Deep Reinforcement Learning Based Real-time AC Optimal Power Flow Considering Uncertainties

Yuhao Zhou, *Student Member*, *IEEE*, Wei-Jen Lee, *Fellow*, *IEEE*, Ruisheng Diao, *Senior Member*, *IEEE*, and Di Shi, *Senior Member*, *IEEE*

***Abstract*—Modern power systems are experiencing larger fluctuations and more uncertainties caused by increased penetration of renewable energy sources (RESs) and power electronics equipment. Therefore, fast and accurate corrective control actions in real time are needed to ensure the system security and economics. This paper presents a novel method to derive real-time alternating current (AC) optimal power flow (OPF) solutions considering the uncertainties including varying renewable energy and topology changes by using state-of-the-art deep reinforcement learning (DRL) algorithm, which can effectively assist grid operators in making rapid and effective real-time decisions. The presented DRL-based approach first adopts a supervised-learning method from deep learning to generate good initial weights for neural networks, and then the proximal policy optimization (PPO) algorithm is applied to train and test the artificial intelligence (AI) agents for stable and robust performance. An ancillary classifier is designed to identify the feasibility of the AC OPF problem. Case studies conducted on the Illinois 200-bus system with wind generation variation and $N-1$ topology changes validate the effectiveness of the proposed method and demonstrate its great potential in promoting sustainable energy integration into the power system.**

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

I. Introduction

ALTERNATING current (AC) optimal power flow (OPF) remains an essential but challenging optimization problem for the operation and control of modern power system with high penetration of renewable energy sources (RESs). Many approaches in the literature have been proposed in recent decades to solve this non-convex and NP-hard problem, the solution of which is typically time-intensive to achieve the convergence for real-time application [1]. With the increasing penetration of RES, modern power systems are experiencing larger fluctuations and more uncertainties, causing grand challenges for operators to make prompt decisions. Thus, there is a compelling need for deriving real-time AC OPF controls to tackle the uncertainties caused by the RES for secure and economic operation of power system.

To address this issue, [1] proposes a single-iteration quasi-Newton method to expedite the real-time AC OPF solutions with the prerequisite for an accurate estimation of the second-order information. In [2], linearized AC power flow equations are applied to achieve real-time OPF in distribution systems. With the recent success of deep learning (DL), several supervised-learning-based methods are proposed to approximate OPF solutions with improved solution speed. In [3] and [4], deep neural networks (DNNs) are utilized to solve the direct current (DC) OPF problem. In [5], the worst-case guarantees of the DNN for DC OPF are analyzed. Reference [6] applies the DNN and [7] uses the graph neural network to approximate optimal generator set-points from the solutions of AC OPF problem. However, the small training and testing loss values cannot guarantee the feasibility of solutions under various operating conditions. To deal with this issue, [8] and [9] utilize the penalized loss function to capture the operational constraints. Reference [10] adopts the zero-order optimization technique on the IEEE 30-bus system that achieves the feasibility among 98% of the testing data. With the recent success of deep reinforcement learning (DRL) algorithms adopted in power system controls, multi-agent $Q(\lambda)$ learning is implemented to perform OPF tasks under discretized action spaces in [11]. In [12], an agent is trained to achieve the optimality while satisfying feasibility under continuous action space by applying the deep deterministic policy gradient algorithm aiming at solving the AC OPF problem. However, the robustness of these methods regarding load variations, uncertainties of RES, and topology changes ($N-1$ contingencies) needs to be further investigated.

Inspired by the efforts above, this paper presents a novel DRL-based approach, the contributions of which are summarized below.

1) It adopts the proximal policy optimization (PPO) algorithm introduced in [13] to first train DRL agents offline for solving the AC OPF problem considering RES and $N-1$ topology changes. The well-trained agents are then applied in real-time applications with periodic updating. The state space uses diagonal elements of the network admittance matrix to represent grid topology; hence, the well-trained agent

Manuscript received: December 21, 2020; revised: March 25, 2021; accepted: August 13, 2021. Date of CrossCheck: August 13, 2021. Date of online publication: September 17, 2021.

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

R. Diao and D. Shi are with the AI & System Analytics Group at GEIRINA, San Jose, CA 95134, USA (e-mail: ruisheng.diao@gmail.com; di.shi@asu.edu).

DOI: 10.35833/MPCE.2020.000885

remains effective and robust during the online implementation regarding the uncertainty of topology changes.

2) To facilitate the agent's learning speed and performance during the offline training process, the supervised-learning regression method is applied to initialize the weights for the DRL agent, serving as an "initial guide".

3) A reward function is carefully designed to tackle the feasibility issue, where the DRL agent learns an optimal stochastic policy. Therefore, compared with running many stochastic scenarios regarding the uncertainties under high penetration of RES and $N-1$ topology changes, the proposed method has the advantage to be applied in real-time security-constrained economic dispatch applications.

Numerical experiments conducted on the Illinois 200-bus system with RES and realistic operational data extracted from [14] demonstrate the effectiveness and robustness of the proposed approach. The online testing results show that a well-trained agent can obtain near-optimal solutions with a computation time of at least one order less than that obtained by the interior point solver (IPS). It manifests a great promise of employing artificial intelligence (AI) techniques in the real-time control of power system, especially with large penetration of RES. Moreover, an ancillary and independent "alarm" function is designed to help system operators rapidly identify the feasibility of the AC OPF problem under various operating conditions.

The remainder of this paper is organized as follows. Section II provides the problem formulation and the preliminaries of DRL algorithms. In Section III, the detailed procedures of the proposed methodology are illustrated. In Section IV, numerical experiments are conducted on the Illinois 200-bus system to demonstrate the performance of DRL agents and the effectiveness of the proposed method. Finally, Section V draws the conclusion and presents future work.

II. Problem Formulation and Preliminaries

A. Problem Formulation of AC OPF

Considering an AC system with a set of buses $N_b=\{1,2,\ldots,n_B\}$, a set of transmission lines L with a total of n_{br} branches, and the generator buses $G\subset N_b$ with a total number of n_G generators, the AC OPF problem can be formulated as:

$$\begin{cases} \min \sum_{k\in G} f_k(P_{gk}) = c_{k2}P_{gk}^2 + c_{k1}P_{gk} + c_{k0} \quad \forall k\in G \\ \text{s.t.} \quad P_{gk}^{\min} \le P_{gk} \le P_{gk}^{\max} \quad \forall k\in G \\ \quad Q_{gk}^{\min} \le Q_{gk} \le Q_{gk}^{\max} \quad \forall k\in G \\ \quad V_k^{\min} \le |V_k| \le V_k^{\max} \quad \forall k\in N_b \\ \quad |S_{lm}| \le S_{lm}^{\max} \quad \forall (l,m)\in L \\ \quad P_{gk}-P_{dk} = \sum_{l\in N_b(k)} \mathrm{Re}\left\{V_k(V_k^*-V_l^*)y_{kl}^*\right\} \\ \quad Q_{gk}-Q_{dk} = \sum_{l\in N_b(k)} \mathrm{Im}\left\{V_k(V_k^*-V_l^*)y_{kl}^*\right\} \end{cases} \tag{1}$$

where y_{kl} is the admittance between buses k and l; subscripts g and d represent the generator and load, respectively; P and Q are the active power and reactive power, respectively; and V_k and S_{lm} are the bus voltage magnitudes at bus k and branch flow limit between bus l and m, respectively. In the model above, the wind farm is also considered as a PV (constant power and constant voltage) bus, and thus the corresponding operational limit of the first constraint in (1) becomes $P_{g\_wind\_k}^{\min}=P_{g\_wind\_k}=P_{g\_wind\_k}^{\max}$, where $P_{g\_wind\_k}$ is the active power output regarding the k^{th} wind power plant. Also, the modeled wind farms follow a real-world protocol for reactive power constraints, which requires the power factor to be 0.95 or less [15]. The objective is to find the optimal set points for all generators in the system, such that the quadratic cost function is minimized subject to operational security limits shown in (1).

B. DL for Solving AC OPF

The motivation of applying DL to solve the AC OPF is to find a mapping function represented by a DNN $\boldsymbol{\beta}_\zeta$ parameterized by ζ between the operating states and optimal generator settings such that the solving speed can be improved significantly. Unlike [8]-[12], $N-1$ topology changes are also considered in this paper to make it more robust for power system operation during the online application process. Therefore, the loads at each bus and admittance information (the magnitude and angle of diagonal elements in the admittance matrix $\boldsymbol{Y}$), $\boldsymbol{s}=[P_d, Q_d, |Y_{diag}|, \angle Y_{diag}]$, are applied as the input, and the optimal generator set-points for each generator $\hat{\boldsymbol{a}}=[P_g, V_g]$ are set as the output. The learning task can be formulated as a supervised regression problem to minimize the $L-2$ norm loss function shown in (2), where the optimal generator set-points shown as the "labels" $\hat{\boldsymbol{a}}$, could be obtained by running the AC OPF solver offline for the training dataset D_{train} with a size of N_{train}.

$$\min_{\zeta} \sum_{(\boldsymbol{s},\hat{\boldsymbol{a}})\in D_{train}} \frac{1}{N_{train}} \|\hat{\boldsymbol{a}}-\boldsymbol{\beta}_\zeta(\boldsymbol{a}|\boldsymbol{s})\|_2^2 \tag{2}$$

However, if the DNN is only trained by adopting (2), the feasibility of the AC OPF problem cannot be guaranteed after running the power flow (PF) solver during online implementation even though the loss is small due to operational security limit violations defined in (1). Although [8] - [10] adopt the penalty function to deal with the feasibility issue, the penalty coefficient needs to be further tuned regarding the training performance. In this paper, the DRL framework is adopted to address the feasibility issue.

C. PPO Algorithm with Clipped Surrogate Loss

The goal of DRL is to train an agent aiming at learning an optimal policy π^* that maximizes the expected reward return by continuously interacting with the environment [16]. Compared with other state-of-the-art policy gradient algorithms, PPO has been verified to have the best or comparable performance in the various DRL benchmark game envrionments with the continuous control spaces while its hyperparameters are simpler to be tuned compared with other DRL algorithms [17]. There are two versions of the PPO algorithm: an adaptive KL-divergence penalty version and a clipped surrogate loss version. And the second version has been validated to have the best performance on all continuous control tasks [17]. Therefore, the PPO with surrogate

loss version is chosen as the DRL algorithm in this paper. A brief introduction of the PPO with the clipped surrogate loss function is shown as follows.

Categorized as one actor-critic type of RL algorithms, the PPO agent consists of two DNNs, where the first DNN, the "actor", is trained to learn the stochastic optimal policy, and the second DNN, the "critic", is designed to estimate the value function. The PPO algorithm ensures an improved performance compared with other policy gradient algorithms due to the following two kinds of enhancement regarding the "actor" updates. Firstly, the generalized advantage estimation (GAE) function $A_t^{GAE(\gamma,\lambda)}$ is utilized during the "actor" training process to reduce the variance of the estimation as shown in (3) [18].

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

where $V^\pi(s_t)$ is the state value representing how good a state is by calculating the expected reward starting from state s_t at time step t following a certain policy, which is the output of the "critic" network; λ controls the average degree of n-step advantage values; r_t is the immediate reward from the environment at time step t; and $\gamma \in [0, 1]$ is the discount factor on the future reward.

Secondly, the PPO algorithm updates the "actor" parameters within an appropriate trust region, and this helps avoid falling off the "cliff" from the hyper-surfaces of the reward functions which may be hard to escape from. Such a safe update is achieved by modifying the objective function L^{PPO} shown in (4). $\hat{E}_t$ is the expectation operator; and $clip(\cdot)$ means when the value of $h_t(\theta)$ is outside the range of $[1-\varepsilon, 1+\varepsilon]$, $h_t(\theta)$ will be forced to be either $1-\varepsilon$ or $1+\varepsilon$.

$$\begin{cases} L^{PPO}(\theta) = \hat{E}_t[\min(h_t(\theta)A_t^{\pi_{\theta_{old}}}, clip(h_t(\theta), 1-\varepsilon, 1+\varepsilon)A_t^{\pi_{\theta_{old}}})] \\ h_t(\theta) = \dfrac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \end{cases} \tag{4}$$

where θ indicates the parameters of the DNN $\pi_\theta(a_t|s_t)$ of the "actor"; ε determines the range of the trust region for the update; and the advantage value A_t is calculated from (3) $A_t^{GAE(\gamma,\lambda)}$. The minimization operator makes sure that the new policy does not benefit from going too far away from the old policy and thus regulates the update of the DNN parameters.

Besides, the policy π_θ in PPO is stochastic, which is parameterized as a conditional Gaussian policy $\pi_\theta \sim N(\mu_\theta(s), \Sigma_{\pi\theta})$. The mean value $\mu_\theta(s)$ is the output of the DNN in the "actor", and the covariance $\Sigma_{\pi\theta}$ is initially assigned manually but will be updated during backpropagation. Besides, θ_{old} indicates the policy parameters before updating the "actor".

As for the DNN in the "critic" parameterized by φ, which is designed to estimate the value function $V_\varphi^\pi(s_t)$, the objective function to update the "critic" is shown in (5).

$$\min_\varphi \sum_{(s_t, R_t)\in D_{batch}} \frac{1}{N_{batch}} \|R_t - V_\varphi^\pi(s_t)\|_2^2 \tag{5}$$

$$R_t = \sum_{k=0}^{el-1}\gamma^k r_{t+k+1} \tag{6}$$

where R_t is the discounted accumulated reward; D_{batch} is the trajectories accumulated from the agent interacting with the environment with batch size N_{batch}; and el is the episode length when the agent interacts with its environment.

III. Proposed DRL-based AC OPF Solutions

The proposed DRL-based framework for AC OPF solutions is illustrated in Fig. 1, which is referenced from the "Grid-Mind" framework [19]. The PPO agent firstly is trained offline by interacting with the power system simulation environment to learn an optimal policy; then the well-trained agent can provide the suggested actions based on the measured state data in the power system to achieve near-optimal AC OPF solutions in real time.

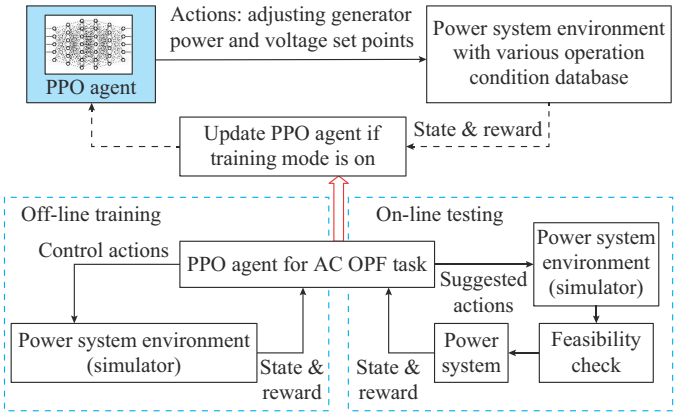

Fig. 1. DRL-based framework to solve AC OPF problem.

A. State and Action Spaces

The ***state***, which is the input for the PPO agent, includes the active and reactive power (P_{di} and Q_{di}) of system loads at all buses ($i \in N_b$), the magnitude and angle of the diagonal elements of the admittance matrix $\boldsymbol{Y}$, and all n_G generators' initial active power setting P_{gj} and voltage setting V_{gj} ($j \in G$), as denoted in (7). The MinMax scaling preprocessing technique [20] is conducted on the [0, $2n_B$) columns and [$2n_B$, $4n_B$) columns of this vector individually before passing it to the agent to handle different scales of various parameters.

$$\begin{aligned} \boldsymbol{state} = [&P_{d1}, P_{d2}, ..., P_{dn_B}, Q_{d1}, Q_{d2}, ..., Q_{dn_B}, |Y_{diag\_1}|, \\ &|Y_{diag\_2}|, ..., |Y_{diag\_n_B}|, \angle Y_{diag\_1}, \angle Y_{diag\_2}, ..., \angle Y_{diag\_n_B}, \\ &P_{g1}, P_{g2}, ..., P_{gn_G}, V_{g1}, V_{g2}, ..., V_{gn_G}] \end{aligned} \tag{7}$$

The action spaces are the incremental adjustments made to generator set-points shown in (8) instead of optimal generator set points due to the training interactions between the DRL agent and its environment. Then, the well-trained DRL agent could adaptively achieve the optimal status with several adjustment steps during the online testing process, although our training target is to achieve the optimality in one step.

$$\boldsymbol{action} = [\Delta P_{g1}, \Delta P_{g2}, ..., \Delta P_{gn_G}, \Delta V_{g1}, \Delta V_{g2}, ..., \Delta V_{gn_G}] \tag{8}$$

B. Neural Network Structure

The DNN structures for the "actor" and the "critic" in PPO are shown in Fig. 2 and Fig. 3, respectively, where the input, ***state***, is denoted in (7).

Due to the consideration of system topology information, the input state dimension is two times larger compared with those in [8]-[10]; therefore, to further effectively extract the features from the inputs for the "actor", one convolutional layer from the convolutional neural network (CNN) is utilized first and then connects a fully connected (FC) layer with the following hidden layers shown in Fig. 2. Similar to the applications of CNN in DL for image processing, a convolutional kernel conducts the convolutional feature extraction calculations, and thus, this structure is more suitable for our application with large input dimension spaces. However, to maintain the information of every bus in the "actor", the pooling layer is not adopted. The convolutional layer parameters are set as follows: ① the "stride" parameter, which is the step size for moving the kernel window (yellow block shown in Fig. 2) in the convolutional calculation, is set to be 1; ② "zero padding" is applied to maintain the width and height of the input. In addition, the kernel filter size is set as [4, 4, 1, k_{cnn}] as shown in Fig. 2, and the rectified linear unit (ReLU) activation function is applied. In the hidden layers for both "actor" and "critic", the ReLU activation function is adopted to effectively prevent the vanishing gradient. Besides, the *sigmoid* activation in the "actor" output layer makes sure that the output can have bounded negative or positive values. And this output of the DNN in the "actor" acts as the mean value of the stochastic policy in the PPO training shown in Section II-C. On the other hand, the one neuron in the output layer of "critic" has no activation, which outputs the state value $V_{\varphi}^{\pi}(s_t)$. Moreover, as different generator set points represent different state values, all elements in (7) are applied as the input for the neural network in "critic". The outputs of the output layer are the optimal generator settings.

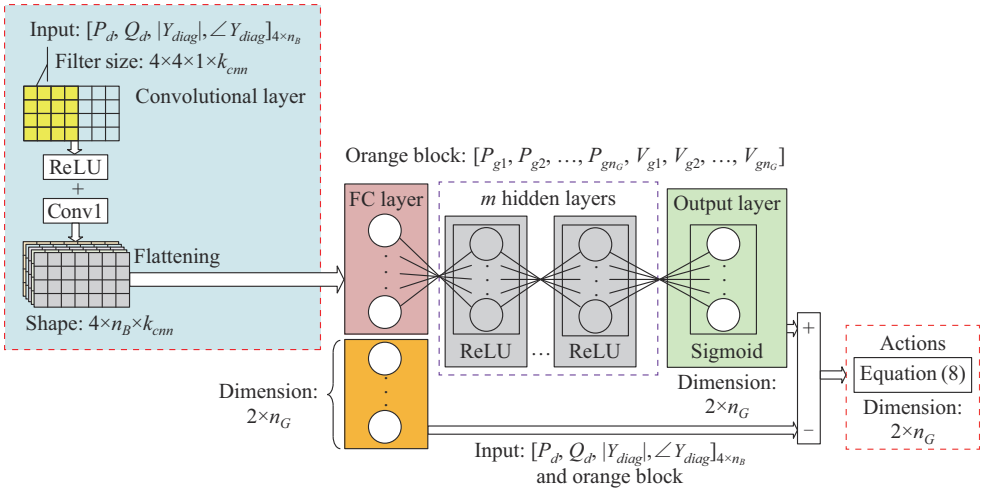

Fig. 2. DNN structure for "actor" in PPO training.

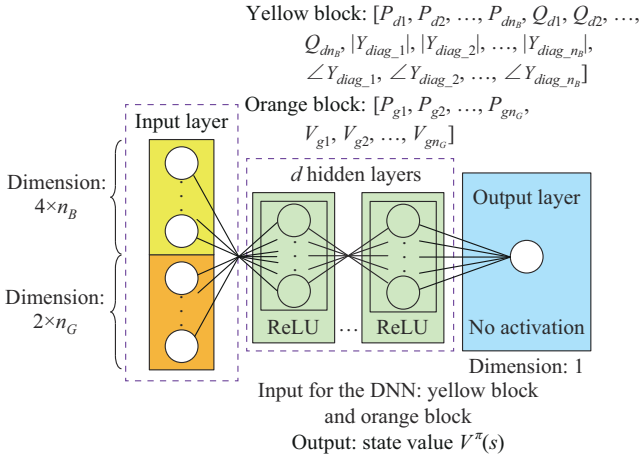

Fig. 3. DNN structure for "critic" in PPO training.

C. DL-based Initialization

To facilitate the DRL training process for solving the AC OPF problem with large state and action spaces, if the agent starts training from a good initial status, it could solve the sample inefficiency caused by numerous trials and errors without experts' demonstration. Therefore, the DRL training process could be sped up and become more effective. On the other hand, the DL training result could serve as a validation process for the structure of the DNN in the "actor". However, the difference here is that the "labels" in the initialization process become optimal generator setting adjustments shown in (8) for further DRL training. After collecting the optimal action labels and states to by the training dataset D_{train} with the size of N_{DL} by running AC OPF solver offline, adopting (9) as the loss function and applying the first-order optimizer such as stochastic gradient descent, the initial mean value $\mu_{\theta}(s)$ of the stochastic policy π_{θ} in PPO agent could be trained to clone the optimal generator settings from AC OPF solution results.

$$\min_{\theta} \sum_{(s_t,\hat{a}_t)\in D_{train}} \frac{1}{N_{DL}} \|\hat{a}_t - \mu_{\theta}(a_t|s_t)\|_2^2 \tag{9}$$

D. Offline Training Process of PPO Agents in Solving AC OPF Problem

Figure 4 illustrates the interaction between the DRL agent and the power system environment within one episode (one training case), which starts from the initialization of the case through "*reset*(·)" until the "end" in the figure with "done" set by "*step*(·)". "*reset*(·)" function initializes a training case by retrieving the loads, generators settings and current system topology information to formulate the initial state s_t; "*step*(·)" function applies the agent's action, runs the AC power flow with enforced generators' reactive power limits, and then provides the agent with the resulting state, "done" signal, the corresponding reward, and updates the generator set-points. Since it is difficult to determine whether the optimal cost has been reached, the "done" signal becomes "true" when ① the reward is positive; ② the PF solver is diverged, which indicates the status of "game over" and thus a large negative reward of −5000 is given to train the agent to avoid such actions; or ③ the maximum number of steps has been reached.

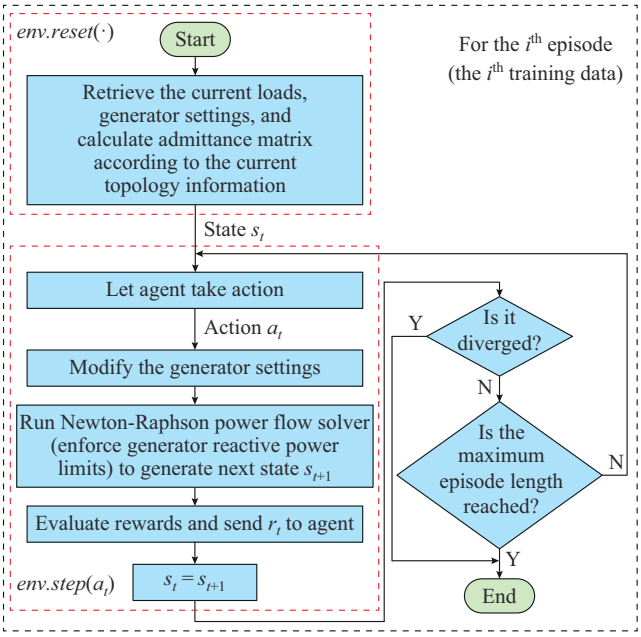

Fig. 4. Flowchart of power system environment interacting with an agent.

The detailed design of the reward function is given in (10).

$$reward = \begin{cases} -5000 & \text{PF solver is diverged} \\ R_{pg\_v} + R_{v\_v} + R_{br\_v} & \text{there are constraint violations} \\ 1000 - 0.01 Costs_{gen} & \text{solutions are feasible} \end{cases} \tag{10}$$

where R_{pg_v}, R_{v_v}, and R_{br_v} are shown in (11) corresponding to negative rewards if violations of any inequality constraints are detected, including: ① the active power limits of generators; ② the voltage magnitude limits of buses; and ③ the thermal flow limits (in both directions) of transmission lines. Variable $Costs_{gen}$ in (10) is the total generation cost value of the power system. Equation (11) corresponds to the operational limits in the original problem shown in (1); on the other hand, running the PF solver with the generator reactive power enforcement in the "*step*(·)" function corresponds to the operational limits in the original problem shown in (1). Therefore, positive rewards suggest feasible solutions. Besides, if solutions are feasible, (10) linearly transforms the convex cost function into a concave reward function for the DRL training, which aims at maximizing the rewards.

$$\begin{cases} R_{pg\_v} = \begin{cases} -\left(\sum_{i=1}^{n_G}(P_{gi} - P_{gi}^{\max}) + \sum_{i=1}^{n_G}(P_{gi}^{\min} - P_{gi})\right) & \forall i \in G \\ P_{gi} - P_{gi}^{\max} > 0 \text{ or } P_{gi}^{\min} - P_{gi} > 0 \\ 0 \quad \text{otherwise} \end{cases} \\ R_{v\_v} = \begin{cases} -\left(\sum_{i=1}^{n_B}(V_i - V_i^{\max}) + \sum_{i=1}^{n_B}(V_i^{\min} - V_i)\right) & \forall i \in N_b \\ V_i - V_i^{\max} > 0 \text{ or } V_i^{\min} - V_i > 0 \\ 0 \quad \text{otherwise} \end{cases} \\ R_{br\_v} = \begin{cases} -\left(\sum_{i=1}^{n_{br}}(S_{lm\_i} - S_{lm\_i}^{\max}) + \sum_{i=1}^{n_{br}}(S_{ml\_i} - S_{ml\_i}^{\max})\right) & \forall (l,m) \in L \\ S_{lm\_i} - S_{lm\_i}^{\max} > 0 \text{ or } S_{ml\_i} - S_{ml\_i}^{\max} > 0 \\ 0 \quad \text{otherwise} \end{cases} \end{cases} \tag{11}$$

With the DL-based initialization, the DRL training can produce more reliable and improved results. A brief illustration of PPO training is shown in Fig. 5 and further illustrated in Algorithm 1. After interacting with the power system environment to collect batch-sized trajectories, the PPO agent is updated/trained accordingly.

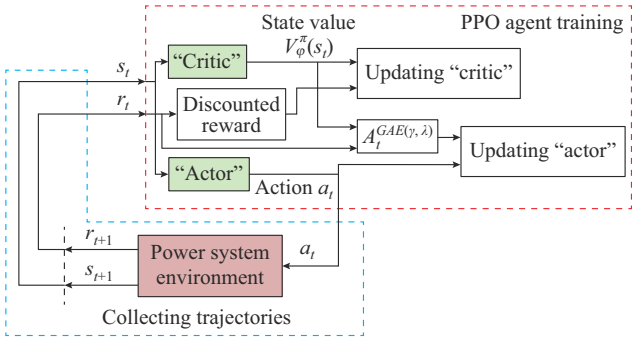

Fig. 5. A brief illustration of PPO training in this paper.

In Algorithm 1, one *epoch* means that all the training data have been trained once in the DRL. The hyper-parameter *KL_tar* controls the dynamic training updates for the "actor", which additionally oversees the balance between exploration and exploitation by PPO agents.

As for the computational time analysis, the process of the proposed approach for solving the AC OPF problem during the online implementation consists of two parts: the feed-forward calculation time only in the well-trained "actor" DNN shown in Fig. 2, and the power flow calculation. The feed-forward neural network calculation takes polynomial time regarding the input dimensions [21]-[23], even though one convolutional layer is adopted. Since the input dimension for the "actor" DNN is $4n_{bus}$, this number is manageable even for a large-scale power system; therefore, this feed-forward

calculation time could be regarded as a small constant. As for the second part (the power flow calculation), the feasible region of the AC OPF problem (NP-hard and non-convex) is the subset of the power flow calculation solution set, and thus, it will require less computational time than solving the non-convex optimization problem by applying the conventional interior-point solver, which has been adopted by many vendors' commercial software. Besides, if the GPU is applied, the feed-forward calculation time in the proposed approach can be further reduced.

Algorithm 1: PPO training for solving AC OPF problem

initialize: the number of training data $E_{p\_\max}$, episode length T, KL_tar, batch size N_{batch}, policy log covariance $\Sigma_{\pi\theta}$, training epoch numbers *epo*, "actor" as *policy* parameterized by θ, "critic" as *value-fn* parameterized by φ, updating numbers of neural networks N_{NN}, and related hyper-parameters in [13]
parse in the training dataset D_{train} containing the information of load and generator settings
for each epoch in *range*(*epo*):
  shuffle the training data and set $index = 0$
  while $index < E_{p\_\max}$
    get a new batch of training data with size N_{batch}
    for each episode e in *range*(N_{batch}):
      collect the trajectories' information for every step including (s_t, a_t, r_t, s_{t+1}) from Fig. 4
    end for
    for $i = 1, 2, ..., N_{NN}$ **do**
      train *policy* w.r.t. θ via Adam optimizer [24]
      break if *KL-divergence* > KL_tar
    end for
    for $i = 1, 2, ..., N_{NN}$ **do**
      train *value-fn* w.r.t. φ via Adam optimizer [24]
    end for
    $index = index + N_{batch}$
  end while
end for
return: *policy*

Because of the improved solving time to obtain the near-optimal solutions, the well-trained agent could run more stochastic scenarios resulted from the RES uncertainties and topology changes compared with the conventional interior-point solver. Since the well-trained agent could learn the stochastic optimal policy of the feasible AC OPF solutions, the agent could be applied online as shown in Fig. 1, which has the advantage in real-time economic dispatch applications.

IV. Case Study

The proposed approach to solve the AC OPF problem considering wind integration and $N-1$ topology changing scenarios is tested on the Illinois 200-bus system (with 200 buses, original 38 generators, 1 wind farm connected with bus 161, and 245 lines) [25]. The simulation platform is developed using Python 3.7, Tensorflow, and PYPOWER [26], which is the python version of Matpower [27] that provides Newton-Raphson AC power flow solver and interior-point AC OPF solver, i. e., IPS. The power factor of the wind farm's output is set to be 0.95 following a real-world protocol in ERCOT, which determines the reactive power limits of the wind farm [15].

1) Data generation: each load is randomly perturbed between [0.6, 1.4] p. u. with uniform distribution, where the original data file is considered as the base case; each generator's set point including the wind farm's output is also randomly perturbed between [$P_{g\min}$, $P_{g\max}$] for active power control and [$V_{g\min}$, $V_{g\max}$] for reactive power control; a transmission line is randomly chosen to be tripped to simulate the $N-1$ topology changing scenarios under the uniform distribution (only including the data with feasible solutions from IPS).

2) Label creation: the IPS is adopted to generate the optimal action labels for the "actor" initialization, and to indicate whether the AC OPF problem is feasible or not.

3) Data arrangement: all the data with feasible AC OPF solutions are collected and divided into 3 datasets: 130000 data with both original system's topology and $N-1$ transmission line tripping conditions forms the training dataset used for "actor" initialization and PPO training; 23489 data with original system topology in the testing dataset I and 11511 data with $N-1$ topology changes form testing dataset II (35000 testing data in total) used for testing the trained agent online and verifying its performance. Besides, to further validate the well-trained agent's performance regarding the realistic operating scenarios with uncertainties, both the real-time load and wind power data per 5 min from CAISO in August 2019 [14] are applied as online testing cases.

The cost comparison in percentage κ, feasibility rate, and the total computation time are chosen as performance evaluation indices during the online testing process. The cost comparison in percentage κ, which describes the optimality shown in (12) [6], is only calculated when the agent's actions are feasible, where $cost_{agent}$ and $cost_{ips}$ are the system cost obtained through the PPO agent and IPS solver, respectively. The feasibility rate denotes the percentage of the online testing datasets that the agent's actions lead to feasible solutions.

$$\kappa = (cost_{ips} - cost_{agent})/cost_{ips} \tag{12}$$

In this paper, a rated 150 MW wind farm is connected to bus 161 and all bus voltage magnitude limits are modified from [0.9, 1.1]p. u. to [0.95, 1.05]p. u.. Accordingly, the dimensions of the state and action space are 878 and 78, respectively. The maximum episode length T is set to be 100. One convolutional layer with [4, 4, 1, 32] filter size connected with eight hidden layers with 1024, 1024, 1024, 512, 512, 512, 512, 512 neurons is applied in the "actor" and five hidden layers with 8770, 781, 128, 128, 64, 32, 1 neurons are applied in the "critic".

The initialization results applying (9) to train the "actor" with the convolutional layer is shown in Fig. 6, where 99% of the data in the training dataset are used as a sub-training dataset and the remaining 1% data is regarded as a sub-testing dataset. During the initialization process, the total training iterations are 100000. For each iteration, a random batch of training data is fed into the neural network, and the training loss values are recorded every 100 iterations. From Fig.

6, it is verified that the neural network structure of the "actor" is valid with small relative errors. However, when the initialized "actor" is applied for the online testing process, the results shown in Table I indicate that the feasibility is not achieved under various testing operating conditions with uncertainties, indicating an overlearning issue. Then the proposed DRL training framework is adopted for 2 epochs. To further demonstrate the necessity and effectiveness of the initialization process, PPO training without initialization is also performed and the results are shown in Fig. 7. It can be observed that the PPO agent can be trained more efficiently and effectively with the help of the initialization. Because of the special design for the agent's neural network utilizing a convolutional layer, after the initialization process is adopted, although the agent's outputs may provide infeasible solutions, the outputs are very close to the true solutions from the interior-point solver. During the PPO training process, as the episode length is set to be 100 (it means that the agent can try up to 100 times to get a positive reward) and the initialization process has already been adopted, the actions sampled by the agent based on the Gaussian distribution can finally provide near-optimal solutions when the agent interacts with the power system environment. That is why the training curve appears to be flat. However, at the early stage of PPO training, the average steps taken by the agent to achieve near-optimal solutions are high (for some data, it may take 30 steps to achieve positive reward), whereas, at the end of the training process, the average step taken by the agent is 1 for most of the training data, which is also the objective of PPO training for providing an optimal stochastic policy.

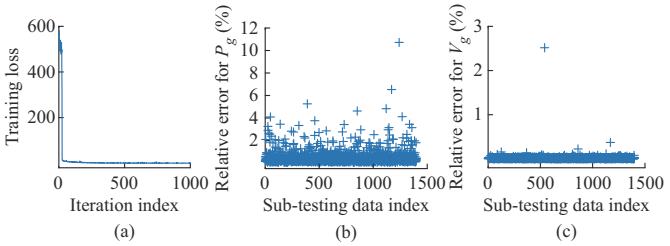

Fig. 6. DL initialization results. (a) Training loss curve. (b) Relative error for P_g. (c) Relative error for V_g.

The well-trained PPO agent is then adopted to perform the online AC OPF task on the testing dataset and the corresponding results are shown in Table I, where "DL-initial" refers to only the initialization process and "initialized PPO" represents the initialization plus PPO training processes. The initialization process aims at minimizing the mean square errors between the outputs of the neural network and the "labels"; therefore, the weights of the neural network are not trained via regularization. After the initialization, the outputs of the neural network, which provides the feasible solutions, are very close to the results from the interior-point solver. That is why the optimality gap is very close to 0. However, as the constraints of the AC OPF problem are not considered in this initialization process while the DRL training further models the constraints in the reward function, the feasibility rate is significantly improved while the optimality gap increases a little bit to improve the generalization of the neural network's performance. From Table I, after adopting the DRL training, the feasibility rate for the PPO agent is improved significantly. More importantly, the well-trained agent can achieve 100% feasibility and near-optimal solutions under the original system topology conditions of the testing dataset I. Compared with the DL-based methods shown in [8]-[10], which can be regarded as directly applying the mean vector of the policy (it will always take one step for the agent to attempt to achieve the near-optimal solution), because the PPO agent learns a stochastic policy, even though it may take several more steps to achieve the near-optimal solutions, it can improve the feasibility rate by adaptively tuning the generator settings. On the other hand, 99.83% of testing dataset II containing $N-1$ topology changes can be solved by the trained agent while achieving near-optimal solutions simultaneously, which suggests that the trained stochastic policy is effective and robust.

Due to a smaller feasible region in $N-1$ scenarios, all violation data in testing dataset II trigger the bus voltage magnitude violation flag. The PPO's on-policy characteristic may be eligible to explain why the agent cannot solve the very small portion of violation data shown in Table I. Therefore, another test is performed by relaxing the bus voltage magnitude constraint from [0.95, 1.05]p.u. to [0.94, 1.06]p.u., which can be regarded as a preventative measure regarding the security concerns of a well-trained agent. Under this relaxed situation shown in Table I, the PPO agent can achieve the feasibility for all the previous violation data, and near-optimal solutions are attained simultaneously. However, the result of the feasibility rate from the initialization is not impacted.

TABLE I
COMPARISON OF TRAINED AGENTS' PERFORMANCE ON TESTING DATASET

Training method	Testing dataset number	Feasibility rate (%)	Maximum $\|\kappa\|$ (%)	Minimum $\|\kappa\|$ (%)	Average $\|\kappa\|$ (%)	Percentage using 1 step (%)
DL-initial	I	51.75	2.95	4.7×10^{-5}	0.046	51.75
DL-initial	II	53.88	3.42	4.8×10^{-5}	0.063	53.88
DL-initial	II (relaxed)	55.77	4.12	3.4×10^{-3}	0.076	55.77
Initialized PPO	I	100.00	1.08	6.5×10^{-4}	0.401	99.88
Initialized PPO	II	99.83	4.33	4.6×10^{-3}	0.402	99.07
Initialized PPO	II (relaxed)	100.00	4.63	2.0×10^{-2}	0.416	99.69

Furthermore, the running time comparison is made by using a desktop equipped with Intel i7-7700 CPU and 8 GB RAM. To obtain near-optimal solutions for the 23489 data in the testing dataset I, the running time from the IPS (the ini-

tial-point vector is set as the mean values of the decision variables' lower and upper bounds as the default in the PYPOWER) costs 6.1 hours, while it only takes 0.41 hours from the proposed method, indicating an average speedup factor of approximately 14 times. It could be even faster if a GPU is used.

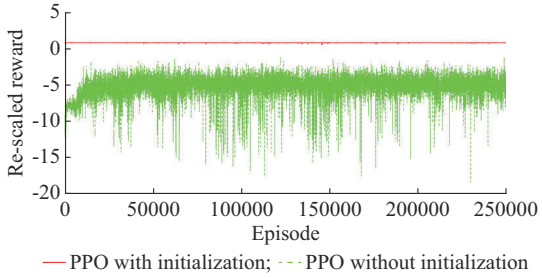

Fig. 7. PPO training process where reward is rescaled 1000 times smaller.

To verify the effectiveness of securing the $N-1$ post-contingencies on the topology changes, another new testing dataset with 1000 data is generated regarding the selective 21 contingency scenarios, which are shown in Table II. In Table II, there are a total of 21 pre-screened contingency scenarios, and the index 0 represents the condition that the system is under the original topology scenario. The corresponding results are shown in Fig. 8.

TABLE II
SELECTIVE $N-1$ CONTINGENCY SCENARIOS

Index	Tripped line (from bus to bus)	Index	Tripped line (from bus to bus)	Index	Tripped line (from bus to bus)
0	None	8	81-178	16	149-87
1	1-119	9	83-146	17	176-88
2	124-1	10	83-186	18	171-190
3	193-1	11	84-113	19	195-171
4	44-42	12	85-120	20	180-172
5	43-84	13	86-101	21	199-172
6	44-200	14	142-86		
7	46-45	15	88-87		

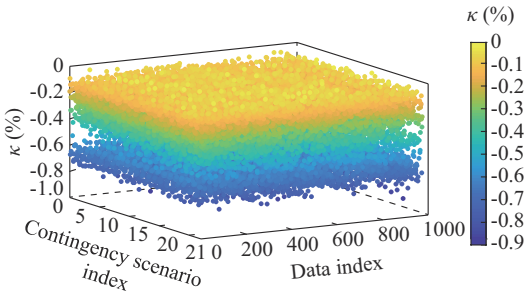

Fig. 8. Online testing results of initialized PPO agent under selective post-contingencies.

As shown in Fig. 8, the well-trained agent is capable of securing the $N-1$ post-contingency scenarios on the topology changes (the average κ value shown in (12) is –0.385%), which validates the effectiveness of the methodology proposed in the paper.

To further show the effectiveness and robustness of the proposed approach, the real-time data with 5 min intervals of CAISO in August 2019 is applied as the new on-line testing data shown in Fig. 9. Besides, to validate the advantages of adopting the convolutional layer for DRL training, another agent is trained with multi-layer perceptron (MLP) structure (2048, 1024, 1024, 1024, 512, 512, 512, 512, 512 neurons) [28], where only the first layer is different from the previous "actor" and the "critic" also has the same structure. The corresponding testing results are shown in Fig. 10.

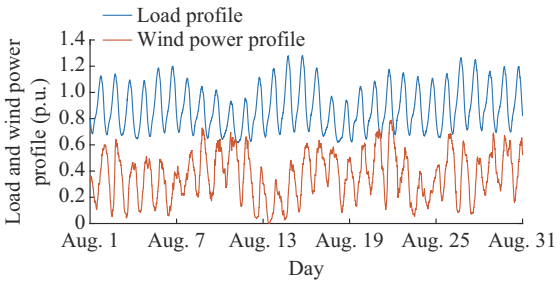

Fig. 9. Real-time load and wind power profiles per 5 min from CAISO in August 2019.

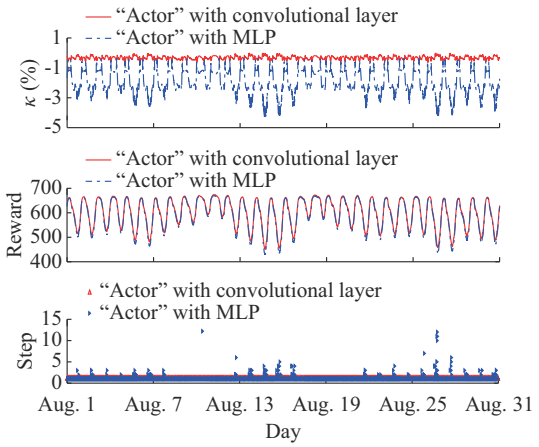

Fig. 10. Online testing results of initialized PPO agents for real-time data from CAISO in August 2019 under original system topology.

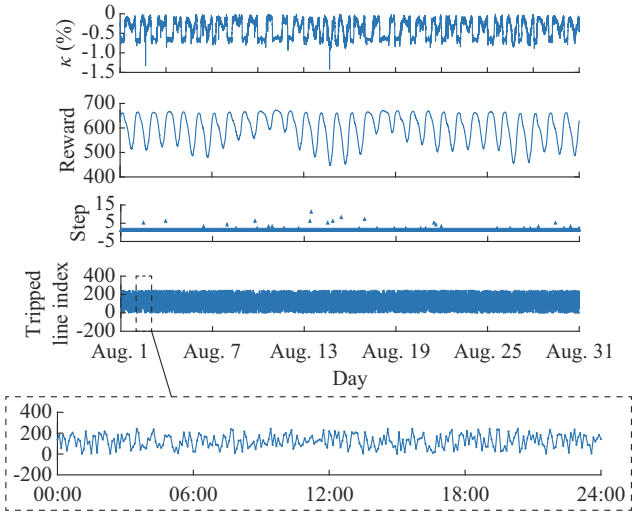

Fig. 11. Online testing results of initialized PPO agent for real-time data from CAISO in August 2019 under topology change conditions (one random transmission line is tripped).

The positive rewards in Fig. 10 reveal that both the well-trained agents with the convolutional layer and MLP structure are capable of solving the real-time data without constraint violations. However, the agent with the convolutional layer has much better performance regarding the optimality (average κ value shown in (12) is −0.321% versus −1.96%) and only takes 1 step to solve all the cases. Besides, to further test the robustness of the DRL agent, one transmission line is randomly chosen to be tripped in the online testing process with the same load data shown in Fig. 9, and the corresponding results are shown in Fig. 10. By comparing the results in Fig. 11 with Fig. 10, it is noticed that with the changes in the network topology introduced by one random line tripping, the agent with the convolutional layer may take more steps to achieve solutions, but the solution quality regarding the optimality (average κ value is −0.397%) is at the similar level. Figures 10 and 11 validate the robustness of the proposed method and demonstrate the advantage of the convolutional layer in the "actor" structure.

To deal with the randomness and uncertainty brought in by high-penetration RESs, it is envisioned that faster control and decision-making are needed in operating the power systems in the future. Therefore, in this paper, we randomly pick real-time data from Fig. 9 on August 2, 2019, and interpolate the data to change the time granularity to 6 s as the base-load and wind power generation profiles. Then we perturb the load at each bus in the system between [0.8, 1.2]p. u. with uniform distribution for active power and [0.95, 1.05]p.u. with uniform distribution for reactive power to simulate load variations and uncertainty. As for the wind power profile, we add Gaussian noise to simulate its uncertainty [29]. The interpolated load and wind power data with noises added are applied as another new testing case shown in Fig. 12, and the corresponding testing results of a well-trained agent under the original system topology scenario and topology change scenarios (one random transmission line is tripped) are shown in Figs. 13 and 14, respectively.

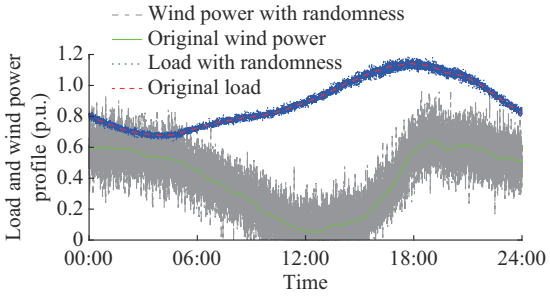

Fig. 12. Real-time load and wind power profiles from CAISO with interpolation per 6 s and added noises on August 2, 2019.

Figure 13 shows that a well-trained agent is capable of controlling the system with uncertainties in real time to achieve optimal costs adaptively (proximal to the IPS results with 99.96% of data taking only 1 step) during a 24-hour period. As shown in Fig. 14, the well-trained agent still effectively provides near-optimal solutions for 99.95% of the data samples, among which 98.82% takes only 1 step (the κ value shown in (12) is manually set to be −2% for violation data and the maximum episode length is set to be 20). Similarly, when the bus voltage magnitude constraint is relaxed from [0.95, 1.05]p. u. to [0.94, 1.06]p. u., the PPO agent could achieve the feasibility for all the previous violation data, and near-optimal solutions are reached simultaneously. Figures 13 and 14 further demonstrate the advantages of the proposed approach for the secure and economic operations of real-time power system when dealing with uncertainties.

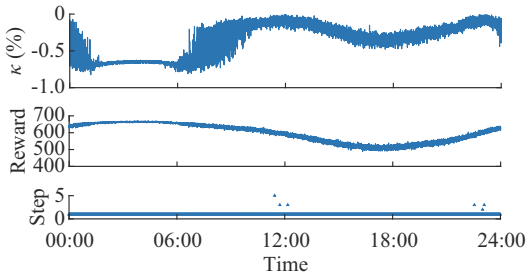

Fig. 13. Online testing results of initialized PPO agent for real-time data from CAISO with interpolation per 6 s and added noises on August 2, 2019 under original system topology.

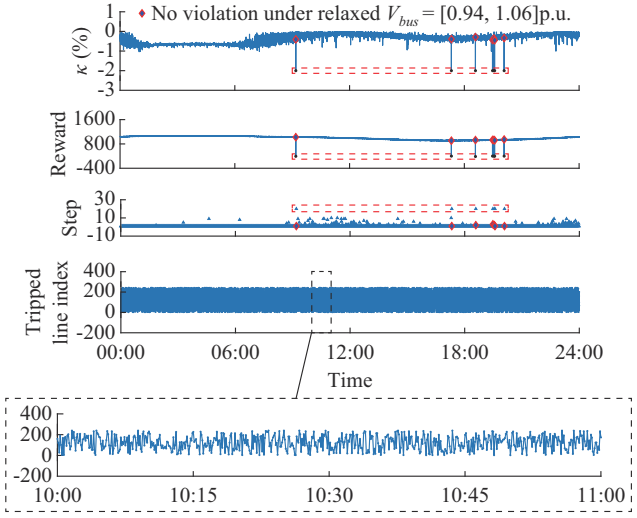

Fig. 14. Online testing results of initialized PPO agent for real-time data from CAISO with interpolation per 6 s and added noises on August 2, 2019 under topology change conditions (one random transmission line is tripped).

Moreover, the selective post-contingencies shown in Table II based on the original CAISO load data on August 2, 2019 are applied for further validating the effectiveness of the well-trained agent, and the corresponding results are shown in Fig. 15.

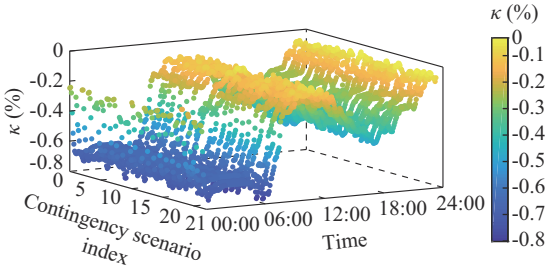

Fig. 15. Online testing results of initialized PPO agent for real-time data from CAISO on August 2, 2019 under selective post-contingencies.

From Fig. 15, the well-trained agent can successfully handle all the selective post-contingencies, where the average κ value shown in (12) is −0.385% compared with IPS results. This further demonstrates the advantages of the proposed well-trained agent for the real-time secure and economic operation of power system when dealing with post-contingencies.

An additional ancillary and independent "alarm" function can be designed to help system operators identify whether the current load, RES power outputs, and system topology information would lead to infeasibility from IPS. This is formulated as a classification problem (the label is 0 if infeasible, otherwise 1). By running the IPS to generate data under various conditions including topology changes (one random transmission line is tripped), 140000 data samples are adopted as a training dataset. Figure 16 shows the DNN structure and its training process. Another 115357 data records are generated as the testing dataset, and the classification accuracy for the testing dataset reaches 98.6%, which demonstrates the effectiveness of the proposed design. This ancillary "alarm" function can be combined with the well-trained DRL agent. If it detects that the current system status is feasible, it will then adopt the well-trained DRL agent to provide the near-optimal solutions; otherwise, it will send an alarm to the system operators.

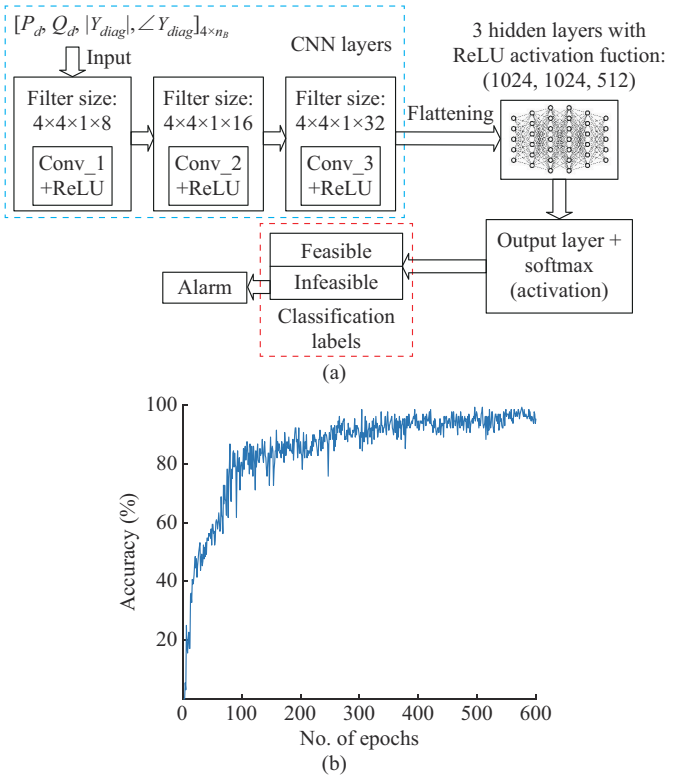

Fig. 16. Training process for feasibility classification of AC OPF problem. (a) DNN structure. (b) Accuracy of training process.

V. Discussion on Full $N-1$ Contingencies on Topology Changes

We choose three representative operating conditions from Fig. 12 and apply the full $N-1$ contingencies on topology changes, respectively.

1) Peak time at 18:02, where the load is around 1.19 p.u. and wind power output is around 0.63 p.u..

2) Off-peak time at 03:11, where the load is around 0.66 p.u. and wind power output is around 0.52 p.u..

3) Time at 12:38, where the wind power output is at the relatively low level with 0.045 p.u. output and the load is around 0.85 p.u..

The corresponding results are shown in Fig. 17, where the horizontal axis represents the index of the tripped transmission line. From Fig. 17, the well-trained agent can provide the near-optimal adjustments of the generator set points compared with the results coming from the IPS. As for the specific infeasible scenarios, the IPS cannot provide feasible solutions under these topology conditions; thus, the results are manually set as −2% for infeasible conditions. From Fig. 17, the well-trained agent could successfully mimic the AC OPF solver results. Therefore, if the solver can successfully solve the full $N-1$ contingencies on topology changes, and is adopted in the training, the well-trained agent should also be capable of mimicking the solver results. However, due to the large number of post-contingency states and the non-convexity of the AC power flow equations, the power system operators are still adopting the DC OPF scheme due to its convexity and computational benefits [30], and thus, it is still very challenging for the AC OPF solver to achieve full $N-1$ security.

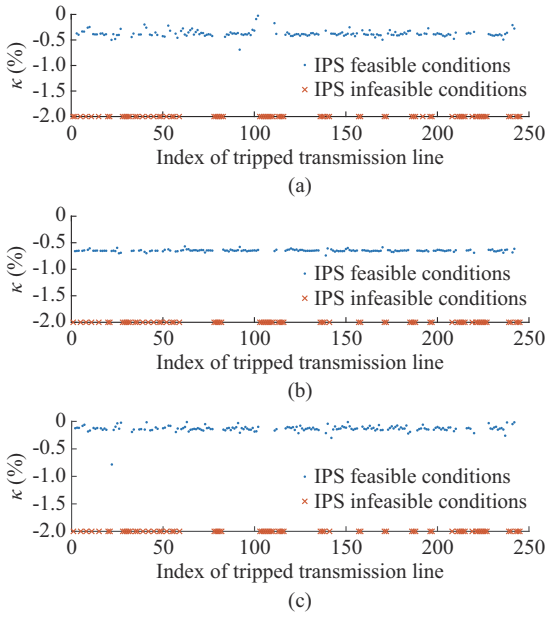

Fig. 17. Cost comparison for all $N-1$ contingencies on topology changes for peak time, off-peak time, and low wind power output time on August 2, 2019 from CAISO profile. (a) Peak time. (b) Off-peak time. (c) Low wind power output time.

VI. Conclusion

This paper proposes a novel framework of deriving fast AC OPF solutions for real-time applications using deep reinforcement learning. Case studies are based on the Illinois 200-bus system, and real-time data from CAISO is also adopted. The testing results demonstrate that after the offline

DRL training, the near-optimal AC OPF solutions can be accomplished with at least 14 times speedup compared with the interior-point method. Moreover, the well-trained DRL agent is robust to achieve near-optimal status to deal with the uncertainties of RES and topology changes, which provides great potential for the operation and control of modern power system with high penetration of renewable energy. Although only the outage of one random transmission line is included as the uncertainty regarding the topology change scenarios, it could be expanded to the outages of multiple transmission lines with higher computation burden. Furthermore, an efficient and robust classifier, which serves as an independent "alarm" function, is designed to help system operators identify the feasibility of the AC OPF problem under the present conditions of loading, RES outputs, and topology.

Future work includes further improvements on the AI agent to gain higher accuracy, application of GPU for process parallelization and better speedup, and test of the proposed approach on larger power systems. Besides, the constraint of the ramping rate limits for the generators will be considered for solving the multi-period AC OPF problem. On the other hand, as the results from the solver are applied for the initialization process of the agent and IPS in this paper only considers pre-contingency states, it requires further investigation to guarantee the security when considering full $N-1$ contingencies for solving the AC OPF problem.

REFERENCES

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

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

[3] X. Pan, T. Zhao, and M. Chen, "DeepOPF: deep neural network for DC optimal power flow," in *Proceedings of 2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids*, Beijing, China, Oct. 2019, pp. 1-12.

[4] D. Deka and S. Misra, "Learning for DC-OPF: classifying active sets using neural nets," in *Proceedings of 2019 IEEE Milan PowerTech*, Milan, Italy, Jun. 2019, pp. 1-6.

[5] A. Venzke, G. Qu, S. Low *et al*., "Learning optimal power flow: worst-case guarantees for neural networks," in *Proceedings of 2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids*, Tempe, USA, Nov. 2020, pp. 1-7.

[6] A. S. Zamzam and K. Baker, "Learning optimal solutions for extremely fast AC optimal power flow," in *Proceedings of 2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids*, Tempe, USA, Nov. 2020, pp. 1-7.

[7] D. Owerko, F. Gama, and A. Ribeiro, "Optimal power flow using graph neural networks," in *Proceedings of 2020 IEEE International Conference on Accoustics, Speech and Signal Processing*, Barcelona, Spain, May 2020, pp. 1-5.

[8] F. Fioretto, T. W. K. Mak, and P. V. Hentenryck, "Predicting AC optimal power flows: combining deep learning and lagrangian dual method," in *Proceedings of 2020 AAAI Conference on Artificial Intelligence*, New York, USA, Feb. 2020, pp. 630-637.

[9] M. Chatzos, F. Fioretto, T. W. K. Mak *et al*. (2020, Jun.). High-fidelity machine learning approximations of large-scale optimal power flow. [Online]. Available: https://arxiv.org/abs/2006.16356

[10] X. Pan, M. Chen, T. Zhao *et al*. (2020, Jul.). DeepOPF: a feasibility-optimized deep neural network approach for AC optimal power flow problems. [Online]. Available: https://arxiv.org/abs/2007.01002v1

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

[12] Z. Yan and Y. Xu, "Real-time optimal power flow: a lagrangian based deep reinforcement learning approach," *IEEE Transactions on Power Systems*, vol. 35, no. 4, pp. 3270-3273, Apr. 2020.

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

[14] CAISO. (2020, Dec.). Today's California ISO website. [Online]. Available: http://www.caiso.com/TodaysOutlook/Pages/default.aspx

[15] ERCOT. (2020, Dec.). ERCOT protocols & operating guides on reactive testing. [Online]. Available: http://www. ercot. com/content/wcm/key_documents_lists/54515/Reactive_Testing__ERCOT_Protocols_Op._Guides.pdf

[16] V. Francois-Lavet, P. Henderson, R. Islam *et al.*, "An introduction to deep reinforcement learning," *Foundations and Trends in Machine Learning*, vol. 11, no. 3-4, pp. 219-354, Dec. 2018.

[17] OpenAI. (2020, Dec.). OpenAI blog: proximal policy optimization. [Online]. Available: https://openai.com/blog/openai-baselines-ppo/

[18] J. Schulman, P. Moritz, S. Levine *et al*. (2018, Oct.). High-dimensional continuous control using generalized advantage estimation. [Online]. Available: https://arxiv.org/abs/1506.02438v4

[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, Sept. 2019.

[20] 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.

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

[22] Kasper Fredenslund. (2020, Dec.). Computational complexity of neural networks. [Online]. Available: https://kasperfred. com/series/introduction-to-neuralnetworks/computational-complexity-of-neural-networks

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

[24] D. P. Kingma and J. Ba. "Adam: a method for stochastic optimization," in *Proceedings of 3rd International Conference on Learning Rresentations (ICLR)*, San Diego, USA, May 2020, pp. 1-15.

[25] Texis A&M University. (2020, Dec.). Electric grid test case repository. [Online]. Available: https://electricgrids.engr.tamu.edu/electric-grid-test-cases/

[26] Pypower. (2020, Dec.). Pypower 5.1.4. [Online]. Available: https://pypi.org/project/PYPOWER/

[27] 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.

[28] Y. Zhou, B. Zhang, C. Xu *et al.*, "A data-driven method for fast AC optimal power flow solutions via deep reinforcement learning," *Journal of Modern Power Systems and Clean Energy*, vol. 8, no. 6, pp. 1128-1139, Nov. 2020.

[29] Y. Dvorkin, M. Lubin, S. Backhaus *et al*., "Uncertainty sets for wind power generation," *IEEE Transactions on Power Systems*, vol. 31, no. 4, pp. 3326-3327, Jul. 2016.

[30] K. Baker, "Solutions of DC OPF are never AC feasible," in *Proceedings of 12th ACM International Conference on Future Energy Systems*, Virtual, Italy, Jun. 2021, pp. 264-268.

Yuhao Zhou received the B.S. and M.S. degrees in electrical engineering from Huazhong University of Science and Technology, Wuhan, China, in 2013 and 2016, respectively, and the Ph.D. degree in electrical engineering from the University of Texas at Arlington, Arlington, USA, in 2021. His research interests include dynamic equivalent modeling of wind farm, AC optimal power flow, and arc flash protection.

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 is a Registered Professional Engineer in the State of Texas. He has been involved in the revision of IEEE Std. 141, 339, 551, 739, and dot 3000 series development. Currently, he is the 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. His research interests include power flow, transient and dynamic stability, voltage

stability, short circuit, relay coordination, power quality analysis, renewable energy, and deregulation for utility companies.

Ruisheng Diao received the Ph.D. degree in electrical engineering from Arizona State University, Tempe, USA, in 2009. He has published nearly 90 peer reviewed journal and conference papers, as well as dozens of technical reports. He is the co-applicant for more than 10 U.S. patents. He 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. Serving as a PM/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, artificial intelligence (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, in 2012. He is the Founder of AINERGY LLC, Santa Clara, USA. He was the Director of Fundamental R&D Center and Department Head of AI & System Analytics at GEIRINA, and a Research Staff Member at NEC Laboratories America. He is an Editor of the IEEE Transactions on Smart Grid and the IEEE Power Engineering Letters. His research interests include data analytics, energy storage systems, and applications of AI and Internet of Things (IoT) in power systems.