A New Reinforcement Learning based Learning Rate Scheduler for Convolutional Neural Network in Fault Classification

Long WEN, Xinyu LI, *Member, IEEE,* and Liang GAO, *Senior Member, IEEE*

***Abstract*—Convolutional Neural Network (CNN) has gained increasing attentions in fault classification. However, the performance of CNN is sensitive to its learning rate. Some previous work has been done to tune the learning rate, including the 'trial and error' and manual search, which heavily depend on the experts' experiences and should be conducted repeatedly on every dataset. Because of the variety of the fault data, it is time-consumption and labor-intensive to use these traditional tuning methods for fault classification. To overcome this problem, this research develops a novel Learning Rate Scheduler based on Reinforcement Learning (RL) for Convolutional Neural Network (RL-CNN) in fault classification, which can schedule the learning rate efficiently and automatically. Firstly, a new RL agent is designed to learn the policies about the learning rate adjustment during the training process. Secondly, a new structure of RL-CNN is developed to balance the exploration and exploitation of the agent. Thirdly, the bagging ensemble version RL-CNN-Ens is presented. Three bearing datasets are used to test the performance of RL-CNN-Ens. The results show that RL-CNN-Ens outperforms traditional DLs and machine learning methods. Meanwhile, RL-CNN-Ens can find the state-of-the-art learning rate schedulers as human designed, showing its potential in fault classification.**

***Index Terms*—Fault Classification, Learning Rate, Reinforcement Learning, Convolutional Neural Network.**

I. Introduction

FAULT diagnosis plays the vital role in modern manufacturing industry [1][2]. The accurate and efficient fault diagnosis methods can significantly reduce the risk of dangerous situations and also save invaluable time to take remedy to reduce the considerable economic loss or even human loss. With the development of smart manufacturing and artificial intelligence, data-driven fault diagnosis has proven to be an effective way [3][4]. The procedure of data-driven fault diagnosis includes data acquisition, feature extraction and fault classification [5]. Various deep learning (DL) techniques based fault classification methods have been studied from both the academia and engineering fields [6].

This work was supported in part by the National Key R&D Program of China (2019YFB1704600), the Natural Science Foundation of China under Grant 51805192 and 51775216, and sponsored by the State Key Laboratory of Digital Manufacturing Equipment and Technology (DMET) of Huazhong University of Science and Technology (HUST) under Grant No. DMETKF2020029. (Corresponding author: Liang GAO)

Long WEN is with School of Mechanical Engineering and Electronic Information, China University of Geosciences, No. 388 Lumo Road, Wuhan, 430074, China (e-mail: wenlong@cug.edu.cn).

Xinyu LI, and Liang Gao are with the State Key Laboratory of Digital Manufacturing Equipment & Technology, Huazhong University of Science and Technology, 1037#, Luoyu Road, Wuhan, 430074, China (lixinyu@mail.hust.edu.cn, gaoliang@mail.hust.edu.cn)

Convolutional Neural Network (CNN) is one of the most popular DL based fault classification in recent years [7]. However, even though CNN methods have achieved the remarkable application improvements, their performances heavily depend on their hyper-parameters. But the default hyper-parameters cannot guarantee the final performances of CNN [8], which makes the hyper-parameter tuning process is essential for the application of CNN on fault classification.

Learning rate is one of the most important hyper-parameters in CNN [9]. Even though there are several suggestions on the adjustment of the learning rate during training (which is also denoted as leaning rate scheduler) [10], but it is still difficult to select the proper leaning rate scheduler with its optimal hyper-parameters. The main barriers include: 1) The tuning processes for learning rate scheduler and its hyper-parameters are mostly by trial and error or manual search, so the tuning results depend on the experts' experience; 2) For a new dataset, it is hard to know that in which ranges the learning rate can be tuned; 3) The good learning rate schedulers are problem specific, which makes that the tuning process should be repeatedly conducted on every application dataset. These situations are more urgent in fault classification. Because the data in fault classification is more variety than other fields, the tuning process on learning rate in CNN based fault classification is very time-consumption and labor-intensive.

To overcome the above drawbacks, some previous works have been conducted attempting to construct the leaning rate scheduler in the automatic way. The learning rate scheduler is modeled as the sequential decision process, and reinforcement learning (RL) is applied to train an RL based agent as the learning rate scheduler to control the learning rate at each training step. These RL based agent can learn the policy from the historical information of the training process to guide the adjustment on the learning rate, and they have been tested on Artificial Neural Network (ANN) [11]. However, these agents are preferred to be designed only to decay the learning rate, or to reset to the initial values, so there is a de-facto pre-defined upper constraints on the learning rate range, making these agents are semi-automatic.

In this research, a novel RL based learning rate scheduler is developed for CNN based fault classification, named RL-CNN. In the proposed method, the constraint on the learning rate range is eliminated and RL-CNN can adjust the learning rate to any value. However, it is well known that too small a learning rate will make a training algorithm converge slowly while too large a learning rate will make the training algorithm diverge [12]. So it is challenge for RL-CNN to find the reasonable and efficient learning rate scheduler. Based on this, the new structure and training algorithm are developed for RL-RNN. Firstly, a new RL agent is designed as the learning rate scheduler. It can automatically increase, decrease or preserve the learning rate, which can explore the whole learning rate space to find the proper learning rate values. Secondly, a new structure of RL-CNN is proposed for fault classification. It applied double CNN and double Deep Q-Network (DDQN) to balance the exploration and exploitation on the learning rate and to avoid the over-estimation on the learning rate trend. Finally, the bagging ensemble version of RL-CNN, which is denoted as RL-CNN-Ens, is presented to enhance its ability. The proposed RL-CNN-Ens is tested on the fault classification of three bearing datasets and the results show that RL-CNN can schedule the learning rate at the reasonable range and it can provide the state-of-the-art performance for fault classification.

The rest of this paper is organized as follows. Section II discusses the related work. Section III presents the background of reinforcement learning. Section IV gives the methodologies of the proposed RL-CNN-Ens. Section V shows the case study results of RL-CNN-Ens on three datasets. Finally, Section VI presents the conclusion and future research.

II. Related Work

The related work is described in this section, including CNN based fault classification and learning rate tuning for CNN.

A. CNN based Fault Classification

Data driven fault classification has gained increasing attentions in recent years. Convolutional Neural Network (CNN) has been widely applied in data-driven fault classification. Khan and Yairi [13] presented the systematic review of artificial intelligence based system health management with the emphasis on the recent trends of DL for fault classification. Zhao et al. [7] studied the DL applications to machine health monitoring, including CNN. Zhao et al. [14] applied the deep residual network for the fault classification of planetary gearboxes. Chen et al. [15] investigated the transferable CNN which is pre-trained based on large source dataset for the intelligent fault classification of rotary machinery. Wen et al. [16] studied the two-level hierarchical classification network based on CNN and validated the proposed method on the different working load. Zhu et al. [17] developed the CNN with capsule network for the fault classification of bearing, and the experiment validated its generalization ability.

The performances of CNN heavily depend on its hyper-parameters, such as learning rate. Many approaches have been investigated aiming at tuning the hyper-parameters efficiently, such as Hyper Parameters Optimization (HPO) [18]. The most prominent HPO approaches are Grid Search, Random Search and Bayesian Optimization [19]. However, even though HPO approaches have achieved remarkable improvements, but they require that the hyper-parameters should be kept as constant. As the learning rate is always varied during the training process, making the HPO approaches are not suitable for tuning learning rate. Snoek et al. [20] studied the pseudo-Bayesian neural network to tune the learning rate of deep neural network, while the learning rate scheduler is fixed to one value in all iterations.

Because the data in fault classification is more variety than other fields, making the tuning process on learning rate is more difficult, so it is promoting to find the automatic way for tuning the learning rate.

B. Learning Rate Tuning for CNN

Learning rate tuning process is essential for CNN based fault diagnoses. Traditionally, the tedious trial and error or manual search are applied until the good choices for learning rate is founded. Xia et al. [21] investigated the fault classification on roller bearing by testing the learning rate values from 0.005 to 0.05 with the interval of 0.005, which uses ten repeatedly running to select the learning rate. Xie et al. [22] tried to trial and error method to select the optimal learning rate, number of kernels, number of weights in each layer, as well as the batch size for the fault classification for rotating machinery based on CNN.

Since the trial and error or manual search are very time-consumption and labor-intensive, some previous works attempted to apply RL to construct an automatic learning rate tuning process. Hansen [11] firstly applied RL to realize the RL based agents as learning rate scheduler for ANN, and achieved the good results. However, the agents are preferred to be designed only to decay or reset learning rate, making them semi-automatic. Even so, the application of RL based learning rate scheduler on CNN based fault classification is few.

In this research, a new RL agent which can control learning rate automatically is constructed as the learning rate scheduler and tested for the CNN based fault classification.

III. Background of Reinforcement Learning

This section presents the background of reinforcement learning and double deep Q-network (DDQN).

A. Reinforcement Learning

Reinforcement learning (RL) is the presiding methodology to train an agent that perform the tasks within the environment. The task can be modeled as a sequence decision problem, and it has a clear underlying goal. At the certain time step t, RL requires the agent to take an action ($a_t \in A$) according its state ($s_t \in S$) and policy ($\pi : S \to A$). Then the agent will receive the reward (r_{t+1}) from environment and arrive at next state s_{t+1} .

The aim of the RL agent is to maximize the cumulative reward, and a successful agent should balance the immediate reward and its overall goal. RL achieves this via the action-value function (also denoted as *Q*-value function) $Q^{\pi}(s,a):(S,A)\rightarrow\mathbb{R}$, which is the discounted expected return of the rewards of given state *s*, action *a* and policy π, as shown in equation (1). *T* is the maximum number of time steps, γ is the discounted factor. The expectation of the discounted expected return is calculated following policy π. The optimal action-value function $Q^{*}(s,a)=\max_{\pi}Q^{\pi}(s,a)$ satisfies the well-known Bellman equations, as shown in equation (2)

$$Q^{\pi}(s,a)=E_{\pi}\left\{\sum_{k=t}^{T}\gamma^{k}r_{k}\,|s_{0}=s,a_{0}=a\right\} \tag{1}$$

$$Q^{*}(s_{t},a_{t})=E\left\{r_{t+1}+\gamma\max_{a'}Q^{*}(s_{t+1},a')\,|\,s_{t}=s,a_{t}=a\right\} \tag{2}$$

When the number of the states and actions are both finite, the *Q*-value function is a look-up *Q* table. Otherwise, this *Q* table should be approximated by a surrogate model. Deep *Q*-network (DQN) belongs to this kind and it applies ANN as the surrogate model. In this research, the state of RL-CNN-Ens is the six-dimension continuous vector, so the DQN variant, named DDQN, is applied to construct the RL agent.

B. Double Deep Q-Learning

DQN is denoted as $QNet(s;\theta):\mathbb{R}^{|S|}\rightarrow\mathbb{R}^{|A|}$,which maps the state to the action as *Q*-value function. θ is the weight of DQN. The training of DQN is to minimize the expected Temporal Difference (TD) error of the optimal Bellman equation $E_{s_t,a_t,r_{t+1},s_{t+1}}\left\|\hat{y}_t-y_t\right\|_2^2$, where the target y_t and its estimate $\hat{y}_t$ are shown in equation (3) and (4).

$$\hat{y}_{t}=QNet(s_{t};\theta) \tag{3}$$

$$y_{t}=\begin{cases}r_{t+1}+\mathbf{1}_{t\neq T-1}\left(\gamma\max_{a'}\left[QNet(s_{t+1};\theta)\right]_{a'}\right) & a=a_{t}\\ QNet(s_{t};\theta) & a\neq a_{t}\end{cases} \tag{4}$$

As shown in equation (4), DQN is applied in the prediction of *Q*-value functions in both the target y_t and its estimate $\hat{y}_t$, so DQN exists the over-estimation of the $\max_{a}Q(s,a)$. To overcome this drawback, double deep Q-learning (DDQN) is introduced. The original of DQN with weight θ^{-} (denoted as Main-QNet) are still applied when DQN is deciding an action, and the second DQN with weight θ (denoted as Target-QNet) is used to train the objective *Q*-value function. After the certain steps, the Target-QNet would be trained using Experience Replay dataset *D* and then it will copy its weight θ to the weight of Main-QNet θ^{-}.

IV. Proposed RL-CNN-Ens for Fault Classification

This section presents the proposed RL based Learning Rate Scheduler for CNN and its ensemble version (RL-CNN-Ens). Firstly, the RL model for CNN is defined, and then the training method of RL-CNN is shown. Finally, the ensemble version RL-CNN-Ens is shown.

A. Reinforcement Learning Model for CNN

In this subsection, the flowchart of RL-CNN is given, and then the definitions of state space, action space, the policy, and the reward are given to construct the RL agent for the learning rate scheduler for CNN.

1) *The Flowchart of RL-CNN for Fault Classification*

In this part, a RL based agent as the learning rate scheduler is trained to control the learning rate of CNN without the constraint on its range. The most challenge for the proposed RL-CNN is to avoid the over-estimation of the learning rate trend, which may adjust the learning rate into too large and too small values.

To find the reasonable learning rate scheduler, the flowchart of RL-CNN method is designed as shown in Fig. 1. Firstly, in spirit of DDQN, the double CNN structure is applied in RL-CNN denoted by Main-CNN and Game-CNN. Main-CNN is the main stream flow in RL-CNN, while Game-CNN is used for conducting the pre-exploration of the agent. Secondly, the RL agent concepts, including the state, action, reward and the policy, are defined to control the learning rate of the CNN models.

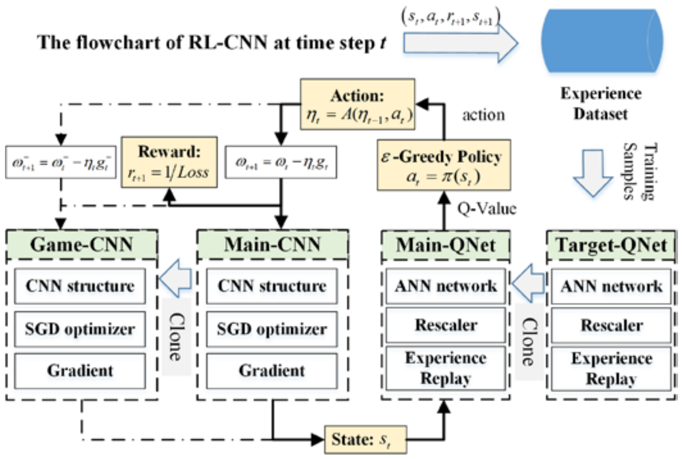

Fig. 1. The flowchart of the proposed RL-CNN

The flowchart of RL-CNN is shown in Fig. 1. At time step *t*, the state s^{t} of Main-CNN/Game-CNN will be generated firstly, and the Q-Value of it would be predicted by Main-QNet. Secondly, the ε-greedy policy is conducted to choose the right action $a_t=\pi(s_t)$ which balances the exploration and exploitation of the agent. Then, the learning rate η would be updated and the Main-CNN or Game-CNN would conduct the Stochastic Gradient Descent (SGD) using the η. Along with the update of Main-CNN or Game-CNN, the agent receives its reward r_{t+1}. Finally, the time step will arrive at *t+1*, and the pair $(s_t,a_t,r_{t+1},s_{t+1})$ is stored in the experience replay dataset.

The RL-CNN trains the RL agent using its historical training information. The environment of the RL agent is observable and dynamic. The underlying optimization goal of the RL agent is to maximize the discounted cumulative reward as shown in equation (1), and it is a single-objective optimization problem. The RL agent is trained and updated with a certain time steps using the experience replay dataset.

2) *State Space*

The state should satisfy the Markov condition to perform the RL agent. The state space is defined as a six statistic features vector as [11] in this research, and it works well in practice [23]. The elements in the state space are: a) the current learning rate, b) the loss value of CNN, c) the norm of gradient $g(x)=\nabla f(x)$,

d) the number of iteration, e) min/max encoding, and f) the state alignment of current gradient. The number of iterations also should be a state feature since the states would not be stationary. The first four states elements can be obtained directly, here present the min/max encoding and state alignment elements.

The min/max encoding indicates whether the current candidate is higher or lower than the *M* lowest achieved loss during the training process. Let F^{t-1} be the list of the *M* lowest loss values obtained up to time *t*-1, the state encoding at time step *t* is given by equation (5).

$$[s_t] = \begin{cases} 1 & f(x_t) \le \min\left(F^{t-1}\right) \\ 0 & \min\left(F^{t-1}\right) \le f(x_t) \le \max\left(F^{t-1}\right) \\ -1 & otherwise \end{cases} \tag{5}$$

The state alignment is a measure between successive descent directions, and it is as shown in equation (6).

$$[s_t]_{alignment} = \frac{1}{n}\sum_{i=1}^{n} sign\left([g_t]_i [g_{t-1}]_i\right) \tag{6}$$

3) *Action Space*

This research designs a new action space to control the learning rate. It contains five actions and they are the large increase, small increase, unchanged, small decrease, large decrease respectively. Two factors, denoted as α_1 and α_2, to control the update step size on the learning rate and it is assumed that $\alpha_1 < \alpha_2$ without loss generalization. The selection of α_1 and α_2 would affect the performance of RL agent. In this research, these two values are determined by using the most common used decay values for learning rate tuning, and α_1 is 0.99 and α_2 is 0.996.

a) Large increase (Action 0): $\eta = \eta / \alpha_1$

b) Small increase (Action 1): $\eta = \eta / \alpha_2$

c) Keep unchanged (Action 2): $\eta = \eta$

d) Small decrease (Action 3): $\eta = \eta \times \alpha_2$

e) Large decrease (Action 4): $\eta = \eta \times \alpha_1$

As RL-CNN eliminates the upper constraints on the learning rate, this action space gives RL-CNN the fully control on the learning rate totally. As shown above, with this action space, the RL-CNN can adjust the learning rate to any value with a consequence combination of the actions.

4) *Reward*

As the underlying aim of the RL agent is to maximize the cumulative reward, it can promote the convergence of agent by giving the higher reward values to the CNN models with smaller loss values. So the reward has a strong negative relationship with the loss of CNN models. In this research, the reward is defined as the reciprocal of loss to ensure that the RL agent can find the optimal objective value within the smallest number of iterations. The reward function can be presented in equation (7).

$$r_{t+1} = \frac{1}{loss} \tag{7}$$

5) ε *-Greedy Policy*

In RL, the policy is always soft, which mean that each action has a certain number of the probability to be selected, i.e. $\pi(a \mid s) > 0, \forall s \in S, \forall a \in A$. The ε -greedy policy is popular soft policy and it is widely applied to select the action. In ε -greedy policy, the probability of selection the action with maximum Q value is $1-\varepsilon+\varepsilon/\left|A(s)\right|$, and the probability of selection the other action is $\varepsilon/\left|A(s)\right|$. ε is the factor to control the selection probability and it is 0.3 in this research. The formulation of ε -greedy policy is shown in equation (8). By using ε -greedy policy, RL-CNN-Ens can enhance its global search ability to train the well performed RL agent.

$$\pi(a \mid s) = \begin{cases} 1-\varepsilon+\dfrac{\varepsilon}{\left|A(s)\right|} & if\ a = \arg\max_a Q(s,a) \\ \dfrac{\varepsilon}{\left|A(s)\right|} & f\ a \ne \arg\max_a Q(s,a) \end{cases} \tag{8}$$

B. *The Training Process of RL-CNN*

The subsection the training process of RL-CNN is presented, including the training of Double CNN and DDQN.

1) *The Training of Double CNN*

As RL agent can adjust the learning rate into any value without the limitation on the upper and lower boundaries, it is challenge for RL-CNN to find the reasonable leaning rate. In this research, the double CNN structure is adopted for the pre-exploration of the RL agent. The Main-CNN is the main stream flow in RL-CNN, as shown in the solid line in Fig. 1. The Game-CNN is the clone version of Main-CNN, which has the same workflow as Main-CNN, and it is shown in the dot dash line in Fig. 1. Game-CNN is used for the pre-exploration. Let ϖ denotes the CNN weights of Main-CNN and ϖ^- denotes the CNN weights of Game-CNN.

The training process of Main-CNN and Game-CNN is defined as in the GP function as shown in Algorithm 1. The CNN in Algorithm 1 denotes the Main-CNN or Game-CNN. During the training process of CNN, the RL components are combined and the pair $(s_{t+k}, a_{t+k}, r_{t+k+1}, s_{t+k+1})$ is stored in the experience dataset *D*, which will be used for training DDQN.

Algorithm 1: GP function for Main-CNN/Game-CNN

```
Input: Current weight $\varpi_t$, mini-batch $x_t$
For k=0, step4game do
    Obtain the current state of CNN: $s_{t+k} = \chi(\varpi_{t+k}, x_{t+k})$
    Predict the Q function $Q(s_{t+k}, a; \theta^-)$ using Main-QNet
    Select the action $a_{t+k}$ with $\varepsilon$ -greedy policy
    Execute $a_{t+k}$ and obtain the current learning rate $\eta_{t+k}$
    Sample the next mini-batch to get $x_{t+k+1}$
    Update the weight of CNN $\varpi_{t+k+1} = \varpi_{t+k} - \eta_{t+k} g_{t+k}$
    Receive the reward $r_{t+k+1}$
    Store the transition $(s_{t+k}, a_{t+k}, r_{t+k+1}, s_{t+k+1})$ in D
End For
```

The one-step training of RL-CNN is denoted as one episode, and it includes the sequencing training of the Game-CNN, DDQN and the Main-CNN. As Game-CNN is used for the pre-exploration, so whenever Main-CNN will be applied, the Game-CNN would be trained firstly and to detect the exploration space and then to update the DDQN. The one episode of RL-CNN is presented in Algorithm 2. *D* is the dataset for experience replay, and *step4game* control the update frequency of DDQN.

Algorithm 2: One episode for RL-CNN

```
Initialize Game-CNN by clone the $\omega$ to $\omega^-$
Initialize the step4game
Clone the mini-batch $x_t$ to $x_t^-$
Execute GP( $\omega^-$, $x_t^-$ ) for Game-CNN
Train the weight $\theta$ of Target-QNet using $D$
Clone the weight $\theta$ of Target-QNet to Main- QNet $\theta^-$
Execute GP( $\omega$, $x_t$ ) for Main-CNN
Output: Main-CNN, Main- QNet
```

2) *Training DDQN with Rescaling Operator*

In this research, the RL agent is implemented by the DDQN algorithm. The structure of DDQN is [6, 16, 16, 5], which maps the six-dimension state space to the five-dimension action space in RL-CNN. The activation function of DDQN is '*sigmoid*' function, so the output of DDQN $QNet(s_{t+1};\theta)$ is limited to (0,1). However, the reward r_{t+1} has the range of $(0,+\infty)$, so it is necessary to rescale r_{t+1} and $QNet(s_{t+1};\theta)$ to improve equation (4).

The rescale process (scaler) has three operators, named *fit*, *scale*, *inv_scale*, are defined, which are same with *fit*, *transform* and *inv_transform* functions in *scikit-learn*. Since the rewards of RL-CNN change sharply along with the training process, the scaler should be dynamic updated, and it is shown in equation (9-11). Equation (9) inversely transforms $QNet(s_{t+1};\theta)$ to and calculate the discounted reward y_scale_t. Equation (10) updates the parameters of the scaler. Equation (11) scales the target discounted reward, and then y_t can be used for training the DDQN by replacing the equation (4). The interval of the scaler is [0.1, 0.9] in this research. This scaler is executed with the training of Target-QNet.

$$y\_scale_t = r_{t+1} + \gamma inv\_scale\{\max_{a'}[QNet(s_{t+1};\theta)]_{a'}\} \tag{9}$$

$$scale = fit(y\_scale_t) \tag{10}$$

$$y_t = scale(y\_scale_t) \tag{11}$$

C.Ensemble of RL-CNN (RL-CNN-Ens)

Cross validation (CV) is a popular technique to obtain the reliable performance evaluation of fault classifier [24]. In this research, the proposed RL-CNN is implemented using five-fold CV to obtain the well-trained CNN models. Then these CNN models in five-fold CV would generate the ensemble version of RL-CNN using bagging ensembles, which is denoted as RL-CNN-Ens.

V. Case Studies and Experimental Results

In this section, the proposed RL-CNN-Ens based fault classification is implement by *Tensorflow* and *scikit-learn* package, and runs on Ubuntu 16.04 with RTX 2080Ti GPU. The proposed RL-CNN-Ens is conducted on three bearing datasets [25][26][27]. In these case studies, the learning rate curves of RL based agent are presented and discussed, as well as the time issue and the convergence of RL-CNN-Ens. Then, the results of RL-CNN-Ens are compared with other published DL and machine learning (ML) methods to validate it potential on fault classification.

A. Algorithm Setup

1) *Data Preprocessing based on S-Transform*

Data preprocessing is a necessary part in ML field. It converts the data into another format, which is more suitable for the subsequent ML/DL algorithms. As CNN is effective to extract the feature of 2D image data, the time-domain fault signals is converted to the time-frequency format using S-Transform technique.

S-Transform is proposed by Stockwell [28]. It can provide frequency-dependent resolution while keeping a direct relationship with the Fourier spectrum. S-Transform can extract time-varying information from non-stationary signals, which is suitable for the machinery fault classification with crossing working load, and it has been shown to be effective. S-Transform can be viewed as an extension of short-time Fourier transform (STFT). The STFT of a signal *x(t)* is defined as equation (12). τ is the time of spectral localization, *f* is the Fourier frequency, and *w(t)* is the window function.

$$STFT(\tau, f) = \int_{-\infty}^{+\infty} x(t)\,w(t-\tau)\,e^{-j2\pi ft}\mathrm{d}t \tag{12}$$

S-Transform replaces the window function using the Gaussian function as $w(t) = \frac{|f|}{\sqrt{2\pi}} e^{-t^2 f^2/2}$. Then S-Transform can be given by equation (13).

$$S(\tau, f) = \int_{-\infty}^{+\infty} x(t) \frac{|f|}{\sqrt{2\pi}} e^{-(t-\tau)^2 f^2/2} e^{-j2\pi ft}\mathrm{d}t \tag{13}$$

2) *CNN Structure of RL-CNN*

Since LeNet-5 has achieved good results on fault classification, the CNN Structures of Main-CNN and Game-CNN are also the LeNet-5 like structure. This CNN structure has six alternative convolutional and pooling layers with two-layer Fully-Connected (FC) layers followed. The detail of network structure is presented in TABLE I. 'Conv(7×7×64)' means that it is a convolution layer with 7×7 filter size and 64 depth. Maxpool layer is adopted and its pool size is 2×2. The stride size of Maxpool layer is 2×2. The hidden neurons of FC layers are 2560 and 512. Since the CNN based fault diagnoses is modeled as the multi-class classification as [25][26][27], the Softmax layer is added at the end, and it predicts the final predictions on the fault type. The cross entropy error function is used for training CNN and the average prediction accuracy (AVG) is selected for performance comparison.

TABLE I
THE LAYER CONFIGURATIONS OF CNN MODELS

Layer Name	CNN Operator	Stride	Layer Name	CNN Operator	Stride
L1	Conv(7×7×64)	1×1	L9	Conv(3×3×256)	1×1
L2	MaxPool(2×2)	2×2	L10	MaxPool(2×2)	2×2
L3	Conv(5×5×96)	1×1	L11	Conv(3×3×256)	1×1
L4	MaxPool(2×2)	2×2	L12	MaxPool(2×2)	2×2
L5	Conv(3×3×128)	1×1	FC1	2560	
L6	MaxPool(2×2)	2×2	FC2	512	
L7	Conv(3×3×256)	1×1	Output	Softmax Layer	
L8	MaxPool(2×2)	2×2			

3) *Parameter Setup*

The parameters setting of Main-CNN and Game-CNN are same, and they are given below: GradientDescentOptimizer is applied, batch size is 40, and its learning rate is controlled by the RL-CNN, *step4game*=5. The parameters of DDQN are: batch size is 50, training epoch is 50, AdamOptimizer is applied and its learning rate is 1e-4.

B. Case Study 1: CWRU Motor Bearing Dataset

In this case study, the proposed RL-CNN is tested on the motor bearing dataset from Case Western Reserve University (CWRU) [25]. In this experiment, the 6205-2RSJEM SKF bearing is install on the motor using electro-discharge machining. There are three kinds of fault patterns which are the roller fault (RF), outer race fault (OF) and inner race fault (IF). Each fault type has three different damage severity, and the damage sizes are 0.18mm, 0.36mm and 0.54mm. So there are totally nine fault patterns. The health conditions are ten, which include nine fault patterns and the normal condition. During the experiments, the fault signals are collected on four working conditions, and the motor loads are 0, 1, 2, 3 *hp* respectively. The vibration signal was collected for bearings for the analysis for fault classification, and the sampling frequency is 12 kHz. During the testing of RL-CNN-Ens, the five-fold CV is applied, and the initial learning rate is 0.01.

1) *The Learning Rate Scheduler Curves in RL-CNN-Ens*

Since there are four load conditions and the five-fold CV is applied in each load conditions, there are total twenty (4*5=20) experiments in this case study. The learning rate curves of the RL agent can be categorized into six types, and they are presented in Fig. 2.

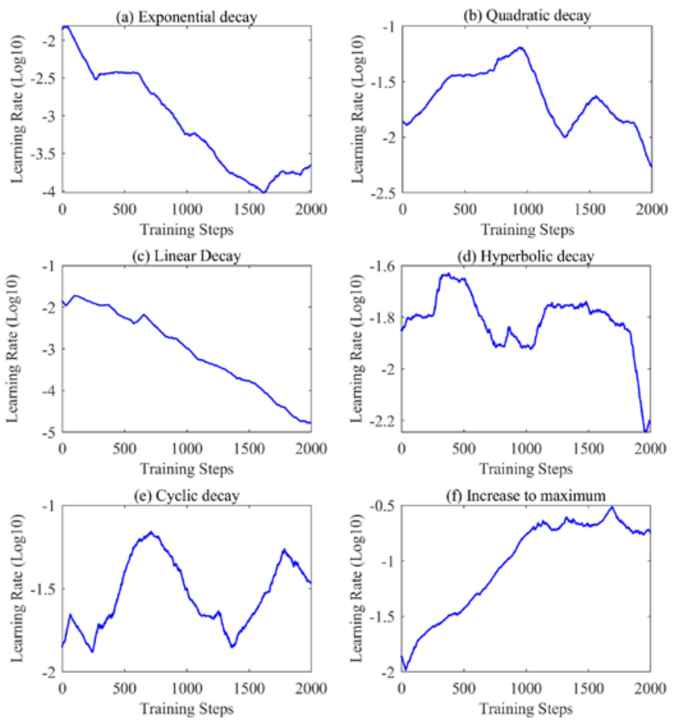

Fig. 2. The learning rate curves of RL-CNN-Ens in Case 1

These results show that the learning rate in all experiments are scheduled within the range of 1e-5 to 0.5 by the proposed RL agent, even though there is no boundary constraint on the learning rate. These results provide a solid support that the learning rate can be successfully scheduled by the RL agent in all experiments, indicating that the proposed RL-CNN-Ens can find the reasonable learning rate values and to adjust the learning rate effectively.

As shown in Fig. 2, the obtained RL agent based learning rate schedulers can be categorized into six types. They are Exponential decay, Quadratic decay, Linear decay, Hyperbolic decay, Cyclic decay and Increase to maximum. It should be noted that Exponential decay, Linear decay and Cyclic decay are also the most popular human-designed learning rate scheduler, showing that RL-CNN-Ens can also find the state-of-the-art learning rate schedulers automatically in this case study.

2) *Results on Training Time and Convergence*

The training time issue is discussed for RL-CNN-Ens. As shown in TABLE II, the time consumption of DDQN, Game-CNN, Main-CNN and the total time consumption are 4773.5, 930.1, 807.4 and 6641.4 seconds. The baseline CNN (the same CNN structure using the Exponential decay learning rate scheduler) is near 860.1 seconds. This result shows that the time consumption of RL-CNN-Ens is almost the 7.7 times than baseline CNN model. However, the baseline CNN needs to tune the learning rate value and the learning rate scheduler, the actual time consumption of baseline CNN should increase by many times in practice applications, and the tuning process always cause more than 10 times [29]. While RL-CNN-Ens can control its learning rate automatically without the tuning process, so it is also efficient in time.

TABLE II
THE TRAINING TIME OF RL-CNN-ENS AND BASELINE CNN IN CASE 1

RL-CNN-Ens			Baseline CNN	
	Mean (s)	**Percentage(%)**		
DDQN	**4773.5**	**71.8**	**Time**	**860.1 (s)**
Game-CNN	**930.1**	**14.0**	**AVG-1**	**96.49%**
Main-CNN	**807.4**	**12.2**		
Others	**130.3**	**2.0**	**AVG-2**	**97.65%**
Total	**6641.4**	**100**		

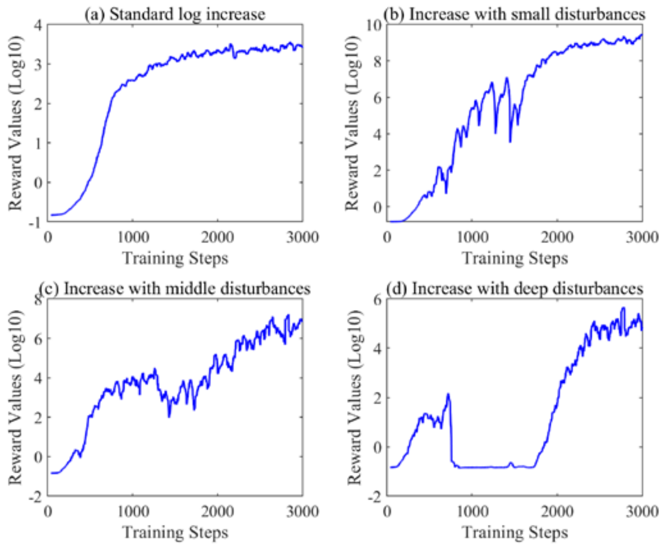

Fig. 3. The Reward curves of RL-CNN-Ens in Case 1

The prediction results of the baseline CNN are also given in TABLE II. AVG-1 and AVG-2 of baseline CNN are 96.49% and 97.65%. These results of RL-CNN-Ens will be given in

TABLE III
THE COMPARISON RESULTS ABOUT AVG ON CONFIGURE 2 IN CASE 1 (%)

No.	0-1	0-2	0-3	1-0	1-2	1-3	2-0	2-1	2-3	3-0	3-1	3-2	AVG-1
RL-CNN-Ens	**99.35**	**99.65**	**94.35**	**98.25**	**99.85**	**98.70**	**96.90**	**98.05**	**99.10**	**90.95**	**93.95**	**98.55**	**97.30**
DCN-SDR	96.60	94.20	97.40	96.20	99.90	99.20	95.80	98.70	99.10	91.60	93.00	98.40	96.7
TICNN	92.00	91.70	94.00	95.80	98.70	98.40	90.00	96.10	98.80	87.50	89.70	95.10	94.0
WDCNN	95.80	92.80	96.70	85.00	95.60	91.50	77.20	90.80	96.50	71.40	86.20	92.20	89.3
NSAE-LCN	75.20	83.00	81.00	76.50	81.30	70.70	86.30	92.80	91.80	80.40	80.20	87.30	82.2
SVM-FFT	62.40	66.20	63.20	78.40	81.40	82.40	71.60	74.80	77.00	84.20	76.40	70.40	74.0
ANN-FFT	85.60	79.40	79.40	99.60	99.80	99.80	91.80	97.60	99.60	84.80	87.20	81.00	90.5

TABLE III and TABLE IV. The results show that RL-CNN-Ens has achieved better results on both AVG-1 and AVG-2 than baseline CNN model.

The convergence curves of the reward achieved by RL-CNN-Ens are shown in Fig. 3. Nine out of twenty experiments belong to the Fig. 3(a), and seven experiments belong to the Fig. 3(b). These two types of reward have a good convergence property, and the reward values are increased quickly from low reward values to high. Three experiments are similar to Fig. 3(c), and only one experiments is similar to Fig. 3(d). These two types of reward curves suffer from some disturbances in a certain degree during the training process. But RL-CNN-Ens can overcome the disturbances and finally converge to the stable high reward values. The learning rate and reward curves validate the convergence of RL-CNN-Ens.

3) *Compared with Other DL and ML Methods*

This subsection presents the comparison results of RL-CNN-Ens with other famous DL and ML methods. There are two comparison configurations under crossing working conditions in literature. The first is from Chen et al. [25] and the comparison results are shown in TABLE III. The second is from Zhu et al. [17] and the comparison results are shown in TABLE IV. These two configurations are almost identical but used by different researchers, so both of them are compared with RL-CNN-Ens. In the comparison, the notation of '0-1' means that RL-CNN-Ens is trained on load 0 and tested on load 1. The average prediction accuracy (AVG) is used for the comparison.

TABLE IV
THE COMPARISON RESULTS ABOUT AVG ON CONFIGURE 1 IN CASE 1 (%)

No.	1-2	1-3	2-1	2-3	3-1	3-2	AVG-2
RL-CNN -Ens	**99.85**	**98.70**	**98.05**	**99.10**	**93.95**	**98.55**	**98.03**
ICN	98.23	97.17	99.80	94.71	94.93	98.10	97.15
HCNN	99.93	98.79	95.15	99.45	89.33	93.69	96.1
WDCNN (AdaBN)	99.4	93.4	97.5	97.2	88.3	99.9	95.9
Ensemble TICNN	99.5	91.1	97.6	99.4	90.2	98.7	96.1
ResNet	99.70	94.40	94.87	94.33	88.70	98.47	94.58
AlexNet	98.93	92.27	95.07	94.40	88.40	96.87	94.32
SVM	68.6	60.0	73.2	67.6	68.4	62.0	66.6
ANN	82.1	85.6	71.5	82.4	81.8	79.0	80.4

In TABLE III, the proposed RL-CNN-Ens has achieved the start-of-the-art results on configuration 1. The AVG on configuration 1 (AVG-1) of RL-CNN-Ens is 97.30%, which outperforms DCN-SDR[25], TICNN[25], WDCNN[25] and NSAE-LCN[25] largely. Two ML methods, including ANN and SVM, are also selected for the comparison. The results show that RL-CNN-Ens has achieved a significant improvement than ANN and Support Vector Machine (SVM) in this case study.

In TABLE IV, the results show that RL-CNN-Ens achieved the best results among these methods. The AVG on configuration 2 (AVG-2) of RL-CNN-Ens is 98.03%, which is better than ICN[17], HCNN[16], WDCNN (AdaBN) [30], Ensemble TICNN[31]. In this comparison, the famous DL and ML models, including ResNet [17], AlexNet [17], SVM [31] and ANN [31] are used as the baseline methods, while the results validate that RL-CNN-Ens has achieved a promoting improvement in this configuration.

C.Case 2: Self-priming Centrifugal Pump Dataset

The second case study is the self-priming centrifugal pump (SPCP) [26] dataset. The data acquisition system and the acceleration sensor are installed above the motor housing. There are five healthy conditions and they are the bearing roller wearing (BR), inner race wearing (IR), outer race wearing (OR), impeller wearing (IW) fault and the normal condition. The vibrational signals are collected for the fault classification. During the experiment, the rotation speed of motor is 2,900 per minute, and the vibration sampling frequency is 10,239Hz. The proposed RL-CNN-Ens is conducted on this dataset to validate its performance. The configuration of RL-CNN-Ens is the same as the case 1.

1) *The Learning Rate Scheduler Curves in RL-CNN-Ens*

The learning rate curves of RL-CNN-Ens are shown in Fig. 4. The values of learning rate are varied from 1e-4 to 1e-1 in this case study. These results indicate that the proposed RL agent can schedule the learning rate in the proper range even its range has no boundary constraint. It shows that RL-CNN can find the reasonable learning rate values in this case study.

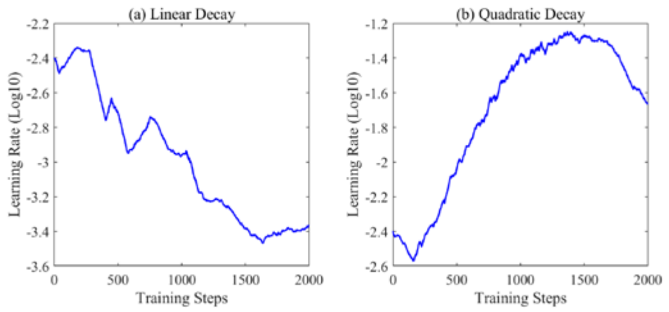

Fig. 4. The learning rate curves of RL-CNN-Ens in Case 2

The RL agent based learning rate schedulers can be categorized into two types in this case study, i.e. Linear decay as shown in Fig. 4(a) and Quadratic decay in Fig. 4(b). These two types of learning rate scheduler are almost the same with Fig. 2(c) and Fig. 2(b) in case 1. These results validate that RL-CNN-Ens can also successfully schedule the learning rate in case study 2.

2) *Results on Training Time and Convergence*

The training time of RL-CNN-Ens is 6880.3(s), and that of baseline CNN is 793.9(s), which means that the time consumption of RL-CNN-Ens is 8.5 times than baseline CNN. The AVG of baseline CNN is 99.98%, which is a high AVG value, and it is slightly inferior than RL-CNN-Ens.

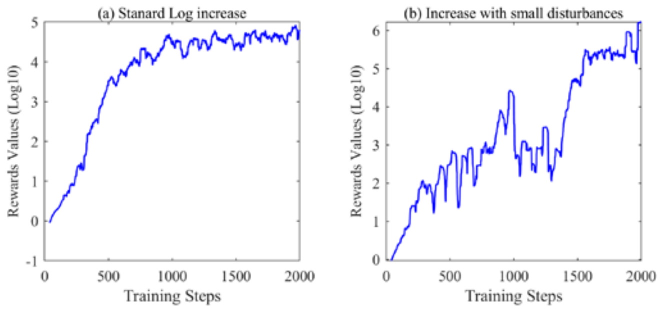

Fig. 5. The Reward curves of RL-CNN-Ens in Case 2

All the rewards of RL-CNN-Ens have the good convergence curve and the most typical two curve are presented in Fig. 5. These reward curves are similar with Fig. 5 (a) and Fig. 5 (b) in Case 1. From these results, it shows that RL-CNN-Ens has shown a good convergence ability on this case study.

3) Compared with other DL and ML methods

The results of RL-CNN-Ens are compared with other DL and ML methods. The comparison methods are Speeded Up Robust Features based Probabilistic Neural Networks (SUPF-PNN) [26], CNN model [32] and Subset based Deep Auto-Encoder (SBTDA) [33]. The comparison results are shown in TABLE V, and the AVG is used for the comparison.

TABLE V
COMPARISON RESULTS ABOUT AVG IN CASE 2 (%)

Methods	AVG
RL-CNN-Ens	**100**
SBTDA	99.60
CNN model	99.48
SUPF-PNN	98.33

The AVG of RL-CNN-Ens is as high as 100%. The RL-CNN-Ens outperforms SBTDA, CNN model and SUPF-PNN in this case study.

D.Case Study 3: MFPT Bearing Dataset

The third case study is the famous bearing dataset from Machinery Failure Prevention Technology (MFPT) Society [27]. In this experiment, the test rig was equipped with a NICE bearing. The acceleration signals of vibration data are collected for the analysis of the health conditions. There are three kind of health conditions in the experiment. The normal baseline conditions (NO) are collected at 270 lbs of load and the sampling frequency is 97,656Hz. The outer-raceway fault (OU) includes ten experiments. Three experiments are conducted at 270 lbs of load and the sampling frequency is 97,656Hz, and the rest seven assessed at varying loads: 25, 50, 100, 150, 200, 250, and 300lbs, and the sampling frequency is 48,828Hz. The inner race faults (IR) were analyzed with varying loads of 0, 50, 100, 150, 200, 250, and 300 lbs and the sampling frequency is 48,828Hz. The proposed RL-CNN-Ens is conducted to validate its performance. During this case study, the configuration of RL-CNN-Ens is the same with case 1 and case 2.

1) *The Learning Rate Scheduler Curves in RL-CNN-Ens*

In this case study, the learning rate controlled by RL-CNN-Ens is varied from 1e-4 to 1e-1. This phenomenon also occurs in case 1 and case 2, and it shows that RL-CNN-Ens can also find the reasonable learning rate values and adjust the learning rate successfully in this case study, as shown in Fig. 6.

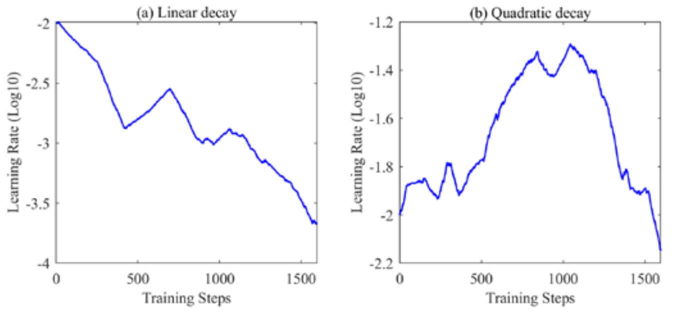

Fig. 6. The learning rate curve of RL-CNN-Ens in Case 3

The learning rate scheduler curves of RL-CNN-Ens can be categorized into Linear decay in Fig. 6(a) and Quadratic decay in Fig. 6(b). These two learning rate schedulers are almost the same with Fig. 2(c) and Fig. 2(b) in case 1, and also same with Fig. 4(a) and Fig. 4(b) in case 2.

2) *Results on Training Time and Convergence*

The training times of RL-CNN-Ens and baseline CNN are 4447.5 and 568.8 seconds, meaning that the time consumption of RL-CNN-Ens is 7.8 times than baseline CNN. The AVG of baseline CNN is 99.77%, which is worse than RL-CNN-Ens.

The reward curves of RL-CNN-Ens are as shown in Fig. 7. It is the 'Standard Log increase' type and the 'Increase with deep disturbance' type, which are similar with Fig. 3(a) and Fig. 3(c) in case 1, and Fig. 5(a) and Fig. 5(b) in case 2. The reward curves indicate that RL-CNN-Ens has achieved a good convergence in this case study.

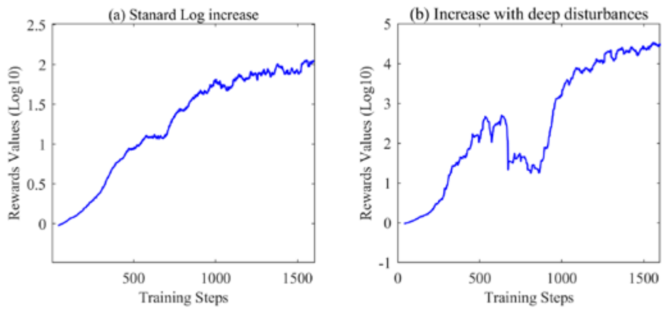

Fig. 7. The Reward curve of RL-CNN-Ens in Case 3

3) Compared with other DL and ML methods

The results of RL-CNN-Ens are also compared with other famous DL and ML methods in the literature. They are CNN, MLP and SVM with Scalogram 32*32, Spectrogram 32*32 and HHT 32*32 from Verstraete et al [27] and CNN-VAE from San Martin [34]. The comparison results are shown on TABLE VI.

The results show that RL-CNN-Ens has achieved the average prediction accuracy (AVG) of 99.95%, which is better than CNN with Scalogram 32*32, CNN with Spectrogram 32*32, CNN with HHT 32*32 and CNN-VAE methods. The prediction of MLP and SVM with Scalogram 32*32, Spectrogram 32*32 and HHT 32*32 are all also given in

TABLE VI, and the results show that RL-CNN-Ens outperforms MLP and SVM based methods.

TABLE VI
COMPARISON RESULTS ABOUT AVG IN CASE 3 (%)

Methods	AVG
RL-CNN-Ens	**99.95**
CNN with Scalogram 32*32	99.7
MLP with Scalogram 32*32	94.0
SVM with Scalogram 32*32	92.7
CNN with Spectrogram 32*32	81.4
MLP with Spectrogram 32*32	70.3
SVM with Spectrogram 32*32	73.0
CNN with HHT 32*32	75.7
MLP with HHT 32*32	49.2
SVM with HHT 32*32	58.5
CNN-VAE	92.58

E. Discussion

In this research, a new RL agent based learning rate scheduler is developed and it can automatically adjust the learning rate for CNN based fault classification without any boundary constraint. The proposed RL-CNN-Ens has improved in the following ways:

Firstly, RL-CNN-Ens has no constraint on the learning rate range, so RL-CNN-Ens can find the proper learning rate values according to its training history. It is prone to obtain the good results, as the results show that RL-CNN-Ens can schedule the learning rate at the reasonable range. The results show that RL-CNN-Ens has achieves good results on all three case studies.

Secondly, RL-CNN-Ens can adjust the learning rate automatically, so it can construct the complex learning rate schedulers to improve its performance, which can make it easier to be used than baseline CNN. As shown in Fig. 2, RL-CNN-Ens has found six types of learning rate schedulers. Three state-of-the-art learning rate schedulers, including Exponential decay, Linear decay and Cyclic decay, are also founded by RL-CNN-Ens. This shows that RL-CNN-Ens can generate the proper learning rate scheduler during the training process.

However, the training time of RL-CNN-Ens is 7.7~8.5 times than baseline CNN. This is because a new DDQN is trained for each experiment and the time consumption of DDQN is as high as 71.8% in the whole training time consumption in case 1. Since DDQN is the core in RL-CNN-Ens, it should be well-trained. Many works also take a long time to train the RL algorithm, such as DQN, such as 150K and 400K episodes in Hansen's work [11]. In future, the well-trained DDQN could be shared and reused across experiments to reduce the training time of RL-CNN-Ens.

VI. CONCLUSION AND FUTURE RESEARCH

This research presents a new RL based learning rate scheduler for CNN based fault classification with bagging ensemble (RL-CNN-Ens). The main contributions of this research are: 1) a new RL agent is designed as the learning rate scheduler for CNN and it can automatically adjust the learning rate effectively. 2) a new structure of RL-CNN-Ens is developed for fault classification. It adopts double CNN and double DDQN to explore and exploit for the RL agent and it can avoid the over-estimation of the trend of the learning rate. The proposed RL-CNN-Ens is tested on CWRU, SPCP and MFPT bearing dataset. The results show that RL-CNN-Ens can find the reasonable learning rate ranges and schedule it learning rate effectively. The results of RL-CNN-Ens outperform other DL and ML methods. Three human-designed learning rate schedulers, including Exponential decay, Linear decay and Cyclic decay, are also found by RL-CNN-Ens automatically. These results validate the potential of RL-CNN-Ens in fault classification. With the automatic learning rate scheduler of RL-CNN-Ens, it can both release the tuning process and prone to obtain the good results.

The limitations of the proposed RL-CNN-Ens include the following aspects. Firstly, RL-CNN-Ens needs almost near 8 times than baseline CNN, and the training of DDQN takes nearly 71.8% in total time consumptions. Secondly, RL-CNN-Ens can learn the learning rate scheduler automatically, but the decay factors α_1 and α_2 still should be selected manually. Thirdly, the proposed method is only tested on CNN structure. Based on these limitations, the future research can be conducted in the following ways. Firstly, the general DDQN can be investigated across dataset, which can reduce the training time of RL-CNN-Ens sharply. Secondly, the automatically selection of the decay factor can be investigated to eliminate expert's experience further. Thirdly, the proposed methods can be extended to other deep learning structures such as auto-encoder, deep belief network.

REFERENCES

[1] L. Guo, Y. Lei, S. Xing, T. Yan and N. Li, "Deep Convolutional Transfer Learning Network: A New Method for Intelligent Fault Diagnosis of Machines with Unlabeled Data," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 9, pp. 7316-7325, 2019.

[2] S. Yin, J. J. Rodriguez-Andina and Y. Jiang, "Real-Time Monitoring and Control of Industrial Cyberphysical Systems: With Integrated Plant-Wide Monitoring and Control Framework," *IEEE Industrial Electronics Magazine*, vol. 13, no. 4, pp. 38-47, Dec. 2019.

[3] C. Sun, M. Ma, Z. Zhao, S. Tian, R. Yan and X. Chen, "Deep Transfer Learning Based on Sparse Autoencoder for Remaining Useful Life Prediction of Tool in Manufacturing." *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2416-2425, 2018.

[4] Y. Jiang, S. Yin and O. Kaynak, "Data-Driven Monitoring and Safety Control of Industrial Cyber-Physical Systems: Basics and Beyond," *IEEE Access*, vol. 6, pp. 47374-47384, 2018.

[5] H. N Liu, C. L. Liu and Y. X. Huang, "Adaptive Feature Extraction Using Sparse Coding for Machinery Fault Diagnosis." *Mechanical Systems and Signal Processing*, vol. 25, no. 2, pp. 558-574, 2011.

[6] G. Jiang, H. He, J. Yan and P. Xie, "Multiscale Convolutional Neural Networks for Fault Diagnosis of Wind Turbine Gearbox," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 4, pp. 3196-3207, 2019.

[7] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang and R. X. Gao, "Deep Learning and Its Applications to Machine Health Monitoring," *Mechanical Systems and Signal Processing*, vol. 115, pp. 213-237, 2019.

[8] P. Probst, A. L. Boulesteix and B. Bischl, "Tunability: Importance of Hyperparameters of Machine Learning Algorithms," *Journal of Machine Learning Research*, vol. 20, no. 53, 1-32, 2019.

[9] L. N. Smith, "Cyclical Learning Rates for Training Neural Networks," *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Santa Rosa, CA, pp. 464-472, 2017.

[10] X. J. Guo, L. Chen and C. Q. Shen, "Hierarchical Adaptive Deep Convolution Neural Network and Its Application to Bearing Fault Diagnosis," *Measurement*, vol. 93, pp. 490-502, 2016.

[11] S. Hansen, "Using Deep Q-Learning to Control Optimization Hyperparameters," *arXiv preprint arXiv:1602.04062*, 2016.
[12] Y. Bengio. Neural Networks: Tricks of the Trade, chapter Practical recommendations for gradient-based training of deep architectures, pages 437–478. Springer Berlin Heidelberg, 2012. 1, 2, 4.
[13] S. Khan and T. Yairi, "A Review on the Application of Deep Learning in System Health Management," *Mechanical Systems and Signal Processing*, vol. 107, pp. 241-265, 2018.
[14] M. Zhao, M. Kang, B. Tang and M. Pecht, "Multiple Wavelet Coefficients Fusion in Deep Residual Networks for Fault Diagnosis," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 6, pp. 4696-4706, 2019.
[15] Z. Chen, K. Gryllias and W. Li, "Intelligent Fault Diagnosis for Rotary Machinery Using Transferable Convolutional Neural Network," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 1, pp. 339-349, 2020.
[16] L. Wen, X. Y. Li and L. Gao L, "A New Two-Level Hierarchical Diagnosis Network based on Convolutional Neural Network," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 2, pp. 330-338, 2020.
[17] Z. Y. Zhu, G. L. Peng, Y. H. Chen and H. J. Gao, "A Convolutional Neural Network based on A Capsule Network with Strong Generalization for Bearing Fault Diagnosis," *Neurocomputing*, vol. 323, pp. 62-75, 2019.
[18] F. Hutter, L. Kotthoff, and J. Vanschoren, "Automated Machine Learning-Methods, Systems, Challenges". *Automated Machine Learning*, 2019.
[19] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams and N. de Freitas, "Taking the Human Out of the Loop: A Review of Bayesian Optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148-175, 2016.
[20] J. Snoek, O. Rippel, K. Swersky, R. Kiros, N. Satish, N. Sundaram and R. Adams, "Scalable Bayesian Optimization Using Deep Neural Networks," *International Conference on Machine Learning*, pp. 2171-2180, 2015.
[21] M. Xia, T. Li, L. Xu, L. Z. Liu and C. W. Silva, "Fault Diagnosis for Rotating Machinery Using Multiple Sensors and Convolutional Neural Networks," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 1, pp. 101-110, 2018.
[22] Y. Xie and T. Zhang, "Fault Diagnosis for Rotating Machinery based on Convolutional Neural Network and Empirical Mode Decomposition," *Shock and Vibration*, 2017.
[23] L. J. Lin, "Reinforcement learning for robots using neural networks," No. CMU-CS-93-103. Carnegie-Mellon Univ Pittsburgh PA School of Computer Science, 1993.
[24] T. W. Raube, F. Assis Boldt and F. M. Varejão, "Heterogeneous Feature Models and Feature Selection Applied to Bearing Fault Diagnosis," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 1, pp. 637-646, 2015.
[25] T. Chen, Z. Wang, X. Yang and K. Jiang, "A Deep Capsule Neural Network with Stochastic Delta Rule for Bearing Fault Diagnosis on Raw Vibration Signals," *Measurement*, vol. 148, pp. 106857, 2019.
[26] C. Lu, Y. Wang, M. Ragulskis and Y. Cheng, "Fault Diagnosis for Rotating Machinery: A Method Based on Image Processing," *PloS one*, vol. 11, no. 10, pp. e0164111, Oct, 2016.
[27] D. Verstraete, A. Ferrada, E. L. Droguett, V. Meruane and M. Modarres, "Deep Learning Enabled Fault Diagnosis Using Time-Frequency Image Analysis of Rolling Element Bearings," *Shock and Vibration*, 2017.
[28] R. G. Stockwell, L. Mansinha and R. P. Lowe, "Localization of The Complex Spectrum: The S Transform," *IEEE Transactions on Signal Processing*, vol. 44, no. 4, pp. 998-1001, 1996.
[29] P. Probst, A. L. Boulesteix and B. Bischl, "Tunability: Importance of Hyperparameters of Machine Learning Algorithms," *Journal of Machine Learning Research*, vol. 20, no. 53, pp. 1-32, 2019.
[30] W. Zhang, G. Peng, C. Li, Y. Chen and Z. Zhang, "A New Deep Learning Model for Fault Diagnosis with Good Anti-Noise and Domain Adaptation Ability on Raw Vibration Signals," *Sensors*, vol. 17, no. 2, pp. 425, 2017.
[31] W. Zhang, C. H. Li, G. L. Peng, Y. H. Chen and Z. J. Zhang, "A Deep Convolutional Neural Network with New Training Methods for Bearing Fault Diagnosis Under Noisy Environment and Different Working Load," *Mechanical Systems and Signal Processing*, vol. 100, pp. 439-453, 2018.
[32] L. Wen, X. Y. Li, L. Gao and Y. Y. Zhang, "A New Convolutional Neural Network based Data-Driven Fault Diagnosis Method," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 7, pp. 5990-5998, 2018.
[33] Y. Y. Zhang, X. Y. Li, L. Gao and P. G. Li, "A New Subset Based Deep Feature Learning Method for Intelligent Fault Diagnosis of Bearing," *Expert Systems with Applications*, vol. 110, pp. 125-142, 2018.
[34] G. San Martin, E. López Droguett, V. Meruane, M. das Chagas Moura, "Deep Variational Auto-Encoders: A Promising Tool for Dimensionality Reduction and Ball Bearing Elements Fault Diagnosis," *Structural Health Monitoring*, 2018, 1475921718788299.

Long Wen received his Ph.D. degree in industrial engineering from Huazhong University of Science and Technology, China, 2014.
He is a Professor in the School of Mechanical Engineering and Electronic Information, China University of Geosciences. He had published more than 20 refereed papers, and his research interests include deep learning, automatic machine learning, fault diagnosis and intelligent algorithm, etc.

Liang Gao (M'08-SM'20) received Ph.D. degree in mechatronic engineering from Huazhong University of Science and Technology, China, 2002.
He is a Professor of the Department of Industrial & Manufacturing System Engineering, State Key Laboratory of Digital Manufacturing Equipment & Technology, School of Mechanical Science & Engineering, HUST. He had published more than 170 refereed papers. His research interests include operations research and optimization, big data and machine learning etc.

Xinyu Li (M'19) received his Ph.D. degree in industrial engineering from Huazhong University of Science and Technology, China, 2009.
He is a Professor of the Department of Industrial & Manufacturing Systems Engineering, State Key Laboratory of Digital Manufacturing Equipment & Technology, School of Mechanical Science & Engineering, HUST. He had published more than 80 refereed papers. His research interests include intelligent algorithm, big data and machine learning etc.