the agent's training performance, the PPO training without IL is also performed. Figure 7 illustrates the PPO training processes, where the entropy is applied to evaluate how deterministic the policy becomes during training (how certain the PPO agent becomes). It can be observed from Fig. 7 that the decreasing policy entropies and increasing rescaled mean rewards verify that both agents are effectively learning control policies during the training process.

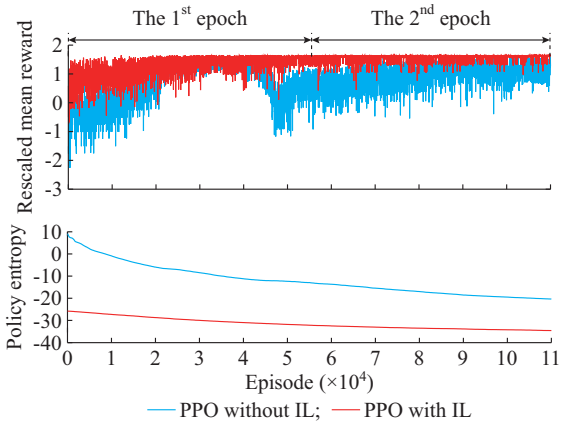

Fig. 7. PPO training process for IEEE 14-bus system.

From Table II, it can be seen that there are still 1420 out of 17364 testing cases that either violate the constraints or diverge after two epochs of training without IL, which suggests that it would take longer training time for the agent to converge under such circumstance. In contrast, the PPO agent with IL performs much better in all cases using the same setting, where the rewards are all positive shown in Fig. 8. More importantly, the total generation costs obtained from the agent are very close to those calculated directly from IPS, where the average and maximum absolute value of κ shown in (19) are only 0.597% and 1.75% higher, respectively. This comparison clearly demonstrates the significance and superior performance when using IL for training PPO agents. In addition, the trained agent is capable of controlling the system to achieve the optimal cost within just one step in 99.76% of the 17364 testing cases. Furthermore, to obtain optimal or sub-optimal solutions for the 17364 testing data, the running time from the IPS costs 6.184 hours, while it only takes 0.863 hour from the proposed method, indicating approximately 7 times of speedup.

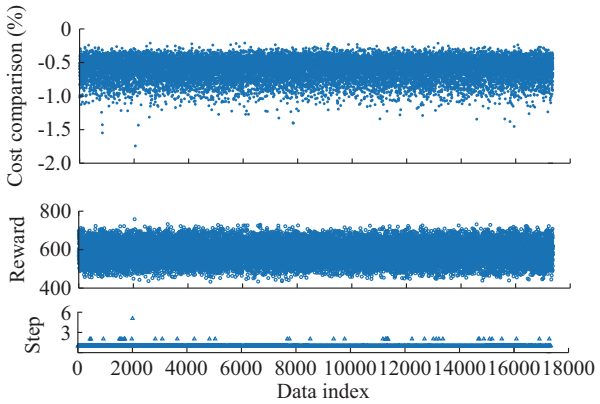

Fig. 8. PPO performance with IL for solving OPF task on testing dataset I for IEEE 14-bus system.

Figure 9 illustrates the performance of the agent with IL in the testing dataset II, which contains the data in which the Newton-Raphson PF solver initially diverges. It can be observed that the trained agent can still solve the OPF problem well under "dangerous" conditions, and that all the cases can be solved within a maximum of 2 steps. From Table II, Fig. 8, and Fig. 9, the effectiveness of the proposed method is verified in the IEEE 14-bus system.

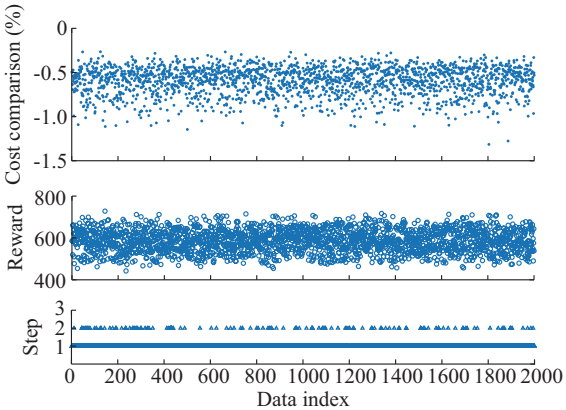

Fig. 9. PPO performance with IL for solving OPF task on testing dataset II for IEEE 14-bus system.

B. Illinois 200-bus System

The Illinois 200-bus system is also used to demonstrate the performance of the proposed approach in a larger and

more complex system. The dimensions of the state and action space are 476 and 76, respectively. The thermal limit of line 124-123 is reduced from 400 MW to 230 MW, which makes the AC OPF problem more difficult to solve. All other thermal limits of lines are not modified. The w and z values in (14) are set to be –0.01 and 1000, respectively. The maximum episode length T is set to be 5.

Since the action space is much larger, a deeper NN containing eight hidden layers (2048, 1024, 1024, 1024, 512, 512, 512, 512) with neurons is applied as the actor NN, and the NN consisting of three hidden layers (4760, 154, 5) with neurons is utilized as the critic NN. Besides, IL is applied to the training dataset as initial weights for training PPO agents.

Table III illustrates the performance of the NN with just IL on the testing dataset I. The results show that 8702 out of 20000 cases still have violations in terms of the system constraints. The PPO agents with and without IL are both trained for four epochs, where KL_{tar} is set as 0.01. The corresponding PPO training process for the 200-bus system is shown in Fig. 10. The agent without IL does not achieve the same level of reward as the agent with IL. The results from Table III indicates that the PPO agent without IL is actually not trained well and it may need more training epochs since it is very difficult for the agent to sample the "good actions" with positive rewards under large and high dimensional action spaces. Figures 7 and 10 imply that the benefits of initialization with IL would be much more obvious in larger systems.

TABLE III
COMPARISON OF PPO AGENTS' PERFORMANCE ON TESTING DATASET I FOR MODIFIED ILLINOIS 200-BUS SYSTEM

Training method	No. of success data	No. of failure data	Success rate (%)	Maximum $\|\kappa\|$ among success data (%)	Minimum $\|\kappa\|$ among success data (%)	Average $\|\kappa\|$ among success data (%)
IL	11298	8702	56.49	1.28	8.60×10^{-4}	0.12
PPO	16004	3996	80.02	10.52	0.83	4.92
PPO with IL	19990	10	99.95	1.65	0.11	0.61

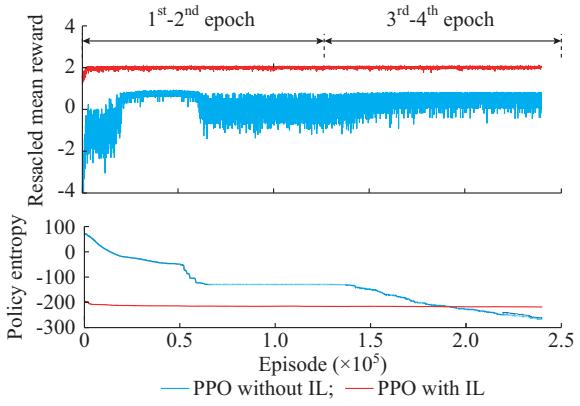

Fig. 10. PPO training process for Illinois 200-bus system.

Figure 11 and Table III illustrate the performance of the trained agent with IL on testing dataset I, which shows that the agent is capable of controlling the system to achieve the optimal cost within 5 steps in 99.95% of 20000 cases (κ is set to be –5% if there are any violations), in which 99.96% can be solved within just one step. The total generation cost resulting from the actions given by the agent is very close to that obtained from the IPS, with the average and maximum absolute value of κ shown in (19) only around 0.61% and 1.65% higher, respectively. Moreover, to solve the 20000 testing data, the running time of the IPS is around 15 hours, whereas it only takes 2 hours for the proposed method, which is around 7.5 times faster than the conventional solver.

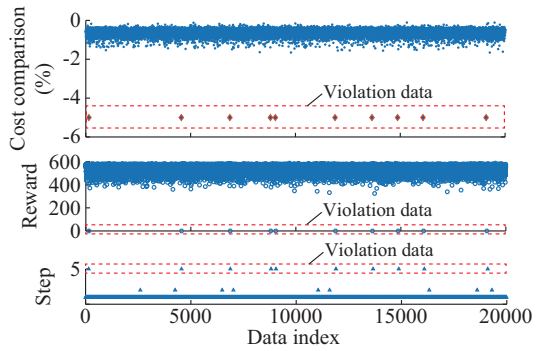

Fig. 11. Performance of PPO agent with IL on testing dataset I for Illinois 200-bus system.

The process of the proposed method for solving the AC OPF problem is divided into two parts: the feed-forward NN calculation and the PF calculation. The feed-forward NN calculation takes polynomial time regarding the input dimensions, which is proven in [44] and [45]. Since the input dimension for the NN shown in Fig. 3 is $2n_{bus}$ (number of buses in the power grid), even for a large-scale power system, this number is manageable that would not lead to high computational complexity. The feed-forward NN calculation time could be seen as a constant time from the "big o" point of view [46]. As for the second part, the PF calculation, which could be seen as "finding the root of the equation", it will definitely cost less computation time than solving the non-convex optimization problem. Besides, if the GPU is applied, the proposed method can be even faster. Although the well-trained agent may take several steps to achieve the near-optimal solutions, from Figs. 8, 9, and 11, the probability of such situation is small. Therefore, the computational burden of the well-trained agent to solve the AC OPF problem is less compared with IPS.

Additionally, the performance of the agent with IL is also tested on the testing dataset II with "dangerous" conditions, and the results are shown in Fig. 12. It can be observed that the trained agent can solve 99.94% of the 5000 cases with "dangerous" conditions (κ is set to be –5% if there are any violations), which attributes to the special design in the actor NN introduced in Section III.

To further validate the effectiveness of the proposed method, a new testing case considering topology changes is per-

formed, where a rated 150 MW wind farm is connected to bus 161. There are 50000 new data generated as a new training dataset including both original topology condition and $N-1$ topology change conditions (one transmission line is randomly chosen if it is tripped or not). Similarly, 10000 new data containing both original topology condition and $N-1$ topology change conditions are generated as a new testing dataset (the system stays in the original topology if the tripped line index is 0). In this testing case, the input for the actor NN is modified as $[P_d, Q_d, |Y_{diag}|, \angle Y_{diag}]$ to reflect the topology changes, where $|Y_{diag}|$ and $\angle Y_{diag}$ are the magnitude and angle of the diagonal elements in the admittance matrix, respectively. The maximum episode length T is set to be 10.

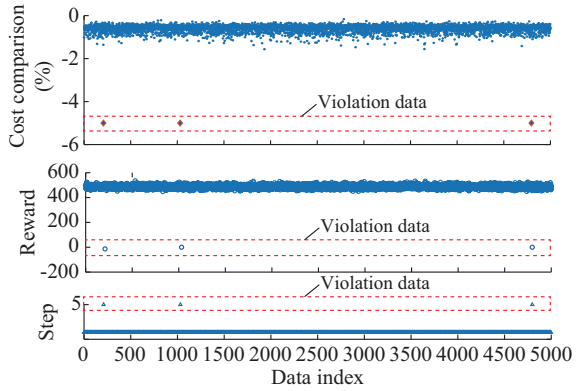

Fig. 12. OPF task results of testing dataset II for Illinois 200-bus system.

Figure 13 shows that the well-trained agent can solve 99.82% of the 10000 cases including both load variation conditions under original topology structure and $N-1$ topology changing contingencies (κ is set to be –5% if there are any violations), where the average and maximum absolute value of κ shown in (19) are 0.772% and 4.2% higher, respectively.

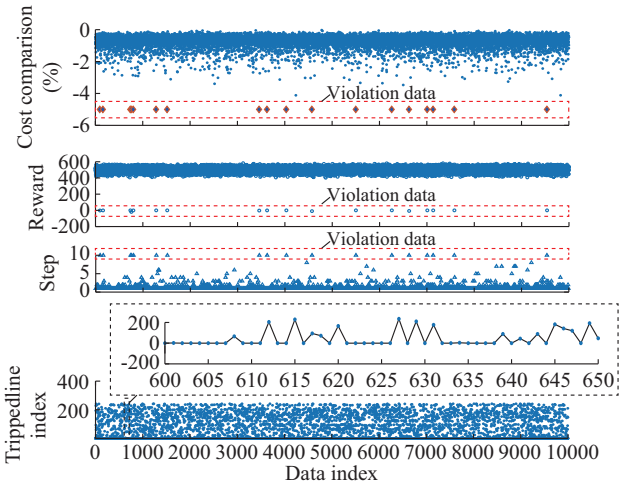

Fig. 13. OPF task results considering topology changes and renewable integration for Illinois 200-bus system.

Although the optimality gap and the number of violation data are slightly larger than the scenarios without topology change as shown in Table III, the well-trained agent is still effective for solving the AC OPF problem in various scenarios, including complex $N-1$ topology changing conditions with renewable energy integration.

Due to the "on-policy" characteristic of the PPO algorithm (the agent cannot be trained to learn to make the optimal actions if the sampled actions do not lead to the positive reward during the training), there still exists a very small portion of violation data, as shown in Figs. 11-13. However, the proposed method shows significant potential for applying it to future power system operation and control, since the negative rewards are small, as shown in Figs. 11-13.

The hyper-parameters adopted in the PPO for both the IEEE 14-bus system and the Illinois 200-bus system are the same due to the robustness of PPO, which can be typical values and suitable for other power grids. These hyper-parameter values are listed in Table IV. The different parameters between the aforementioned two systems are: ① the w value shown in reward function (14) due to the different system costs; ② the training epoch number *epo* since the larger system may require more epochs to achieve the expected performance. Therefore, if the proposed approach is applied in other power grids, the w and z in (14) may need to be updated because of the different generation costs, and the training epoch number *epo* needs further to be tuned. Besides, the structure of DNNs applied in the PPO may be further deeper if the size of test systems is larger, and the IL training loss values could validate the design of the DNNs. The bigger the loss values are, the deeper the DNNs are required.

TABLE IV
HYPER-PARAMETERS ADOPTED IN PPO FOR TEST SYSTEMS

Hyper-parameter	Value
Discount factor γ	0.95
λ in (11)	0.98
KL_{tar}	0.04
Batch size N_{batch}	20
Initial policy log covariance $\Sigma_{\pi\theta}$	–8
Clipping range ε in (12)	0.2

V. CONCLUSION

In this paper, a novel method is proposed for deriving fast AC OPF solutions by using DRL. IL is adopted to initialize the weights of NNs for expediting the training process of the AI agent. The proposed method has been validated on both the IEEE 14-bus and the Illinois 200-bus systems. While the optimal costs from the proposed method are very close to those directly calculated from the interior-point solver, the solution time is improved by at least 7 times compared with a conventional solver. The proposed method also works under $N-1$ contingency on changing network topology, which further validates its effectiveness for power system real-time operation and control.

Future work includes exploring improvements in the performance of the agent in terms of how to further improve the optimality while maintaining the feasibility simultaneously. Furthermore, more realistic system operation conditions should be applied to validate the robustness of the proposed approach. Besides, in terms of expediting the training speed,

the process of selecting the representative training data needs to be considered, or the semi-supervised learning method is taken into account. Also, the parallel distributed coding techniques should be further applied to improve the efficiency of the DRL training process, etc.

REFERENCES

[1] Y. Tang, K. Dvijotham, and S. Low, "Real time optimal PF," *IEEE Transactions on Smart Grid*, vol. 8, no. 6, pp. 2963-2973, Nov. 2017.

[2] R. A. Jabr, "A primal-dual interior-point method to solve the optimal PF dispatching problem," *Optimization and Engineering*, vol. 4, no. 4, pp. 309-336, Dec. 2003.

[3] R. A. Jabr, "Radial distribution load flow using conic programming," *IEEE Transactions on Power Systems*, vol. 21, no. 3, pp. 1458-1459, Aug. 2006.

[4] J. Lavaei and S. H. Low, "Zero duality gap in optimal PF," *IEEE Transactions on Power Systems*, vol. 27, no. 1, pp. 92-107, Feb. 2012.

[5] D. K. Molzahn, J. T. Holzer, B. C. Lesieutre *et al*., "Implementation of a large-scale optimal PF solver based on semidefinite programming," *IEEE Transactions on Power Systems*, vol. 28, no. 4, pp. 3987-3998, Nov. 2013.

[6] R. Madani, S. Sojoudi, and J. Lavaei, "Convex relaxation for optimal PF problem: mesh networks," *IEEE Transactions on Power Systems*, vol. 30, no. 1, pp. 199-211, Jan. 2015.

[7] R. Madani, M. Ashraphijuo, and J. Lavaei, "Promises of conic relaxation for contingency-constrained optimal PF problem," *IEEE Transactions on Power Systems*, vol. 31, no. 2, pp. 199-211, Jan. 2015.

[8] E. Dall'Anese and A. Simonetto, "Optimal PF pursuit," *IEEE Transactions on Smart Grid*, vol. 9, no. 2, pp. 942-959, Mar. 2018.

[9] B. Donon, B. Donnot, I. Guyon *et al*., "Graph neural solver for power systems," in *Proceedings of 2019 International Joint Conference on NNs (IJCNN)*, Budapest, Hungary, Jul. 2019, pp. 1-8.

[10] X. Pan, T. Zhao, and M. Chen. (2019, May). Deep OPF: deep NN for DC optimal PF. [Online]. Available: https://arxiv.org/abs/1905.04479

[11] D. Deka and S. Misra. (2019, Feb.). Learning for DC-OPF: classifying active sets using neural nets. [Online]. Available: https://arxiv.org/abs/1902.05607

[12] A. S. Zamzam and K. Baker. (2019, Sept.). Learning optimal solutions for extremely fast AC optimal PF. [Online]. Available: https://arxiv.org/abs/1910.01213

[13] F. Fioretto, T. W. K. Mak, and P. V. Hentenryck. (2019, Sept.). Predicting AC optimal PFs: combining deep learning and Lagrangian dual method. [Online]. Available: https://arxiv.org/abs/1909.10461

[14] V. Francois-Lavet, P. Henderson, R. Islam *et al.* (2018, Nov.). An introduction to deep reinforcement learning. [Online]. Available: https://arxiv.org/abs/1811.12560

[15] Y. Xu, W. Zhang, W. Liu *et al*., "Multiagent-based reinforcement learning for optimal reactive power dispatch," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1742-1751, Nov. 2012.

[16] T. Yu, J. Liu, K. W. Chan *et al*.,"Distributed multi-step $Q(\lambda)$ learning for optimal PF of large-scale power grids," *Electrical Power and Energy Systems*, vol. 42, no. 1, pp. 614-620, Nov. 2012.

[17] E. Mocanu, D. C. Mocanu, P. H. Nguyen *et al*., "Online building energy optimization using deep reinforcement learning," *IEEE Transactions on Smart Grid*, vol. 10, no. 4, pp. 3698-3708, Jul. 2019.

[18] R. Diao, Z. Wang, D. Shi *et al.* (2019, Apr.). Autonomous voltage control for grid operation using deep reinforcement learning. [Online]. Available: https://arxiv.org/abs/1904.10597

[19] J. Duan, D. Shi, R. Diao *et al*., "Deep-reinforcement-learning-based autonomous voltage control for power grid operations," *IEEE Transactions on Power Systems*, vol. 35, no. 1, pp. 814-817, Jan. 2020.

[20] D. Cao, W. Hu, J. Zhao *et al*., "A multi-agent deep reinforcement learning based voltage regulation using coordinated PV inverters," *IEEE Transactions on Power Systems*, vol. 35, no. 5, pp. 4120-4123, Jun. 2020.

[21] Z. Yan and Y. Xu, "Data-driven load frequency control for stochastic power systems: a deep reinforcement learning method with continuous action search," *IEEE Transactions on Power Systems*, vol. 34, no. 2, pp. 1653-1656, Mar. 2019.

[22] Q. Huang, R. Huang, W. Hao *et al*., "Adaptive power system emergency control using deep reinforcement learning," *IEEE Transactions on Smart Grid*, vol. 11, no. 2, pp. 1171-1182, Mar. 2020.

[23] Y. Ji, J. Wang, J. Xu *et al*., "Real-time energy management of a microgrid using deep reinforcement learning," *Energies*, vol. 12, no. 12, pp. 1-21, Jun. 2019.

[24] V. François-Lavet, D. Taralla, D. Ernst *et al*., "Deep reinforcement learning solutions for energy microgrids management," in *Proceedings of European Workshop on Reinforcement Learning (EWRL 2016)*, Barcelona, Spain, Sept. 2016, pp. 1-7.

[25] R. Rocchetta, L. Bellani, M. Compare *et al*., "A reinforcement learning framework for optimal operation and maintenance of power grids," *Applied Energy*, vol. 241, pp. 291-301, May 2019.

[26] J. Zhang, C. Lu, J. Si *et al*., "Deep reinforcement learning for short-term voltage control by dynamic load shedding in China southern power grid", in *Proceedings of IEEE 2018 International Joint Conference on NNs (IJCNN)*, Rio de Janeiro, Brazil, Jul. 2018, pp. 1-8.

[27] Z. Wan, H. Li, H. He *et al*., "Model-free real-time EV charging scheduling based on deep reinforcement learning," *IEEE Transactions on Smart Grid*, vol. 10, no. 5, pp. 5246-5257, Sept. 2019.

[28] G. Zhang, W. Hu, D. Cao *et al*., "Deep reinforcement learning based approach for proportional resonance power system stabilizer to prevent ultra-low-frequency oscillations," *IEEE Transactions on Smart Grid*, vol. 11, no. 6, pp. 5260-5272, Nov. 2020.

[29] J. Schulman, F. Wolski, P. Dhariwal *et al*. (2017, Jul.). Proximal policy optimization algorithms. [Online]. Available: https://arxiv. org/abs/1707.06347

[30] R. S. Sutton and A. G. Barto, *Reinforcement Learning – An Introduction*, 2nd ed., Cambridge: MIT Press, 2018.

[31] Y. LeCun, Y. Bengio, and G. Hinton. "Deep learning," *Nature*, vol. 521, pp. 436-444, May 2015.

[32] H. Daumé III (2017, Jan.). *A Course in Machine Learning.* [Online]. Available: http://ciml.info/dl/v0_99/ciml-v0_99-all.pdf

[33] V. Kurin. (2017, Aug.). *Introduction to Imitation Learning*. [Online]. Available: https://blog. statsbot. co/introduction-to-imitation-learning-32334c3b1e7a

[34] J. Schulman, P. Moritz, S. Levine *et al*. (2015, Jun.). High-dimensional continuous control using generalized advantage estimation. [Online]. Available: https://arxiv.org/abs/1506.02438

[35] J. Schulman, S. Levine, P. Moritz *et al*. (2015, Feb.). Trust region policy optimization. [Online]. Available: https://arxiv.org/abs/1502.05477

[36] G. Brockman, V. Cheung, L. Pettersson *et al*. (2016, Jun.). OpenAI gym. [Online]. Available: https://arxiv.org/abs/1606.01540

[37] F. Pedregosa, G. Varoquaux, and J. Vanderplas. "Scikit-learn: machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, Oct. 2011.

[38] D. J. C. MacKay, *Information Theory, Inference and Learning Algorithms*, Cambridge: Cambridge University Press, 2003.

[39] D. P. Kingma and J. Ba. (2014, Dec.). Adam: a method for stochastic optimization. [Online]. Available: https://arxiv.org/abs/1412.6980

[40] Illinois Center for a Smarter Electric Grid (ICSEG) (1993, Aug.). IEEE 14-bus system. [Online]. Available: https://icseg. iti. illinois. edu/ieee-14-bus-system/

[41] Electric Grid Test Case Repository (2020, Jul.). Illinois 200-bus system: ACTIVSg200. [Online]. Available: http://electricgrids. engr. tamu. edu/ele ctric-grid-test-cases/activsg200/

[42] PYPOWER (2018, Jun.). PYPOWER 5.1.4. [Online]. Available: https://pypi.org/project/PYPOWER/

[43] R. D. Zimmerman, C. E. Sanchez, and R. J. Thomas, "MATPOWER: steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Transactions on Power Systems*, vol. 26, no. 1, pp. 12-19, Feb. 2011.

[44] G. Serpen and Z. Gao, "Complexity analysis of multilayer perceptron NN embedded into a wireless sensor network," *Procedia Computer Science*, vol. 36, pp. 192-197, Nov. 2014.

[45] Kasper Fredenslund (2018, Mar.). Computational complexity of NNs. [Online]. Available: https://kasperfred.com/series/introduction-to-neural-networks/computational-complexity-of-neural-networks

[46] T. H. Cormen, C. E. Leiserson, R. L. Rivest *et al*., *Introduction to Algorithms*, 3rd ed., Cambridge: MIT Press, 2009.

Yuhao Zhou received the B.S. and M.S. degrees both in electrical engineering from Huazhong University of Science and Technology, Wuhan, China, in 2013 and 2016, respectively. Currently, he is pursuing the Ph. D. degree in the University of Texas at Arlington, Arlington, USA, as a member of the Energy Systems Research Center (ESRC). He was a summer research intern with GEIRI North America, San Jose, USA, in 2019. His research interests include AI-related applications in power system operation and control, dynamic equivalent modeling of wind farms, and arc flash protection.

Bei Zhang received the B.S. and M.S. degrees from Shanghai Jiao Tong University, Shanghai, China, in 2009 and 2012, respectively, both in electrical engineering. She received the Ph. D. degree in electrical engineering from Texas A&M University, College Station, USA, in 2017. Her research interests include power system reliability and resilience, electric vehicle, and electricity market.

Chunlei Xu received the B.S. degree in electrical engineering from Shanghai Jiao Tong University, Shanghai, China, in 1999. He currently leads the Dispatching Automation Department at Jiangsu Electrical Power Company, Nanjing, China. His research interests include power system operation and control, and wide-area measurement system (WAMS).

Tu Lan received the B.S. degree from The University of Nottingham, Ningbo, China, in 2012, and the M.S. degree from San Jose State University, San Jose, USA, in 2018. He was a Machine Learning Engineer at GEIRI North America, San Jose, USA, in 2019. He is currently pursuing the Ph.D. Degree in Southern Methodist University, Dallas, USA. His research interests include DRL-based applications in power system operation and controls, computer vision based power grid fault detection, and data mining in power system stability and reliability.

Ruisheng Diao received his Ph.D. degree in electrical engineering from Arizona State University, Tempe, USA, in 2009. Serving as a Principle Investigator (PI)/co-PI, he has been managing and supporting a portfolio of research projects in the area of power system modeling, dynamic simulation, online security assessment and control, dynamic state estimation, integration of renewable energy, AI and high performance computing (HPC) implementation in power systems, etc. He has published about 90 peer reviewed journal and conference papers, as well as dozens of technical reports. Dr. Diao is the recipient of the 2018 R&D 100 Awards, 2018 IEEE PES Conference Prize Paper Award, and multiple IEEE PES best paper awards. He is a senior member of IEEE, Editor for IEEE Transactions on Power Systems, IEEE Power Engineering Letters, IEEE Access, IET Generation, Transmission & Distribution, and a registered Professional Engineer (PE) in Washington State, USA. Dr. Diao is now with GEIRI North America, San Jose, USA, as Deputy Department Head, AI & System Analytics. His research interests include power system modeling, dynamic simulation, online security assessment and control, dynamic state estimation, integration of renewable energy, AI and high performance computing (HPC) implementation in power systems, etc.

Di Shi received the Ph. D. degree in electrical engineering from Arizona State University, Tempe, USA. He is currently the Director of Fundamental R&D Center and Department Head of the AI & System Analytics Group, GEIRI North America, San Jose, USA. Prior to joining GEIRI North America, he was a Research Staff Member with NEC Laboratories America. He is an Editor of the IEEE Transactions on Smart Grid and the IEEE Power Engineering Letters. His research interests include phase measurement unit (PMU) data analytics, AI for power systems, energy storage systems, and IoT for power systems.

Zhiwei Wang received the B.S. and M.S. degrees in electrical engineering from Southeast University, Nanjing, China, in 1988 and 1991, respectively. He is President of Global Energy Interconnection Research Institute North America, San Jose, USA. Prior to this assignment, he served as President of State Grid US Representative Office, New York, USA, from 2013 to 2015, and President of State Grid Wuxi Electric Power Supply Company, Wuxi, China, from 2012-2013. His research interests include power system operation and control, relay protection, power system planning, and WAMS.

Wei-Jen Lee received the B.S. and M.S. degrees in electrical engineering from National Taiwan University, Taipei, China, in 1978 and 1980, respectively, and the Ph.D. degree in electrical engineering from The University of Texas at Arlington, Arlington, USA, in 1985. In 1985, he joined The University of Texas at Arlington, where he is currently a Professor with the Department of Electrical Engineering and the Director of the Energy Systems Research Center. He has been involved in research on PF, transient and dynamic stability, voltage stability, short circuits, relay coordination, power quality analysis, renewable energy, and deregulation for utility companies. Prof. Lee is a Registered Professional Engineer in the State of Texas. Dr. Lee has been involved in the revision of IEEE Std. 141, 339, 551, 739, and dot 3000 series development. He is the Vice President of the IEEE Industry Applications Society (IEEE-IAS), a distinguished lecturer of the IEEE-IAS, a member of IEEE Fellow Committee, the Associate Editor of IEEE/IAS, and International Journal of Power and Energy Systems. He is the project manager of IEEE/NFPA Collaboration on Arc Flash Phenomena Research Project. His research interests include power system operation and control, utility deregulation, renewable energy, arc flash hazard and electrical safety, etc.