Improving Personalized IoT Services Providing by Edge-Cloud Collaborated Reinforcement Learning: A Case Study on Smart Lighting

Authors Name/s per 1st Affiliation (Author)
line 1 (of Affiliation): dept. name of organization
line 2: name of organization, acronyms acceptable
line 3: City, Country
line 4: Email: name@xyz.com

Authors Name/s per 2nd Affiliation (Author)
line 1 (of Affiliation): dept. name of organization
line 2: name of organization, acronyms acceptable
line 3: City, Country
line 4: Email: name@xyz.com

***Abstract*—Internet of Things (IoT) is an important platform in bringing people, processes, data, and things together. As a typical IoT application, the smart home aims to improve the comfort and efficiency of occupants by monitoring, controllingand automating functions through connected, intelligent devices in a home. Despite the research and products of the smart home are being readily available, it is still challenging to realize a real smart home that has self-awareness, self-management, and self-learning. Considering the limitations of using cloud computing or edge computing alone, we try to explore a hybrid approach by combining the two computing paradigms and propose a reinforcement learning-based edge-cloud collaborative learning framework. In the edge environment, i.e., in a home, we apply the basic Q-learning or lightweight Deep Q-Network(DQN) to learn users' local personalized usage habits and make proper real-time decisions. In the cloud center, we adopt the regular DQN to dig out global consensus to improve edge learning and decision making. We implemented two different decision-making strategies by fusing both the locally personalized model and the global consensus model within a smart lighting system as a proof-of-concept. The experiment results show that the smart device can make decisions more efficiently than only use online learning in the edge environment.**

***Keywords*-edge-cloud collaborative learning; personalized services; reinforcement learning; Q-learning; Deep Q-Network (DQN); smart lighting**

I. Introduction

Services have been migrating from traditional areas and old IT infrastructure to the more advanced service-oriented architecture (SOA) pattern and the Cloud [1]. It is accelerating the trend of providing everything as a service (XaaS or asS) and leading to a new Internet, i.e., the Internet of Things (IoT) [2], [3]. IoT enables all kinds of real-world objects (including human beings) to be connected to the cyber world and interact with each other through providing and consuming services [4]. During the interaction between the physical and the cyber world, computer programs in the cyber world play the role of an intelligent agent. They try to extract useful knowledge from the data generated in the physical world and produce proper reactions to the physical world. To process the enormous amount of raw data efficiently, it needs a powerful data center which has enough storage and computing resources. Although cloud computing is an excellent platform to handle the enormous IoT data, pushing all the raw data to the cloud is inefficient in terms of response latency, network bandwidth cost, and possible privacy concerns. Especially, it is not suitable for applications like smart home, which demands a quick response, high usability, and privacy-preserving [5], [6].

A smart home system comprises sensors, monitors, interfaces, appliances, and devices networked together to enable automation as well as localized and remote control of the domestic environment [7], [8]. As human-in-the-loop is higher than any other IoT application, usability that encompasses easy to use and learn plays a vital role in the success of a smart home, and intelligence becomes an essential integrant to get automation smarter. Intelligence in smart home refers to the ability to predict human behavior from the collection of raw data, the management of information, the learning of experience, the understanding of the surroundings, and the adaptation to dynamic environments [9]. Currently, products such as Amazon Echo, Apple HomeKit, Google Home, etc. can play the role of a central gateway to connect devices and provide an interface for human-home interactions in a smart home. However, the current smart home systems are still far from being "smart" that has self-awareness, self-management, and self-learning abilities [10]. The main obstacle is how to accomplish real-time data stream analytics and event processing more efficiently in the resource-constrained home environment [11], [12], [13].

To solve the above massive real-time data processing problem, the emerging edge computing paradigm [14], also known as fog computing [15], is becoming a good solution and get more attention in both research and industry domain. Edge computing enables data processing closer to data creators by extending the cloud computing functions to the edges of the network. It is suitable for applications requiring low-latency, location-awareness, and privacy protection [16]. Nevertheless, although edge computing can address some of the drawbacks of cloud computing, it usually faces limitations in storing and computing the massive and continuously generated data in a non-stop running smart home system.

Additionally, considering the sample sparseness resulted from unbalanced distribution, there is a clear need to investigate suitable architecture and interaction mechanisms for improving local user experience in a smart home by global consensus learning and sharing among different homes.

In this regard, establishing coordination between edge computing and cloud computing is proposed as a promising way, and a few works have recently paid attention to such a hybrid computing model [13], [17], [18], [19]. Most of them focus on the basic principles and mechanisms of data processing. They do not consider the issue of how to enhance the ability of learning and adaptation by edge-cloud collaboration. Besides, since the successful application in playing Atari and Go games, Deep Reinforcement Learning (DRL) is flourishing in providing personalized services [20], [21], [22], [23]. However, most of the deep learning revolution has been limited to the cloud as it requires numerous computational resources for developing new machine learning models. Edge nodes are often in charge of executing the pre-trained models due to their limited computing resources. Considering the dynamic nature of the physical environment and human behavior, the latency from data uploading to get a new model might reduce the effectiveness of local decision making based on the outdated model.

To improve the efficiency and effectiveness of providing personalized services in the edge environment, we propose a reinforcement learning-based edge-cloud collaborative learning and decision making framework. In the edge environment, i.e., in a home, we apply the basic Q-learning or lightweight Deep Q-Network(DQN) to learn users' local personalized usage habits and make proper real-time decisions. In the cloud center, we adopt the regular DQN to dig out global consensus to improve edge learning and decision making. We implemented two different decision-making strategies by fusing both the locally personalized model and the global consensus model within a smart lighting system as a proof-of-concept. The main contributions of this paper are as follows

- A reinforcement learning-based edge-cloud collaborative framework.
- Two decision-making strategies by fusing the global model and the local model.
- An edge-cloud collaborated reinforcement learning based smart lighting system as a proof-of-concept.

The rest of the paper is organized as follows. We describe the background and review related work in Section II, introduce the framework and operations of the edge-cloud collaborative reinforcement learning in Section III, present the implementation details of the proposed smart lighting system in Section IV and the experiments in Section V, and offer our conclusion in Section VI.

II. Background and Related Work

A. Background

Within the many subsystems of a smart home, lighting plays a significant role in our daily lives. Generally, lighting includes the use of both natural illumination in the form of daylight and electric illumination provided by various light sources, such as incandescent, fluorescent, and light-emitting diodes (LED). Nowadays, LED is widely adopted as it is energy-efficient and can be controlled digitally. Together with the flourishing of IoT, a new generation of LED lighting systems are emerging, i.e., LED-based intelligent lighting systems where LEDs are integrated with sensors and actuators. They can also work with other smart things to improve our lifestyles in terms of convenience, ambiance, customizability, and power savings [24]. For example, Philips Hue is a wireless lighting product, which can cooperate with a range of smart devices such as Amazon Echo, Apple HomeKit, Google Home, etc. to provide a convenient and comfortable way for occupants to control and experience light.

A lighting control strategy is the core of an intelligent lighting system that describes how the illumination can be modified under given goals. For example, switches, dimmers, and scene setters are basic strategies to satisfy individual preferences and save energy through manual control [25]. And to enhance the quality of user experience, light control strategies need to be more flexible to be able to adjust the light for providing comfortable illuminance with minimized energy cost automatically. Especially, due to the varying nature of daylight and diversity of user requirements, self-learning and adaptation are highlighted by many works [9], [10], [26] as crucial capabilities of a smart lighting system.

To adapt to users' explicit or implicit wishes dynamically, a smart lighting system needs to establish a closed-loop as Fig.1. Sensors perceive changes occurred in the home while residents perform their daily routines. And the sensor readings are collected to generate useful knowledge, which will help to make more accurate decisions. For instance, readings of motion sensors are collected for predicting residents' behaviors, while lightness/color readings are combined with manual adjusting commands to model user's preferences. Based on such information, a central controller produces a target lightness or color to adjust the LEDs. As a result, the ambient brightness and color are changed, which triggers a new perception/action cycle.

B. AI based Smart Lighting System

In this loop, AI and data-mining technologies are widely adopted to seek useful information on resident behavior and the state of the environment for generating satisfactory reactions. [27] describes a method for intelligently controlling home lights using the ANN-IMC network. [28] implemented

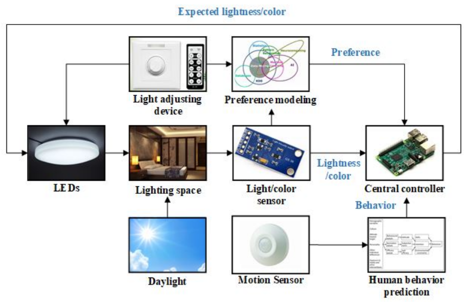

Figure 1. System structure of a conventional smart lighting system with closed-loop feedback.

a Raspberry Pi-based system that can use mobile phones and the web for light control. [29] describes a light adjustment method that calculates the light intensity of the user's location based on the user's positioning and the brightness of the light and adjusts the light. [30] introduced an intelligent lighting system based on user behavior patterns, using online learning neural networks for decision-making, and using sliding windows to avoid infinite data growth.

These approaches usually storing and analyzing the continuous human-system interactions during the non-stop system running. As mentioned above, considering the limitation of using cloud computing or edge computing alone, it is crucial to assign tasks reasonably in terms of latency, bandwidth, privacy-preserving, etc..

C. Edge-Cloud Collaboration Frameworks

Among most of the current studies, virtualizing the resources and services over WAN networks becomes the shared premises. Researches such as Pcloud [31], CoTware [32], FocusStack [33], etc. emphasize to virtualize resources of individual devices, edge nodes and cloud to build a distributed resource pool for supporting resource-limited front-end devices. While, SpanEdge [34], CloudPath [35], ECHO [18], etc. focus on the data stream processing across different layers seamlessly.

These works establish an elastic data analysis environment. However, most of them pay attention to leveraging resources on higher layers along the path from front-end devices to cloud with fewer considerations on how these analysis results will reflect behaviors provided on the front layers. Besides, as presented in several works [8], [14], [36], most existing studies lack data sharing among multiple stakeholders, while the sharing may help these edge nodes to make smarter decisions. Thus, it is still challenging to support more diversified, personalized, and delay-sensitive system behaviors in smart homes.

D. Edge-Cloud Collaborated Model Training

At present, research on improving computing power and effects based on edge-cloud collaboration is still in its early stages [36]. Among them, there are three primary schemes for training models using edge-cloud collaboration.

1) Gradient sharing: Reduce the transmission size of a single model by compressing the gradient, so that the model update results are transmitted frequently and multiple times to make up for the lack of computing power of edge nodes [37]. The training effect in the network is independent of the same and distributed data. As a result, the sharing effect of multi-edges in heterogeneous networks with different data sets cannot be guaranteed.
2) Parameter sharing: The edge side conducts preliminary training of the model and transmits the parameters to the parameter server. The parameter aggregation method in the cloud helps to improve the accuracy of the edge side model [38]. This scheme can reduce the transmission volume. It also protects the privacy and improves model accuracy, but in scenarios such as smart homes with high personalization requirements, the method of parameter aggregation still has challenges.
3) Data sharing: When it is necessary to collect the original data and perform parameter aggregation or training directly on the parameter server, noise can be added to the data on the edge side or privacy leakage can be reduced by preprocessing [39]. At the same time, there are some methods to study how to enhance the processing capabilities of edge nodes through algorithms or model hardware [40], [41].

Existing work focuses more on improving the efficiency of data analysis and model training and protecting privacy. However, the issue of how to improve the personalized intelligence at the edge through the edge-cloud collaboration still needs further studies.

III. System Model for Edge-Cloud Collaborative Reinforcement Learning

A. Basic Ideas

Taking the smart lighting, for example, different users have varying degrees of comfort in ambient light (including the brightness, color, and color temperature, etc.). On the one hand, it requires capturing users' personalized explicit or implicit requirements by self-learning. On the other hand, considering the influences from weather, season, and other external factors, even the user change his/her preference, the system should be able to adapt to these new situations for providing better services. Therefore, we focus on two key aspects to achieve smarter automation, learning, and adaptation.

1) How to share knowledge among different edge nodes with the assistance of cloud: Single edge environments always face the data sparsity problem, for example, lack of various states of weather, season, and system

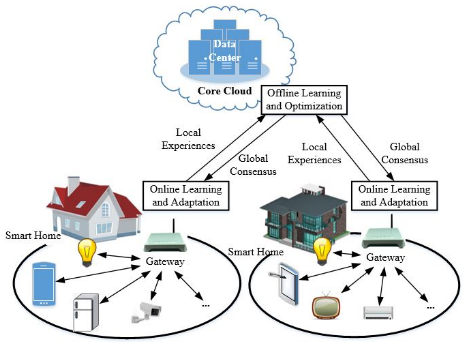

Figure 2. The overview of collaboration between edge systems and the cloud for learning and adaptation.

deployment. It is necessary to share knowledge among different edge systems, which will enable an edge system to make more smart decisions by taking advantage of the situations shared by others that haven't already appeared but might happen in the future.

2) How to utilize the historical experiences/knowledge generated on cloud in making real-time decisions on edge nodes to create more accurate reactions efficiently: Learning performed on cloud is off-line based on the historical data submitted by different edge systems. Therefore, the resulted experiences/knowledge usually reflect past situations that may be outdated to current user requirements and environment states. Thus, it needs an appropriate mechanism to integrate the downloaded historical experiences with instantaneous rules of the local environment to improve the accuracy of reactions generated by an edge decision-maker.

To provide large-scale personalized services under the limitations of using cloud computing or edge computing alone, we propose a hybrid framework combing these two computing paradigms and focus on how to realize and improve the self-learning and adaptive ability of an edge system. An overview of the proposed collaboration between edge systems and the cloud is shown in Fig.2. We try to divide the learning and adaptation into two layers: online learning and adaptation performed on edge systems and offline learning and optimization performed on the central cloud.

B. Model Definition for Using Reinforcement Learning

According to the definition of reinforcement learning, we use a quadruple $< S, A, P, R >$ to represent a reinforcement learning model, where S represents an environmental state, A represents an action, P represents a state transition probability, and R represents a reward value.

In reinforcement learning, the state comes from the agent's observation of the environment. We installed four kinds of sensors around a light, which are light sensors, ultrasonic sensors, and infrared sensors. Thus, we define the state as a 5-tuple $< Bright1, Bright2, Distance, PIR, Time >$, which includes the values of two light sensors, the distance data obtained by the ultrasonic sensor, the thermal signal sensed by the infrared sensor, and the current time.

For an LED, the action represents the adjustment of the lamp output by the model. For simplicity, we only consider the brightness in this paper. There should be two kinds of actions. One is a determined value of brightness or a predefined gear. The other is one of the operations up, down, and hold.

Reinforcement learning needs to construct reward functions for training the model. To achieve higher user satisfaction, we define the reward function as follows.

$$r = \gamma * R_{positive} + I_{control} * R_{negative} + penalty \quad (1)$$

where, γ is the discount rate, that is, the reward decay rate, and $I_{control}$ is whether the user has operated the lamp. If so, a negative return $R_{negative}$ will be added. To avoid the situation that the lights are frequently adjusted by the model, a $penalty$ term is added.

C. The Model-Training and Decision-Making Process

To apply the reinforcement learning algorithm in a smart lighting system, the training process of the model is different from the training process of the traditional reinforcement learning algorithm. Currently, most of the training of reinforcement learning algorithms run in a simulated environment. For providing personalized lighting services, using simulation software can not generate data reflecting users' personalized characteristics. It is hard to obtain real-time data and adjust the lights in real-time during the model training process. Thus, we propose a training method that trains the model during interactions between the user and the smart light.

The basic flow of our training method is in the following order, initializing a model, obtaining sensor data to construct the input state, putting the state into the model and getting output from the model, applying the model output to the environment, adjusting the brightness of the light, continuing to obtain the relevant data in the next state, obtaining the user's feedback information, and calculating the corresponding reward value based on the feedback information. Different algorithms train the model according to their iterative formula and loss functions. Then it gets the next state and repeats the above steps. At last, we can get a model when the process converges to a stable state.

Based on the basic ideas of transfer learning and integrated learning, we propose two different collaboration strategies by fusing both the local model and the global model.

1) Collaboration by Adjusting the Global Model's Output: Because the global consensus model is achieved by training on data uploaded by multiple users, it contains user consensus with similar characteristics. However, when the model is delivered to an edge environment, it may not be able to meet the individual features of a given user. To satisfy users' personalized usage habits and requirements, we need to modify the output of the global model properly. To this end, we take the output of the global model as an additional input in training the local personalized model, as shown in Algorithm 1, which both accelerates the training of local model but also improves the effectiveness of the decision making.

Algorithm 1 Collaborative algorithm by adjusting cloud model's output.

Require:
The environment state, $S =< Bright1, Bright2, Distance, Time >$;
User's operation on the light, $I_{control}$
The globel model, $Model1$
Ensure:
The adjustment action to the light, $Action$;

```
Set the training episode to n;
for i = 1 to n do
    Achieve the initial state S;
    Set the max adjustment time to m;
    for j = 1 to m do
        Input S to Model1 and get the output o1;
        Combine S and o1 to a new state S^-;
        Input the state S^- into the local model Model2.
        Train the model and get the output Action;
        Perform the generated Action;
        Achieve the next state S';
        Achieve use's operation I_control;
        Compute the Reward based on I_control;
        Perform related iterative formula or loss function
        to optimize Model2.
        Update the state to the next iteration, S = S';
    end for
end for
```

2) Collaboration by Fusing the Outputs of the Two Models: As the global model and the local model have different data samples, the trained models have specific differences. Data in the edge environment is obtained based on the habits of limited users and has precise accuracy. The global model is based on the patterns of many users, which reflects diverse usage habits. It is possible to get a better solution by fusing the output of two models according to a certain proportion. Such fusing can be realized as follows.

$$H(x) = \sum_{i=1}^{N} w_i * h_i(x) \tag{2}$$

where, w_i is the weight of the output $h_i(x)$.

Under the edge-cloud collaboration framework, both the cloud and the edge model might have some information not learned by the counterpart model. So the setting of weights needs to balance the advantages of both parties. Assuming that the accuracy of both parties is the same, the weights of the two are the same, and the advantages of both parties can be guaranteed to be balanced. While, if they have different degrees of accuracy, it is necessary to ensure that the model with higher accuracy has a higher weight. With this in mind, we define the following equation to compute the weight w_i based on the corresponding accuracy Acc_i.

$$w_i = \frac{Acc_i}{\sum_{i=1}^{N} Acc_i} \tag{3}$$

It should be noted that if the weight is set in this way, the accuracy of the cloud model must be high enough. Because if the model's accuracy is low, the weighted output makes no sense. Only under the guidance of a more accurate cloud model, local decision making can be improved effectively.

On this basis, suppose the accuracy of the edge model is Acc_{edge}, and the accuracy of the cloud model is Acc_{cloud}. The corresponding model output value formula is

$$\begin{aligned} H(x) = & \frac{Acc_{edge}}{Acc_{edge} + Acc_{cloud}} * h_{edge}(x) \\ & + \frac{Acc_{cloud}}{Acc_{edge} + Acc_{cloud}} * h_{cloud}(x) \end{aligned} \tag{4}$$

Initially, the edge model has just started training and is still in the process of exploration. At this time, the accuracy of the local model should be set to 0, while the weight of the cloud model should be set to 1. After training for a while, as the accuracy of the edge model improves, its weight will gradually increase.

IV. Smart Lighting System based on the Proposed Edge-Cloud Collaboration Architecture

As a proof-of-concept, we implement an edge-cloud collaborative reinforcement learning-based smart lighting system, whose framework is shown in Fig.3. There are three layers in the proposed framework, cloud layer, edge layer, and device layer. On the cloud layer, we deploy a k8s master node to control the operation of all containers and a k8s node to run cloud services, including global model-related services and sample-related services. We deploy a Docker in the Raspberry Pi on edge, which is responsible for running edge services, including edge-side data collection, lamp control services, and local model-related services. The cloud selects container images and deploys to the edge

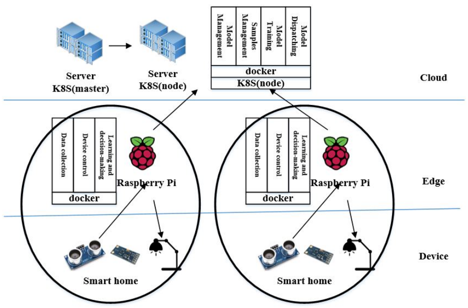

Figure 3. Framework of the Smart Lighting System

Table I
CONFIGURATION OF HARDWARE USED IN THE EXPERIMENTS

Hardware/	Configuration
Raspberry Pi	Quad-core A53,64 bit, 1.4GHz, 1G Memory
Sensors	Light sensors, Ultrasonic sensors, Infrared sensors
NodeMCU	With wifi module
Desktop Computer	Intel i7-9700k, 16G memory, GTX1070Ti
Light	Yeelight

device. After the edge node downloads the required image, it starts the container for related services. Services on the edge nodes connect sensors and front-end devices, collect data, and console lights. At the same time, decisions and console lights are made based on real-time data.

V. EXPERIMENTS

A. Experimental Setup

The experiment uses a desktop computer to write programs and configure a Raspberry Pi to collect state-related data. The relevant software and hardware configurations used in the experiment are shown in Table I.

B. Evaluation Criteria Setting

To evaluate the quality of the given model, we adopt the average squared error in the regression problem as an evaluation standard. It is the average of the squares of the differences between all observed values and the predicted values output by the model. Its calculation formula is as follows.

$$MSE = \frac{1}{n} * \sum (y_i - \hat{y_i})^2 \tag{5}$$

where y_i is an observed value and $\hat{y_i}$ is a predicted value.

Besides, $Reward$ received after each episode is also adopted to compare the effectiveness of different strategies.

C. Performance Evaluation on the GYM Platform

The experiment was first performed in a simulated environment. The CartPole environment in GYM was selected for the experiment. First, experiment with the independent training and decision-making of the edge model from scratch. The experimental results are shown in Fig.4.

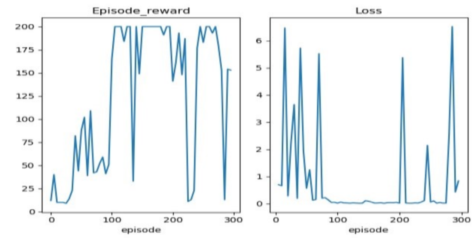

Figure 4. Performance of Independent Training and Decision-Making on the Edge Part

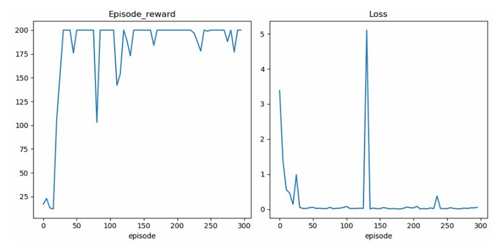

Figure 5. Performance of Taking the Cloud Model Output as Input of Local Model

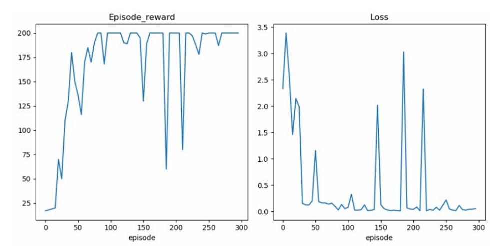

Figure 6. Performance of Fusing both Model's Outputs

It can be seen from the figure that during the independent training and decision-making process of the edge model, the overall revenue is on the rise. However, due to the model's exploration mechanism and the uncertainty of the environment, the training process of the model is unstable.

Fig.5 and Fig.6 show the episode rewards and loss of the two collaboration strategies.

From these two pictures, we can see that our collaboration strategies have higher convergence speed and more reliable stability than training a model from scratch. But compared with the policy that modifies the output of the cloud model, the convergence speed of the fusing outputs strategy is relatively slow, and the stability is relatively weak. The reason is that the two models are trained independently. If there is any state that has not been encountered before, the local model cannot make a correct decision, which will, in turn, decrease the accuracy of the final system behavior.

D. Performance Evaluation on the Smart Lighting System

First, we need to train a good enough model for adjusting the brightness of the smart light and save the relevant model parameters as a cloud model for edge loading. When the

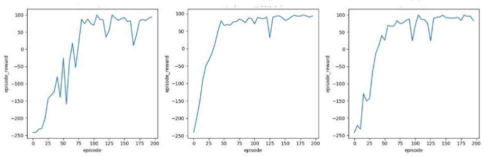

Figure 7. Episode Reward Comparison among Independent Training, Modifying the Global Model Output, and Fusing Both Models Outputs.

edge is used for the first time, a relatively simple structure model is initialized locally, and the pre-trained cloud model is loaded by the edge. Then, the user's usage data will be recorded every 300 milliseconds or when the user makes adjustments. These data include the value of each sensor at each time point and the user's feedback (that is, the user's adjustment of the light). Such real-time data is further input into the model to make an adaptation decision and start training with the user's feedback. The episode reward of independent training, modifying the global model output, and fusing both models outputs are shown in Fig.7 from left to right.

Experimental results in smart lighting scenarios lead to similar conclusions to which we have got in the simulated environment. The effect of collaboration is better than the model's independent decision. The method of output fusion is a little unstable in the early stage, but it can also achieve better results in the later stage.

VI. Conclusion

Considering the limitation by using cloud computing or edge computing alone, we propose an edge-cloud collaboration approach to improve the effectiveness and efficiency of providing personalized services in smart homes. We first establish a hybrid architecture to support edge-cloud collaborative learning and decision-making. Then, we present its application in a smart lighting scenario. From the experimental results, we can see that using the proposed collaborative learning and decision-making strategies would result in shorter convergence time and more stably execution.

Acknowledgment

The authors would like to thank... more thanks here

References

[1] Y. Duan, X. Sun, A. Longo, Z. Lin, and S. Wan, "Sorting terms of "aas" of everything as a service," *International Journal of Networked and Distributed Computing*, vol. 4, no. 1, pp. 32–44, 2016.

[2] F. Mattern and C. Floerkemeier, "From the internet of computers to the internet of things," in *From active data management to event-based systems and more*. Springer, 2010, pp. 242–259.

[3] L. Atzori, A. Iera, and G. Morabito, "Understanding the internet of things: definition, potentials, and societal role of a fast evolving paradigm," *Ad Hoc Networks*, vol. 56, pp. 122–140, 2017.

[4] I.-L. Yen, G. Zhou, W. Zhu, F. Bastani, and S.-Y. Hwang, "A smart physical world based on service technologies, big data, and game-based crowd sourcing," in *2015 IEEE International Conference on Web Services*. IEEE, 2015, pp. 765–772.

[5] P. Corcoran and S. K. Datta, "Mobile-edge computing and the internet of things for consumers: Extending cloud computing and services to the edge of the network," *IEEE Consumer Electronics Magazine*, vol. 5, no. 4, pp. 73–74, 2016.

[6] S. A. Osia, A. S. Shamsabadi, S. Sajadmanesh, A. Taheri, K. Katevas, H. R. Rabiee, N. D. Lane, and H. Haddadi, "A hybrid deep learning architecture for privacy-preserving mobile analytics," *IEEE Internet of Things Journal*, 2020.

[7] M. Alaa, A. A. Zaidan, B. B. Zaidan, M. Talal, and M. L. M. Kiah, "A review of smart home applications based on internet of things," *Journal of Network and Computer Applications*, vol. 97, pp. 48–65, 2017.

[8] B. L. R. Stojkoska and K. V. Trivodaliev, "A review of internet of things for smart home: Challenges and solutions," *Journal of Cleaner Production*, vol. 140, pp. 1454–1464, 2017.

[9] T. K. Hui, R. S. Sherratt, and D. D. Sánchez, "Major requirements for building smart homes in smart cities based on internet of things technologies," *Future Generation Computer Systems*, vol. 76, pp. 358–369, 2017.

[10] J. Cao, L. Xu, R. Abdallah, and W. Shi, "Edgeos_h: A home operating system for internet of everything," in *2017 IEEE 37th international conference on distributed computing systems (ICDCS)*. IEEE, 2017, pp. 1756–1764.

[11] Y. Qin, Q. Z. Sheng, N. J. Falkner, S. Dustdar, H. Wang, and A. V. Vasilakos, "When things matter: A survey on data-centric internet of things," *Journal of Network and Computer Applications*, vol. 64, pp. 137–153, 2016.

[12] E. Ahmed, I. Yaqoob, I. A. T. Hashem, I. Khan, A. I. A. Ahmed, M. Imran, and A. V. Vasilakos, "The role of big data analytics in internet of things," *Computer Networks*, vol. 129, pp. 459–471, 2017.

[13] S. K. Sharma and X. Wang, "Live data analytics with collaborative edge and cloud processing in wireless iot networks," *IEEE Access*, vol. 5, pp. 4621–4635, 2017.

[14] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[15] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for internet of things and analytics," in *Big data and internet of things: A roadmap for smart environments*. Springer, 2014, pp. 169–186.

[16] C. Vallati, A. Virdis, E. Mingozzi, and G. Stea, "Mobile-edge computing come home connecting things in future smart homes using lte device-to-device communications," *IEEE Consumer Electronics Magazine*, vol. 5, no. 4, pp. 77–83, 2016.

[17] X. Masip-Bruin, E. Marín-Tordera, G. Tashakor, A. Jukan, and G.-J. Ren, "Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems," *IEEE Wireless Communications*, vol. 23, no. 5, pp. 120–128, 2016.

[18] P. Ravindra, A. Khochare, S. P. Reddy, S. Sharma, P. Varshney, and Y. Simmhan, "Echo: An adaptive orchestration platform for hybrid dataflows across cloud and edge," in *International Conference on Service-Oriented Computing*. Springer, 2017, pp. 395–410.

[19] Q. Zhang, Q. Zhang, W. Shi, and H. Zhong, "Firework: Data processing and sharing for hybrid cloud-edge analytics," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 9, pp. 2004–2017, 2018.

[20] H. Li, T. Wei, A. Ren, Q. Zhu, and Y. Wang, "Deep reinforcement learning: Framework, applications, and embedded implementations," in *2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2017, pp. 847–854.

[21] T. Wei, Y. Wang, and Q. Zhu, "Deep reinforcement learning for building hvac control," in *Proceedings of the 54th Annual Design Automation Conference 2017*, 2017, pp. 1–6.

[22] Q. Fu, L. Hu, H. Wu, F. Hu, W. Hu, and J. Chen, "A sarsa-based adaptive controller for building energy conservation," *Journal of Computational Methods in Sciences and Engineering*, vol. 18, no. 2, pp. 329–338, 2018.

[23] Z. Cheng, Q. Zhao, F. Wang, Y. Jiang, L. Xia, and J. Ding, "Satisfaction based q-learning for integrated lighting and blind control," *Energy and Buildings*, vol. 127, pp. 43–55, 2016.

[24] S. Tang, V. Kalavally, K. Y. Ng, and J. Parkkinen, "Development of a prototype smart home intelligent lighting control architecture using sensors onboard a mobile computing system," *Energy and Buildings*, vol. 138, pp. 368–376, 2017.

[25] A. Pandharipande and G. R. Newsham, "Lighting controls: Evolution and revolution," *Lighting Research & Technology*, vol. 50, no. 1, pp. 115–128, 2018.

[26] M. T. Imam, S. Afshari, and S. Mishra, "An experimental survey of feedback control methodologies for advanced lighting systems," *Energy and Buildings*, vol. 130, pp. 600–612, 2016.

[27] N. K. Kandasamy, G. Karunagaran, C. Spanos, K. J. Tseng, and B.-H. Soong, "Smart lighting system using ann-imc for personalized lighting control and daylight harvesting," *Building and Environment*, vol. 139, pp. 170–180, 2018.

[28] C. Cheuque, F. Baeza, G. Marquez, and J. Calderon, "Towards to responsive web services for smart home led control with raspberry pi. a first approach." in *2015 34th International Conference of the Chilean Computer Science Society (SCCC)*. Ieee, 2015, pp. 1–4.

[29] T. Yu, Y. Kuki, G. Matsushita, D. Maehara, S. Sampei, and K. Sakaguchi, "Design and implementation of lighting control system using battery-less wireless human detection sensor networks," *Ieice Transactions on Communications*, 2016.

[30] A. Paulauskaite-Taraseviciene, N. Morkevicius, A. Janaviciute, A. Liutkevicius, A. Vrubliauskas, and E. Kazanavicius, "The usage of artificial neural networks for intelligent lighting control based on resident's behavioural pattern," *Elektronika ir Elektrotechnika*, vol. 21, no. 2, pp. 72–79, 2015.

[31] M. Jang, K. Schwan, K. Bhardwaj, A. Gavrilovska, and A. Avasthi, "Personal clouds: Sharing and integrating networked resources to enhance end user experiences," in *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE, 2014, pp. 2220–2228.

[32] J. Al-Jaroodi, N. Mohamed, I. Jawhar, and S. Mahmoud, "Cotware: a cloud of things middleware," in *2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW)*. IEEE, 2017, pp. 214–219.

[33] B. Amento, B. Balasubramanian, R. J. Hall, K. Joshi, G. Jung, and K. H. Purdy, "Focusstack: Orchestrating edge clouds using location-based focus of attention," in *2016 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 2016, pp. 179–191.

[34] H. P. Sajjad, K. Danniswara, A. Al-Shishtawy, and V. Vlassov, "Spanedge: Towards unifying stream processing over central and near-the-edge data centers," in *2016 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 2016, pp. 168–178.

[35] S. H. Mortazavi, M. Salehe, C. S. Gomes, C. Phillips, and E. de Lara, "Cloudpath: A multi-tier cloud computing framework," in *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*, 2017, pp. 1–13.

[36] Q. Zhang, Q. Zhang, W. Shi, and H. Zhong, "Distributed collaborative execution on the edges and its application to amber alerts," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 3580–3593, 2018.

[37] C. Hardy, E. Le Merrer, and B. Sericola, "Distributed deep learning on edge-devices: feasibility via adaptive compression," in *2017 IEEE 16th International Symposium on Network Computing and Applications (NCA)*. IEEE, 2017, pp. 1–8.

[38] S. Lu, Y. Yao, and W. Shi, "Collaborative learning on the edges: A case study on connected vehicles," in *2nd {USENIX} Workshop on Hot Topics in Edge Computing (HotEdge 19)*, 2019.

[39] T. Zhang, Z. He, and R. B. Lee, "Privacy-preserving machine learning through data obfuscation," *arXiv preprint arXiv:1807.01860*, 2018.

[40] Q. Xu, S. Varadarajan, C. Chakrabarti, and L. J. Karam, "A distributed canny edge detector: algorithm and fpga implementation," *IEEE Transactions on Image Processing*, vol. 23, no. 7, pp. 2944–2960, 2014.

[41] J. Lee, H. Tang, and J. Park, "Energy efficient canny edge detector for advanced mobile vision applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 4, pp. 1037–1046, 2016.