KICS The Korean Institute of Communications and Information Sciences

Available online at www.sciencedirect.com

ScienceDirect

ICT Express 9 (2023) 803–808

www.elsevier.com/locate/icte

Reward-based participant selection for improving federated reinforcement learning

Woonghee Lee

Department of Applied Artificial Intelligence, Hansung University, Seoul 02876, Republic of Korea

Received 5 May 2022; received in revised form 31 July 2022; accepted 26 August 2022
Available online 31 August 2022

Abstract

Federated reinforcement learning (FRL) has recently received a lot of attention in various fields. In FRL systems, the concept of performing more proper actions with better experiences exists, and we focused on this unique characteristic. Motivated by such inherent property of FRL, in this paper, we propose the reward-based participant selection scheme for improving FRL. The FRL system with the proposed scheme performs learning effectively by putting a priority on utilizing better experiences of agents performing outstanding actions. We conducted various experiments, and the results show that it is possible to accelerate learning and require fewer agents when using the proposed scheme, which means that the proposed scheme improves the performance and efficiency of learning.

Keywords: Federated learning; Reinforcement learning; Federated reinforcement learning; Participant selection

1. Introduction

Supported by the advance in artificial intelligence- (AI-) related technologies, the Internet of Things (IoT) has developed into the intelligent IoT. AIoT is a compound word of AI and IoT, which refers to an autonomous IoT system in which objects are connected to Internet to exchange data, as well as learn and develop by leveraging AI. Federated learning (FL) is a very suitable learning technique for AIoT. Unlike most conventional learning methods that collect distributed data in one place and perform learning, FL involves a number of IoT devices working together to build a global model in a decentralized manner. FL was first introduced by Google in 2016 [1], and many studies proposed various techniques for improving IoT systems by using FL. Mothukuri et al. proposed the FL-based approach to recognize intrusion in IoT networks using decentralized on-device data [2]. Yuan et al. devised an advanced FL framework for healthcare IoT [3].

Reinforcement learning (RL) is machine learning in which machines interact with environments and perform self-learning. Many studies [4,5] utilized deep RL (DRL) to solve various problems that were difficult to resolve using existing methods. Federated reinforcement learning (FRL) is a fusion of FL and RL, and FRL is a relatively recently proposed technique [6]. Different from typical FL where each device conducts supervised or unsupervised learning as local training, every device in FRL takes on the role of an agent and performs RL. In FRL, it is possible to perform faster and more unbiased learning by sharing RL results based on many devices' diverse experiences without exchanging raw data. Thus, FRL can be an effective technique to build the intelligence in AIoT. For instance, autonomous vehicles are able to use FRL to obtain cooperative perception which extends their sensing range beyond line-of-sight [7]. Also, FRL can be utilized for IoT devices to learn optimal control policy collaboratively [8]. Another example is to extract the knowledge from electronic medical records across edge devices using FRL [9].

In FL including FRL, all devices may not be able to participate in learning on all rounds due to diverse reasons, such as the status of devices, communication situations, and network problems. Thus, in actual FL, some of the devices are selected for each round, and only the selected ones participate in the learning of the round. Therefore, the proper participant selection is important to improve the performance of FL, and some studies have been conducted. Lai et al. devised a guided participant selection scheme to improve the performance of federated training and testing [10]. Cho et al. conducted the convergence analysis of federated optimization

E-mail address: whlee@hansung.ac.kr.
Peer review under responsibility of The Korean Institute of Communications and Information Sciences (KICS).

https://doi.org/10.1016/j.icte.2022.08.008

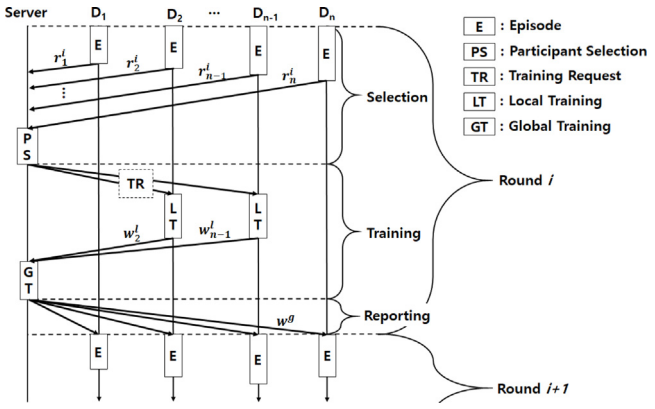


Fig. 1. The overall procedure of the proposed system.

for biased client selection strategies [11]. Also, some studies [12–14] utilized RL to improve the participant selection. Note that such techniques just used RL to improve FL and did not leverage the characteristic of FRL.

In FRL systems, multiple devices perform actions repeatedly to achieve the goal of applications. Thus, the concept of performing better operations exists in FRL systems. In other words, among the devices participating in FRL, there are some superior devices who get higher reward values. Motivated by such unique property of FRL, in this paper, we propose the reward-based participant selection scheme for improving FRL. Our study is distinct from the aforementioned researches that utilized RL to optimize the participant selection in FL. We did not devise an RL technique for the participant selection, but focused on leveraging the FRL's inherent property explained above. The FRL system with the proposed scheme performs the participant selection by considering each agent's reward value to put a priority on utilizing better experiences of the agents that performed outstanding actions and got higher rewards. We conducted various evaluations to analyze the proposed system, and the results show that the proposed system enhances the efficiency and performance of learning.

The remainder of this paper is organized as follows. In Section 2, we explain the proposed system and give detailed explanations about the reward-based participant selection scheme. In Section 3, we describe the implementation, experiments, and performance evaluation results. Finally, Section 4 concludes this paper with explaining remarks and future directions.

2. System design

In this section, we explain the details of the FRL system with our proposed technique. We first describe the concept of the proposed system. After that, we explain the reward-based participant selection, the RL algorithm for local training, and the operations in global training.

2.1. Overall procedure in proposed FRL system

Fig. 1 shows the overall procedure in the proposed system. To explain the procedure, we assume that there are n devices, $D_1, \ldots, D_n$, and one server. In the proposed FRL system, the server initially sends a global model to all the devices, and the system includes the following steps in every round. The round is the unit of performing learning from the perspective of FL system.

Episode is the unit of learning in RL. For each episode, every device starts a sequence of interactions with its environment. At every time step t, the device observes a state, s_t, in the environment. In the given state, the policy chooses an action, a_t, and the device takes the selected action. Then, the state transitions to a new state, s_{t+1}, and the device gets a reward, r_{t+1}, from the environment as feedback. The device stores the trajectory segment $<s_t, a_t, r_{t+1}, s_{t+1}>$ into its trajectory memory as its experience, and the trajectory segments are maintained throughout all rounds. Every device repeats taking actions and getting rewards until the end condition of episode is met. The end condition varies depending on applications of IoT systems.

In the participant selection step, the server in general FL systems selects some devices to participate in the learning in a round randomly or by taking into account various factors, such as communication situation, the devices' workload, battery status, and data distribution. However, in the proposed system, the server considers rewards gained by each device for the participant selection. The proposed technique can be compatible with existing selection techniques, not in a way that replaces the existing ones. The detailed process of the reward-based selection will be described in Section 2.2.

After the selection, the server requests the selected devices to perform learning, and the devices conduct local training. We will describe the detailed explanation about the learning algorithm used in local training in Section 2.3. After each device finishes the local training, every device transmits the locally trained model, w^l, to the server. After receiving the trained models, the server integrates the models to create a global model, w^g, and delivers the aggregated global model back to the devices for the following episode in the next round.

The above operations take place on a round and are repeated over and over again until the global model is sufficiently trained to achieve the required performance.

2.2. Reward-based participant selection

The appropriate selection of participant devices is important for better performance of FL. In the FRL system, devices repeatedly perform actions to accomplish the purpose of the application. Thus, naturally, there are some devices achieving better results compared to the others. The fact that devices with better rewards or scores exist is a unique property of FRL, and such characteristic is not found in typical FL. Therefore, motivated by such characteristic of FRL, we devise the system that utilizes the reward for participant selection.

In the proposed FRL system, the selection and training steps in Fig. 1 are the parts to which the proposed technique is applied. In typical FL, devices that can participate in the FL transmit the set of status values, $<i_1, i_2, i_3, \ldots, i_n>$, including various information, such as battery status, amount of

Algorithm 1 Reward-based participant selection.

Input: status information, reward information
Output: the list of selected participation devices
1: S = [], R = []
2: **Collecting devices' status and reward information:**
3: **for** i in range(n) **do**
4: S.append($s_i[0:-2]$)
5: R.append($s_i[-1]$)
6: **end for**
7: **Checking the qualification of participation:**
8: R_q = []
9: **for** i in range(n) **do**
10: **if checker**($S[i]$, $H[i]$) **then**
11: R_q.append(i,$R[i]$)
12: **end if**
13: **end for**
14: **Prioritization and Selection:**
15: D_p = **prioritizer**(R_q)
16: D_s = $D_p[0:f*n]$
17: **return** D_s

Table 1
Variables used for reward-based participant selection.

Notation	Description
S	Set of devices' status value set
R	Set of devices' reward value
s_i	ith device's set of status values
R_q	Set of qualified devices' index and reward value
H	List of devices' history information
D_p	List of devices arranged in order of reward value
D_s	List of devices selected to participate in the training

computation being performed, and communication situation, to the server at the start of every round. However, unlike the existing systems, each device in the proposed system transmits the set of status values including the sum of reward values gained by the device (i.e. the score), $<i_1, i_2, i_3, \ldots, i_n, r>$, in the recent episode. r_n^i in Fig. 1 means the sum of reward values of nth device in the ith round. Algorithm 1 shows the pseudo code of reward-based participant selection technique, and Table 1 lists the variables used for Algorithm 1. Lines 3 to 6 in Algorithm 1 are relevant to collecting the devices' status and reward information. The server separates the status and reward values from the information sent by the devices and stores them in S and R, respectively.

After that, the server in typical FL selects some devices to participate in the learning in this round by considering the devices' status and history information. However, in the proposed system, such information is used only to determine whether each device has the qualification to participate in the training in this round. If one device has the qualification, the server stores the device's index and reward value in R_q. Lines 8 to 13 in Algorithm 1 are relevant to these operations.

Then, based on the reward value of the qualified devices, the server prioritizes the devices. In other words, the devices with higher score have higher priorities. In addition to the priority, the server considers the participation ratio, f, to select participant devices. For example, if there are n devices available for FRL and f is 0.7, the server finally selects only $0.7\times n$ devices with high priority among the qualified devices. Lines 15 to 17 in Algorithm 1 are relevant to these operations.

Through the above processes, the server finally chooses the participant devices and then transmits messages requesting the devices to perform RL, represented as the training request step in Fig. 1.

2.3. Local training: reinforcement learning

This subsection explains the RL algorithm used as the local training in the proposed system with reference to [8,15,16]. Proximal Policy Optimization (PPO) was utilized in the system, and PPO uses two separate networks based on the actor–critic concept [15]. PPO was inspired by Trust Region Policy Optimization (TRPO) [16]. PPO provides a more direct approach to coordinating tasks for learning compared to TRPO. In addition, PPO is known to require simpler implementation and provide better performance in many applications in IoT [17]. For these reasons, we chose PPO as the RL algorithm used in the proposed FRL system. However, note that the proposed system was not designed to be specific to a particular RL algorithm, so the system can also be used with any RL algorithm.

In the proposed system, the actor model, π_θ, determines an action, whereas the critic model, V_μ, evaluates the action and provides a feedback to optimize the actor model. As we explained in Section 2.1, the trajectory memory contains the trajectory segments $<s_t, a_t, r_{t+1}, s_{t+1}>$ for every time step t. Using the trajectory segments, the gradients for the optimization of the actor and critic models are calculated whenever a certain number of time steps proceed. The objective function, L^V, is as follows:

$$L^V(\mu) = \hat{\mathbb{E}}_t\big[|\hat{V}_\mu(s_t) - V_\mu(s_t)|\big]. \tag{1}$$

The target value of time-difference error, $\hat{V}_\mu$, is as follows:

$$\hat{V}_\mu(s_t) = r_{t+1} + \gamma V_\mu(s_{t+1}), \tag{2}$$

where γ indicates the discount factor. The parameters of V_μ are updated using a stochastic gradient descent (SGD) algorithm as follows:

$$\mu = \mu - \eta_\mu \nabla L^V(\mu), \tag{3}$$

where η_μ is the learning rate used for the critic model optimization.

In the actor model, the importance sampling is used to obtain the expectation of samples gathered from the old policy under the new policy, and the surrogate objective function, L^{CPI}, can be maximized as follows:

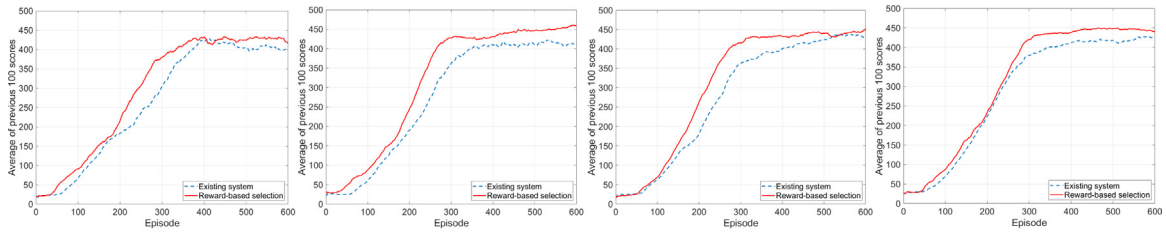
$$L^{CPI}(\theta) = \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}\hat{A}_t\right] = \hat{\mathbb{E}}_t\big[R_t(\theta)\hat{A}_t\big]. \tag{4}$$

CPI is the conservative policy iteration [18], $\hat{A}_t$ means the estimator of the advantage function, and $R_t(\theta)$ is the probability ratio. L^{CPI} is optimized subject to the constraint on the amount of the policy update with δ as follows:

$$\hat{\mathbb{E}}_t\big[\mathrm{KL}[\pi_{\theta_{old}}(\cdot|s_t), \pi_\theta(\cdot|s_t)]\big] \le \delta. \tag{5}$$

KL indicates the Kullback–Leibler divergence [19]. The objective function of PPO, L^{CLIP}, is as follows:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min\Big(R_t(\theta), \mathrm{clip}\big(R_t(\theta), 1-\epsilon, 1+\epsilon\big)\Big)\hat{A}_t\right], \tag{6}$$



(a) 4 agents (b) 6 agents (c) 8 agents (d) 10 agents

Fig. 2. The performance comparison between the proposed system with reward-based participant selection and the existing system.

Table 2
Hyperparameters and values of PPO used for the proposed system.

Hyperparameter	Value
Horizon value	32
Minibatch size	16
Number of epochs	4
Discount factor gamma	0.99
Learning rate	0.0003
Generalized advantage estimator	0.95
Clipping parameter	0.2
Value function coefficient	0.5
Optimizer algorithm	Adam
Actor network dimension	4*256*256*2
Critic network dimension	4*256*256*1

where ϵ is the clipping parameter. By using the SGD algorithm, the actor model's parameters are updated as follows:

$$\theta = \theta - \eta_\theta \nabla L^{CLIP}(\theta). \tag{7}$$

η_θ is the learning rate for the actor model optimization.

Using the above algorithms, the devices requested to learn from the server perform the local training and then transmit the trained model's parameters to the server, represented in the training step in Fig. 1.

2.4. Global training: federated learning

After the server receives the trained models from the participant devices, the server integrates the models to build the global model. In detail, the server aggregates the local actor models' parameters into the global actor model, $\pi_\theta^g = \sum_{i=1}^{n} \pi_\theta^i / n$, where n is the number of the participant devices. Similarly, the server also integrates the local critic models' parameters into the global critic model, V_μ^g. Then, the server delivers the global actor and critic models back to the devices for the following episode and learning in the next round, represented as the reporting step in Fig. 1.

3. Performance evaluation

In this section, we first describe the implementation of the proposed system. After that, we explain the experiments of learning performance and efficiency, and we show the evaluation results.

3.1. Implementation

We implemented the proposed FRL system in Python using PyTorch library [20] by referring to [21]. We used PPO explained in Section 2.3 as the RL algorithm with reference to the work in [22], and Table 2 shows the hyperparameters used in the algorithm. We utilized OpenAI Gym's CartPole-v1 environment [23] to evaluate the proposed system since the environment is famous and widely used in RL researches. In the environment, there is a pole attached by a passive pivot joint to a cart, and the goal is to prevent the pole from falling over by pushing the cart to the left or right. A reward of 1 is given for each timestep that the bar remains upright. The state vector for this system is a four dimensional vector having components, the cart's position and velocity, and the pole's angle and angular velocity.

As explained in Section 2.1, by using the proposed FRL system, we trained the learning models and tested the agents in the environment. For each episode, every agent repeated taking actions based on the policy network and getting rewards until the episode ends. After the episode ended, the local and global trainings were performed, and then every agent with the updated policy network started the next episode. For implementation, we built the system on Ubuntu 20.04 LTS using the desktop equipped with AMD Ryzen™ 7 5800X, 32 GB RAM, and NVIDIA GeForce RTX 3070.

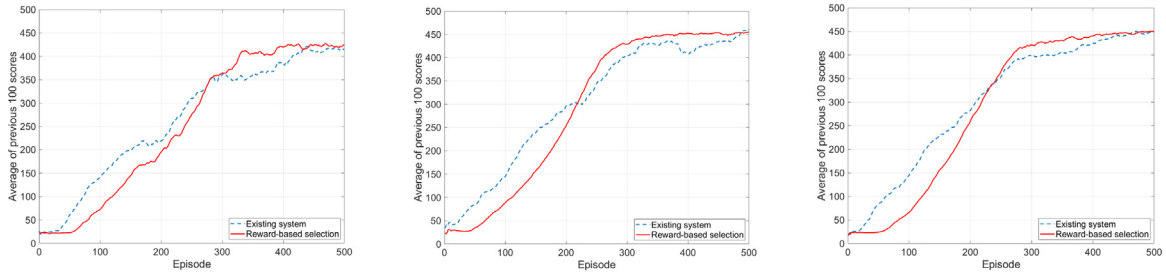
3.2. Evaluation of learning performance

We conducted the experiment to compare the performances of FRL systems with or without the reward-based participant selection. We performed the evaluation by varying the number of agents from 4 to 10, and the participation ratio of FRL system, f, was set to 0.75. We measured the sum of final scores of the last 100 episodes to evaluate the learning performance.

As shown in Fig. 2, the average score increased faster in the early stage of learning, and the final performance was also higher when the proposed technique was used. In addition, Table 3 shows that it took a smaller number of episodes required for the score to exceed 400 and 450 when using the proposed scheme. The above results mean that the learning performance was improved by using the propose FRL system.

3.3. Analysis of learning efficiency

In addition to the learning performance, we analyzed the proposed technique in terms of learning efficiency. Similar



(a) 4 agents (b) 6 agents (c) 8 agents

Fig. 3. The analysis of learning efficiency when using the proposed FRL system with f of 0.5 and the existing system with f of 0.75.

Table 3
The number of episodes required to exceed the target performance.

Target score	Average number of episodes	
	w/o	w/
400	366.2	288.3
450	780.0	520.3

to the experiments in Section 3.2, we performed FRL by changing the number of agents. However, unlike the previous experiments, there was a difference in the value of f when the proposed technique was used or not. The learning in the existing system was performed with f of 0.75, whereas the value of f was set to 0.5 in the learning when using the proposed system. In other words, a smaller number of agents were used to perform learning in the proposed system than in the existing system.

Fig. 3 shows the analysis result. In the beginning of learning, there was little difference in the superiority of the devices' experiences because the network was barely trained. Thus, it was more important to utilize as much experience as possible to train the model in the early stage of learning. Therefore, naturally, the existing system using more agents performed learning faster at the beginning. However, as the learning progressed, some agents that got higher scores by performing more proper actions started to appear, which means that the learning performance could be improved by putting a priority on utilizing their better experiences. As shown in the figures, after some episodes had passed, the proposed system with the reward-based participant selection outperformed the existing system in all cases despite using fewer agents. Naturally, after a sufficiently large number of episodes passed, the existing system also got high scores. However, the proposed system reached the saturation point much earlier, which means that the proposed system performed learning better. These results show that it can be more efficient for training models in FRL systems to selectively utilize agents with superior experiences than unconditionally using as many agents as possible.

4. Conclusion

In this paper, we proposed the reward-based participant selection scheme which leverages the FRL's unique property that the concept of performing more proper actions with better experiences exists. The FRL system with the proposed scheme performs the participant selection by considering rewards to put a priority on utilizing the better experiences of agents that perform the outstanding actions for learning. We conducted various experiments, and the results show that the learnings were performed faster and the fewer agents were required when using the proposed scheme, which means that the proposed scheme improves the performance and efficiency of learning.

As a future work, we plan to apply the proposed participant selection scheme to various IoT systems and perform diverse evaluations in different IoT applications. In addition, we will analyze the case of applying the proposed scheme to FRL with devices operating in very different environments.

CRediT authorship contribution statement

Woonghee Lee: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Roles/Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was financially supported by Hansung University, Republic of Korea.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, B.A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial Intelligence and Statistics, PMLR, 2017, pp. 1273–1282.

[2] V. Mothukuri, P. Khare, R.M. Parizi, S. Pouriyeh, A. Dehghantanha, G. Srivastava, Federated learning-based anomaly detection for IoT security attacks, IEEE Internet Things J. (2021).

[3] B. Yuan, S. Ge, W. Xing, A federated learning framework for healthcare iot devices, 2020, arXiv preprint arXiv:2005.05083.

[4] M. Chen, A. Liu, W. Liu, K. Ota, M. Dong, N.N. Xiong, RDRL: A recurrent deep reinforcement learning scheme for dynamic spectrum access in reconfigurable wireless networks, IEEE Trans. Netw. Sci. Eng. 9 (2) (2021) 364–376.
[5] M. Chen, W. Liu, T. Wang, S. Zhang, A. Liu, A game-based deep reinforcement learning approach for energy-efficient computation in MEC systems, Knowl.-Based Syst. 235 (2022) 107660.
[6] H.H. Zhuo, W. Feng, Q. Xu, Q. Yang, Y. Lin, Federated reinforcement learning, 2019, arXiv preprint arXiv:1901.08277.
[7] M.K. Abdel-Aziz, C. Perfecto, S. Samarakoon, M. Bennis, W. Saad, Vehicular cooperative perception through action branching and federated reinforcement learning, 2020, arXiv preprint arXiv:2012.03414.
[8] H.-K. Lim, J.-B. Kim, J.-S. Heo, Y.-H. Han, Federated reinforcement learning for training control policies on multiple IoT devices, Sensors 20 (5) (2020) 1359.
[9] Z. Xue, P. Zhou, Z. Xu, X. Wang, Y. Xie, X. Ding, S. Wen, A resource-constrained and privacy-preserving edge-computing-enabled clinical decision system: A federated reinforcement learning approach, IEEE Internet Things J. 8 (11) (2021) 9122–9138.
[10] F. Lai, X. Zhu, H.V. Madhyastha, M. Chowdhury, Oort: Efficient federated learning via guided participant selection, in: 15th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 21), 2021, pp. 19–35.
[11] Y.J. Cho, J. Wang, G. Joshi, Client selection in federated learning: Convergence analysis and power-of-choice selection strategies, 2020, arXiv preprint arXiv:2010.01243.
[12] H. Wang, Z. Kaplan, D. Niu, B. Li, Optimizing federated learning on non-iid data with reinforcement learning, in: IEEE INFOCOM 2020-IEEE Conference on Computer Communications, IEEE, 2020, pp. 1698–1707.
[13] P. Zhang, P. Gan, G.S. Aujla, R.S. Batth, Reinforcement learning for edge device selection using social attribute perception in industry 4.0, IEEE Internet Things J. (2021).
[14] Y. Dong, P. Gan, G.S. Aujla, P. Zhang, RA-RL: Reputation-aware edge device selection method based on reinforcement learning, in: 2021 IEEE 22nd International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM, IEEE, 2021, pp. 348–353.
[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, 2017, arXiv preprint arXiv:1707.06347.
[16] J. Schulman, S. Levine, P. Abbeel, M. Jordan, P. Moritz, Trust region policy optimization, in: International Conference on Machine Learning, PMLR, 2015, pp. 1889–1897.
[17] M. Chen, H.K. Lam, Q. Shi, B. Xiao, Reinforcement learning-based control of nonlinear systems using Lyapunov stability concept and fuzzy reward scheme, IEEE Trans. Circuits Syst. II: Express Briefs 67 (10) (2019) 2059–2063.
[18] S. Kakade, J. Langford, Approximately optimal approximate reinforcement learning, in: In Proc. 19th International Conference on Machine Learning, Citeseer, 2002.
[19] S. Kullback, Information Theory and Statistics, Courier Corporation, 1997.
[20] Pytorch, 2022, https://pytorch.org/.
[21] A.R. Jadhav, Federated-learning, 2021, https://github.com/AshwinRJ/Federated-Learning-PyTorch.
[22] P. Tabor, Youtube-code-repository, 2020, https://github.com/philtabor/Youtube-Code-Repository/tree/master/ReinforcementLearning/PolicyGradient/PPO/torch.
[23] OpenAI, Cartpole-v1, 2022, https://gym.openai.com/envs/CartPole-v1/.