Protecting privacy in microgrids using federated learning and deep reinforcement learning

Wenzhi Chen
Department of Engineering
Durham University
Durham, UK, DH1 3LE
wenzhi.chen@durham.ac.uk

Hongjian Sun*
Department of Engineering
Durham University
Durham, UK, DH1 3LE
hongjian.sun@durham.ac.uk

Jing Jiang
Department of Mathematics, Physics and Electrical Engineering
Northumbria University
Newcastle upon Tyne, UK, NE1 8ST
jing.jiang@northumbria.ac.uk

Minglei You
Department of Electrical and Electronic Engineering
University of Nottingham
Nottingham, UK, NG8 1BB
Minglei.You@nottingham.ac.uk

Piper, William J.S.
Department of Engineering
Durham University
Durham, UK, DH1 3LE
william.j.piper@durham.ac.uk

***Abstract*—This paper aims to improve the energy management efficiency of home microgrids while preserving privacy. The proposed microgrid model includes energy storage systems, PV panels, loads, and the connection to the main grid. A federated multi-objective deep reinforcement learning architecture with Pareto fronts is proposed for total carbon emission and electricity bills optimization. The privacy of data is protected by federated learning, by which the original data will not be uploaded to the server. Numerical results show that compared with the traditional single Deep-Q network, using the proposed method the accumulated carbon emission decreased by 3% and the electricity bills decreased by 21%.**

***Index Terms*—Microgrids, Privacy, Deep learning, Multi-objective**

I. Introduction

A. Background

1) Home microgrids: Due to the concern of fossil fuel depletion, integrating renewable and distributed energy sources in power grids is needed. The concept of microgrid is a promising integration solution due to its potentials of improving the grid operation efficiency, realizing low carbon emission, enabling high renewable energy penetration, and protecting the privacy of consumers (or prosumers) [1]. The home microgrid is a kind of small-scale microgrid for families, in which the privacy issues become more prominent. How to improve the operation efficiency of home microgrids, realizing low carbon, low cost and high renewable energy penetration while protecting the privacy of residents is a challenge.

2) Deep Q-network and federated learning: In a real environment, the carbon emissions and electricity price change over time. For example, carbon emissions are higher during peak demand hours, and lower in the middle of the night. So it is complicated to acquire the best electricity purchase opportunities. Machine learning methods like Deep Q-Network (DQN) [2] can capture the best energy purchase opportunities through learning from experience, helping reduce the total carbon emission and electricity costs of home microgrids. In DQN, the optimal operating parameters in the next time step could be estimated, by using a prediction algorithm. In home microgrid system, most data (e.g. photovoltaic, carbon emission, electricity price data) are time-series data. When predicting them, Long-Short Term Memory (LSTM) algorithm often performs better than other machine learning algorithms [3]. Traditionally, DQN and LSTM are centralized due to the easier deployment structure and the limitation of the computing resource. However, there are some limitations in this centralized learning framework such as data island, privacy, and the challenge of storage as well as data congestion. These challenges can be addressed by federated learning [4], which is a distributed machine learning technology that allows participants to build distributed models without sharing data.

B. Literature review

1) Deep Q-network: In a traditional reinforcement learning method such as Q-learning, according to the explore experience, Q values were set for state-action pairs, forming a Q table. In a DQN [2], a Deep Neural Network (DNN) was used to fit the Q table. This is because in traditional Q-learning, when the dimension increases, the Q table occupies a large amount of storage space and making it difficult to train.

2) Federated learning: McMahan *et al*. proposed FedAvg, which is the first paper of federated learning [4]. In FedAvg, instead of uploading the original data, the neural network weights were uploaded from clients to the server after local training, the data of each client can be used by the server on the premise of protecting privacy.

3) Microgrids optimization with deep reinforcement learning: Deep reinforcement learning has been applied to different aspects of microgrids scheduling, mainly including:

*Prof. Hongjian Sun is the corresponding author.

- Uncertainties: In [5], Ji *et al*. used the proximal policy optimization algorithm to minimize the total costs considering the uncertainty of renewable energy generation, load demand, and electricity costs. In [6], Li *et al*. determined the spinning reserve while minimizing the total costs with reinforcement learning, which has a better performance than the traditional solver CPLEX.
- Stability: In [7], Li *et al*. proposed 'safe reinforcement learning', where 'safe' refers to the consideration of power flow constraints. In [8], Guo *et al*. proposed a real-time dynamic optimal energy management based on a deep reinforcement learning algorithm, maintaining the safety and stability of microgrids.
- Demand response: In [9], Nakabi *et al*. tested 7 different deep reinforcement learning optimization algorithms considering the demand response of loads.

C. Motivation and paper structure

There still exists some research questions to be answered, particularly: 1) In a real environment, the carbon emissions in power grids and electricity prices change over time, how to acquire the best electricity purchase opportunity to optimize overall carbon emissions and electricity bills together? 2) The information from different microgrids should be utilized while the privacy of the prosumers needs to be protected, how to concatenate different strategies from multiple microgrids without sharing personal data? 3) In microgrids, there may exist many objectives to be optimized. How each microgrid should choose the coefficients/weights of different objectives? To answer these questions, we have to achieve the following objectives:

- Development of a smart energy scheduling algorithm.
- Development of a privacy-preserving distributed algorithm for collaborations between smart homes.
- Development of a multi-objective optimization algorithm.

In this paper, a multi-objective method for the distributed deep-Q network is proposed, including a DQN-based method to reduce the carbon emission and electricity bills of the home microgrids, a federated learning algorithm to make the DQN algorithm more efficient while considering privacy, and a Pareto front for the multi-objective DQN.

The remainder of the paper is as follows: Section II explains the system composition, with the development of algorithms presented in Section III. Section IV explains the simulation setup, with the conclusions and future work in Section V.

II. System model

This section introduces the system composition, including photovoltaic (PV), battery, load and scheduling models.

A. System composition

The proposed system consists of 4 home microgrids. Each home microgrid consists of 2 sub-systems, i.e., the supply sub-system and the load sub-system. The supply sub-system consists of a PV panel, batteries, and the power grid. The load sub-system contains home loads.

B. Photovoltaic model

PV energy generation can be modeled by [9]:

$$E_{\text{pv}}(n) = A \times SR(n) \times \eta_{\text{pv}} \times t_{\text{n}} \tag{1}$$

where n refers to the $n\text{-}th$ time interval, t_{n} is the length of the time interval. A is the effective contact area (m^2), η_{pv} is the solar-electric energy conversion efficiency. $SR(n)$ is the averaged solar irradiance (W/m^2). $E_{\text{pv}}(n)$ is the energy generated from PV (Wh). The constraints are as follows:

$$A>0, \eta>0, t_{\text{n}}>0, SR(n) \geq 0 \tag{2}$$

C. Rules for energy supply and batteries

The battery model can be given by [9].

$$E_{\text{B}}(n) = E_{\text{B}}(n-1) \times (1-\eta_{\text{s}}) + (E_{\text{ch}}(n) - \frac{E_{\text{dis}}(n)}{\eta_{\text{c}}})\eta_{\text{b}} \tag{3}$$

$$SoC(n) = E_{\text{B}}(n)/(C_{\text{B}} \times V_{\text{Ra}}) \tag{4}$$

where $E_{\text{B}}(n)$ is the battery energy (Wh), η_{s} is the charging efficiency, $E_{\text{ch}}(n)$ is the charging energy (Wh), η_{c} is the inverter efficiency. η_{b} is the battery efficiency. $E_{\text{dis}}(n)$ is the discharging energy for the load (Wh). C_{B} is the maximum battery capacity (Wh) and V_{Ra} is the battery voltage (V). The constraints are as follows:

$$E_{\text{ch}}(n) = E_{\text{pv}}(n) + E_{\text{bought}}(n) \tag{5}$$

$$SoC(n) \geq 0, E_{\text{dis}}(n) \geq 0, E_{\text{ch}}(n) \geq 0 \tag{6}$$

$$SoC(n) \leq SoC_{\text{lim}}, E_{\text{dis}}(n) \leq E_{\text{maxd}}, E_{\text{ch}}(n) \leq E_{\text{max}} \tag{7}$$

where $E_{\text{bought}}(n)$ is the amount of energy bought from the main power grid to charge the battery. SoC_{lim}, E_{max} and E_{maxd} are the limitations of the battery, maximum energy charging and discharging in a time interval, respectively.

D. Load model

A widely used normal distribution is adopted for describing load fluctuations [10]. Its probability density function is:

$$f_{\text{l}}(E_{\text{L}}(n)) = \frac{1}{\sqrt{2\pi}\sigma_{\text{L}}} e^{\frac{1}{2}(\frac{(E_{\text{L}}(n)-\mu_{\text{L}}}{\sigma_{\text{L}}})^2} \tag{8}$$

where $E_{\text{L}}(n)$ is the load active power, μ_{L} and σ_{L} are the mean and standard deviation of the active power.

E. Scheduling model

The objective of scheduling function is to minimize the total carbon emission and electricity costs with biases $\lambda(i)$, that is to minimize $R(i)$ as below:

$$\min R(i) = \sum_{t=1}^{n} r(i,n) \tag{9}$$

where

$$r(i,n) = \lambda(i)R_{\text{cost}}(i,n) + (1-\lambda(i))R_{\text{carbon}}(i,n) \tag{10}$$

$$R_{\text{cost}}(i,n) = Reward(F_{\text{cost}}(i,n)) \tag{11}$$

$$R_{\text{carbon}}(i,n) = Reward(F_{\text{carbon}}(i,n)) \tag{12}$$

$$\lambda(i) \in [0,1], \forall i \in \mathbb{N} \tag{13}$$

where i represents the i th client, $\lambda(i)$ is the bias towards carbon emission and electricity costs, belonging to $[0,1]$. $F_{\text{cost}}(i,n)$ and $F_{\text{carbon}}(i,n)$ are the electricity cost and carbon emission of the ith client. $Reward(\cdot)$ is a function to generate the rewards $R_{\text{cost}}(i,n)$ and $R_{\text{carbon}}(i,n)$ according to $F_{\text{cost}}(i,n)$ and $F_{\text{carbon}}(i,n)$ and historic data in the database, items with lower electricity cost and carbon emission can get higher rewards, and vise versa. The $F_{\text{cost}}(i,n)$ and $F_{\text{carbon}}(i,n)$ are shown as follows:

$$F_{\text{cost}}(i,n) = (E_{\text{MG}}(i,n))Pr(n) \tag{14}$$

$$F_{\text{carbon}}(i,n) = (E_{\text{MG}}(i,n))Ca(n) \tag{15}$$

where

$$E_{\text{MG}}(i,n) = E_{\text{bought}}(i,n) + e(i,n)E_{\text{L}}(i,n) \tag{16}$$

$$e(i,n) = \{1,0\}, \forall i \in \mathbb{N}, \forall n \in \mathbb{N} \tag{17}$$

$$Pr(n), Ca(n), E_{\text{bought}}(i,n), E_{\text{L}}(i,n) \geq 0, \forall n \in \mathbb{N}^{+} \tag{18}$$

Here $e = 1$ if the main power grid is used to power the loads and $e = 0$ if the batteries are used. $E_{\text{MG}}(i,n)$ is the total energy bought from main power grid during time interval n. $Pr(n)$ and $Ca(n)$ are the electricity price and carbon emission data during time interval n. The optimization in (9) turned to choose the best opportunity (when $Pr(n)$ and $Ca(n)$ are relatively lower) to purchase electricity ($E_{\text{MG}}(i,n)$), thus minimizing $F_{\text{cost}}(i,n)$ and $F_{\text{carbon}}(i,n)$. For this optimization, the constraints of power flow are as follows:

$$P_{\text{net}}(i,t) = P_{\text{charge}}(i,t) - P_{\text{dis}}(i,t) \tag{19}$$

$$P_{\text{net}}(i,t) + P_{\text{L}}(i,t) = P_{\text{pv}}(i,t) + P_{\text{MG}}(i,t) \tag{20}$$

that means the power supply meets the power demand. Where the symbol P means power, t refers to current time. $P_{\text{net}}(i,t)$ is the net power of batteries.

III. Proposed Algorithm

We use LSTM for time series data forecasting, which can be easily implemented with [3] [4]. DQN is used for scheduling, that is to acquire the best electricity purchase opportunity, and federated learning works with these two algorithms to form a distributed privacy-protection machine learning environment.

A. Markov decision process

We propose that reinforcement learning gets a policy, mathematically, the policy is a mapping as follows:

$$\pi(x) : State \rightarrow Action \tag{21}$$

where $\pi(x)$ is the policy, that is a mapping between state and action space. Given any state to $\pi(x)$, the optimal action can be obtained by the mapping of $\pi(x)$ in real-time. While traditional solvers like the generic algorithm or bayesian optimization only get a solution at a time, whenever the state is changed, re-optimizations are needed. In a distributed optimization problem like (9), the policy can be reused by transferring to different nodes with similar tasks, so reinforcement learning is chosen and (9) is transferred and described as a Markov decision process, the elements are described as follows:

- Environment: A home microgrid system with loads, PV panels and energy storage system (batteries).
- State: The state space $x(i,n)$ can be described as:

$$x(i,n) = [SoC(i,n), Ca(n), Pr(n), E_{\text{pv}}(i,n), E_{\text{MG}}(i,n)] \tag{22}$$

- Action: The action space can be described as:

$$a(i,n) = [e(i,n), E_{\text{bought}}(i,n)] \tag{23}$$

  where $E_{\text{bought}}(i,n)$ decides whether to buy electricity into the battery or not, and if so how much electricity to buy. Also, the $e(i,n)$ decides to use batteries or electricity from the grid to power the home microgrids.
- Reward: Calculate the ranking of carbon emission and electricity cost according to historic data. If it is in the high position of low carbon emission and low electricity cost, the more electricity purchased, the higher the reward, and vice versa.

The proposed Markov decision process can be solved by the federated learning-based distributed algorithm in Section III-C.

B. Pareto fronts

In a multi-objective optimization, the Pareto front is the set of all non-dominated solutions. Consider a system with function $f : X \rightarrow \mathcal{R}^M$, where X is a set of feasible decisions in the metric space $\mathcal{R}^M$. and Y is the feasible set of criterion vectors in $\mathcal{R}^M$, such that $Y = y = f(x), \forall x \in X$. If a point y'' strictly dominates another point y', written as $y'' \succ y'$. The Pareto frontier is thus written as:

$$P(Y) = \{y' \in Y : \{y'' \in Y : y'' \succ y', y'' \neq y'\} = \emptyset\} \tag{24}$$

C. Federated learning-based distributed algorithm

The pseudocode of the proposed algorithm is described in Algorithm 1, where lines 1-8 are the initialization, the proposed network will randomly select a client as the server. Line 10 is the initialization of the environment and states for the deep-Q network. Line 12 is the epsilon-greedy algorithm. Line 13-14 is a step move for the deep-Q network, the action is decided by the online neuro network. Line 15 is to store the experience for future training. Line 16 is to calculate the total carbon emission and electricity bills. Line 18-19 judge whether this online network is a non-dominated solution according to carbon emissions and electricity bills. If so, store the results. 21-23 is to update the online network with the target network regularly (every Tr_{gap} rounds). 24-28 train the target network with the stored memory regularly (every Tr_{renew} rounds), with line 26 using Bellman's equation to estimate the possible best Q-value $y(i,j)$ and line 27 perform gradient descent. The server performs line 33-40 to perform federated learning regularly (every Tr_{upload} rounds), including aggregation and generating overall Pareto fronts, and otherwise, the clients perform line 30-32 to upload the weights of deep neuro network and Pareto fronts to the server.

Algorithm 1: Proposed federated multi-objective deep Q-learning algorithm with Pareto fronts

1 Load data from datasets
2 Initialize the state: Client i_1 - i_3 or Server
3 Initialize home microgrid loads with (8)
4 Initialize replay memory $\mathcal{D}$ to capacity N
5 Initialize pareto memory $\mathcal{P}$ to capacity V
6 Initialize target network Q with random weights θ
7 Initialize online network Q^* with random weights θ^*
8 Initialize Tu_{gap}, Tu_{renew} and Tu_{upload}
9 **for** *episode* $= 1, M$ **do**
10   Initialise sequence $s(i,1) = \{x(i,1)\}$ with datasets and preprocessed sequenced $\phi(i,1) = \phi(s(i,1))$
11   **for** $n = 1, N$ **do**
12   With probability ϵ select a random action $a(i,n)$ otherwise try all actions a and select $a(i,n) = \max_a Q^*(\phi(s(n)), a; \theta^*(i,n))$
13   Execute action $a(i,n)$ and observe reward $r(i,n)$ from (9) then get $x(i,n+1)$
14   Set $s(i,n+1) = s(i,n), a(i,n), x(i,n+1)$ and preprocess $\phi(i,n+1) = \phi(s(i,n+1))$
15   Store $(\phi(i,n), a(i,n), r(i,n), \phi(i,n+1))$ in $\mathcal{D}$
16   Calculate accumulated carbon emission $CE_{\text{total}}(i)$ and electircity cost $EC_{\text{total}}(i)$ with $F_{\text{cost}}(i,n)$ and $F_{\text{carbon}}(i,n)$
17   **end for**
18   **if** *($CE_{total}(i)$ and $EC_{total}(i)$ non-dominated)* **then**
19   Store and update Pareto fronts with θ in $\mathcal{P}$
20   **end if**
21   **if** *(!episode%Tr_{gap})* **then**
22         $Q^* = Q$
23   **end if**
24   **if** *(!episode%Tr_{renew})* **then**
25   Sample random minibatch of transitions $(\phi(i,j), a(i,j), r(i,j), \phi(i,j+1))$ from $\mathcal{D}$
26   Set $y(i,j) = r(i,j) + \gamma \max_{a'} Q(\phi_{j+1}, a'; \theta)$
27   Perform a gradient descent step on $(y(i,j) - Q(\phi(i,j), a(i,j); \theta(i)))^2$ for Q.
28   **end if**
29   **if** *(!episode%Tr_{upload})* **then**
30   **if** *(State==Client)* **then**
31   Upload θ and $\mathcal{P}$ to Server
32   **end if**
33   **if** *(State==Server)* **then**
34   Collect weights $\theta(i,n+1)$ from Clients
35   Calculate the data volumn $v(i)$ of Client i
36   Calculate the total data volumn v
37             $\theta(n+1) \leftarrow \sum_{i=1}^{I} \frac{v(i)}{v} \theta(i,n+1)$
38   Generate Pareto fronts with $\mathcal{P}$
39   Return global model $\theta(n+1)$
40   **end if**
41   **end if**
42 **end for**
43 Output: The Q^* of Pareto fronts

IV. Simulations

A. Simulation environment

The proposed LSTM, DQN and federated learning were developed in MATLAB, using a PC with CPU Intel Core i7 6600u and 16GB memory capacity. The simulation parameters are shown in Table I.

TABLE I
SIMULATION PARAMETERS

Parameters	Values
Default time interval	half an hour
PV panel area	$1m^2$
Conversion efficiency	20%
Capacity of the battery	$4 * 50Ah$
Rated voltage	$12V$
Conversion or storage loss	0%
Maximum energy bought every half an hour	each battery $5Ah$
Conversion or storage loss	0%
DNN input Nodes	7
DNN output Nodes	1
DNN hidden Nodes	30×30

B. Datasets and predictions

The PV datasets (2005-2021) [11], [12] of Durham were used, with LSTM and federated learning, the prediction RMSE =11.70, R =0.943. The electricity price datasets (2015-2020) and carbon emission datasets (2017-2020, with predicted data) were used [13], [14]. The real electricity price data in the next half an hour was used instead of the predicted data. For load profile, the typical UK household electricity demand curve with 15% variation is used [15].

C. Scheduling tests

In this part, 200 half-an-hours (100 hours) were chosen to evaluate the performance of the trained DQN. There are three subgraphs in Fig. 1. As shown in the Y axis labels, the first and second subgraphs are the carbon emission coefficient and the electricity price every half an hour. The third subgraph is the decision made by the DQN, that is, when and how much electricity to buy. It is observed that the DQN only bought energy when the carbon emission or electricity price was low, such as in 0-5, around 20, 40-55, 90-100, around 140 and 190-200, these time slots are in accordance with the time slots having lower carbon emission and electricity costs. Therefore, the energy efficiency is improved by the DQN.

It can be found that the proposed DQN can seize the opportunity to buy electricity when carbon emissions or electricity prices are lower than other times.

D. Case studies

Three scenarios are considered in this part. The first one is a single DQN without sharing data, this scenario has the highest privacy. The second scenario is shown in Fig. 2 (Algorithm 1), the DQN with federated learning. Instead of sharing data, the weights and biases of the deep Q-Network are shared, and privacy is protected because there is no need to upload private data. As shown in Fig. 3, the third scenario is a distributed model with shared memory data in an authorized third-party database. This means the clients share their memory database with the server, the server train a global model, then updates

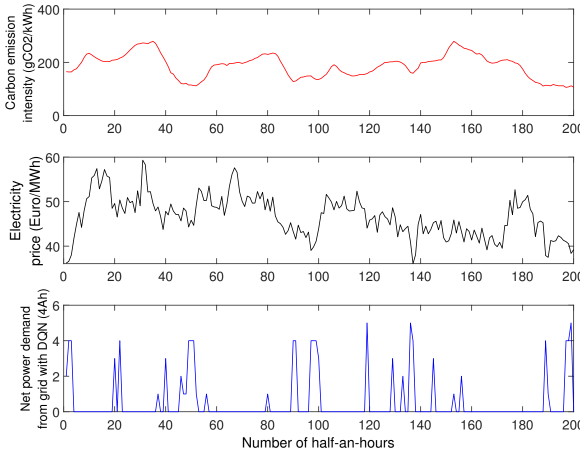

Fig. 1. DQN is used to acquire the best electricity purchase opportunities.

the global model for each client. The clients generate experience based on local data and provide that to the server for future training. The privacy of these scenarios is not as good as the first two, however, only the trusted server has private data, so privacy is protected if there is no data leakage from the trusted server.

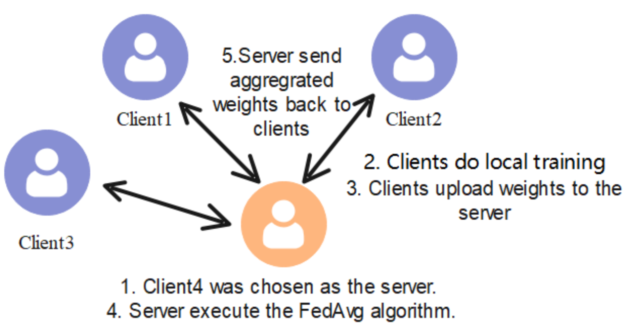

Fig. 2. Federated learning in the scenario 2.

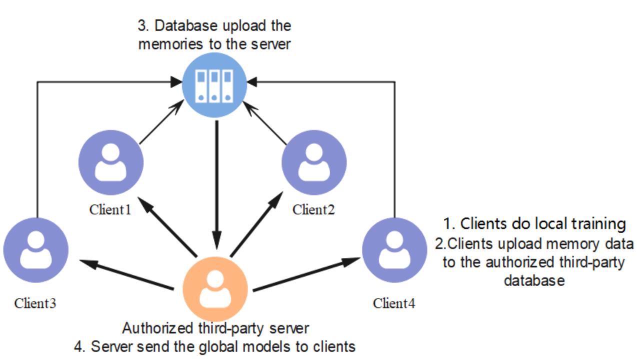

Fig. 3. Centralized learning in the scenario 3.

Scenario1: The Pareto fronts of a single DQN are shown in Fig. 4. Each node was the average carbon emission or electricity costs in 100 hours of 400 rounds of training iterations, the total training iteration is $400 \times 75 = 30000$. Iterations 0-800 are the observation period and shouldn't be compared. Compared with the 800-1200 iterations (average carbon emission was 7514 gCO_2, average electricity bill was 0.98 $Euro$), in the Pareto fronts solutions (with stars, the average carbon emission was 6910 gCO_2, average electricity bill was 0.86 $Euro$), the carbon emission decreased by 8%, the electricity bill decreased by 12.2%.

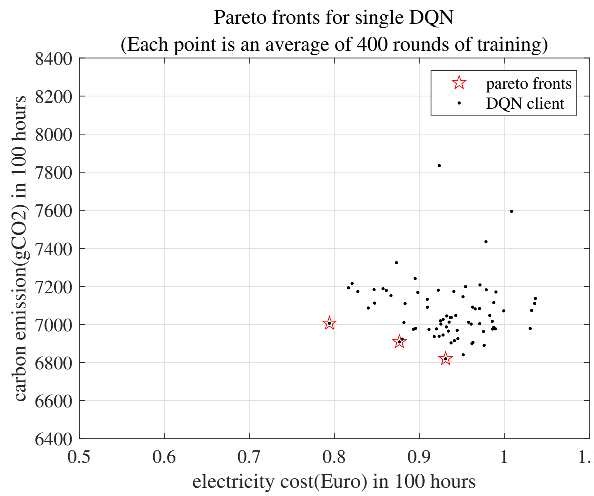

Fig. 4. Pareto fronts for single DQN

Scenario2: With the help of federated learning, the privacy of homes can be protected by uploading client weights. 10 Pareto fronts of a federated DQN are shown in Fig. 5, in which the leftmost front is invalid because the carbon emission is too high. Compared with a single DQN in Scenario 1 (the average carbon emission was 6910 gCO_2, average electricity bill was 0.86 $Euro$), the performance of Pareto fronts increased significantly (the average carbon emission was 6700 gCO_2, average electricity bill was 0.68 $Euro$). The carbon emission decreased by 3%, the electricity bill decreased by 21%.

For different Pareto fronts, the carbon emission and electricity bill of the right most valid Pareto front is 6960 gCO_2 and 0.6 $Euro$, for left most Pareto front, that is 6468 gCO_2 and 0.74 $Euro$. For different DQN models of Pareto fronts, The amplitude of carbon emission varies about 7%, and for the electricity bill, that is 18.9%. The client can choose their bias towards carbon emission and electricity bills by using different DQN models with different Pareto fronts, which were stored during the training steps according to Algorithm 1.

Scenario3: The Pareto fronts of a centralized DQN are shown in Fig. 6, like scenario 2, the left-most front is invalid. The performance of the electricity bill optimization is the best among the three scenarios (the average carbon emission was 6675 gCO_2, average electricity bill was 0.64 $Euro$). Compared with a single DQN in scenario 1, the carbon emission decreased by 3.4%, the electricity bill decreased by 25.6%. Compared with scenario 2, they have a similar optimization effect with better performance in electricity bills. The simulations show that the optimization degree of the electricity bill exceeded carbon emission, a potential reason is that the real price data was used instead of predicted data.

From scenarios 2 and 3, it can be found that federated learning can achieve similar performance to centralized learning, the Pareto fronts provide biases towards electricity costs or carbon emission optimization.

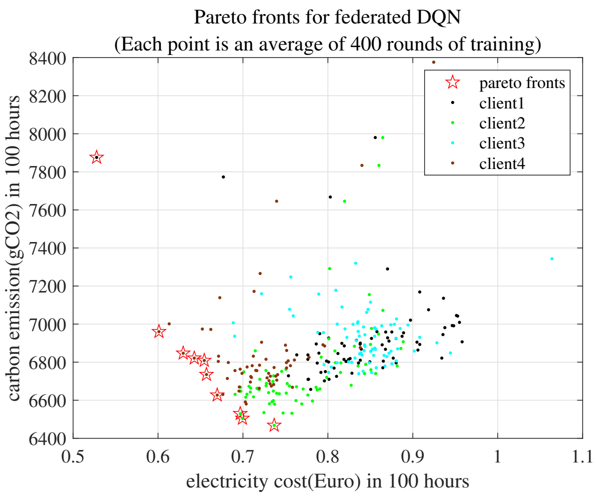

Fig. 5. Pareto fronts for federated DQN.

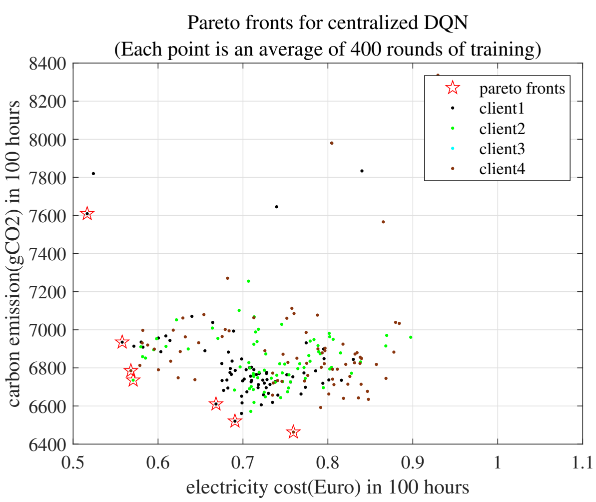

Fig. 6. Pareto fronts for centralized DQN.

V. Conclusion and Future Work

This paper introduces the multi-objective federated scheduling algorithm to optimize carbon emissions in the home microgrid system. Simulations show that with the single DQN, the carbon emission decreased by 8%, and the electricity bill decreased by 12.2%. The proposed federated method can capture lower carbon emissions or electricity prices than the single DQN, the carbon emission decreased further by 3%, and the electricity bill further decreased by 21%. The proposed algorithm achieved similar performance as the centralized Deep-Q network which however has privacy information leakage concerns. On the premise of protecting privacy, the clients can choose their bias toward carbon emission and electricity bills by using different DQN models in the Pareto fronts.

In the future, other control algorithms like Deep Deterministic Policy Gradient [16] and Asynchronous Advantage Actor-Critic [17] will be investigated in the future to compare performance with this work.

References

[1] D. E. Olivares, A. Mehrizi-Sani, A. H. Etemadi, C. A. Cañizares, R. Iravani, M. Kazerani, A. H. Hajimiragha, O. Gomis-Bellmunt, M. Saeedifard, and R. Palma-Behnke, "Trends in microgrid control," *IEEE Transactions on smart grid*, vol. 5, no. 4, pp. 1905–1919, 2014.

[2] M. Volodymyr, K. Koray, S. David, A. A. Rusu, V. Joel, M. G. Bellemare, G. Alex, R. Martin, A. K. Fidjeland, and O. Georg, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–33, 2019.

[3] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," *computer science*, vol. 1, no. 1, pp. 338–342, 01 2014.

[4] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," *Artifical intelligence and statistics*, vol. 1, no. 1, pp. 1273–1282, 2017.

[5] Y. Ji, J. Wang, J. Xu, and D. Li, "Data-driven online energy scheduling of a microgrid based on deep reinforcement learning," *Energies*, vol. 14, no. 8, p. 2120, 2021.

[6] Y. Li, R. Wang, and Z. Yang, "Optimal scheduling of isolated microgrids using automated reinforcement learning-based multi-period forecasting," *IEEE Transactions on Sustainable Energy*, vol. 13, no. 1, pp. 159–169, 2021.

[7] H. Li, Z. Wang, L. Li, and H. He, "Online microgrid energy management based on safe deep reinforcement learning," in *Proc. 2021 IEEE Symposium Series on Computational Intelligence (SSCI)*. Orlando, Florida, USA, 2021, pp. 1–8.

[8] C. Guo, X. Wang, Y. Zheng, and F. Zhang, "Real-time optimal energy management of microgrid with uncertainties based on deep reinforcement learning," *Energy*, vol. 238, no. 1, p. 121873, 2022.

[9] T. A. Nakabi and P. Toivanen, "Deep reinforcement learning for energy management in a microgrid with flexible demand," *Sustainable Energy, Grids and Networks*, vol. 25, no. 1, p. 100413, 2021.

[10] S. Xia, X. Luo, K. W. Chan, M. Zhou, and G. Li, "Probabilistic transient stability constrained optimal power flow for power systems with multiple correlated uncertain wind generations," *IEEE Transactions on Sustainable Energy*, vol. 7, no. 3, pp. 1133–1144, 2016.

[11] NASA, "PV data in Durham," [Online] https://power.larc.nasa.gov/data-access-viewer//, accessed November 16, 2021.

[12] European Commission, "PV data in Durham, 2005-2016," [Online] https://re.jrc.ec.europa.eu/pvg-tools/en/tools.html/, accessed November 16, 2021.

[13] Nord pool, "Electricity price data, 2015-2020," [Online] https://www.nordpoolgroup.com/, accessed November 16, 2021.

[14] National grid ESO, "Carbon emission data, 2017-2020," [Online] https://carbonintensity.org.uk/collapseData, accessed November 16, 2021.

[15] A. J. Pimm, T. T. Cockerill, and P. G. Taylor, "The potential for peak shaving on low voltage distribution networks using electricity storage," *Journal of Energy Storage*, vol. 16, no. 1, pp. 231–242, 2018.

[16] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *Proc. ICLR 2019 conference*. New Orleans, USA, 2019, pp. 1–14.

[17] V. Mnih, A. P. Badia, M. Mirza, A. Graves, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. PMLR International conference on machine learning*. New York City, NY, USA, 2016, pp. 1928–1937.