Smart Home's Energy Management through a Clustering-based Reinforcement Learning Approach

Ioannis Zenginis, John Vardakas, *Senior Member, IEEE,* Nikolaos E. Koltsaklis, *Member, IEEE,* and Christos Verikoukis, *Senior Member, IEEE*

***Abstract*—Smart homes that contain renewable energy sources, storage systems and controllable loads will be key components of the future smart grid. In this paper, we develop a reinforcement learning-based scheme for the real-time energy management of a smart home that contains a photovoltaic system, a storage device, and a Heating Ventilation and Air Conditioning (HVAC) system. The objective of the proposed scheme is to minimize the smart home's electricity cost and the residents' thermal discomfort by appropriately scheduling the storage device and the HVAC system on a daily basis. The problem is formulated as a Markov decision process, which is solved using the Deep Deterministic Policy Gradient (DDPG) algorithm. The main contribution of our study compared to the existing literature on RL-based energy management is the development of a clustering process that partitions the training dataset into more homogeneous training subsets. Different DDPG agents are trained based on the data included in the derived subsets, while in real-time, the test days are assigned to the appropriate agent, which is able to achieve more efficient energy schedules when compared to a single DDPG agent that is trained based on a unified training dataset.**

***Index Terms*—smart home, energy management, reinforcement learning, clustering.**

I. Introduction

Internet of Things (IoT) technologies are the key drivers contributing to the development of Microgrids (MGs) [1], which are local energy networks comprising Distributed Energy Resources (DER), Energy Storage Systems (ESSs), and controllable loads[2]. MGs' adoption aims to make the operation of traditional energy systems more efficient, economic and environment-friendly [3]. Hence, large amounts of Renewable Energy Sources (RES) will be placed close to electricity consumers [4], while energy management schemes should be developed for achieving proper coordination of the various MGs' components through the utilization of IoT platforms [5].

MGs appear at various scales, from a single building to a whole neighborhood [6]. For several reasons, including government incentives, decreasing costs, and environmental awareness, there is a rapid increase in the number of PV installations at residential buildings around the world. The cost of ESSs also follow a decreasing trend, making the investment in PV-storage systems popular [7]. ESS is a valuable component because it mitigates the impacts of load fluctuations and PVs' stochastic generation, while it also provides with energy management flexibilities [8], such as taking advantage of the energy arbitrage; when dynamic pricing schemes are applied, the ESS can store energy during low price periods and provide it back to the demand over peak pricing [9]. Moreover, dynamic pricing offers opportunities to reduce energy cost through the smart scheduling of controllable loads [10], such as Heating, Ventilation, and Air Conditioning (HVAC) systems, which account for about 40% of total energy consumption in a household [11]. An effective energy management scheme should minimize the operating cost of HVAC systems while maintaining the residents' thermal comfort.

Traditionally, the MGs' energy management issue is formulated as a sequential optimization problem that determines appropriate set points for the controllable devices so that energy costs are minimized. Several optimization frameworks have been proposed in [12]-[15], including Mixed Integer Linear Optimization (MILP) in [12], non-linear optimization in [13], and Model Predictive Control (MPC) in [14]-[15], which target to minimize the operating cost of buildings' HVAC systems, while also considering the occupants' thermal comfort. MPC and MILP frameworks have been also applied in [16] and [17], respectively, for the energy cost optimization of smart homes that contain RES, ESSs, and controllable loads. The optimal solutions in [12]-[17] are obtained through the utilization of commercial solvers. In contrast, this is avoided in [18]-[20] where approximate dynamic programming techniques are developed for minimizing the energy costs of MGs, whose operation is modeled as a Markov Decision Process (MDP).

The energy management schemes in [12]-[20] are model-based, which means that detailed domain knowledge is required to construct accurate system models describing the MGs' dynamics and their components' interactions. In addition, accurate forecasts of stochastic variables are required to obtain the optimal control decisions. Any inaccuracies of the employed system models and forecasting methods may deteriorate the solutions' quality.

On the contrary, Reinforcement Learning (RL) techniques can leverage the deployment of IoT devices to develop model-free energy management schemes [21], which do not require the analytical design of system models, the accurate forecasting of stochastic variables and the utilization of expensive commercial solvers. RL is a process where an agent interacts with an environment in order to learn what to do (actions) in given situations (states) so that numerical returns (rewards) are maximized. RL agents are trained through the utilization of historical data, and then they are able to deal with unknown situations based on the gained experience.

In recent years, there is a large number of studies that de-

I. Zenginis and J. Vardakas are with Iquadrat, Barcelona, Spain e-mail: ({izenginis, jvardakas}@iquadrat.com).

N. Koltsaklis is with Czech Technical University (CTU), Prague, Czech Republic, (e-mail: nikkoltsak@gmail.com)

C. Verikoukis is with CEID, University of Patras, Patras, Greece, (e-mail: cveri@upatras.gr).

velop RL-based energy management schemes. The Q-learning algorithm is applied in [22]-[24] for the scheduling of smart homes' controllable appliances, while the Deep Q-learning (DQN) algorithm is employed in [25]-[26] for the same purpose. In all cases, the main objective is the minimization of the smart homes' energy costs, while the minimization of users' dissatisfaction and the reduction of peak demand are additional objectives that are considered in [23]-[24] and [25], respectively. DQN is also applied in [27] for minimizing the daily operating cost of a MG that is equipped with controllable DER, RES and ESSs, while the double DQN algorithm is proposed in [28] for optimizing the cooperation between a MG and an external storage system.

In RL-based approaches, the energy scheduling problem, which is characterized by continuous variables, is modeled as an MDP. However, Q-learning is applicable to MDPs with discrete state and action spaces, while DQN is applicable to MDPs with continuous state spaces and discrete action spaces. As a consequence, these methods suffer from the curse of dimensionality [29]. To resolve the dimensionality issue, energy management schemes have been developed in [30]-[35] that utilize RL algorithms compatible with continuous state and action spaces. The Deep Deterministic Policy Gradient (DDPG) algorithm is employed for the optimal control of a building's HVAC system in [30], and in [31], where it is found that DDPG reduces the HVAC system's energy consumption by 4.31% and 8.95% while improving the occupants' thermal comfort by 13.6% and 17.6%, compared to the DQN and Q-learning algorithms, respectively. In [32], Li $et\ al.$ apply the Trust Region Policy Optimization (TRPO) algorithm for the real-time scheduling of controllable appliances in a smart home to minimize the electricity cost and to maximize the occupants' thermal comfort.

Contrary to the studies in [30] - [32], where the system models do not include any energy sources, a DDPG-based energy management scheme is developed in [33] for the optimal control of an isolated MG that is equipped with a diesel generator, a PV system, and an ESS. A DDPG-based control strategy is also proposed in [34] with the objective to minimize the daily operating cost of a residential multi-energy system that contains electrical and thermal energy sources, as well as electrical and thermal storage units. However, real-time pricing is not taken into account in [33] and [34]. As opposed to that, in [35], Yu $et\ al.$ consider dynamic pricing, and apply the DDPG algorithm to minimize the energy cost while maintaining a comfortable temperature range for the occupants of a smart home that contains RES, an ESS, and an HVAC system. However, only the cooling mode of the HVAC system is taken into account by the authors.

Motivated by the interest of the scientific community on RL methods, in this paper we develop a DDPG-based scheme for the real-time energy management of a smart home that is equipped with a PV system, an ESS and an HVAC system. Under uncertainties induced by the not-controllable load demand, the PV generation, the real-time electricity prices, and the outdoor temperatures, the proposed method targets to obtain effective energy schedules for the ESS and the HVAC system on a daily basis so that the smart home's electricity cost and the residents' thermal discomfort are minimized. Contrary to [35], both the heating and the cooling mode of the HVAC system are considered in our study.

Moreover, our proposed scheme targets to improve the effectiveness of RL agents when dealing with energy scheduling issues by reducing the variance of the training data, as well as by increasing the similarity degree between the training set and the test set. In the existing literature ([22]-[35]), the RL agents are trained based on a unified training dataset that includes stochastic variables' values, and then they use the obtained knowledge to deal with unknown situations encountered in the test set. In our work, a clustering process is developed based on the K-means algorithm that partitions the training dataset into day-type subsets consisting of more homogeneous price and outdoor temperature data points (e.g. subsets that contain clusters of high-price curves and clusters of high-temperature curves, or subsets that contain clusters of high-price curves and clusters of low-temperature curves, etc.). Then, a separate DDPG agent is trained based on the data included in each day-type subset. In this way, the training process becomes more case-oriented since the agents gain experience from days that have similar price and temperature profiles.

In addition, the agents' ability to generalize their experience becomes more effective because any day in the test set is first assigned to one of the predetermined subsets, and then, the agent that has been trained based on the information of that subset is loaded to the smart home's Energy Management System (EMS) for taking real-time decisions regarding the ESS's and HVAC system's set points. For the assignment of test days into the appropriate day-type subset, a simple forecast model is used that predicts, before the beginning of the decision horizon, the next day's price and temperature curves based on past data (e.g. a week). The predicted curves are then assigned to the closest price and temperature clusters, and hence to the appropriate day-type subset. It should be noted that predictions do not need to be extremely accurate as in optimization-based models, and so a complex forecast model is not required. Instead, a simple Long Short-Term Memory (LSTM) network proved to be adequate for effectively matching the predicted curves with the closest clusters. Our clustering-based energy management scheme can achieve up to 24.7% lower electricity costs than a no-clustering approach, while in terms of thermal discomfort, the performance gap between the two methods can reach up to 1914%, depending on the examined test sets.

The rest of the paper is organized as follows: Section II describes the various smart home's components, and the way the energy scheduling issue is formulated as a RL problem. Section III describes the DDPG algorithm's training process, the way the training dataset is partitioned into day-type subsets, and the way real-time energy management is implemented based on the trained agents. Section IV contains a case study for testing the performance of the proposed method, while Section V concludes the outcomes of our work.

II. System Model and RL Formulation

A. System Model

We consider a smart home that contains a PV system, an ESS, an HVAC system, and an EMS, which are interconnected

through an IoT-based infrastructure. The EMS is responsible for implementing daily power scheduling of the controllable devices so that electricity cost is minimized and thermal comfort is maintained, i.e. the indoor temperature is kept within a minimum Θ_{min} and a maximum Θ_{max} level. The intra-day scheduling takes place over a decision horizon of T time slots t of duration Δt. Under this convention, power is used interchangeably with energy in this paper. At every time slot, the power balance is described by:

$$P_t^G = P_t^L - P_t^{PV} + |P_t^{HVAC}| + P_t^{ESS} \tag{1}$$

where P_t^G denotes either the power imported from the main grid, if $P_t^G \geq 0$, or the power exported to the main grid, if $P_t^G \leq 0$. P_t^L is the smart home's not-controllable load, P_t^{PV} is the generated PV power, P_t^{HVAC} is the HVAC system's power, and P_t^{ESS} denotes either the power transferred to the ESS, if $P_t^{ESS} \geq 0$, or the power discharged from the ESS, if $P_t^{ESS} \leq 0$.

The indoor temperature at the next time slot is a function of the indoor temperature Θ_t^{in}, the outdoor temperature Θ_t^{out} and the HVAC system's power at the current time slot:

$$\Theta_{t+1}^{in} = \epsilon\, \Theta_t^{in} + (1-\epsilon)\left(\Theta_t^{out} - \frac{P_t^{HVAC} n_{HVAC}}{W}\right) \tag{2}$$

where positive values of P_t^{HVAC} denote that the HVAC system operates at the cooling mode and negative values denote that it operates at the heating mode. In addition, ϵ is a constant factor, n_{HVAC} is the efficiency of the HVAC system and W stands for the thermal conductivity [36]. Based on (2), the smart home's thermal comfort is maintained by appropriately adjusting the HVAC system's power up to its rated value P_{max}^{HVAC}:

$$| P_t^{HVAC} | \leq P_{max}^{HVAC} \tag{3}$$

The ESS's State of Charge (SoC) SoC_t at t is given by:

$$SoC_t = SoC_{t-1} + m_t^{ESS}\left(P_t^{ESS}/N_{ESS}\right) \tag{4}$$

where N_{ESS} is the nominal capacity of the ESS and m_t^{ESS} stands for the charging or discharging losses. When the ESS is charged, m_t^{ESS} is described by the ESS's charging efficiency n_c^{ESS} ($m_t^{ESS}=n_c^{ESS}$). In case the ESS is discharged, m_t^{ESS} is expressed as the reversed discharging efficiency n_d^{ESS}, i.e. $m_t^{ESS} = 1/n_d^{ESS}$. The ESS's SoC ranges within a minimum SoC_{min} and a maximum SoC_{max} level:

$$SoC_{min} \leq SoC_t \leq SoC_{max} \tag{5}$$

while P_t^{ESS} is bounded by a maximum power rate P_{max}^{ESS} [7]:

$$| P_t^{ESS} | \leq P_{max}^{ESS} \tag{6}$$

B. RL Formulation

The smart home's daily power scheduling is formulated as an RL problem where an agent is trained through an RL algorithm to learn how to interact with an environment. During training, the agent observes the environment's current state s_t and performs a set of actions $a_t \in A_{s_t}$. The action space A_{s_t} stands for the range of all possible actions that can be taken at state s_t, and it is defined by the rules that govern the environment's transition from one state to another. Depending on the performed actions, the environment responds with a reward signal r_t and moves to the next state s_{t+1}, where the agent performs another action, receives a new reward and so on. The sequence of states, actions and rewards, as well as the rules for transitioning from one state to another over a decision horizon, compose an MDP episode. The mapping from states to actions during an episode is defined as the agent's policy π. The objective of an RL algorithm is to train the agent so that an optimal policy π^* is achieved, where the agent's actions at every observed state maximize the total discounted reward R_t^π over an MDP episode of T steps, which is expressed as:

$$R_t^\pi = \sum_{t=0}^{T-1} r_t \gamma^t \tag{7}$$

where $0 \leq \gamma \leq 1$ is a discount factor that determines the importance of future rewards. When $\gamma = 0$, the agent considers only the current reward, while when $\gamma = 1$ the agent weighs equally both the current and the future long-term rewards.

When the smart home's power scheduling is modeled as an MDP, the state of the system $s_t = \left\{P_t^L - P_t^{PV}, \Theta_t^{in}, \Theta_t^{out}, SoC_t, \phi_t, t\right\}$ at time slot t is described by a set of variables that include the not-controllable load P_t^L minus the PV generation P_t^{PV}, the indoor Θ_t^{in} and outdoor Θ_t^{out} temperatures, the ESS's state of charge SoC_t, as well as the electricity price ϕ_t and the time slot's incremental number t. In addition, the agent's taken actions $a_t = \left\{P_t^{HVAC}, P_t^{ESS}\right\}, a_t \in A_{s_t}$ refer to the set points of the system's controllable variables, which are the HVAC system's power P_t^{HVAC} and the ESS's power P_t^{ESS}, while the action space A_{s_t} is defined by the operational constraints in (1)-(6).

The reward r_t that the agent receives from the environment at every time interval consists of three terms I_t^{elec}, I_t^{comf} and I_t^{ess} related to the electricity cost, the thermal comfort and the ESS's operation, respectively:

$$r_t = I_t^{elec} + I_t^{comf} + I_t^{ess} \tag{8}$$

where

$$I_t^{elec} = \begin{Bmatrix} -P_t^G \phi_t, & if\ P_t^G \geq 0 \\ -P_t^G \phi_t \rho, & if\ P_t^G < 0 \end{Bmatrix}, \tag{9}$$

$$I_t^{comf} = \begin{Bmatrix} 0, & if\ \ \Theta_{min} \leq \Theta_t^{in} \leq \Theta_{max} \\ -\delta\left(\Theta_{min} - \Theta_t^{in}\right), & if\ \ \Theta_t^{in} < \Theta_{min} \\ -\delta\left(\Theta_t^{in} - \Theta_{max}\right), & if\ \ \Theta_t^{in} > \Theta_{max} \end{Bmatrix} \tag{10}$$

and

$$I_t^{ess} = -\zeta U_t^{ESS} \tag{11}$$

I_t^{elec} signifies that higher rewards are obtained either when less electricity is imported ($P_t^G \geq 0$), or when more electricity is exported ($P_t^G < 0$). It should also be noted that the electricity selling price is assumed to be a fraction of the buying price ϕ_t, i.e. $0 < \rho < 1$ in (9). I_t^{comf} denotes that the agent receives a penalty when the indoor temperature deviates from the acceptable limits. The penalty depends on a weighting factor δ and the amount of deviation. Finally, I_t^{ess} represents also a penalty consisting of a weighting factor ζ and of U_t^{ESS}, which is associated with the ESS's proper operation. In case the ESS's SoC ranges within the acceptable limits of (5), $U_t^{ESS} = 0$. However, as far as the SoC limits are violated, it is computed by $U_t^{ESS} = \beta + (1-\beta) U_{t-1}^{ESS}$, where β is a weighting factor.

III. Proposed Energy Management Algorithm

A. Training process of the DDPG algorithm

The DDPG algorithm uses four Deep Neural Networks (DNNs); a critic, which is a Q-network with parameters θ^Q, an actor, which is a policy network with parameters θ^μ, a target Q-network with parameters $\theta^{Q'}$ and a target policy network with parameters $\theta^{\mu'}$. The target networks are copies of the original ones, and they are used for making the training process more stable [29]. The algorithm updates the parameters of the four DNNs so that the trained actor represents the agent's optimal policy. The training process takes place by considering MDP episodes i.e. days with a T-length decision horizon, in our case, where stochastic variables' values are derived from a hyperset of historical data that includes load demand, PV generation, outdoor temperature and price datasets over a long period of time. Specifically, the initial hyperset is firstly divided into day-type subsets that include days with similar outdoor temperature and price profiles, and then a separate agent is trained for each subset.

The process of training an agent based on a day-type subset consisting of D days' data is described in Algorithm 1. Firstly, the DNNs' parameters are initialized in line 1, while a replay buffer of size *B* is initialized in line 2. The algorithm runs for *M* iterations; at every iteration, a random episode is selected, and a random process Ξ_t is initialized, which is used for exploration (line 4). The main part of the agent's training takes place within the selected episode (lines 5-15), and consists of the following steps: firstly, the actor observes the system's current state s_t and performs an action a_t to which exploration noise Ξ_t is added for enabling the agent to gain more experience from the environment (line 6). Given the performed action, the environment responds with a reward signal r_t and moves to the next state s_{t+1} (line 7). The transition (s_t, a_t, r_t, s_{t+1}) is stored in the replay buffer, while a mini-batch of *N* transitions $\left(s_\tau^{(i)}, a_\tau^{(i)}, r_\tau^{(i)}, s_{\tau+1}^{(i)}, i \in N, \tau \in (T-1)\right)$ is randomly sampled (line 8). Following that, a forward pass takes place at the critic for deriving the Q-values of the sampled state-action pairs $(s_\tau^{(i)}, a_\tau^{(i)}, \forall i \in N)$ (line 9). A forward pass also happens at the target Q-network for obtaining the Q-values of the pairs $(s_{\tau+1}^{(i)}, a_{\tau+1}^{'(i)}, \forall i \in N)$ (line 11), where $a_{\tau+1}^{'(i)}$ is obtained by the target policy network (line 10). Given that the Q-function represents the discounted reward when action a_t is performed in state s_t, and then a policy π is followed till the end of the decision horizon, the critic's parameters are updated in line 12 by minimizing the loss function $L\left(\theta^Q\right)$, while the actor's parameters are updated in line 13 by maximizing $J\left(\theta^\mu\right)$, which represents the expectation of the Q-function [29]. Finally, the target networks' parameters are updated by slowly tracking the parameters of the original networks (line 14).

Algorithm 1: DDPG agent's training process

1: Randomly initialize the critic's and actor's parameters θ^Q and θ^μ, respectively, and set the target networks' parameters equal to them: $\theta^{Q'} \leftarrow \theta^Q, \quad \theta^{\mu'} \leftarrow \theta^\mu$.
2: Define the size *B* of the replay buffer.
3: for $ep = 1$ to M :
4: Select a random day from D, and initialize Ξ_t.
5: for $t = 0$ to $T-1$:
6: Observe s_t and perform action $a_t = \pi\left(s_t, \theta^\mu\right) + \Xi_t$.
7: Execute action a_t in the environment and observe the instant reward r_t and the next state s_{t+1}.
8: Store $(s_t, \ a_t, \ r_t, \ s_{t+1})$ in the replay buffer, and sample from it a random mini-batch of *N* transitions $\left(s_\tau^{(i)}, \ a_\tau^{(i)}, \ r_\tau^{(i)}, \ s_{\tau+1}^{(i)}\right)$, where $i \in N$ and $\tau \in (T-1)$.
9: Given $(s_\tau^{(i)}, a_\tau^{(i)}, \forall i \in N)$, do a forward pass of the critic for obtaining their Q-values $Q\left(s_\tau^{(i)}, a_\tau^{(i)}, \theta^Q\right), \forall i \in N$.
10: Given $s_{\tau+1}^{(i)}$, do a forward pass of the target policy network for obtaining the actions $a_{\tau+1}^{'(i)} = \pi'\left(s_{\tau+1}^{(i)}, \theta^{\mu'}\right)$.
11: Given $(s_{\tau+1}^{(i)}, \ a_{\tau+1}^{'(i)})$, do a forward pass of the target Q-network for obtaining $Q'\left(s_{\tau+1}^{(i)}, \ a_{\tau+1}^{'(i)}, \ \theta^{Q'}\right), \forall i \in N$.
12: Update the critic's parameters by minimizing $L\left(\theta^Q\right)$:

$$L\left(\theta^Q\right) = \frac{1}{N}\sum_i \left(y_\tau^{(i)} - Q\left(s_\tau^{(i)}, \ a_\tau^{(i)}, \ \theta^Q\right)\right)^2$$

$$y_\tau^{(i)} = r_\tau^{(i)} + \gamma \ Q'\left(s_{\tau+1}^{(i)}, \ a_{\tau+1}^{'(i)}, \ \theta^{Q'}\right)$$

13: Update the actor's parameters by maximizing $J\left(\theta^\mu\right)$ using the following sampled policy gradient $\nabla_{\theta^\mu} J\left(\theta^\mu\right)$:

$$\nabla_{\theta^\mu} J(\theta^\mu) = \frac{1}{N}\sum_i \left(\nabla_{a_\tau^{(i)}} Q\left(s_\tau^{(i)}, a_\tau^{(i)}, \theta^Q\right) \nabla_{\theta^\mu} \pi\left(s_\tau^{(i)}, \theta^\mu\right)\right)$$

14: Update the target networks:
$\theta^{Q'} \leftarrow \omega\, \theta^Q + (1-\omega)\, \theta^{Q'}, \theta^{\mu'} \leftarrow \omega\, \theta^\mu + (1-\omega)\, \theta^{\mu'}, \omega \ll 1$
15: end
16: end

B. Day-type subsets' derivation and real-time implementation

The day-type subsets, required for the DDPG algorithm's training process are derived by the proposed Algorithm 2, which is based on K-means, an iterative algorithm that tries to partition a dataset into K pre-defined clusters. K-means assigns data points (T-dimensional price and outdoor temperature curves in our case) to a cluster such that the Euclidean distance between the data points and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is minimized. The optimum number of clusters is obtained by using the silhouette score, which is a measure of similarity of a point to the other points in its own cluster when compared to the points in other clusters. After finding the optimum number of k_ϕ price and k_θ temperature clusters (lines 2-7), the initial historical hyperset is divided into day-type subsets that include more homogeneous price and temperature curves (lines 8-12). Then, a separate DDPG agent is trained for each subset following the process of Algorithm 1.

After the training process is completed, the policy networks can be used for the smart home's real-time energy management as described in Algorithm 3. Before the beginning of the decision horizon, the next day's price and temperature curves are predicted based on data of the past G days (line 1). An LSTM network is used for this purpose since it is appropriate for multi-step time series forecasting [37]. After the predicted curves are obtained, they are assigned to the closest price and temperature clusters (lines 2-3). Then, the corresponding policy network is loaded to the EMS (line 4) for implementing

Algorithm 2: Derivation of day-type subsets

1: Input the historical hyperset of training data which includes the load, PV generation, price, and outdoor temperature datasets Λ_Ω, Π_Ω, Φ_Ω and Θ_Ω, respectively, where each of which contains Ω data-points indexed by date.
2: for $k = 2$ to K :
3: Implement K-Means on the price dataset to obtain an array of labels $E_{\Phi k}$ that indexes to which of the k clusters each one of the Ω data-points belongs, as well as an array $C_{\Phi k}$ that contains the centroids of the k clusters: $E_{\Phi k}, C_{\Phi k} = KMeans(\Phi_\Omega, k)$
4: Calculate the silhouette score $S_{\Phi k}(\Phi_\Omega, E_{\Phi k})$
5: Apply steps 3 and 4 for the outdoor temperature dataset.
6: end
7: Given the silhouette scores obtained in step 4 for the price dataset, keep the maximum one $S_{\Phi k_\phi}$, which corresponds to k_ϕ price clusters, where $2<k_\phi<K$. Likewise, given the silhouette scores obtained in step 5 for the temperature dataset, keep the maximum one $S_{\Psi k_\theta}$, which corresponds to k_θ temperature clusters, where $2<k_\theta<K$.
8: Implement K-means on the price dataset by setting k=k_ϕ to obtain $E_{\Phi k_\phi}$ and $C_{\Phi k_\phi}$.
9: Given $E_{\Phi k_\phi}$, create k_ϕ subsets that are indexed by date and contain price data-points that belong to the same cluster.
10: Apply steps 8 and 9 for the outdoor temperature dataset to partition it into k_θ subsets.
11: Create $k_\phi \cdot k_\theta$ day-type subsets by taking the intersections of the indices (dates) of the k_ϕ price subsets obtained in step 9 with the indices of the k_θ temperature subsets obtained in step 10.
12: Take the intersections of the indices of the obtained day-type subsets with the indices of the load and PV generation datasets to obtain the final day-type subsets.

Algorithm 3: Real-time energy management

1: Predict the next day's price curve based on the previous G days' curves by using an LSTM network.
2: Calculate the squared Euclidean distance between the predicted price curve and the clusters' centroids $C_{\Phi k_\phi}$ and assign it to the closest price cluster $k_{\phi\_closest}$.
3: Do steps 1 and 2 for the next day's temperature curve to assign it to the closest temperature cluster $k_{\theta\_closest}$.
4: Load to the EMS the policy network of the agent that has been trained based on the day-type subset $(k_{\phi\_closest}, k_{\theta\_closest})$.
5: for $t = 0$ to $T - 1$:
6: Observe s_t and perform action $a_t = \pi(s_t, \theta^\mu)$.
7: Execute action a_t in the environment, observe the instant reward r_t and transit to the next state s_{t+1}.
8: end

real-time power scheduling (lines 5-8).

IV. Case Study

For the evaluation of the proposed energy management scheme, a smart home is considered that contains a 3 kWp PV system, an ESS with nominal capacity $N_{ESS} = 6$ kWh, and an HVAC system with nominal power $P_{max}^{HVAC} = 2$ kW. Other parameters regarding the smart home's components and the RL formulation of the problem are reported in Table I. The decision horizon is divided into $T = 24$ time slots of duration $\Delta t = 1$ hour, while stochastic variables' data for the load, the prices, as well as the PV generation and the outdoor temperature are obtained from [38], [39] and [40], respectively.

TABLE I: Parameters' values

ESS's parameters	P_{max}^{ESS}=3kW, n_c^{ESS}=n_d^{ESS}= 0.95, SoC_{min}=SoC_0=0.2
HVAC system's parameters	$\Theta_{min} = 20^oC$, $\Theta_{max} = 24^oC$, $\epsilon = 0.7$, $W = 0.252$ $kW/^oC$, n_{HVAC}=2.5
Discount factor, electricity selling price factor, weighting factors	$\gamma = 1$, $\rho = 0.5$, $\delta = 0.9$ €/^oC, $\zeta = 0.5$ €, $\beta = 0.4$
Episodes' number, Replay buffer's size, Mini-batch size	M=15000, B=10^6, N=480
Noise parameters	ξ=0.15, $m = 0$, $dt = 0.01$, σ=0.2

Our method is tested over September and November when the HVAC system operates at the cooling mode and the heating mode, respectively. In both cases, data of 12 months before the test periods are used for obtaining the day-type subsets required for training the DDPG agents, as well as for training the LSTM network that is necessary for the assignment of test days to the appropriate subset. By using Algorithm 2, it is derived that the 12-month price datasets and the 12-month outdoor temperature datasets before both September and November are optimally divided into two price ($k_\phi = 2$) and two temperature ($k_\theta = 2$) clusters. Hence, four different agents are trained for each case that correspond to High Price (HP) - High Temperature (HT), HP - Low Temperature (LT), Low Price (LP)-HT and LP-LT day-type subsets.

Each agent consists of two DNNs representing the actor and the critic. The actor's input layer consists of six neurons, which correspond to the smart home's state s_t. The critic's input layer consists of eight neurons because, besides the state, it also takes as input the actor's output. Both DNNs have three hidden layers consisting of 256 neurons and relu activation functions. The critic's output layer has a single neuron with a linear activation function, while the actor's output layer consists of two neurons which are passed through a tanh activation function. Furthermore, an Ornstein-Uhlenbeck process, which is defined as $\Xi_t = \Xi_{t-1} + \xi(m - \Xi_{t-1})dt + \sigma\sqrt{dt}\ \mathcal{N}(0, 1^2)$, is added to the actor's output for exploration [29]. After that, the actor's output is truncated in the interval [-1, 1], while it is also multiplied by P_{max}^{HVAC} and P_{max}^{ESS}. In this way, the actor outputs a value for P_t^{HVAC} that ranges in the interval $[-P_{max}^{HVAC}, P_{max}^{HVAC}]$, and a value for P_t^{ESS} that ranges in the interval $[-P_{max}^{ESS}, P_{max}^{ESS}]$, satisfying (3) and (6), respectively. The above architecture is implemented using Pytorch [41] in Python. Table I reports the parameters' values considered for each agent's training process, which lasts for about 5 hours being executed on a computer with an Intel Core i7 processor at 2.3 GHz and 8 GB RAM. It should also be noted that the values for the DNNs' parameters, as well as for the SoC of the ESS at the beginning of every day (SoC_0) have been selected after a parameters' tuning process, using as a criterion the overall cost of the training days.

The smart home's energy management, in real-time, is accomplished through Algorithm 3, which requires about 1 second to implement steps 1-4 and several milliseconds for

TABLE II: Assignment of test days to the price and temperature clusters

	High Price	Low Price	High Temp.	Low Temp.
September	1, 2, 5, 6, 7, **8**, 9, 12, 13, 14, 15, 16, 19, 20, 21, 22, 23, 26, 27, 28, 29, 30	3, 4, 10, 11, 17, 18, 24, 25	All days	None
November	All days	None	None	All days

TABLE III: Overall results

	No Clustering	Clustering	Difference
September			
Electricity Cost (€)	65.35	52.42	24.7 %
Thermal discomfort (^oC)	3.06	2.84	7.6 %
Total Cost (€)	68.11	54.98	23.9 %
November			
Electricity Cost (€)	290.29	283.4	2.4 %
Thermal discomfort (^oC)	48.54	2.41	1914 %
Total Cost (€)	333.98	285.57	16.9 %

executing steps 6 and 7. According to Algorithm 3, every day in the test set is assigned, before the beginning of the scheduling horizon, to one of the predetermined price and temperature clusters. An LSTM network (line 1) is used for this purpose that predicts the day's price and temperature curves based on the previous week's data i.e. $G = 7$. The LSTM network's architecture is implemented in Keras [42], and includes an input layer of $G{\cdot}T = 7{\cdot}24$ neurons, which correspond to the past values of the curve that is to be predicted (price or temperature). The input layer is fed to an LSTM layer of 200 units, which in turn is connected to a conventional neural network layer of 100 neurons. Finally, the output layer contains $T = 24$ neurons that represent the desirable predicted curve (next day's price or temperature). Table II summarizes the classification of test days to the derived price and temperature clusters. To confirm the effectiveness of the forecast model, we have also made the assignment by using the actual curves of the test days. It has been found that only the 8^{th} day of September is misclassified.

Based on Table II, the HP-HT agent is loaded to the EMS for the energy management of 22 September days, and the LP-HT agent for the remaining 8 days. For November, only the HP-LT agent is used because all days belong to the same price and temperature cluster. Our clustering-based method is compared with state-of-the-art approaches where no clustering is applied, such as in [35], where the data of two months before the beginning of the test period are used for training (i.e. a single agent is trained for September based on the data of July-August and another agent is trained for November based on the data of September-October).

Table III compares the results of the two approaches for the two test periods. In both cases, the proposed method performs better; the total cost is by 23.9% and 16.9% lower for September and November, respectively. However, the reason for our method's superiority is different over the two test periods. In September, both approaches achieve similar levels of thermal comfort, but the electricity cost is 24.7% higher under the no-clustering case. This is mainly because the majority of days (49/62) in the training set, when this method is applied, belong to the low price level, while the majority of days in the test set (21/30) belong to the high price cluster. The opposite outcomes are observed in November, where although the electricity costs are comparable, the thermal discomfort index is noticeably higher under the no-clustering case because the majority of days (56/61) in the training set (September-October) belong to the high-temperature level, while all November days belong to the low-temperature cluster. The aforementioned issues are treated by our clustering-based approach, which increases the degree of similarity between the training data and the test data, achieving more efficient energy management.

Next, we compare the power scheduling during two September days that belong to different price clusters. Figs. 1 and 2 show the outdoor temperatures and the electricity prices, respectively, for the examined days. The ESS's power P_t^{ESS}, the HVAC system's power P_t^{HVAC}, the not-controllable load minus the PV power $P_t^L - P_t^{PV}$, as well as the power P_t^G exchanged with the main grid on the 4^{th} of September, when the power scheduling is performed by the LP-HT agent is presented in Fig. 3. The ESS is mainly charged during 07:00-15:00 taking advantage of the high PV production ($P_t^L - P_t^{PV} \leq 0$). The stored energy is later utilized to cover part of the load during 18:00-24:00 when the prices are higher than most part of the rest of the day. A different scheduling pattern is observed on the 9^{th} of September (Fig. 4), which is determined by the HP-HT agent. In this case, the ESS is initially charged up to a significant level during the low price period 03:00-06:00, and offers the stored energy back to the load during 06:00-08:00, when the prices show a sharp rise and the load shows a morning peak. The ESS is then recharged at no cost during 09:00-16:00, taking advantage of the energy excess (the not-controllable load plus the HVAC power are lower than the PV power). The stored energy is transferred to the load during 17:00-24:00, which coincides with the second peak price interval of that day.

The results of Figs. 3. and 4, in terms of the ESS's scheduling, can be generalized. Figs. 5 and 6 present the ESS's SoC for the days of September that are assigned to the LP-HT agent, as well as for the days of September that

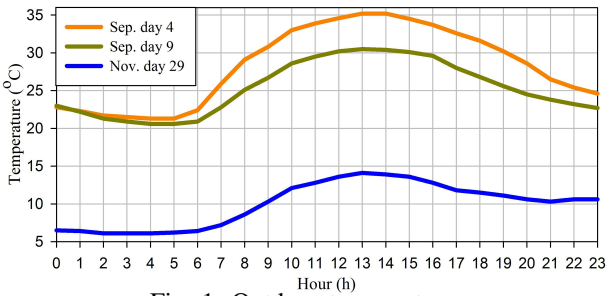

Fig. 1: Outdoor temperatures

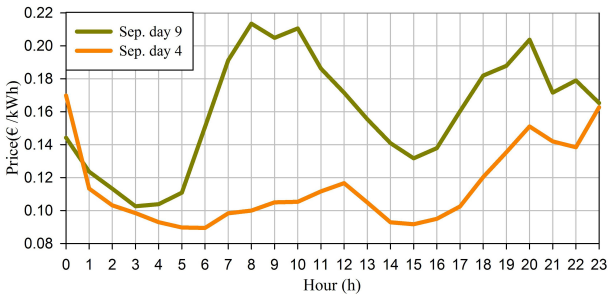

Fig. 2: Real-time prices

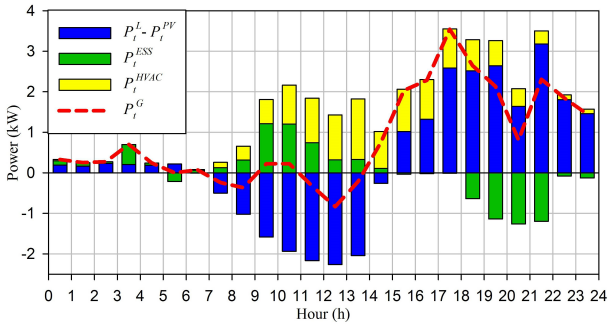

Fig. 3: Power scheduling for September's day 4 (LP-HT agent)

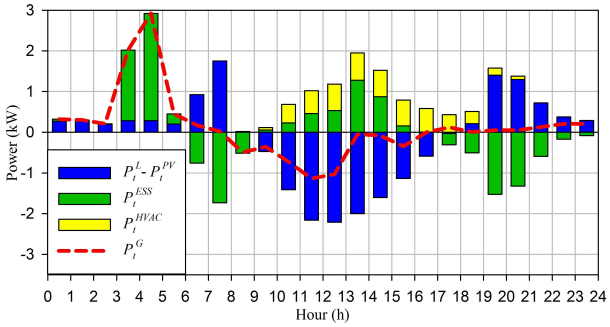

Fig. 4: Power scheduling for September's day 9 (HP-HT agent)

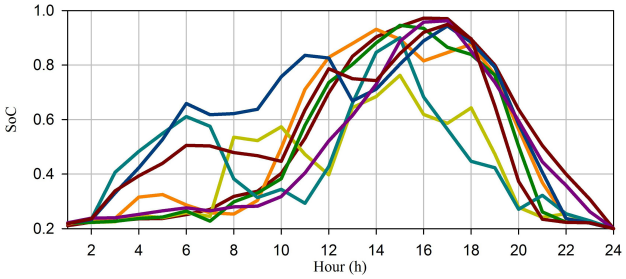

Fig. 5: ESS's schedules for September (LP-HT agent)

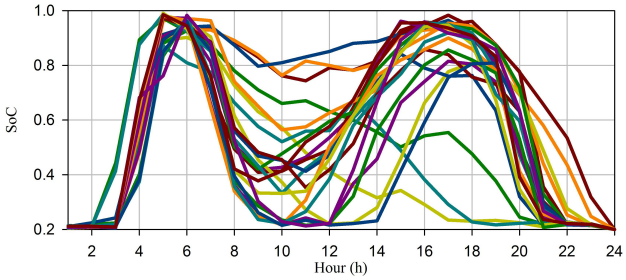

Fig. 6: ESS's schedules for September (HP-HT agent)

are assigned to the HP-HT agent, respectively, based on Table II. As Fig. 5 denotes, for the 8 days that are assigned to the LP-HT agent, the ESS is mainly charged during afternoon and discharged in the evening. On the other hand, for the 22 days that are assigned to the HP-HT agent, the ESS's schedules in Fig. 6 show an additional early-morning charging followed by a morning discharging. The flexibility of obtaining different ESS's schedules depending on the test day's expected price levels is the main reason why the proposed method achieves lower electricity costs (by 24.7% according to Table III) than the no-clustering candidate, in September. The performance gap between the two methods is higher in days that belong to the high price cluster, such as day 9. As mentioned earlier, this is because the majority of days (49/62) in the training set of the no-clustering approach are characterized by low price levels. In November, all days are assigned to the same agent (HP-LT), while the majority of training days (45/61), when the no-clustering approach is applied, are characterized by high price levels. For this reason, the performance gap between the two methods, in terms of electricity cost, is low (2.4% according to Table III).

In Fig. 7 we present the power scheduling on the 9^{th} of September under the no-clustering approach, in order to compare it with the corresponding clustering-based scheduling of Fig. 4. The total benefit obtained from the ESS's utilization under the no-clustering case is 0.69 € (0.89 € are saved through the ESS's discharging in the intervals 06:00-09:00 and 16:00-24:00, while 0.2 € are spent for charging the ESS during 00:00-06:00). On the other hand, the total benefit from the ESS's usage under the clustering-based approach is 0.85 € (1.34 € are saved through discharging the ESS in the intervals 06:00-09:00 and 17:00-24:00, while 0.49 € are spent for charging it during 03:00-06:00). Moreover, the clustering-based approach achieves a higher profit from energy exports and a cheaper operating cost for the HVAC system. During 09:00-17:00 there is energy excess, which under both approaches is used for charging the ESS and operating the HVAC system at no cost. Over the same period an amount of energy is exported to the grid; the exports' profit is by 0.07 € higher under the clustering-based method, while the total cost for the HVAC system's operation is by 0.07 € lower. In summation, the smart home's electricity cost is 0.3 € lower under our proposed method (0.43 €) compared to the no-clustering method's cost (0.73 €), i.e., a difference of 70%.

Opposite to the electricity cost, both methods achieve satisfying levels of thermal comfort in September, according to Table III. Fig. 8 indicates that the indoor temperature lies within the acceptable limits ($\Theta_{min} = 20^oC$, $\Theta_{max} = 24^oC$) under the two methods on the 4^{th} of September, which is the warmest day in the test set. However, the no-clustering approach fails to achieve satisfying thermal comfort in November. As mentioned earlier, this is because the majority of days in the training set (56/61), when this method is applied, are characterized by high temperature levels, while all November days belong to the low-temperature cluster. As Fig. 9 shows, for the 29^{th} of November, which is the coldest day in the test set, there is a lack of thermal comfort under the no-clustering approach, especially during the morning hours when the outdoor temperatures are very low. On the other hand, the indoor temperature is kept within the acceptable values under our clustering-based approach.

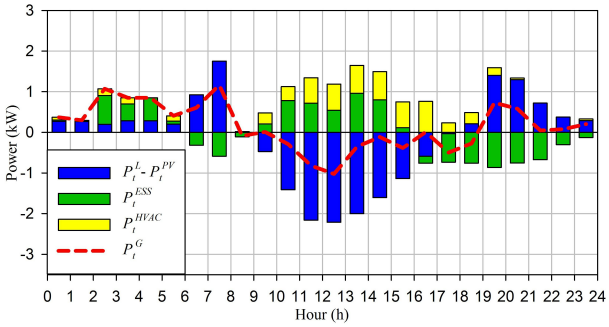

Fig. 7: Power scheduling for September's day 9 (no-clustering)

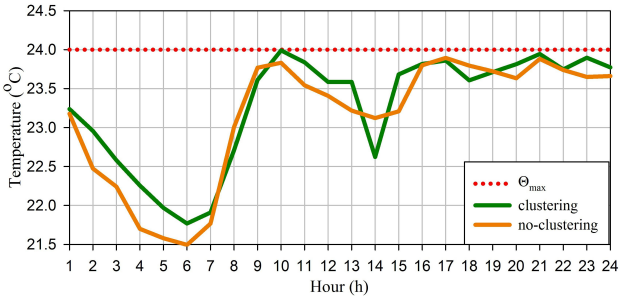

Fig. 8: Indoor temperature for September's day 4

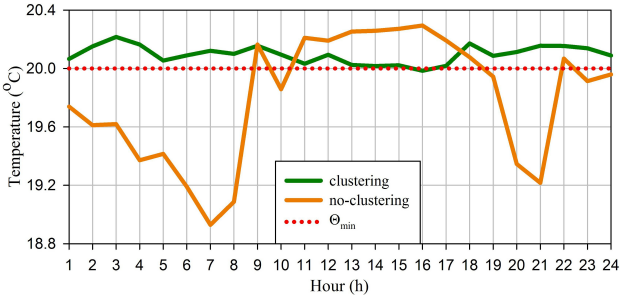

Fig. 9: Indoor temperature for November's day 29

V. Conclusion

The objective of the proposed energy management scheme is to minimize the electricity cost and the thermal discomfort in a smart home by selecting appropriate set points for the ESS's and HVAC system's power in real-time. The problem is formulated as a Markov decision process, and it is solved using the DDPG algorithm, which is compatible with continuous state and action spaces. The main advantage of our clustering-based method compared to existing RL-based energy management schemes is that it increases the similarity between the training and the test data, and by doing so achieves more efficient schedules for the controllable devices.

In our future work, we are planning to extend our approach, and apply it to a system of cooperative microgrids that will be equipped with sources of electrical and thermal energy, such as combined heat and power units, renewable energy sources, and controllable loads. In such a system, the objective will not only be to optimize the schedules of the microgrids' components but also the exchanges of electrical and thermal energy among them. Moreover, we are planning to use and evaluate the efficiency of other continuous RL algorithms, such as the Soft Actor Critic (SAC) and the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithms.

Acknowledgement

This work has been created in the context of the PROGRESSUS project. This project has received funding from the Electronic Components and Systems for European Leadership Joint Undertaking under grant agreement No 876868.

References

[1] G. Bedi, G. K. Venayagamoorthye, R. Singh, R. Brooks, and K. C. Wang, "Review of Internet of Things (IoT) in electric power and energy system," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 847–870, Feb. 2018.

[2] I. Zenginis et al., "Optimal power equipment sizing and management for cooperative buildings in microgrids," *IEEE Trans. Ind. Inform.*, vol. 15, no. 1, pp. 158–172, Jan. 2019.

[3] J. S. Vardakas et al., "Electricity savings through efficient cooperation of urban buildings: the smart community case of Superblocks in Barcelona," *IEEE Com. Mag.*, vol. 56, no. 11, pp. 102–109, Nov. 2018.

[4] I. Zenginis et al., "Cooperation in microgrids through power exchange: An optimal sizing and operation approach," *Appl. Energy.*, vol. 203, pp. 972–981, Oct. 2017.

[5] D. Minoli, K. Sohraby, and B. Occhiogrosso, "IoT considerations, requirements, and architectures for smart buildings – Energy optimization and next-generation building management systems," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 269–283, Jan. 2017.

[6] N. E. Koltsaklis, M. Giannakakis, and M. C. Georgiadis, "Optimal energy planning and scheduling of microgrids," *Chem. Eng. Res. and Des.*, vol. 131, pp. 318–332, March 2018.

[7] C. Keerthisinghe, A. Chapman, and G Verbič, "PV and demand models for a Markov decision process formulation of the home energy management problem," *IEEE Trans. Ind. Elec.*, vol. 66, no. 2, pp. 1424–1433, Feb. 2019.

[8] X. Luo, J. Wang, M. Dooner, and J. Clarke "Overview of current development in electrical energy storage technologies and the application potential in power system operation," *Appl. Energy.*, vol. 137, pp. 511–536, Jan. 2015.

[9] K. Bradbury, L. Pratson, and D. Patino-Echeverri "Economic viability of energy storage systems based on price arbitrage potential in real-time US electricity markets," *Appl. Energy.*, vol. 114, pp. 512–519, Feb. 2014.

[10] J. S. Vardakas, N. Zorba, and C. V. Verikoukis, "A survey on demand response programs in smart grids: Pricing methods and optimization algorithms," *IEEE Comm. Surveys & Tutorials*, vol. 17, no. 1, pp. 152–178, July 2014.

[11] A. Afram, and F. Janabi-Sharifi, "Effects of dead-band and set-point settings of on/off controllers on the energy consumption and equipment switching frequency of a residential HVAC system," *Journal of Process Control*, vol. 47, pp. 161-174, Nov. 2016.

[12] Y. J. Kim, "Optimal price based demand response of HVAC systems in multizone office buildings considering thermal preferences of individual occupants buildings," *IEEE Trans. Ind. Inform.*, vol. 14, no. 11, pp. 5060–5073, Jan. 2018.

[13] A. Vishwanath, V. Chandan, and K. Saurav, "An IoT-based data driven precooling solution for electricity cost savings in commercial buildings," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 7337–7347, Feb. 2019.

[14] G. Mantovani, and L. Ferrarini, "Temperature control of a commercial building with model predictive control techniques," *IEEE Trans. Ind. Elec.*, vol. 62, no. 4, pp. 2651–2660, Dec. 2014.

[15] Y. Ma, J. Matusko, and F. Borrelli, "Stochastic model predictive control for building HVAC systems: Complexity and conservatism," *IEEE Trans. Control Syst. Tech.*, vol. 23, no. 1, pp. 101–116, Apr. 2014.

[16] L. Langer, and T. Volling, "An optimal home energy management system for modulating heat pumps and photovoltaic systems," *Appl. Energy*, vol. 278, Nov. 2020.

[17] N. Koltsaklis, I. P. Panapakidis, D. Pozo, and G. C. Christoforidis, "A Prosumer Model Based on Smart Home Energy Management and Forecasting Techniques," *Energies*, vol. 14, no. 6, Jan. 2021.

[18] C. Keerthisinghe, G Verbič , and A. Chapman, "A fast technique for smart home management: ADP with temporal difference learning," *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 3291–3303, July 2018.

[19] H. Shuai et al., "Stochastic optimization of economic dispatch for microgrid based on approximate dynamic programming," *IEEE Trans. Smart Grid*, vol. 10, no. 3, pp. 2440–2452, Jan. 2018.

[20] H. Shuai, J. Fang, X. Ai, J. Wen, and H. He, "Optimal real-time operation strategy for microgrid: An ADP-based stochastic nonlinear optimization approach," *IEEE Trans. Sustain. Energy*, vol. 10, no. 2, pp. 931–942, July 2018.

[21] L. Lei et al., "Deep reinforcement learning for autonomous internet of things: Model, applications and challenges," *IEEE Comm. Surveys & Tutorials*, vol. 22, no. 3, pp. 1722–1760, Apr. 2020.

[22] S. Kim, and H. Kim, "Reinforcement learning based energy management algorithm for smart energy buildings," *Energies*, vol. 11, no. 8, Aug. 2018.

[23] R. Lu, S. H. Hong, and M. Yu, "Demand response for home energy management using reinforcement learning and artificial neural network," *IEEE Trans. Smart Grid*, vol. 10, no. 6, pp. 6629–6639, Apr. 2019.

[24] X. Xu et al., "A multi-agent reinforcement learning-based data-driven method for home energy management," *IEEE Trans. Smart Grid*, vol. 11, no. 4, pp. 3201–3211, Feb. 2020.

[25] E. Mocanu et al., "On-Line Building Energy Optimization Using Deep Reinforcement Learning," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 3698–3708, July 2019.

[26] P. Lissa et al., "Deep reinforcement learning for home energy management system control," *Energy and AI*, vol. 3, March 2021.

[27] Y. Ji, J. Wang, J. Xu, X. Fang, and H. Zhang, "Real-time energy management of a microgrid using deep reinforcement learning," *Energies*, vol. 12, no. 12, Jan. 2019.

[28] V. H. Bui, A. Hussain, and H. M. Kim, "Double deep Q-learning-based distributed operation of battery energy storage system considering uncertainties," *IEEE Trans. Smart Grid*, vol. 11, no. 1, pp. 457–469, June 2019.
[29] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015
[30] Y. Du et al., "Intelligent multi-zone residential HVAC control strategy based on deep reinforcement learning," *Appl. Energy*, vol. 281, Nov. 2021.
[31] G. Gao, J. Li, and Y. Wen, "DeepComfort: Energy-Efficient Thermal Comfort Control in Buildings via Reinforcement Learning," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8472–8484, May 2020.
[32] H. Li, Z. Wan, and H. He, "Real-time residential demand response," *IEEE Trans. Smart Grid*, vol. 11, no. 5, pp. 4144–4154, March 2020.
[33] L. Lei, Y. Tan, G. Dahlenburg, W. Xiang, and K. Zheng, "Dynamic Energy Dispatch Based on Deep Reinforcement Learning in IoT-Driven Smart Isolated Microgrids," *IEEE Internet of Things Journal*, vol. 8, no. 10, pp. 7938–7953, May 2021.
[34] Y. Ye, D. Qiu, X. Wu, G. Strbac, and J. Ward, "Model-free real-time autonomous control for a residential multi-energy system using deep reinforcement learning," *IEEE Trans. Smart Grid*, vol. 11, no. 4, pp. 3068–3082, Feb. 2020.
[35] L. Yu et al., "Deep reinforcement learning for smart home energy management," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2751–2762, Dec. 2019.
[36] R. Deng, Z. Zhang, J. Ren and H. Liang, "Indoor temperature control of cost-effective smart buildings via real-time smart grid communications," *Proc. IEEE Globecom*, pp. 1–6, Feb. 2016.
[37] J. Brownlee, "Deep learning for time series forecasting: predict the future with MLPs, CNNs and LSTMs in Python, "Machine Learning Mastery, 2018.
[38] OPSD. [Online]. Available: https://data.open-power-system-data.org/household_data/, acc. Dec. 13, 2021
[39] Zenodo. [Online]. Available: https://sandbox.zenodo.org/record/632147#.YMuWub4zaUm, acc. Dec. 13, 2021
[40] SOLARGIS. [Online]. Available: https://solargis.com/products/evaluate/useful-resources, acc. Dec. 13, 2021
[41] PyTorch. [Online]. Available: https://pytorch.org/, acc. Dec. 13, 2021
[42] Keras. [Online]. Available: https://keras.io/, acc. Dec. 13, 2021