2024 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm) | 979-8-3503-1855-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/SmartGridComm60555.2024.10738082

EMS-env: A Reinforcement Learning Framework for Residential Energy Efficiency Recommendations

Spiros Chadoulos*†, Odyssefs Diamantopoulos*, Iordanis Koutsopoulos*, George C. Polyzos*, Nikolaos Ipiotis†
*Mobile Multimedia Laboratory, Department of Informatics, School of Information Sciences and Technology,
Athens University of Economics and Business, Athens, Greece
†Plegma Labs, Athens, Greece

Abstract—**Personalized device-level energy consumption recommendations towards energy efficiency can have a notable impact both on electricity bills and on the overall energy supply-demand balance. End-user behavior regarding device activation is usually unknown a priori, thus giving rise to a highly dynamic environment. Hence, Reinforcement Learning (RL) can be utilized for device scheduling and consumption recommendations since it constitutes an Artificial Intelligence (AI) framework that learns a control policy in a dynamic environment through trying actions and observing incurred rewards. However, existing works on energy consumption recommendations do not explicitly take into account human feedback and preferences regarding the issued recommendations, and they train a single RL agent per device, hence missing the human behavior interdependencies in using different devices. In addition, a flexible open-source RL environment model that integrates user behavior in a Markov Decision Process (MDP) model is missing. In this paper, we propose an MDP-driven RL framework for energy efficiency recommendations that jointly learns the user's behavior for multiple devices. The proposed model is wrapped as an open-source customizable Gymnasium environment, named *EMS-env*, for multi-device energy efficiency recommendations. *EMS-env* can simulate different types of consumer behavior profiles based on the MDP model and supports different device types as well as user feedback. Validation experiments demonstrate the framework's merits and hyperparameters for diverse use cases in terms of user simulation models and RL training policies, resulting in decreased energy costs while maintaining end-user satisfaction.**

I. Introduction

Consumers play a critical role in determining grid demand load and patterns, while buildings are responsible for roughly one-third of the total energy consumption, with residential buildings contributing to 22% of the global energy demand [1]. Hence, engaging residential consumers towards energy efficiency through appropriate recommendations is crucial for the overall energy transition [2]. The use of device scheduling recommendations as an implicit means of minimizing energy costs based on parameters such as grid energy prices and user preferences, can have a positive impact on the energy consumption of a building. In addition, incorporating consumer feedback is crucial for such systems to retrain their algorithms and make more relevant recommendations without disturbing end users, hence shaping demand patterns towards energy efficiency [3].

However, most existing approaches for device scheduling in smart buildings do not take into account end-user feedback and only focus on finding and recommending a near-optimal schedule, regardless of whether the occupants will follow it or not. Such methods may lead to consumer fatigue and limited response in accepting recommendations. In addition, even in works where some form of user feedback is incorporated into the recommendation mechanism, a separate agent is trained for each device, hence missing the underlying correlations of consumer behavior when interacting with multiple devices. For example, turning on the boiler in a household can be directly and causally related to the operation of a dehumidifier since the occupants may jointly use these two devices for showering. Hence, capturing such device use interdependencies is needed.

Moreover, existing works mostly use different proprietary experimental methodologies to evaluate the performance of the proposed scheduling and recommendation modules, with different simulation setups and data, while open-source implementations of such environments do not exist. Thus, the need for a fully customizable open-source environment for energy efficiency recommendations for multiple devices in residential environments is prominent.

In this work, a Reinforcement Learning (RL) approach is proposed that integrates an MDP which models transition probabilities between different device states based on the recommended actions. The framework is wrapped as a customizable open-source[1] Gymnasium environment, named *EMS-env*. The proposed RL model allows researchers to customize the simulated consumer behavior in terms of recommendation acceptance at a device level, supporting both intermittent and uninterruptible devices. Intermittent devices can be turned on and off as needed while uninterruptible ones should stay on for a set time once activated. Namely, the number and power consumption of different device types can be customized when *EMS-env* is initialized, along with the energy price signals, discomfort parameters in the reward function, and system transition probabilities. A high-level representation of the proposed RL pipeline is shown in Fig.1. Overall, the contributions of this work are the following:

- We devise an MDP model for user recommendation acceptance, which can capture different types of building occupant behavior by adjusting the underlying transition probabilities respectively.
- We propose a framework that gives the agents the ability to handle multiple devices simultaneously, including

[1]GitHub repository: https://github.com/SpirosChadoulos/EMS-env

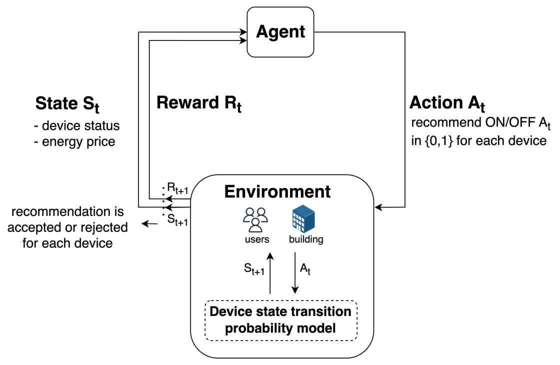

Agent
State S_t
- device status
- energy price
Reward R_t
Action A_t
recommend ON/OFF A_t in {0,1} for each device
R_{t+1}
S_{t+1}
recommendation is accepted or rejected for each device
Environment
users
building
S_{t+1}
A_t
Device state transition probability model

Fig. 1: *EMS-env* high-level RL pipeline.

intermittent and uninterruptible devices, i.e. devices that can be activated intermittently and devices that should run for a specific set of time slots respectively.

- We wrap the proposed RL model as an open-source Gymnasium environment for smart homes and buildings, offering full customizability at a device level.
- We conduct a set of experiments to demonstrate how *EMS-env* can be utilized effectively, increasing the energy efficiency of a building, using different types of user profiles and environment parameters.

The rest of the paper is organized as follows: In Section II, the related work is presented, and the contributions of *EMS-env* compared to the literature are discussed. In Section III, the RL model is presented. In Section IV, the functionalities offered by the environment are demonstrated through data experiments, and in Section V, conclusions are drawn.

II. Related Work

Reinforcement Learning (RL) is widely used for recommendations and asset control with the goal of residential energy efficiency from different perspectives in terms of feedback.

A. Approaches without user feedback

In [4], a deep RL approach is proposed to solve the problem of online scheduling of energy sources both at building and aggregated level. The agent makes multiple actions at each time-step related to switching ON/OFF events of 3 types of electrical devices: "time-scaling" loads (e.g. AC), "time-shifting" loads (e.g. dishwasher), and both "time-scaling" and "time-shifting" loads (e.g. EV). In [5], a multi-agent RL solution for residential energy management is proposed. A Q-learning approach with 4 agents is adopted for different devices to minimize energy costs. In [6], an RL environment for demand response in office spaces is proposed. The goal of the RL agent in this case is to offer points to office workers to incentivize them to shift their energy consumption to times of the day when the energy prices are lower. In [7] an RL environment is proposed for automated control of the Heating, Ventilation, and Air Conditioning (HVAC) system within a building based on heat transfer and weather conditions. It is evident that future research directions should include actual human feedback which RL agents can use in the learning process [3].

B. Approaches with user feedback

In [8] and [9], deep RL is utilized to develop a RecSys for buildings that proposes energy-saving actions, such as thermostat setpoint changes, reductions in heating and lighting in a specific area of the building, and recommendations for occupants to move to a different area of the building. The system also co-optimizes user comfort and air quality within the building. In [10], a residential energy recommendation system that utilizes deep RL along with occupant feedback and activities is designed and validated, by training a separate agent for each device. Specifically, different types of labeled human activities are integrated into the environment state, such as "cooking", "having breakfast", "sleeping", etc., along with electricity prices, device status, device activation duration, and EV-related parameters.

Our proposed RL model differs from existing works since it incorporates human feedback in the form of a fully customizable MDP user behavior simulation framework. It also provides recommendations for multiple intermittent and uninterruptible devices simultaneously, capturing the correlations of user behavior patterns when operating different devices by utilizing a joint action space and a combined reward function that considers all device types. In a nutshell, *EMS-env* builds on prior works to offer a novel energy efficiency recommendations RL environment for smart homes and buildings, incorporating human feedback, different types of users, different types of devices, and multi-device recommendations.

III. Environment model

The overall goal of the *EMS-env* framework is to make an energy efficiency recommendation for each device at each time slot, so as to minimize the energy consumption cost for the building while preserving occupant satisfaction without causing fatigue to them. Time is divided into time slots $t \in \{1, \dots, T\}$ and κ is the duration of a time slot in hours. *EMS-env* handles multiple devices simultaneously, with P^d denoting the mean power consumption of device $d \in \{1, \dots, D\}$ in kW (when activated), and ϕ_t^d standing for the status of device d at time slot t (1 for ON, and 0 for OFF). The proposed environment supports two types of devices:

(a) *Intermittent devices:* Most household and building devices fall under this category, meaning that they can be activated or deactivated intermittently. Examples of intermittent devices include lights, Air Conditioners (ACs), fans, ventilation systems, and dehumidifiers.

(b) *Uninterruptible devices:* These devices should stay ON for a specific number of time slots when they are activated (e.g. washing machines). The usage duration (in hours) of an uninterruptible device d is $l_d \in \mathbb{R}^+$. Also, the usage countdown $u_t^d \in [0, l_d]$ represents the residual time for an uninterruptible device after it is activated.

The total number of devices in the building or household is $D = \hat{D} + \tilde{D}$, where $\hat{D}$ is the number of intermittent devices and $\tilde{D}$ is the number of uninterruptible devices. The environment includes the non-stationary price evolution process $\{p_1, \ldots, p_t, \ldots, p_T\}$, from the electricity utility company for each time slot t. The agent can be trained and deployed within the household to preserve user privacy since device energy consumption data do not leave the house premises.

A. State

The state S_t at time slot t includes:

- The time (hour) at time slot t: h_t
- The energy consumption of device $d \in \{1, \ldots, D\}$ at time slot t: $P_t^d = \kappa P^d \phi_t^d$
- The energy price from the retailer at time slot t: p_t
- The usage countdown u_t^d for each uninterruptible device.

Thus, the state S_t at time slot t is: $S_t = (P_t^1, \ldots, P_t^D) \cup h_t \cup p_t \cup (u_t^1, \ldots, u_t^{\tilde{D}})$. The state is fully observable, i.e. the RL agent has full access to the state parameters. Hence, the agent can utilize the state parameters to make actions in the form of recommendations.

B. Action

Based on the observed state, the agent takes an action in the form of an energy efficiency recommendation regarding the activity status of each device. In other words, the agent's action is a D-dimensional binary vector representing the recommended state for D devices. Hence, the agent's action A_t at time slot t is defined as: $A_t = (A_t^1, \ldots, A_t^d, \ldots, A_t^D)$ where A_t^d is the recommended action regarding device $d \in \{1, \ldots, D\}$ for time slot t. A_t^d is 1 if device d is recommended to be ON at time t, and 0 if it is recommended to be OFF. This holds both for intermittent and uninterruptible devices.

C. System transition probability model

An RL agent requires a significant volume of data, i.e. interactions with the environment, to learn a policy. However, in the case of energy efficiency recommendations, it is not always feasible to train the RL agent in an online manner from scratch with real feedback from building occupants. For that reason, we define a user behavior simulation module that can represent different types of users with a device state transition probability model. As presented in Fig. 2, the MDP for each device is defined by the states $S = \{s_1, s_2\}$, where s_1 and s_2 represent the device being ON or OFF respectively (the device index d is not included for simplicity).

For each device (the device index is dropped again for simplicity), the initial formulation of the action space includes three different actions, $A' = \{a_1, a_2, a_3\}$ representing the agent's recommendation to turn the device ON, turn the device OFF, or to do nothing respectively. The transition probabilities are derived from user responses based on the current device state and the agent's recommendation: $\omega = P(s_1|s_1, a_3)$, $p = P(s_2|s_1, a_2)$, $\theta = P(s_2|s_2, a_3)$, $q = P(s_1|s_2, a_1)$. To make the MDP model simpler, the action space presented in Fig. 2 can be transformed into a binary action space $A = \{a_1, a_2\}$ where the only actions are the recommendations to turn the device ON or OFF respectively. Action a_3 can

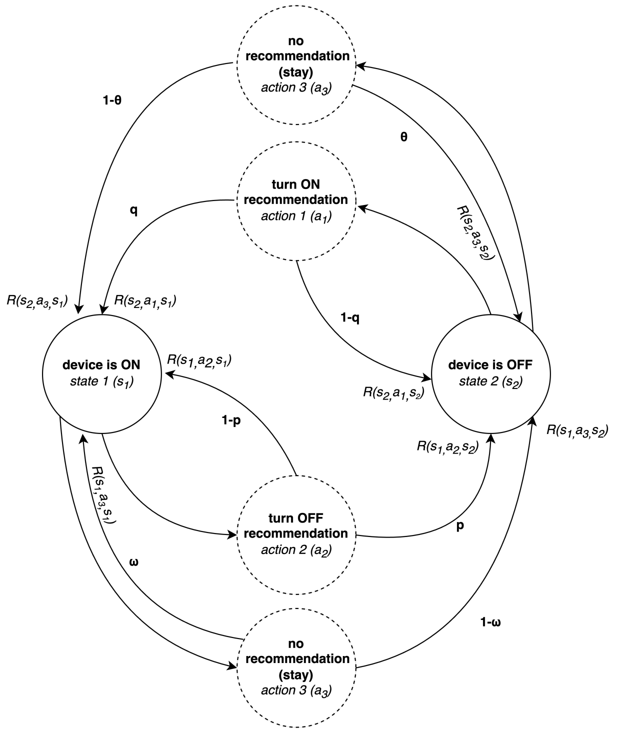

no recommendation (stay) action 3 (a_3)
1-θ
θ
turn ON recommendation action 1 (a_1)
q
$R(s_2,a_3,s_2)$
$R(s_2,a_3,s_1)$
$R(s_2,a_1,s_1)$
1-q
device is ON state 1 (s_1)
$R(s_1,a_2,s_1)$
device is OFF state 2 (s_2)
$R(s_2,a_1,s_2)$
1-p
$R(s_1,a_2,s_2)$
$R(s_1,a_3,s_2)$
$R(s_1,a_3,s_1)$
turn OFF recommendation action 2 (a_2)
p
ω
1-ω
no recommendation (stay) action 3 (a_3)

Fig. 2: Device state transition probability model

be replaced using actions a_1 and a_2 as follows: $a_3 = a_1$ when $s = s_1$, and $a_3 = a_2$ when $s = s_2$. Namely, when the current state of the device is ON ($s = s_1$), a recommendation to turn the device ON (a_1) is the same as an action of no recommendation (a_3) since the agent's intention is to remain in the same state in both cases. The same is the case when the current state of the device is OFF ($s = s_2$), since a recommendation to turn the device OFF (a_2) is the same as an action of no recommendation (a_3).

A binary action space $A = \{a_1, a_2\}$ for each device is adopted in the proposed environment since the reduced action space can improve the RL training complexity, while keeping the formulation simple. In terms of user experience, when the agent offers a recommendation that is the same as the current state, e.g. turn a device ON when it is already ON, the front-end of the system will refrain from notifying the end users to avoid spamming, especially when the recommendations frequency is high.

D. Reward

Based on the selected action A_t, the agent receives a reward and tries to maximize the average reward throughout episodes. The negative energy cost is the main parameter of the reward function to make the agent select actions that maximize total monetary gains (or minimize total costs). The cost for time slot t is defined as: $C_t = p_t \sum_{d=1}^{D} P_t^d$, with: $C_t = \hat{C}_t + \tilde{C}_t$, $\hat{C}_t = p_t \sum_{d=1}^{\hat{D}} P_t^d$, and $\tilde{C}_t = p_t \sum_{d=1}^{\tilde{D}} P_t^d$, where $\hat{C}_t$ and $\tilde{C}_t$ are the intermittent and uninterruptible device costs respectively. In addition, human feedback regarding recommendation acceptance is incorporated into the reward model to make the agent select actions that the user has a high probability of accepting. The total reward for time slot t, R_t, consists of two sub-rewards $\hat{R}_t$ and $\tilde{R}_t$ for intermittent and uninterruptible devices respectively. A variation of the reward model proposed in [10] is integrated and adapted to the *EMS-env* setting.

1) Intermittent devices reward: The intermittent devices reward for time slot t is:

$$\hat{R}_t = -\hat{C}_t - \sum_{d=1}^{\hat{D}} f_t^d \delta_d \tag{1}$$

where $f_t^d = \begin{cases} 1 & \text{if } \phi_t^d \neq A_t^d \\ 0 & \text{if } \phi_t^d = A_t^d \end{cases}$ is the user feedback and $\delta_d \in \mathbb{R}^+$ is a cost coefficient that represents the user's discomfort for device d. In *EMS-env*, δ_d is a hyperparameter that can be modified according to the target household or building.

2) Uninterruptible devices reward: The uninterruptible device reward $\tilde{R}_t$ is defined as:

$$\tilde{R}_t = \sum_{d=1}^{\tilde{D}} \tilde{\delta}_d \tag{2}$$

when $\phi_{t-1}^d = 1$, $A_t^d = 0$ and $u_t^d > 0$, otherwise: $\tilde{R}_t = -\tilde{C}_t - \sum_{d=1}^{\tilde{D}} f_t^d \delta_d$, where $\tilde{\delta}_d$ is the uninterruptible device deactivation discomfort penalty. This parameter is utilized to punish the agent when its action is to turn off an uninterruptible device while there is still time left in the device's usage countdown. For example, turning a washing machine off while the washing cycle is ongoing. Hence, the uninterruptible discomfort $\tilde{\delta}_d$ gets large values depending on the respective device and use case. The total reward for time slot t when both intermittent and uninterruptible devices are integrated is: $R_t = \hat{R}_t + \tilde{R}_t$.

In [10], the formulation of a similar problem using the δ_d parameters was introduced, while in our work we adapt it by introducing a joint reward function with sub-rewards for intermittent and uninterruptible devices using δ_d and the $\tilde{\delta}_d$ penalty respectively, along with the proposed device state MDP model for different user behavior profiles. In practice, the values of δ_d and $\tilde{\delta}_d$ can be determined via manual user preferences.

IV. Demonstration experiments

The proposed *EMS-env* environment is offered as an open-source one while it also utilizes state-of-the-art open-source libraries. It is developed using Python 3.10 and Gymnasium [11], an RL framework for working with existing environments, as well as for creating customized ones. Gymnasium is based on OpenAI's Gym library [12], essentially being a fork of Gym. *EMS-env* is developed using the offered custom environment definition capabilities of Gymnasium, by implementing the required probability transition functions, as well as the action-observation spaces and rewards, as modeled in Section III. In addition, Ray [13], RLlib [14], and Tune [15] are utilized for parallel processing with popular RL algorithms. To demonstrate the applicability of the proposed RL model and the customizability of the *EMS-env* environment, a set of simulation experiments with different setups are conducted, including different reward parameters, user types, and baseline RL algorithms.

A. Environment configurations

1) Device setup: The proposed environment is fully customizable in terms of device setup since it can support different combinations of intermittent and uninterruptible devices, with specific power consumption and usage duration values provided either when the environment is initialized or in an online manner in cases where real energy meters are utilized. In order to demonstrate the capabilities of *EMS-env*, we initialize the environment using the following devices (9 in total):

(a) *Intermittent devices*: two Air Conditioners (ACs) with a power consumption of 1 kW and 2.5 kW respectively, two 70W ceiling fans, one 3kW boiler, and one 70W dehumidifier.

(b) *Uninterruptible devices*: one 1.3 kW dishwasher with a usage duration of 2.5 hours, one 0.5 kW washing machine with a usage duration of 1 hour, and one 2.4 kW clothes dryer with a usage duration of 30 minutes.

All the power consumption values are based on realistic data from residential devices. An episode horizon of 7 days is adopted, along with a time step duration of 0.5 hours (30 minutes), with both parameters being fully configurable at the initialization stage of the environment.

TABLE I: User types of the experiments for each device.

	$\boldsymbol{\omega}_d$	$\boldsymbol{p}_d$	$\boldsymbol{\theta}_d$	$\boldsymbol{q}_d$
Receptive	0.9	0.9	0.9	0.9
Neutral	∼U(0.4, 0.6)	∼U(0.4, 0.6)	∼U(0.4, 0.6)	∼U(0.4, 0.6)
Resistant	0.1	0.1	0.1	0.1
Frugal	0.2	0.8	0.9	0.2

2) User types: EMS-env can support different types of user behaviors, as well as an interface to an external system that fetches real-time device state data from a building, depending on the needs of each use case. *EMS-env* comes by default with the device state transition probability MDP model described in section III-C, which can be fully modified. By adjusting the MDP transition probabilities when the environment is initialized, *EMS-env* can simulate different types of consumer behavior. The following user types are defined and utilized in the demonstration experiments:

(a) *Receptive:* A user that in general is receptive and positive towards the recommendations received from the agent. All transition probabilities for all devices are set to 0.9, meaning that for each device, there is a 90% probability that the agent recommendation is accepted.

(b) *Neutral:* A user that is neutral toward the recommendations. For each device, the transition probabilities are generated using a random uniform distribution between 0.4 and 0.6, i.e $\omega_d \sim U(0.4, 0.6), p_d \sim U(0.4, 0.6), \theta_d \sim U(0.4, 0.6), q_d \sim U(0.4, 0.6)$.

(c) *Resistant:* A user that is resistant to the recommendations. All transition probabilities are set to 0.1 in this case,

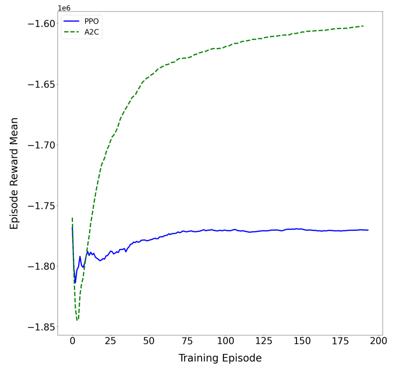
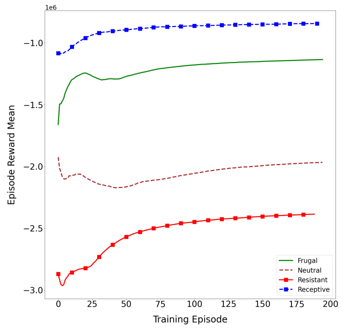
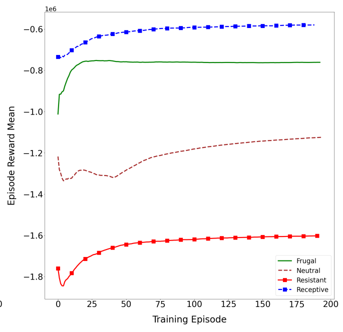


Fig. 3: A2C and PPO reward comparison



(a) $\delta_d = 100$ and $\tilde{\delta}_d = 1000$



(b) $\delta_d = 100 \cdot P^d$ and $\tilde{\delta}_d = 1000 \cdot P^d$

Fig. 4: User types reward comparison.

hence for each device, there is a 10% probability that the recommendation is accepted.

(d) *Frugal:* A user that is frugal with consumption and does not want to waste excess energy. To simulate such a user, the transition probabilities toward ON device states are smaller compared to the probabilities toward OFF device states. The specific probabilities used in the experiments are: $\omega_d = 0.2, p_d = 0.8, \theta_d = 0.9, q_d = 0.2$.

The user types above along with the respective transition probabilities are presented in Table I.

3) Energy Prices: The offered environment is fully customizable in terms of energy prices depending on the studied region or even the specific building contract. By default, the environment comes with a dataset consisting of real-time energy price signals from the New York Independent System Operator (NYISO) [16], which is also utilized in the demonstration experiments. Specifically, real-time energy price data (in cents per kWh) regarding the zone of Long Island are used from January 2018 to December 2018.

4) Rewards: By default, *EMS-env* utilizes the reward mechanism described in Section III-D, which includes the hyperparameters δ_d and $\tilde{\delta}_d$, representing the device cost coefficient for user discomfort and uninterruptible discomfort penalty respectively. For the experiments, the following reward parameter values are tested: (a) $\delta_d = 100$ and $\tilde{\delta}_d = 1000$, (b) $\delta_d = 100 \cdot P^d$ and $\tilde{\delta}_d = 1000 \cdot P^d$. In the first case, we assign equal δ_d and $\tilde{\delta}_d$ values respectively for each device d in the demonstration experiments. *EMS-env* can support a more refined δ_d and $\tilde{\delta}_d$ allocation with different values for each device depending on the cost coefficient for discomfort. For that reason, the second option is also demonstrated in the experiments by including the device consumption P^d in the coefficients, meaning that in this case, devices with greater consumption have a greater impact on the occupant discomfort.

5) RL algorithms evaluated: The Advantage Actor-Critic (A2C) [17] and Proximal Policy Optimization (PPO) [18] algorithms are utilized and compared within the experiments.

B. Results

In Fig. 3, A2C is compared with PPO in terms of mean reward throughout 200 episodes, using a 30-minute time slot duration ($\kappa = 0.5$) and an episode horizon of 7 days. In addition, the environment is initialized using the *Resistant* user behavior type in order to observe how the algorithms perform in a use case where most of the recommendations get rejected. In the first 10 episodes, both algorithms have performed a lot of exploration, which has led to a steady decrease in the achieved reward. Beyond the 10-15 episode mark, they proceed to exploit their knowledge, which leads to a great increase in reward. Overall, it is clear that A2C outperforms PPO in terms of the mean reward achieved. The same is the case for the other three user types, but the respective figures are not included in the paper due to size constraints.

In Fig. 4, a set of experiments is conducted to compare the episode reward mean achieved between the four user behavior types described in Section IV-A2. Specifically, two subfigures are presented, for two different δ_d and $\tilde{\delta}_d$ setups. In both cases, the agent (A2C) achieves the highest reward values with the *Receptive* user type environment since the recommendations made are accepted with a higher probability compared to the other three user types. As expected, when interacting with the *Resistant* user type environment, the agent achieves the lowest reward since most of the recommendations get rejected. The *Frugal* user type environment yields the second highest reward, while the *Neutral* one results in the third highest reward across episodes since the *Neutral* user type has a more random behavior while the *Frugal* one is positive toward not wasting energy hence leading to higher rewards.

Overall, the mean reward increases throughout episodes in most cases for all the user behavior types tested, meaning that training an agent with *EMS-env* leads to decreased energy costs throughout time while the occupant behavior patterns are captured. In addition, the comparative results between the four different user behavior types, demonstrate the relevance and applicability of the proposed MDP user transition probability model.

In Fig. 5, we present an A2C agent trained on *EMS-env* for 200 episodes with an episode horizon of 7 days and 30-minute time slots, using the *Frugal* user behavior type. Specifically,

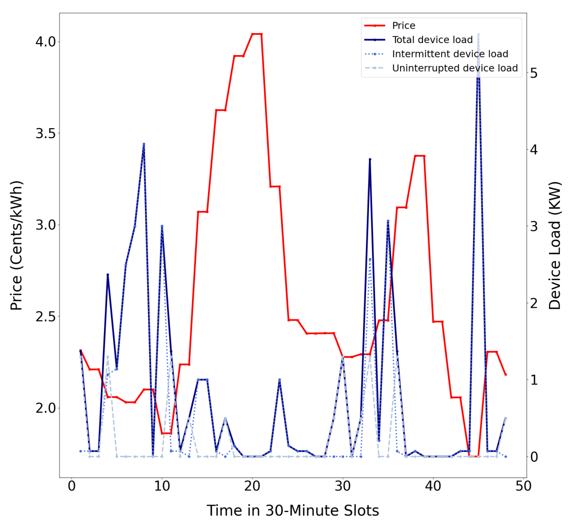


Fig. 5: Device load after recommendations compared to prices

after the training phase is completed, the agent is utilized for inference within a single day (48 time slots). Fig. 5 shows that the total device load after the agent's recommendations is low during time slots where the energy price is high and vice versa. Hence, an agent trained using *EMS-env* can be applied in similar cases to offer energy consumption recommendations that follow user behavior and avoid high energy prices.

V. Conclusion

In this paper, we proposed an RL model and approach for energy efficiency recommendations in multi-device setups that captures consumer behavior and feedback with an MDP-based user transition probability model. The model is customizable in terms of intermittent and uninterruptible devices, rewards, and energy prices, as well as user behavior types. The RL framework is wrapped as an open-source Gymnasium RL environment named *EMS-env*. Simulation experiments show that RL agents trained on EMS-env can learn different types of user behavior and achieve decreased energy costs over time. The framework is presented for a single-building setup, however, the methodology can be extended to an energy community with multiple buildings to coordinate consumption behavior toward overall energy efficiency which is part of our future work. In addition, our future work includes deploying *EMS-env* at a real building setup with multiple real users providing feedback and using Reinforcement Learning with Human Feedback (RLHF) approaches. RLHF can be utilized to learn different reward parameters, such as the discomfort values for each device and each user, which at the moment are assigned manually. This manual process can be unclear to consumers who might face difficulties accurately modeling such parameters.

Acknowledgements

This work was supported by the EMERGENT project which has received funding from the 2nd I-NERGY open call under Grant 101016508.

References

[1] B. Dean, J. Dulac, K. Petrichenko, and P. Graham, "Towards zero-emission efficient and resilient buildings.: Global status report," 2016.
[2] S. Chadoulos, I. Koutsopoulos, and G. C. Polyzos, "Mobile apps meet the smart energy grid: A survey on consumer engagement and machine learning applications," *IEEE Access*, vol. 8, pp. 219 632–219 655, 2020.
[3] J. R. Vázquez-Canteli and Z. Nagy, "Reinforcement learning for demand response: A review of algorithms and modeling techniques," *Applied energy*, vol. 235, pp. 1072–1089, 2019.
[4] E. Mocanu, D. C. Mocanu, P. H. Nguyen, A. Liotta, M. E. Webber, M. Gibescu, and J. G. Slootweg, "On-line building energy optimization using deep reinforcement learning," *IEEE transactions on smart grid*, vol. 10, no. 4, pp. 3698–3708, 2018.
[5] M. Ahrarinouri, M. Rastegar, and A. R. Seifi, "Multiagent reinforcement learning for energy management in residential buildings," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 1, pp. 659–666, 2020.
[6] L. Spangher, A. Gokul, J. Palakapilly, U. Agwan, M. Khattar, W.-J. Ma, and C. Spanos, "Officelearn: An openai gym environment for reinforcement learning on occupant-level building's energy demand response," in *Tackling Climate Change with Artificial Intelligence Workshop at NeurIPS*, 2020.
[7] C. Zhang, Y. Shi, and Y. Chen, "Bear: Physics-principled building environment for control and reinforcement learning," in *Proceedings of the 14th ACM International Conference on Future Energy Systems*, 2023, pp. 66–71.
[8] P. Wei, S. Xia, R. Chen, J. Qian, C. Li, and X. Jiang, "A deep-reinforcement-learning-based recommender system for occupant-driven energy optimization in commercial buildings," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6402–6413, 2020.
[9] S. Xia, P. Wei, Y. Liu, A. Sonta, and X. Jiang, "Reca: A multi-task deep reinforcement learning-based recommender system for co-optimizing energy, comfort and air quality in commercial buildings," in *Proceedings of the 10th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, 2023, pp. 99–109.
[10] S. S. Shuvo and Y. Yilmaz, "Home energy recommendation system (hers): A deep reinforcement learning method based on residents' feedback and activity," *IEEE Transactions on Smart Grid*, 2022.
[11] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.
[12] M. Towers, J. K. Terry, A. Kwiatkowski, J. U. Balis, G. d. Cola, T. Deleu, M. Goulão, A. Kallinteris, A. KG, M. Krimmel, R. Perez-Vicente, A. Pierré, S. Schulhoff, J. J. Tai, A. T. J. Shen, and O. G. Younis, "Gymnasium," Mar. 2023. [Online]. Available: https://zenodo.org/record/8127025
[13] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan *et al.*, "Ray: A distributed framework for emerging ai applications," in *13th USENIX symposium on operating systems design and implementation (OSDI 18)*, 2018, pp. 561–577.
[14] E. Liang, R. Liaw, R. Nishihara, P. Moritz, R. Fox, K. Goldberg, J. Gonzalez, M. Jordan, and I. Stoica, "Rllib: Abstractions for distributed reinforcement learning," in *International conference on machine learning*. PMLR, 2018, pp. 3053–3062.
[15] R. Liaw, E. Liang, R. Nishihara, P. Moritz, J. E. Gonzalez, and I. Stoica, "Tune: A research platform for distributed model selection and training," *arXiv preprint arXiv:1807.05118*, 2018.
[16] "New York Independent System Operator (NYISO)," https://www.nyiso.com/energy-market-operational-data, accessed: 2024-04-09.
[17] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1928–1937.
[18] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.