Fault Identification in Power Network Based on Deep Reinforcement Learning

Mengshi Li, *Member*, *IEEE*, Huanming Zhang, *Student Member*, *IEEE*, Tianyao Ji, *Senior Member*, *IEEE*, and Q. H. Wu, *Fellow*, *IEEE*

***Abstract*—With the integration of alternative energy and renewables, the issue of stability and resilience of the power network has received considerable attention. The basic necessity for fault diagnosis and isolation is fault identification and location. The conventional intelligent fault identification method needs supervision, manual labelling of characteristics, and requires large amounts of labelled data. To enhance the ability of intelligent methods and get rid of the dependence on a large amount of labelled data, a novel fault identification method based on deep reinforcement learning (DRL), which has not received enough attention in the field of fault identification, is investigated in this paper. The proposed method uses different faults as parameters of the model to expand the scope of fault identification. In addition, the DRL algorithm can intelligently modify the fault parameters according to the observations obtained from the power network environment, rather than requiring manual and mechanical tuning of parameters. The methodology was tested on the IEEE 14 bus for several scenarios and the performance of the proposed method was compared with that of population-based optimization methods and supervised learning methods. The obtained results have confirmed the feasibility and effectiveness of the proposed method.**

***Index Terms*—Artificial intelligence, deep Q network, deep reinforcement learning, fault diagnosis, fault identification, parameter identification, power network.**

I. Introduction

POWER system is a general term for an organic system composed of electric energy generation, transformation, transmission, distribution, and consumption. If the power grid is compared to the human body, then the power network is the network of main arteries supporting the whole body, and customers are the blood at the end of the capillaries, supplied by the network of arteries. It is thus clear that the stability of the power network in the whole transmission system is of key importance. Nonetheless, increasing power demand, penetration of renewables, and limitations of necessary grid expansion are putting forth new challenges to existing power system protection and control strategies [1], [2]. Therefore, accurate and rapid fault identification is necessary to reduce the economic losses caused by power outages and to protect the safety of the grid as well as to avoid the cascading of faults [3].

The accuracy of traditional methods decreases when faced with the new challenges brought about by the diversification of fault modes and emergence of new forms [4]. The traditional fault identification method is unable to effectively handle the complex relationship between measurement data and faults, which means that a combination of expert knowledge [5] or a combination of multiple methods [6] are required. Moreover, conventional intelligent methods are based on a large amount of fault data and labels, which fail if the data and labels are missing. Therefore, it is urgent that we need a more universal and effective method to challenge the problem of missing or sparse training data in the fault identification process is urgently needed.

Different types of fault identification techniques have been explored to fix the power network fault identification challenges. The earlier methods used for fault identification and location were the impendence-based methods and the traveling wave-based methods, which have been popular among electric power utilities for their high performance-price ratio [7]. Besides, some scholars regard power system fault identification as an optimization problem and have tried to solve it with optimization algorithms, such as genetic algorithms (GA) [8], particle swarm optimization (PSO) [9], etc.

The artificial intelligence-based fault identification technology has been popular in the power system identification research field for a long time. The intelligent methods do not rely on mathematical calculation of physical quantities but establish a function between some measured physical quantities and fault types, to identify the fault. Early intelligent methods include expert systems (ES) [10], support vector machine (SVM) [11], and the artificial neural network (ANN) [12], which suffer from the incapacity of generalization and the difficulty of validating and maintaining a large rule base.

With the popularity of deep learning (DL), new fault identification methods based on DL have emerged in recent years. Representative algorithms include long short-term memory (LSTM) [13], convolutional neural network (CNN) [14], and deep belief network (DBN) [15]. Although these methods have performed well, their interpretability is often criticized by the industry for their "black-box" phenomenon [16]. Moreover,

Manuscript received August 30, 2020; revised November 1, 2020; accepted December 17, 2020. Date of online publication April 30, 2021; date of current version September 1, 2021. This work is supported by Fundamental Research Funds Program for the Central Universities (No. 2019MS014) and Key-Area Research and Development Program of Guangdong Province (No. 2020B010166004).

M. S. Li, H. M. Zhang, T. Y. Ji (corresponding author, email: tyji@scut.edu.cn; ORCID: https://orcid.org/0000-0003-2788-1349), and Q. H. Wu are with School of Electric Power Engineering, South China University of Technology, Guangzhou 510641, China.

DOI: 10.17775/CSEEJPES.2020.04520

these methods are highly data hungry, and sometimes overfitting of the benchmark data occur [17]. In addition, the handcrafted data preprocessing and training process is laborious and time-consuming. Therefore, a new fault identification method that can solve the problems mentioned above is necessitated.

Reinforcement learning (RL) is another approach in the field of artificial intelligence (AI), which is praised as the real AI and has great of prospects when combined with deep learnin [18]. RL is an intelligent computing method for understanding and automating the problem of goal-oriented learning and decision-making [23]. It emphasizes that the agent learns through direct interaction with the environment, without the need for imitable supervision signals or complete modeling of the surrounding environment, so it has a different paradigm compared with other intelligent methods.

When combined with DL, RL becomes deep reinforcement learning (DRL), which has achieved huge success in games [19], [20], recommendation systems [21] and robotics control [22]. However, DRL is rarely mentioned in the field of fault identification, which is occupied by DL. Thus, we propose a novel fault identification method by parameterizing the fault in a power network model and identifying them through DRL. The process of fault parameterization will help improve the interpretability of intelligent methods and the implementation of DRL will make them more universal.

In this paper, different faults are parameterized individually, which assist DRL in converting a classification problem into a sequential decision problem related to parameter optimization. The environment related to power networks after parameterization is built in a MATLAB/Simulink platform. The proposed method based on DRL was implemented on IEEE 14 bus as test cases as well as compared with population-based method and supervised learning method.

The rest of the paper is organized as follows: a brief introduction of DRL and fault parameter identification is presented in Section II. Section III details the implementation of power networks fault parameter identification based on DRL; Test cases and results are shown in Section IV. Conclusions and future directions for research are given in Section V.

II. Overview of Deep Reinforcement Learning and Fault Parameter Identification

A. Reinforcement Learning

RL is a goal-directed computational approach where an agent learns to make optimal decision by interacting with an unknown dynamic environment through exploration and exploitation [23]. The agent is governed by a decision-making policy that is used to determine what the agent should perform in a given state. The environment is typically described by a Markov Decision Process (MDP) with the following components:

- States S: snapshots of the environment; observed by the agent; could be continuous or discrete;
- Actions A: means by which the agent interacts with its environment; could be continuous or discrete;
- Transition function ($P : S \times A \rightarrow S'$): the probability that state S transitions to state S' when performing action from action space A;
- Reward function ($R : S \times A \rightarrow R$): determines the reward resulting from action A taken in state S.

According to this rule, at each time step t, the agent observes the state $s_t \in S$ and receive the reward $r_t \in R$ from the environment. At the same time, it chooses an action $a_t \in A$ according to its policy $\pi(s)$ to interact with the environment. At the next time step, the agent repeats the above steps until the termination. The goal of an RL agent is to learn an optimal policy $\pi^*(s)$ that maximizes the long-term expected reward that is defined and discounted as follow:

$$G_t = \sum_{k=t+1}^{T} \gamma^{k-t-1} r_t \tag{1}$$

where T refers to the time step that ends the interaction, and γ is the discount factor which prevent long-term rewards from increasing in an unbounded manner (for cases of some environments that do not have terminal state). To evaluate the result of action a_t in state s_t, an action-value function $Q(s, a)$, known as Q value function, is proposed. The action-value function $Q(s, a)$ is updated at each time step by utilizing the Bellman equation and is defined as [23]:

$$Q_{t+1}(s, a) = E[r + \gamma \max_{a'} Q_t(s', a')|(s, a)] \tag{2}$$

The above equation can converge due to the Markov property of state signal when $t \rightarrow \infty$ [24].

The conventional RL algorithm is represented by Q-learning, which is a value-based algorithm. Q-learning finds the optimal policy $\pi^*(s)$ by using

$$\begin{aligned} &Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \eta[r_{t+1} + \\ &\gamma \max_{a} Q(s_{t+1}, a) - Q(s_t, a_t)] \end{aligned} \tag{3}$$

where η represents the learning rate that can be chosen properly to avoid overfitting.

The conventional Q-learning algorithm is effective when the observation space is finite and discrete. But for large-scale problems such as power systems, the information grows exponentially, which is a huge challenge for conventional Q-learning and makes it more time-consuming and even divergent. At the same time, the signal noise in the real world could also make it hard for the convention RL algorithm to capture the pattern behind the state, action and reward [25].

B. Deep Reinforcement Learning

To use RL successfully in a real-world scenario, an agent must be able to derive efficient representations of the environment from high-dimensional sensory inputs, and use these inputs to generalize past experience to accommodate new situations [26]. DRL learns from the agent's interaction with the environment through trial and error. Unlike other fault identification methods used in power systems, which require the researchers to label a vast mess of training data manually, DRL can obtain the training data autonomously via the simulation model. Extreme failures that occur with very low

probability in some industrial applications can be simulated and data obtained in the simulation model. Therefore, DRL can master these to obtain fault experience and expand the scope of fault identification by observing part of the measurable power system parameters and taking corresponding actions to modify the fault parameter subset under the guidance of a reward.

The most commonly used DRL algorithm is Deep Q Network (DQN), which can learn optimal policy directly from high-dimensional sensory inputs using end-to-end reinforcement learning [26]. DQN utilize a deep neural network and its weight θ to substitute the state-action table in Q-learning. The neural network utilizes state and action as inputs and outputs long-term expected rewards, which represents the Q value function (Q) in the DQN algorithm. DRL can solve the problems of RL, which are limited to domains where useful features can only be made by hand, and overcome the limitation of domains that have fully observed low-dimensional state spaces.

There are some vital improvement mechanisms that make the existing DQN algorithm performs so efficiently.

1) The experience replay mechanism

Since deep learning requires independently distributed training data, the strong correlation between training data due to the exploration behavior of the RL agent is not applicable. Therefore, DQN employs an experience replay mechanism inspired biologically to break the correlation between the data [27].

2) Asynchronous update of target Q network

A small update of θ can cause significant fluctuations in policy, resulting in instability and changing the distribution of experience data. Thus, DQN constructed network names target Q network (Q') for evaluating learning effects [26]. This network and the Q network share common weight values ($\theta'\&\theta$). The difference is that θ' values are updated at every time step using the smoothing factor τ based on the latest θ, while θ is always kept the latest.

3) Double DQN mechanism

The maximization operation (argmax) used by DQN in calculating long-term expected reward overestimates the Q-value function. To reduce over-estimation, van Hasselt *et al.* (2016) proposed a double DQN mechanism, which takes advantage of Q' to decouple the selection and evolution [28]. Q is used to select the action with the largest value, and Q' is used to estimate its value, i.e., when calculating the target value, the original equation $y_i = r_i + \gamma Q'(s_i{}', a'|\theta_{Q'})$ is replaced by (4).

$$\begin{aligned} a_{\max} &= \arg\max_{a'} Q(s_i{}', a'|\theta_Q) \\ y_i &= r_i + \gamma Q'(s_i{}', a_{\max}|\theta_{Q'}) \end{aligned} \tag{4}$$

where y_i represents the target value.

4) *The n step bootstrapping*. The Q-learning accumulates a single reward and then uses the greedy action at the next step to bootstrap. Alternatively, forward-view multi-step targets can be used in DQN. While employing n-step bootstrapping, the target value y_i changes from (4) to (5).

$$a_{\max} = \operatorname*{argmax}_{a'} Q(s_i{}', a'|\theta_Q)$$

$$y_i = \sum_{k=0}^{n-1} \gamma_t{}^{(k)} r_{t+k+1} + \gamma_t{}^{(n)} Q'(s'_i, a_{\max}|\theta_{Q'}) \tag{5}$$

To perform experience replay, the agent stores experiences $e_t = (s_t, a_t, r_t, s'_t)$ at each time-step t in a data set $D_t = \{e_1, \ldots, e_t, \ldots, e_C\}$, where C is the length of the experience buffer. The Q network is trained with a sample of data (s, a, r, s') that is drawn randomly from the experience buffer by minimizing the following loss function at each iteration i:

$$L_i(\theta_i) = \mathbb{E}_{(s,a,r,s')U(D)}[(y_i - Q(s, a|\theta_i))^2] \tag{6}$$

where y_i is the target value for iteration i computed by Q'. Compared with the labels that are determined before the start of learning in supervised learning, the target of DQN depends upon the network weight. The network weights are updated using equation (7) with stochastic gradient descent.

$$\nabla_{\theta_i} L(\theta_i) = \mathbb{E}_{s,a,r,s'}[(y_i - Q(s, a|\theta_i))\nabla_{\theta_i} Q(s, a|\theta_i)] \tag{7}$$

In this paper, we adopt an advanced DQN algorithm that utilizes experience replay mechanism, asynchronous update of target network, double DQN and n step bootstrapping. The details are illustrated in Algorithm 1.

Algorithm 1: Deep Q Network

Set hyperparameters: γ, $\epsilon_{\min}$, τ, η, C, M
Initialize network with random weights θ, make $\theta = \theta'$
Initialize the experience buffer D as an empty set
for *episode* $n = 1$ *to terminal* **then**
    Reset state s_t randomly
    for $t = 1$, T **then**
        With probability ϵ select a random action a_t, otherwise select $a_t = \text{argmax}_{a \in A} Q(s_t, a|\theta)$
        Receive next state s'_t and reward r_t
        Store the experience (s_t, a_t, r_t, s'_t) in D
        if s'_t *is terminal state,* T **then**
            $y_i = r_i$
        else
            Calculate according to formula (5)
        end if
        Sample random mini-batch (s_i, a_i, r_i, s'_i) of size M from the experience buffer
        $Loss_i = \frac{1}{M} \sum_{i=1}^{M} (y_i - Q(s_i, a_i|\theta_Q))^2$
        $\theta = \theta - \eta \nabla Loss(\theta)$
        $\theta' = \tau\theta' + (1 - \tau)\theta'$
        $N_t = N_t + 1$
    end for
    if $\epsilon > \epsilon_{\min}$ **then**
        $\epsilon \leftarrow \varphi(N_t, \epsilon)$
    else
        $\epsilon = \epsilon_{\min}$
    end if
end for

As we can see, the ϵ-greedy policy was introduced to enable selection of an action. The exploration rate ϵ is not constant, but linearly decreases from 0.9 to a smaller constant value $\epsilon_{\min}$

within the number of steps N_t, which is defined as function $\varphi(N_t, \epsilon)$. φ can balance the relationship between exploration and exploitation, agent explores more in the beginning and exploit more in the end. Frequent exploration operations in the early stage can help the agent obtain more diversified training data, while later exploitation operations can help the agent learn the characteristics of maximizing long-term rewards from the previous training data.

More details of the DQN algorithm interacting with the fault parameter identification model are discussed in Section III.C

C. Fault Parameter Identification

The time-domain digital simulation model is an indispensable part in the development of the power system discipline, and the accuracy of its parameters has a significant influence on the simulation results. The conventional parameter identification task is mainly used to solve the problem of discrepancy between the simulation model and its associated real physical device. The research work in this paper aims to identify the parameters of the faults rather than solve the conventional component parameter identification problem. Inspired by the unexplainable nature of machine learning based fault identification algorithms, the idea is to formulate failure occurrence as the system's parameters. This solves the problem of some rare faults that cause traditional fault identification methods to fail due to lack of data.

As illustrated in Fig. 1, a real system is simulated as a model whose parameter set has two subsets. Subset α includes the parameters of normal components, such as generator parameters, power line parameters, load parameters, which are considered to be preciseand tangible, in our research. Subset β involves fault parameters that indicate whether the fault occurs and where the fault occurs. The domain of β is an $N+1$-dimensional vector: $\beta_i \in (\beta_0, \beta_1, \beta_2, \cdots, \beta_{N-1}, \beta_N)$, where N is the number of faults to be recognized; β_0 represent a normal situation while others represents fault type 1 to N.

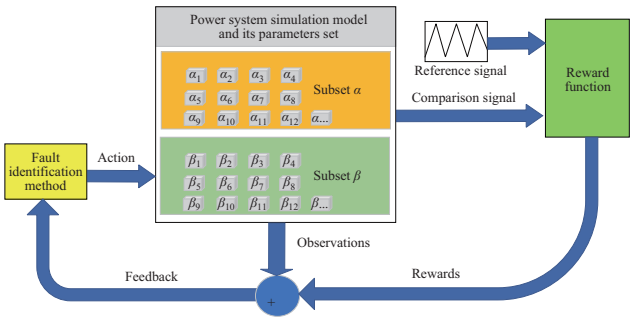


Fig. 1. The proposed fault parameter identification process.

A signal ω_{ref} recorded during fault disturbance and collected from field data sets is used to as input to the parameter identification process. The objective function, ϖ, which is also the reward evaluation index, describes the differences between ω_{ref} and the real simulation model output ω.

By adjusting the value of parameter β, the identification method can minimize ϖ and make the fault parameters closer to the real situation progressively and intelligently.

III. Implementation Details for PowerNet Network Fault Identification

In this section, the detailed implementation of fault identification based on DRL are discussed, including the simulation model, the architecture of the Q-value function approximator (structure of neural network) and the specific interaction details between the DQN algorithm and the fault parameter identification model. The flow chart of the proposed method is shown in Fig. 2.

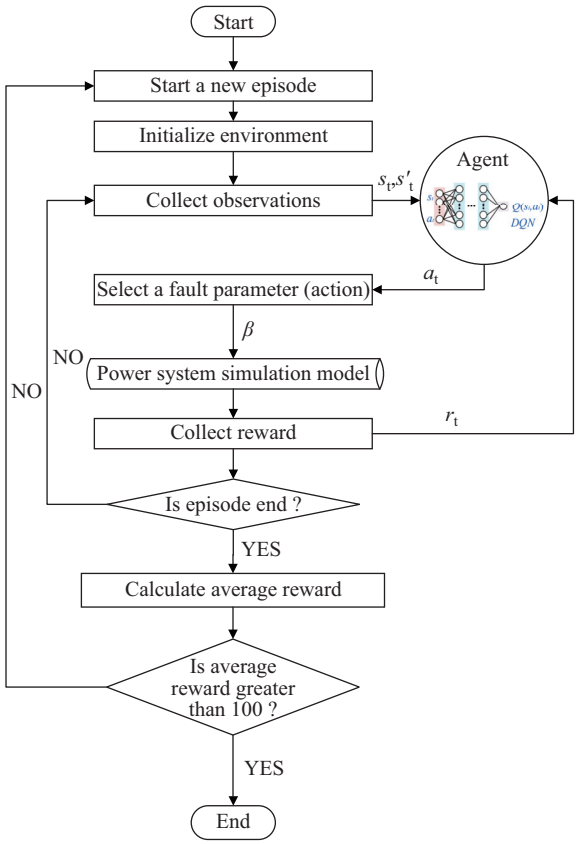


Fig. 2. Flow chart of the proposed fault identification method.

A. Simulation Model and Fault Description

The IEEE 14 bus is employed as a test platform in our experiment. It is a typical power network model that is generally used to test a new theory or a new method, as it is convenient for testing the performance a large number of experiments. The base capacity of IEEE 14 bus is 100 MVA and the base voltage is 23 kV. It is a simple representation of the American power grid as of 1962 which consists of 5 generators, 11 loads, and 14 buses [29].

The probability of power network failure is very small and there actually few fault data samples. Among them, Three-Line-to-Ground (LLLG) faults have the lowest probability of occurrenc [30]. The lowest probability of occurrence of LLLG fault leads to data sparseness or loss in the training data set while employing supervised learning method, which will make the supervised learning method invalid. Therefore, to identify faults like LLLG, which brings about data sparseness, we only discuss LLLG fault in our research. In addition, only one node

of the entire system has such a fault at the same time, while others do not. The power network with a simulation time of 2 s is deployed on MATLAB/Simulink.

B. Q-value Function Approximator

The DQN algorithm adopts a neural network, which is a function approximator, to estimate the value function $Q(s, a)$, while taking both observation S and action A as inputs, and outputs the corresponding expectation of the long-term reward. There are three designed paths to develop this neural network as shown in Fig. 3.

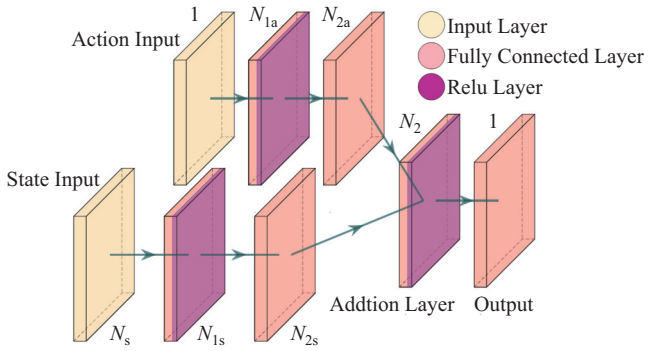


Fig. 3. Network architecture for DRL.

The specific network structure used for the fault parameter identification system is as follows: the input of state-path consists of only current state information s_t with dimension N_s, and the action a_t with dimension 1 is used as the input value of the action-path. The common-path receives the sum of the outputs of the state-path and the action-path, and outputs the expectation of the long-term reward corresponding to the current state and action pairs (s_t, a_t). The structures of the three paths are similar. There are two hidden layers and one layer in between each path's input and the output. The two hidden layers have $N_{1s}, N_{1a}, N_{2s}, N_{2a}$ hidden neural unit, and they are followed by a rectified linear unit (ReLU) layer. The output layer of common-path is a fully connected layer with one unit which outputs a scalar of Q-values. Note that all three paths have their own roles. The state-path obtains the mapping between states and rewards from the raw data. The action-path combines the current actions with rewards and represents the functional relationship between them. The common-path combines the functions of the previous two, and further extracts the relationship between state-action-reward through a neural network. The DRL is superior to other algorithms because it is deep enough to directly explore the environment from the current original environmental data, control behavior and the relationship between the ultimate expected reward.

C. Fault Parameter Identification

The DQN algorithm is an off-policy, value-based and model-free RL algorithm, i.e., the agent observes the real system model partially. It is impossible to completely portray the simulation model only by observing a few states. However, we can improve the accuracy of the model description by extracting more quantities that can describe the characteristics of the system, and use them as observations of the fault identification model. Therefore, we choose a subset of the power system measurable parameter set as the state to observe and choose a subset of the simulation parameter set as the action to act upon. The specific states and actions for fault identification are listed in Table I. The action of the agent is the value of fault parameter β. We assume that β_0 represents a normal and fault-free scenario, and make fault parameter (β_i) represent its own fault state. For example, β_1 represents a LLLG fault that occurs at bus 1, while β_2 represents the same fault but occurring at bus 2.

TABLE I
SPECIFIC STATES AND ACTIONS

Symbol	Description
V	State: the voltage of each node
Pe	State: the active power output by the generator node
Pe_{ref}	State: reference active power (reference signals)
ϖ	State: reward evaluation index
β	Action: fault parameter

As we illustrated in Section II, a reference signal needs to be collected to teach the agent how to distinguish between good and bad outcomes. Corresponding to different failure scenarios, different fault reference signals are needed. The active power of 5 generators extracted from the fault data set have been employed as the reference signal in our experiment. The objective of fault parameter identification is to accurately detect the time and location of failure. To achieve the above objectives, the agent must adjust the fault parameters to make the model close to the real situation. Therefore, a reward function is designed to instruct the agent:

$$r(\varpi) = \begin{cases} 10 & \varpi \leqslant 0.6 \\ -20 & \varpi > 0.6 \end{cases} \tag{8}$$

where ϖ is reward evaluation index calculated using a sliding window with a fixed length of time. As shown in the following formula:

$$\varpi = \frac{\sum_{i=1}^{5} |Pe_i - Pe_{\text{ref}}|}{WL} \tag{9}$$

where WL is the sliding window length. Unlike the final results in the control task, which often have an acceptable margin, the fault parameter identification results are either black or white. Therefore, the reward function must have a clear boundary condition. The cut-off point of the reward function is set empirically. Similarly, the length of *WL* is also set to 1/60 s based on experimental experience.

The agent first initializes the weights of Q and Q' network as well as other training hyperparameters. Then, in each step of the episode, the measurable quantities $(V, Pe, Pe_{\text{ref}}, \varpi)$ of the power system are collected as the state s_t. According to the state s_t, an action a_t that modifies the fault parameter β is chosen to interact with the environment. The action is selected according to ϵ-greedy policy, which is shown in line 7 of Algorithm 1. After a time step, the agent gets a reward r_t given by the reward function and can observe the next

state s'_t at the same time. The experience vector composed of (s_t, a_t, r_t, s'_t) is stored in the fault experience replay buffer D for updating the parameters of the Q network later. The output of the Q network y_i is used to evaluate the result of the action a_t in the state s_t. The greater the value of y_i, the closer the fault parameter value represented by the agent's action in this step to the actual fault situation. When the amount of data in the fault experience replay buffer is greater than M, the agent takes out the fault data of size M from the buffer and updates the network parameters θ and θ', as shown by the formulas in lines 16-18 of Algorithm 1 with stochastic gradient descent at each step of the episode. Parameters θ and θ' contain the relationship between some observations of the power system and the fault parameters in the fault identification model. If s'_t is the terminal state, a new episode is started, the environment is randomly initialized, and the steps mentioned above are repeated.

IV. Test Results and Analysis

In this section, we show some cases and results. The training and testing process of tests were conducted on a Windows PC equipped with an NVIDIA GeForce GTX 1070 GPU processor, an Intel Core i7-6700 3.40 GHz CPU Processors and 32 GB RAM. To illustrate the capabilities of the proposed DRL fault parameter identification method and the population-based optimization method, the supervised learning method are compared in some of these scenarios stand on IEEE 14 bus system.

The observation states are the three-phase voltage magnitude of 13 nodes and the active power of 5 generators as well as the reference signal and reward evaluation index. The action is the value of fault parameter, β. As illustrated in Section II.B, there are 14 action values, which are suffixed from 0 to 13, where '0' represents a normal situation, while each other value corresponds to the disturbance of the corresponding number of nodes. The training process terminates when the average reward reaches 100, which is the same in every scenario. The window length for calculating the average reward is different for each scenario, and the details are discussed in the following subsection. Other hyperparameters are shown in Table II.

TABLE II
TRAININGTRAINING HYPERPARAMETERS

Symbol	Description	Value
N_{2a}	Units in hidden layers	128
η	Learning rate	0.005
$\epsilon_{\min}$	Minimum exploration rate	0.01
τ	Target smooth factor	0.1
γ	Discount factor	0.9
n	Steps to lookahead	10
C	Experience buffer length	10000
M	Sample minibatch size	128
f	Dynamic simulation frequency (Hz)	60

During each episode, the agent interacts with the environment at a time step of 0.2 s. The environment terminates at 2 s. Therefore, each episode has 10 steps. Note that the hyperparameters are the same and the interaction interval is identical to other cases tested in the rest of this paper, and all these cases are deployed using a MATLAB/Simulink platform.

A. Scenario I

In Scenario I an LLLG fault is applied at bus 5 at 1.0 s with a fault duration 0.2 s. The goal in this scenario is limited to use a network to detect one fault only, so the length of the window used to calculate the average reward is set to 3 s. In other words, it is when the reward equals to 100 three times in a row, that means the agent has learned this fault parameter mode. In addition, we also employed genetic algorithm (GA) and particle swarm optimization (PSO) to compare with the DRL algorithm. The moving average reward during the training is shown in Fig. 4.

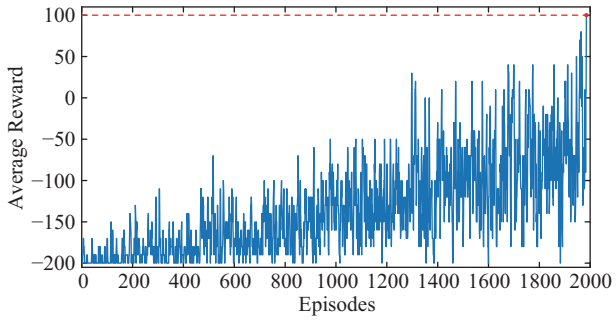


Fig. 4. Moving average reward during training for Scenario I.

The entire training process took 20774 s, with a total of 1985 episodes. This training process may look messy, but in fact, it reflects a normal RL training process. The reason is that the DRL agent completes training in a continuous exploration process. When exploring, the agent has a certain chance, i.e., ϵ to make a random action. When the random action is incorrect, the reward naturally becomes a penalty. It can be seen from Fig. 4 that in the early stage, most of the rewards are negative, as the agent has not accumulated enough successful experiences and can only explore in the penalty space. Once the agent gets a correct action through exploration, it can quickly apply those experiences to training, which corresponds to the second half of the process. The trend of the entire average reward gradually increase, which further shows that the DRL method has a strong learning ability.

For the sake of fairness, we spent the same training time on using GA and PSO algorithm based on Scenario I. GA and PSO were made to do the same thing as DRL, and the fault parameter was changed every 0.2 s to make the outputs conform to the reference signal.

The experimental results are shown in Fig. 5. It can be deduced from Fig. 5 that DRL can reproduce the fault situation successfully by adjusting the fault parameters. However, the performance of GA and PSO is poor, and they may even be considered ineffective. An inference can be possibly used to explain why the population-based approach is not satisfactory. First, the way it obtains the policy is quite different. The proposed method based on DRL can extract the useful feature information through the environment by observing and interacting with the environment and map the feature information to the correct actions through continuous exploration. However, the population-based method only carries out operations such as mutation, selection, inheritance, and iteration within the

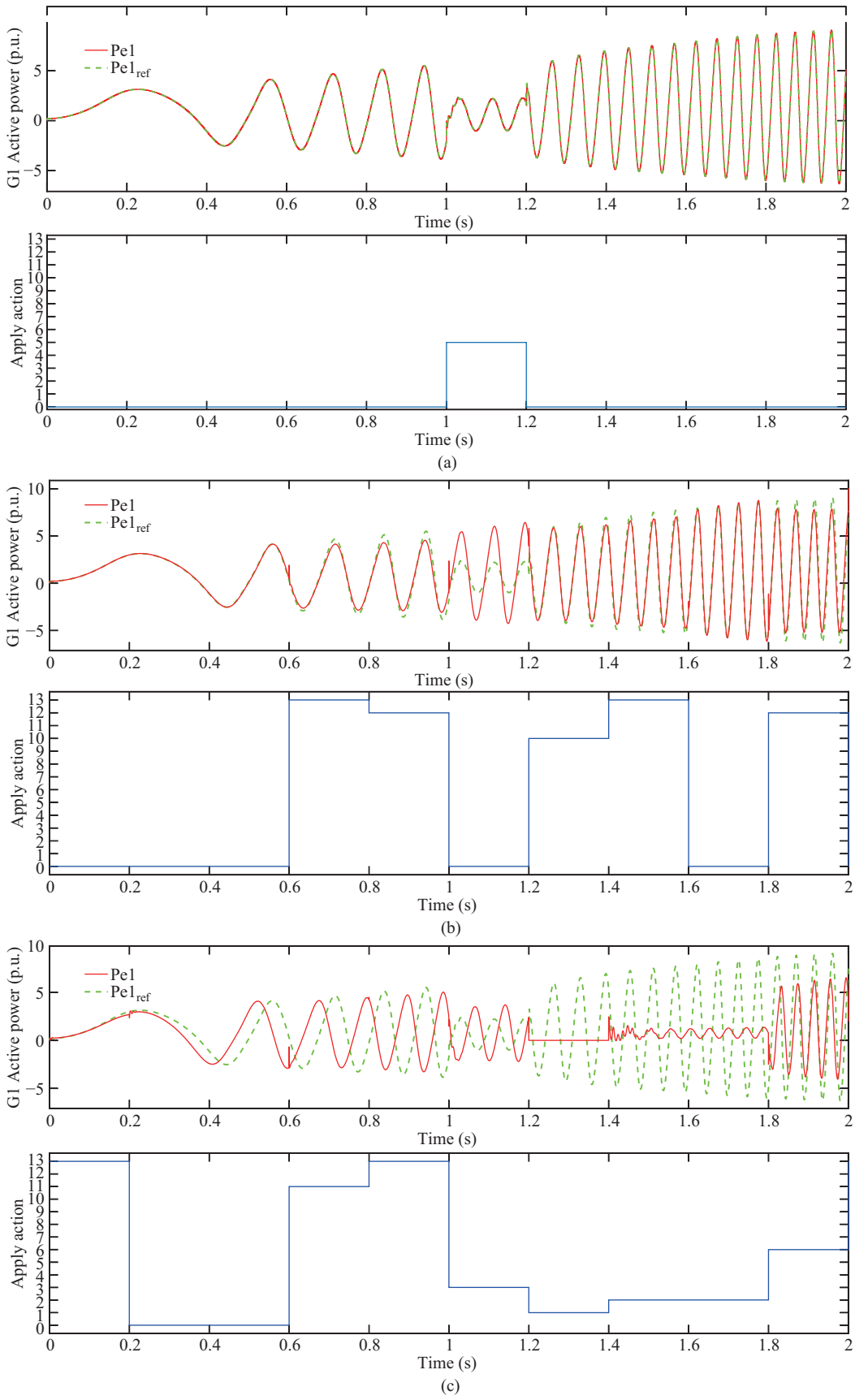


Fig. 5. The change process of the active power of Generator 1 and the reference signal: (a) DRL algorithm; (b) GA algorithm; (c) PSO algorithm.

population. It is just a self-renewal and repair method rather than an intelligent method. The entire selection process does not interact with the environment at all.

In fact, the problem of fault parameter identification can be simplified into an optimization problem of finding a minimum point. The population-based method is dependent to a great extent on the selection of the initial population. In further experiments, we found that if the initial population is selected well, that is, the starting point is very close to the minimum, the population-based method also achieves good results. But this violates our original intention to choose an intelligent approach without human interference. Therefore, in Scenario I, we tested the DRL and the population-based methods based on a single fault, and concluded that the DRL method can effectively extract faulty features and apply them to update the parameters of the DQN neural network.

B. Scenario II

In Scenario I, the proposed method can identify one fault successfully. However, in the actual industrial production process, the classes of fault faced by the power grid are diverse. If we follow Scenario I, we must train for each fault separately, which is very time-consuming and impractical. It is worth nothing than a good fault identification model should have the ability to identify as many faults as possible under one network architecture and set of hyperparameters.

In this scenario, five faults were introduced to node 1, 2, 3, 6, and 8, which are the corresponding nodes of the 5 generators. Similar to Scenario I, an LLLG fault is applied at 1.0 with a duration 0.2 s. The window length for calculating average reward is set to 10 s for the reason that we have 5 reference signals that are needed to be traced. If the reward for 10 consecutive episodes is 100, we denote that the agent has learned the feature of these 5 faults.

It took 37458 s with a total of 3432 episodes to finish the training process. The moving average reward during DRL training for Scenario II and the change process of the active power of the generator 1 to 5 with their reference signal are shown in Figs. 6, 7. The number of episodes is greater than for Scenario I as it more time is needed to explore and experience, and for the fault capture feature. Although there may be some gaps in the middle due to the ϵ greedy strategy, this does not affect the overall upward trend.

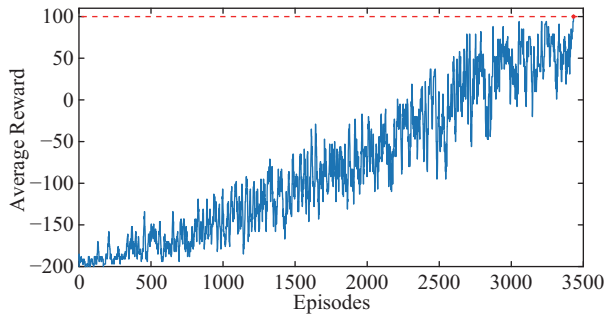


Fig. 6. Moving average reward during training for Scenario II.

The first five figures in Fig. 7 correspond to the active output power of bus 1 (generator 1) when the fault occurs at bus 1,

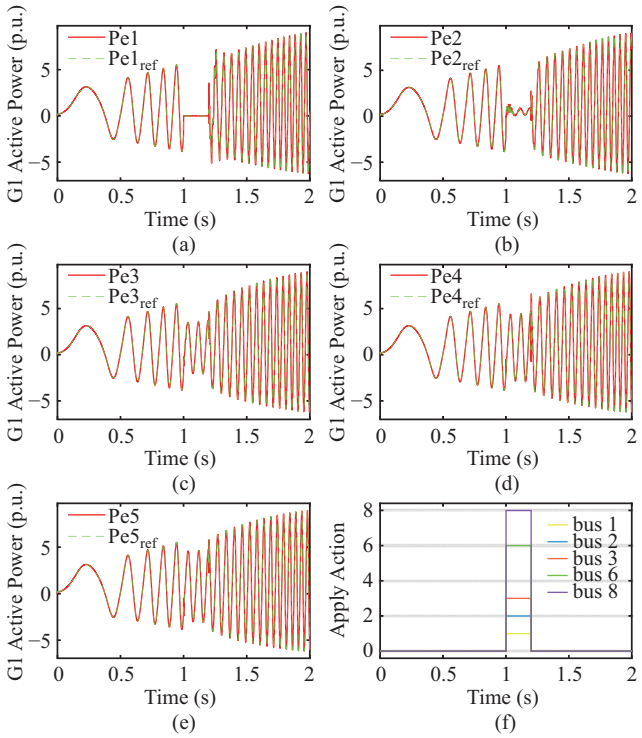


Fig. 7. The change process of the active power of the generator 1 with its reference signal: (a) The location of the fault is at bus 1; (b) The location of the fault is at bus 2; (c) The location of the fault is at bus 3; (d) The location of the fault is at bus 6; (e) The location of the fault is at bus 8; (f) is the corresponding action taken by the agent when it encounters the corresponding reference signal.

bus 2, bus 3, bus 6, and bus 8, respectively. The five reference signals are reproduced through fault parameter identification at their respective fault location. The last figure (Fig. 7(f)) shows the actions taken by the agent in the respective failure conditions. When the bus 1 fault occurs, the agent has ability to adjust the fault parameter to 1 correctly, that is, the agent can inform the operator that an LLLG fault occurred at bus 1 during the period of 1 s to 1.2 s. The result shows that the DRL agent can manage not only one type of fault but also many types of faults through a neural network architecture with fixed hyperparameters.

C. Scenario III

In this scenario, two issues need to be considered.

1) An excellent fault identification method should not only have a strong fault feature extraction ability, but also have certain anti-interference ability. The observations are often noisy due to the inherent characteristics of the data collector. Therefore, it is necessary for the fault parameter identification method to have a certain anti-noise ability.

2) Although the traditional data-driven fault identification method based on supervised learning has achieved good results in many fields, there are still two factors hindering its development.

The first is that the data-driven supervised learning method is deeply dependent on the quality of data. In other words, regardless of whether there is only one or several faults in the training data provided to this method, the method can merely

learn these faults. For some very low probability failures which rarely occur during working condition, the traditional data-driven supervised learning method will not have the ability to analyze this kind of fault.

The second is that the supervised learning method need to improve the overall efficiency. The data provided for training usually requires a lot of time to prepare, including data generation, cleaning and category-labelling. On the other hand, the process of training a deep neural network is also time-consuming.

To solve these problems and verify the superiority of the proposed method, we tested two situation that involves 13 faults in a noisy environment and compared the effectiveness between the proposed method with other state of art method in the field of machine learning. These cases correspond to a LLLG fault between bus 1 and bus 13 whose duration is 0.2 s from 1 s to 1.2 s. In order to be closer to the actual situation, a Gaussian white noise with an intensity of 30 dB shown in Fig. 8 was added to the observations in one case. In another case, the proposed method was deployed in a sparse data situation, which was used to simulate the sparse target label data encountered in the actual application of supervised learning. The data set of the second case consists of 10000 fault data. Among them, 90% are data for single-phase ground faults, 9% are data for double phase-grounded faults, and 1% are data for LLLG faults, studied in this paper. The LSTM and CNN are relatively mature methods in the field of machine learning fault identification and have been proven to be effective in most cases. Therefore, we employed them to represent the supervised learning method in this paper. The details of the LSTM and CNN can be found in [31], [32].

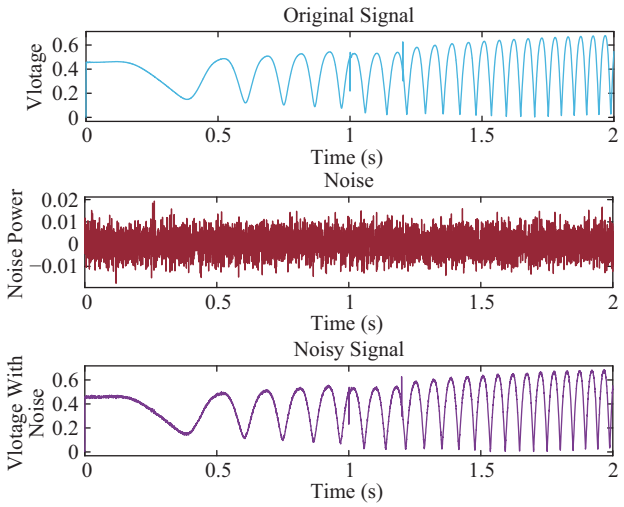


Fig. 8. The noise and noisy signal.

The window length of this scenario for the proposed method is set to 19. It took 50888 s with a total of 4637 episodes to finish training process. The moving average reward during training for scenario III is shown in Fig. 9. It took a longer time than that for Scenario II due to its fault mode being more complicated. Like all the other training processes, the agent behaved clumsily at the beginning but grasped the key to the fault pattern gradually with the enlarging fo the experience buffer.

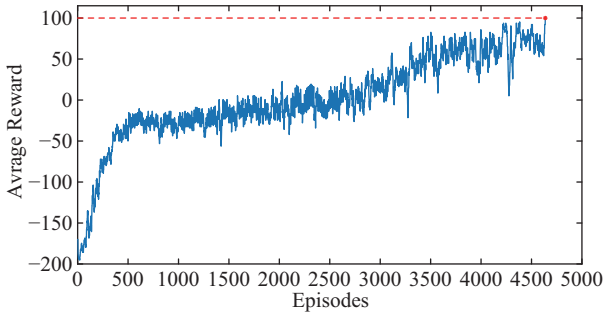


Fig. 9. Moving average reward during training for scenario III.

The results of different methods under different conditions can be viewed in Table III. In terms of identification accuracy, the proposed method is 100% accurate and outperform the LSTM-based and CNN-based methods with respect to both noise condition and noise-free condition. It is obvious from the results that the proposed method can still exert excellent results in the case of sparse data, while the LSTM-based method can merely maintain an accuracy of 8.16% in the validation data set, thus, it fails to carry out identification of the objective fault. This is because the proposed DRL-based method pays more attention to the fault data than can generate rewards. Once the rewards of such fault data are obtained from the environment, it reviews these experiences more in the future. Therefore, it is more sensitive to the target type of fault data and finds it easier to grasp the connotative fault mode from it.

TABLE III
TESTING RESULT OF SCENARIO III

Method	SNR (dB)	Accuracy (%)	Overall time (s)
DQN	0	100	21743
LSTM	0	99.84	67380
LSTM(missing data)	0	8.16	66421
CNN	0	99.69	62640
DQN	30	100	50888
LSTM	30	97.97	78965
CNN	30	99.82	76585

In terms of anti-noise ability, the proposed method can maintain the consistency of accuracy, but the overall time increases. In contrast, the LSTM-based and CNN-based methods performs worse in terms of accuracy and overall efficiency due to the introduction of noise.

In terms of overall efficiency, the total time cost of the proposed method is lower than that of the LSTM-based and CNN-based methods. The main differences are that the training time of the proposed method includes data collection and data analysis, and, the experience replay mechanism is sufficiently powerful, whereas, the supervised learning methods require additional time for data collection and data preprocessing, and need independent training time.

The results show that both the DRL method and the supervised learning method can correctly and effectively classify the states of different fault with a sufficient dataset. However, the DRL method outperform supervised learning in the aspect of overall efficiency as well as the case of sparse datasets.

Results show that the DRL method has the ability to mine the features related to the fault modes from the raw observation signals even in a noisy condition or with a sparse dataset.

V. Conclusion

To enhance the ability of the intelligent fault identification method as regards both overall efficiency and accuracy, this research sheds new light on a novel power network fault identification method. The proposed method is based on deep reinforcement learning, with the power network's fault being regarded as a subset of parameters of the simulation model, which are continuously adjusted in the simulation process by means of DRL, to achieve fault identification.

The key advance between the presented method and supervised learning is that while supervised learning requires fault labels for training, the presented method takes the similarity of simulation model output between known reference faults and sampled fault/action as a reward signal for training, which can help existing data-driven machine learning methods get rid of their dependence on data labels.

Furthermore, we compared the proposed method with traditional population-based optimization algorithms, such as GA and PSO, and supervised learning methods, such as LSTM and CNN. The satisfactory results of the proposed method demonstrate the robustness and effectiveness of the newly developed DRL-based fault identification method, as well as its advantages over conventional population-based optimization methods and supervised learning methods.

This research has broadened the application scope of reinforcement learning in power systems, which could motivate researchers to consider that the problem of power system fault identification can be transformed from a classification problem that is based on large amounts of data into a special parameter identification and optimization problem that can be solved even with limited amount of data. Moreover, this work is conducted automatically by the DRL agent without excessive manual intervention. This represents a significant advance in the development of artificial intelligence in the field of power systems.

Future studies could fruitfully explore this issue further by conducting experiments on large-scale power grids with more fault types.

References

[1] S. Chatterjee and B. K. S. Roy, "Fast identification of symmetrical or asymmetrical faults during power swings with dual use line relays," *CSEE Journal of Power and Energy Systems*, vol. 6, no. 1, pp. 184–192, Mar. 2020, doi: 10.17775/CSEEJPES.2019.01440.

[2] V. Telukunta, J. Pradhan, A. Agrawal, M. Singh, and S. G. Srivani, "Protection challenges under bulk penetration of renewable energy resources in power systems: a review," *CSEE Journal of Power and Energy Systems*, vol. 3, no. 4, pp. 365–379, Dec. 2017, doi: 10.17775/CSEEJPES.2017.00030.

[3] S. Hasan, "A hybrid circuit breaker based on current commutation approach for multi-feeder DC railway substations," *CSEE Journal of Power and Energy Systems*, vol. 5, no. 2, pp. 234–239, Jun. 2019, doi: 10.17775/CSEEJPES.2017.00290.

[4] J. Ding, X. M. Bai, W. Zhao, Z. Fang, Z. H. Li, and Z. Z. Zhong, "Fault information analysis and diagnosis method of power system based on complex event processing technology," *Proceedings of the CSEE*, vol. 27, no. 28, pp. 40–45, Jan. 2007.

[5] T. Minakawa, Y. Ichikawa, M. Kunugi, K. Shimada, N. Wada, and M. Utsunomiya, "Development and implementation of a power system fault diagnosis expert system," *IEEE Transactions on Power Systems*, vol. 10, no. 2, pp. 932–940, May 1995, doi: 10.1109/59.387936.

[6] H. C. Shu, Z. Gong, and X. C. Tian, "Fault-section identification for hybrid distribution lines based on principal component analysis and Euclidean distance," *CSEE Journal of Power and Energy Systems*, vol. 7, no. 3, pp. 591–603, May 2021, doi: 10.17775/CSEEJPES.2018.00850.

[7] S. S. Gururajapathy, H. Mokhlis, and H. A. Illias, "Fault location and detection techniques in power distribution systems with distributed generation: a review," *Renewable and Sustainable Energy Reviews*, vol. 74, pp. 949–958, Jul. 2017.

[8] X. N. Lin, S. H. Ke, Z. T. Li, H. L. Weng, and X. H. Han, "A fault diagnosis method of power systems based on improved objective function and genetic Algorithm-Tabu search," *IEEE Transactions on Power Delivery*, vol. 25, no. 3, pp. 1268–1274, Jul. 2010, doi: 10.1109/TPWRD.2010.2044590.

[9] Y. L. Wang, S. J. Wen, and D. Y. Wang, "Fuzzy fault diagnosis method based on particle swarm optimization algorithm," in *Proceedings of the 9th International Conference on Fuzzy Systems and Knowledge Discovery*, Chongqing, 2012, pp. 307–310, doi: 10.1109/FSKD.2012.6233896.

[10] A. Abel Hafez, W. A. Omran, and Y. G. Hegazy, "A decentralized technique for autonomous service restoration in active radial distribution networks," *IEEE Transactions on Smart Grid*, vol. 9, no. 3, pp. 1911–1919, May 2018, doi: 10.1109/TSG.2016.2602541.

[11] Z. S. Hosseini, M. Mahoor, and A. Khodaei, "AMI-enabled distribution network line outage identification via multi-label SVM," *IEEE Transactions on Smart Grid*, vol. 9, no. 5, pp. 5470–5472, Sep. 2018, doi: 10.1109/TSG.2018.2849845.

[12] F. Dehghani and H. Nezami, "A new fault location technique on radial distribution systems using artificial neural network," in *Proceedings of the 22nd International Conference and Exhibition on Electricity Distribution (CIRED 2013)*, Stockholm, 2013, pp. 1–4, doi: 10.1049/cp.2013.0697.

[13] S. L. Wen, Y. Wang, Y. Tang, Y. Xu, P. F. Li, and T. Y. Zhao, "Real-time identification of power fluctuations based on LSTM recurrent neural network: a case study on Singapore power system," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 9, pp. 5266–5275, Sep. 2019, doi: 10.1109/TII.2019.2910416.

[14] M. Fan, Y. L. Liu, X. Zhang, H. Chen, Y. Q. Hu, L. B. Fan, and Q. Yang, "Fault prediction for distribution network based on CNN and LightGBM algorithm," in *Proceedings of the 14th IEEE International Conference on Electronic Measurement & Instruments (ICEMI)*, Changsha, China, 2019, pp. 1020–1026, doi: 10.1109/ICEMI46757.2019.9101423.

[15] H. T. Shan, Y. Y. Sun, W. J. Zhang, A. Kudreyko, and L. J. Ren, "Reliability analysis of power distribution network based on PSO-DBN," *IEEE Access*, vol. 8, pp. 224884–224894, Jul. 2020, doi: 10.1109/ACCESS.2020.3007776.

[16] S. Chakraborty, R. Tomsett, R. Raghavendra, D. Harborne, M. Alzantot, F. Cerutti, M. Srivastava, A. Preece, S. Julier, R. M. Rao, T. D. Kelley, D. Braines, M. Sensoy, C. J. Willis, and P. Gurram, "Interpretability of deep learning models: a survey of results," in *Proceedings of 2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, San Francisco, CA, 2017, pp. 1–6, doi: 10.1109/UIC-ATC.2017.8397411.

[17] A. L. Yuille and C. X. Liu, "Deep nets: what have they ever done for vision?" *International Journal of Computer Vision*, vol. 129, no. 3, pp. 781–802, Mar. 2021.

[18] Z. D. Zhang, D. X. Zhang, and R. C. Qiu, "Deep reinforcement learning for power system applications: an overview," *CSEE Journal of Power and Energy Systems*, vol. 6, no. 1, pp. 213–225, Mar. 2020, doi: 10.17775/CSEEJPES.2019.00920.

[19] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. T. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017.

[20] OpenAI Five. (2018, Oct. 30). OpenAI. [Online]. Available: https://blog.openai.com/openai-five/.

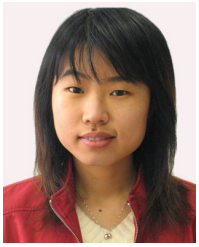
[21] N. Golovin and E. Rahm, "Reinforcement learning architecture for Web recommendations," in *Proceedings of 2004 International Conference on Information Technology: Coding and Computing*, Las Vegas, NV, USA, 2004, pp. 398–402, doi: 10.1109/ITCC.2004.1286487.

[22] S. X. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy

updates," in *Proceedings of 2017 IEEE International Conference on Robotics and Automation (ICRA)*, Singapore, 2017, pp. 3389–3396, doi: 10.1109/ICRA.2017.7989385.
[23] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge: MIT Press, 1998.
[24] C. J. C. H. Watkins and P. Dayan, "*Q*-learning," *Machine Learning*, vol. 8, no. 3–4, pp. 279–292, May 1992.
[25] Q. H. Huang, R. K. Huang, W. T. Hao, J. Tan, R. Fan, and Z. Y. Huang, "Adaptive power system emergency control using deep reinforcement learning," *IEEE Transactions on Smart Grid*, vol. 11, no. 2, pp. 1171–1182, Mar. 2020, doi: 10.1109/TSG.2019.2933191.
[26] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
[27] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," arXiv: 1312.5602, 2013.
[28] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, Phoenix, Arizona, USA, 2016, pp. 2094–2100.
[29] J. A. Boudreaux, "Design, simulation, and construction of an IEEE 14-bus power system," M.S. thesis, Louisiana State University, Louisiana, 2018.
[30] P. K. Lim and D. S. Dorr, "Understanding and resolving voltage sag related problems for sensitive industrial customers," in *Proceedings of 2000 IEEE Power Engineering Society Winter Meeting*, Singapore, 2000, pp. 2886–2890, doi: 10.1109/PESW.2000.847343.
[31] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.
[32] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 1, pp. 142–158, Jan. 2016.

Mengshi Li received the M.Sc. (Eng) degree with distinction in Information and Intelligence Engineering from the Department of Electrical Engineering and Electronics, the University of Liverpool, UK, in 2005. He received the Ph.D. degree in Electrical Engineering from University of Liverpool, UK, in 2010. He is currently a Lecturer in School of Electric Power, South China University of Technology. His research interests include computational intelligence and their applications in power systems.

Huanming Zhang received the B.S. degree in Electrical Engineering from Beijing Jiaotong University, Beijing, China, in 2015. He is currently pursuing the M.S. degree in Electrical Engineering at South China University of Technology, Guangzhou, China. His research interests include the application of artificial intelligence in power systems, pattern recognition and fault diagnosis, and power grid aided decision-making design based on reinforcement learning.



Tianyao Ji received the B.Eng. degree in Information Engineering in 2003, the B.A. degree in English in 2003 and the M.Sc. degree in Signal and Information Processing in 2006 from Xi'an Jiaotong University, Xi'an, China. In 2009, she obtained the Ph.D. degree in Electrical Engineering and Electronics from University of Liverpool, Liverpool, UK from 2010 to 2011, she worked as a Research Associate in University of Liverpool for two years. She is now an associate professor at School of Electric Power Engineering, South China University of Technology. Her research interests include mathematical morphology, signal and information processing, power system protection and evolutionary computation.

Qinghua Wu obtained the M.Sc. (Eng) degree in Electrical Engineering from Huazhong University of Science and Technology, Wuhan, China, in 1981. From 1981 to 1984, he was appointed Lecturer in Electrical Engineering in the University. He obtained the Ph.D. degree in Electrical Engineering from The Queen's University of Belfast (QUB), Belfast, UK in 1987. He worked as a Research Fellow and subsequently a Senior Research Fellow in QUB from 1987 to 1991. He joined the Department of Mathematical Sciences, Loughborough University, Loughborough, UK in 1991, as a Lecturer, subsequently he was appointed Senior Lecturer. In September, 1995, he joined The University of Liverpool, Liverpool, UK to take up his appointment to the Chair of Electrical Engineering in the Department of Electrical Engineering and Electronics. Since then, he has been the Head of Intelligence Engineering and Automation Research Group working in the areas of systems control, computational intelligence and electric power and energy. He is now with South China University of Technology and has authored and coauthored more than 350 technical publications, including 160 journal papers, 20 book chapters and 3 research monographs published by Springer. Professor Wu is a Fellow of IEEE, Fellow of IET, Chartered Engineer and Fellow of InstMC. His research interests include nonlinear adaptive control, mathematical morphology, evolutionary computation, power quality and power system control and operation.