On-Line Building Energy Optimization Using Deep Reinforcement Learning

Elena Mocanu, Decebal Constantin Mocanu, Phuong H. Nguyen, *Member, IEEE*, Antonio Liotta, *Senior Member, IEEE*, Michael E. Webber, *Member, IEEE*, Madeleine Gibescu, *Member, IEEE*, and J. G. Slootweg, *Member, IEEE*

***Abstract*—Unprecedented high volumes of data are becoming available with the growth of the advanced metering infrastructure. These are expected to benefit planning and operation of the future power systems and to help customers transition from a passive to an active role. In this paper, we explore for the first time in the smart grid context the benefits of using deep reinforcement learning, a hybrid type of methods that combines reinforcement learning with deep learning, to perform on-line optimization of schedules for building energy management systems. The learning procedure was explored using two methods, Deep Q-learning and deep policy gradient, both of which have been extended to perform multiple actions simultaneously. The proposed approach was validated on the large-scale Pecan Street Inc. database. This highly dimensional database includes information about photovoltaic power generation, electric vehicles and buildings appliances. Moreover, these on-line energy scheduling strategies could be used to provide real-time feedback to consumers to encourage more efficient use of electricity.**

***Index Terms*—Deep reinforcement learning, demand response, deep neural networks, smart grid, strategic optimization.**

I. Introduction

THERE is an energy transition underway since the start of the millennium, comprised primarily of a push towards replacing large, fossil-fuel plants with renewable and distributed generation. It results in increased uncertainty and complexity in both the business transactions and in the physical flows of electricity in the smart grid. Because the built environment is the largest user of electricity, a deeper look at building energy consumption holds a promise for improving energy efficiency and sustainability. Understanding such individual consumption behavior based on the knowledge transfer from the fusion of extensive data collected from the Advanced Metering Infrastructure (AMI) is an essential step to optimize building energy consumption and consequently the effects of its use.

This work is motivated by the hypothesis that an optimal resource allocation of end-user patterns based on daily smart electrical device profiles could be used to smoothly reconcile differences in future energy consumption patterns and the supply of variable sources such as wind and solar [1]–[3]. It is expected that a cost minimization problem could be solved to activate real-time price responsive behavior [4]. A wide-range of methods have been proposed to solve the building energy and cost optimization problems, including linear and dynamic programing, heuristic methods such as Particle Swarm Optimization (PSO), game theory, fuzzy methods and so on [1]–[7]. Therein, both centralized and decentralized solutions exist, but they fail to consider on-line solutions for large-scale, real databases [6]. More concretely, any time when an optimization is needed, these methods have to compute completely or partially all the possible solutions and to choose the best one. This procedure is time consuming. In the big data era, more and more machine learning methods appear to be suitable to overcome this limitation by automatically extracting, controlling and optimizing the electrical patterns. This can be done by performing successive transformation of the historical data to learn powerful machine learning models to cope with the high uncertainty of the electrical patterns. Then, these models will be capable of generalization and they could be exploited in an on-line manner (i.e., few milliseconds) to minimize the cost or the energy consumption in newly encountered situations. Among all these machine learning models, the ones belonging to the Reinforcement Learning (RL) area are the most suitable for the cost minimization problem, as they are capable to learn an optimal behavior, while the global optimum is not known.

Thus, in the remaining of this paper we focus on RL methods, such as Q-learning [8], and their latest developments. The

Manuscript received June 7, 2017; revised November 18, 2017 and February 18, 2018; accepted April 17, 2018. Date of publication May 8, 2018; date of current version June 19, 2019. This work was supported in part by NL Enterprise Agency through the TKI SG-BEMS project of Dutch Top Sector and in part by the European Union's Horizon 2020 project INTER-IoT under Grant 687283. Paper no. TSG-00805-2017. *(Corresponding author: Elena Mocanu.)*

E. Mocanu is with the Department of Electrical Engineering, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands, and also with the Department of Mechanical Engineering, Eindhoven University of Technology, Eindhoven, 5600 MB, The Netherlands (e-mail: e.mocanu@tue.nl).

D. C. Mocanu is with the Department of Mathematics and Computer Science, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands (e-mail: d.c.mocanu@tue.nl).

P. H. Nguyen, M. Gibescu, and J. G. Slootweg are with the Department of Electrical Engineering, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands (e-mail: p.nguyen.hong@tue.nl; m.gibescu@tue.nl; j.g.slootweg@tue.nl).

A. Liotta is with the Data Science Centre, University of Derby, Derby DE1 3HD, U.K. (e-mail: a.liotta@derby.ac.uk).

M. E. Webber is with the Department of Mechanical Engineering, The University of Texas at Austin, Austin, TX 78712-1591, USA (e-mail: webber@mail.utexas.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSG.2018.2834219

building environment is modeled using a Markov Decision Process [9] and it can be used to find the best long-term strategies. Prior studies showed that RL methods are able to solve stochastic optimal control problems [10] in the power system area as well as an energy consumption scheduling problem [11] with dynamic pricing [12]. A batch reinforcement learning method was introduced in [13] to schedule a cluster of domestic electric water heaters, and further on applied for smart home energy management [14]. Owing to the curse of dimensionality, these methods fail for large-scale problems. More recently, there has been a revival of interest in combining deep learning with reinforcement learning. Lately, in 2015, an application of Q-learning to deep learning has been successful at playing Atari2600 games at expert human levels [15]. In 2016, another one has defeated for the first time in history the world champion at the game of Go. Complementary with our work, François-Lavet *et al.* [16] has proposed the use of Deep Q-learning for storage scheduling in microgrids. The above methods represent the starting point of a new research area, known as Deep Reinforcement Learning (DRL), which has evolved through the intersection of reinforcement learning and neural networks. At the same time, in our previous work, we showed that Reinforcement Learning using Deep Belief Networks for continuous states estimation can successfully perform unsupervised energy prediction [17].

Our Contribution: In this paper, inspired by the above research developments, we propose for the first time the use of the Deep Policy Gradient method, as part of Deep Reinforcement Learning algorithms, in the large-scale physical context of smart grid - smart building, as follows.

- We propose for the first time an approach to optimize directly on-line the building energy consumption and the cost.
- We propose a new way to adapt DRL algorithms to the smart grid context, with the aim of conceiving a fast algorithm to learn the electrical patterns and to optimize on-line either the building energy consumption or the cost.
- We investigate two DRL algorithms, namely Deep Q-learning (DQN) [15] and Deep Policy Gradient (DPG).
- DPG in its current state-of-the-art form is capable to take just one action at a specific time. As in the building context multiple actions have to be taken at the same moment, we propose a novel gradient method to enhance DPG with the capability of handling multiple actions simultaneously.

We evaluate our proposed methods on the PecanStreet database at both the building and aggregated level. In the end, we prove that our proposed methods are able to efficiently cope with the inherent uncertainty and variability in the generation of renewable energy, as well as in the peoples' behavior related with their use of electricity (i.e., charging of electric vehicles). Specifically, we show that the enhanced DPG is more appropriate to solve peak reductions and cost minimization problems than DQN.

The remaining of this paper is organized as follows. Section II describes the problem formulation and in Section III we introduce the background and preliminary concepts. Section IV describes our proposed method followed by implementation details in Section V. Results and discussions are provided in Section VI. Finally, we conclude with some directions for future research.

II. Problem Formulation

In this context, we aim to reduce load peaks as well as to minimize the cost of energy.[1] Let $\mathcal{B}$ denote the set of buildings, such that $B_i \in \mathcal{B}, \forall i \in \mathbb{N}$ representing the index of the building analyzed. The total building energy consumption E_i is a sum over all power generation P^+ and consumption in a specific interval of time Δt. Therein, based on the shifting capabilities of appliances present in a building we differentiate between flexible power P_d^-, e.g., electric devices $d \in \{1, \ldots, m_i\}$, and fixed consumption P^-.

A. Cost Minimization Problem

In this paper, we assume two price components over the space of $\mathcal{B}$, such that λ_t^- is the price value set by the utility company for the time-slot t and λ_t^+ represents the price value at which the utility company buys energy from end-users at time-slot t. Therefore, the optimal cost associated with customer i at time t for an optimization time horizon T can be calculated as

$$\min \quad \sum_{t=1}^{T}\left(\lambda_t^+ \sum_{i=1}^{n} P_{i,t}^+ - \lambda_t^- \sum_{i=1}^{n}\left(P_{i,t}^- + \sum_{d=1}^{m_i} a_{i,d,t} P_{i,d,t}^-\right)\right) \tag{1}$$

$$\text{s.t.} \quad \sum_{t=1}^{T} P_i^- \Delta t = E_i, \ \forall i \in \mathbb{N}, \forall t \in \mathbb{N}, \tag{2}$$

$$\sum_{t=1}^{T} P_d^- \Delta t = E_d, \ \forall d \in \mathbb{N}, \forall t \in \mathbb{N}, \tag{3}$$

$$a_{i,d,t} = \{1, 0\}, \forall a \in \mathcal{A}, \forall i \in \mathbb{N}, \forall d \in \mathbb{N}, \forall t \in \mathbb{N}, \tag{4}$$

$$P_{i,t}^+, P_{i,t}^-, P_{i,d,t}^- \geq 0, \forall t = [1{:}T] \in \mathbb{N}, \tag{5}$$

$$\lambda_t^+, \lambda_t^- \geq 0, \forall t = [1{:}T] \in \mathbb{N}. \tag{6}$$

where $a_{i,d,t} = 1$ if the electrical device is *on* at that specific moment in time, and 0 otherwise. Please note that, in our proposed method, computing $a_{i,d,t}$ is equivalent with the estimation of the actions (see Fig. 1).

B. Peak Reduction Problem

In the special case of constant price, for electricity generation and consumption, with $\lambda_t^+ = \lambda_t^-$, the cost minimization problem becomes a peak reduction problem, defined as

$$\min \quad \sum_{t=1}^{T}\left(\sum_{i=1}^{n} P_{i,t}^+ - \sum_{i=1}^{n}\left(P_{i,t}^- + \sum_{d=1}^{m_i} a_{i,d,t} P_{i,d,t}^-\right)\right). \tag{7}$$

Consequently, the constraints following Eq. (1) will remain valid for both problems. However, based on the differences between different types of electrical devices the full range of constraints is larger as explained in the next sections.

[1] Please note that the main symbols and notation used in this paper are summarized in Table V.

C. Electrical Device Constrains

We are assuming three types of consumption profiles. Firstly, we consider the *time-scaling load*. In respect to this we confine our analysis to the air conditioning load (d_{AC}), as a representative part of a larger set of electrical devices in every building which could be switched on-off for a limited number of times during an optimization horizon, e.g., lights, television, refrigerator. Prior studies show that short-term air conditioning curtailments have a negligible effect on end-user comfort [18]. Secondly, we include the *time-shifting load*, also called deferrable load, that must consume a minimum amount of power over a given time interval. Therein, we model the dishwasher (d_{DW}) as an uninterruptible load, which requires a number of consecutive time steps. Finally the electric vehicle (d_{EV}) was considered as both a *time scaling and shifting load*. A more rigorous formulation of the building electrical components and their associated constrains could be found in [19]. In our case, a complementary probabilistic perspective over the time dependent devices constrains $a_{d,t}$ give us the following assumptions:

A1: For all d, with P_d^- time-scaling loads, there $\exists \delta_d \in \mathbb{R}_+$ constants over the optimization horizon such that

$$\begin{cases} \sum_t P_d^- \leq \delta_d & \text{if } p\left(P_d^- = 0|t\right) \in (0,1] \\ \sum_t P_d^- = \delta_d & \text{if } p\left(P_d^- = 0|t\right) = 0 \end{cases} \tag{8}$$

where $p(P_d^- = 0|t)$ is the probability of the electrical device d to be active at any moment in time t, for all $t = [1{:}T] \in \mathbb{N}$.

A2: For all d, with P_d^- time-shifting loads, there $\exists \delta_d$ constants such that $\sum_t P_d = \delta_d$, for all $t = [1{:}T] \in \mathbb{N}$.

Observation 1: In this paper, P^+ (e.g., PV generation) is considered a non-curtailable resource.

Observation 2: All electrical vehicles, d_{EV}, and their associated consumption P_d^-, were considered as time scaling and shifting loads working under the conditions imposed by Assumption 1 and 2.

III. Background and Preliminaries

In this section, we provide a brief overview of reinforcement learning, Markov decision formalism, and deep neural networks.

A. Reinforcement Learning

In a Reinforcement Learning (RL) [9] context, an agent learns to act using a (Partial Observable) Markov Decision Process (MDP) formalism. MDPs are defined by a 4-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}.(\cdot,\cdot), \mathcal{R}.(\cdot,\cdot) \rangle$, where:

- $\mathcal{S}$ is the state space, $\forall s \in \mathcal{S}$,
- $\mathcal{A}$ is the action space, $\forall a \in \mathcal{A}$,
- $\mathcal{T}{:}\mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0,1]$ is the transition function given by the probability that by choosing action a in state s at time t, the system will arrive at state s' at time $t+1$, such that $p_a(s,s') = p(s_{t+1} = s'|s_t = s, a_t = a)$, and
- $\mathcal{R}{:}\mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ is the reward function, were $\mathcal{R}_a(s,s')$ is the immediate reward received by the agent after it performs the transition to state s' from state s.

The agent aims to optimize a stochastic policy $\pi{:}\mathcal{S} \times \mathcal{A} \times \mathcal{R} \rightarrow \mathbb{R}_+$. Under structure assumption of the environment (i.e., finite states and actions) the Markov decision problem is typically solved using dynamic programing. However, in our built environment, the model has a large (continuous) states space. Therein, the state space is given by the building energy consumption and price at every moment in time, while the action space is highly dependent on the *electric device constrains*. The success of every action a is measured by a reward r. Learning to act in an environment will make the agent to choose actions to maximize future rewards. The value function $Q^\pi(s,a)$ is an expected total reward in state s using action a under a policy π. Currently, one of the most popular reinforcement learning algorithm is Q-learning [8].

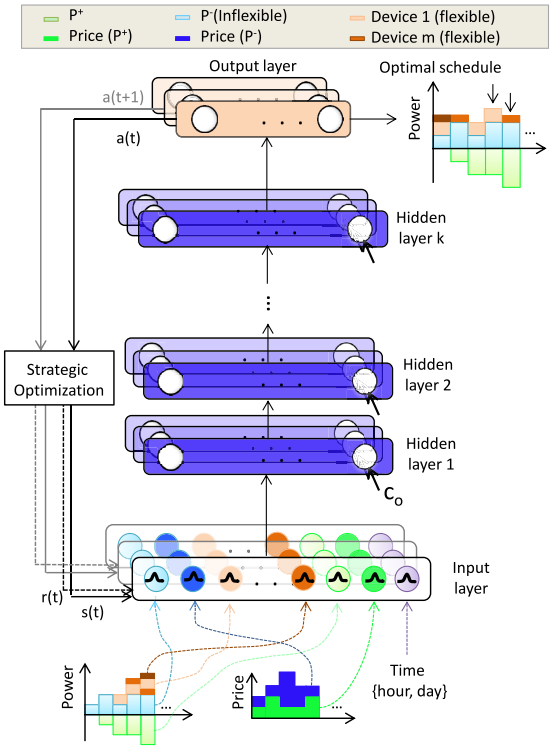


Fig. 1. The closed loop general architecture of Deep Reinforcement Learning, built as a combination of Reinforcement Learning and Deep Neural Network.

B. Deep Neural Networks

The topology of a Deep Neural Network (DNN) architecture is based on multiple layers of neurons. In general, a neuron is a non-linear transformation of the linear sum of its inputs. The first layer models directly the data. A hidden layer in the neural network architecture is build as an array of neurons taking the inputs from the previous layer. The activation function of a neuron on top of k stacked layers in the architecture is using composite functions, such as $x \otimes h_1 \otimes h_2 \otimes \cdots \otimes h_k$. In 2011, it was shown that supervised training of a very deep neural network with hard non-linearities is faster if the hidden layers are composed of Rectified Linear Units (ReLU) [20]. Recently, the logistic sigmoid and the hyperbolic tangent activation are outperformed by ReLU [15], [21], [22]. Formally, ReLU is defined as a function $f(x_i) = max(0, x_i)$, where x_i is its input. However, to avoid a non-zero gradient when the

hidden units are not active, we used a slightly relaxed form proposed in [23], given by

$$f(x_i) = \begin{cases} x_i & \text{if } x_i > 0, \forall i \in \mathbb{N} \\ \eta x_i & \text{if } x_i \leq 0, \forall i \in \mathbb{N} \end{cases} \quad (9)$$

where η is a coefficient controlling the slope of the negative part. If $\eta = 0$ then Eq. (9) becames ReLU.

Still, deep learning methods applied to the power system are in an incipient phase. In 2014, Conditional Restricted Boltzmann Machine (CRBM) were used to increase the energy prediction accuracy [24]. Later on, for the energy prediction problem various deep learning methods have been successfully explored and extended [25]–[27]. For example, Ryu *et al.* [28] used the Long Short-Term Memory (LSTM) method for building energy prediction. Another paper related is the work of Marino *et al.* [29], which is followed by the work of Manic *et al.* [30]. Further on, the Multilayer Perceptron (MLP) enhanced with deep learning capabilities was proposed in [31]. Possible benefits of deep learning methods in terms of superior accuracy start to be investigated also for other types of signals in smart grids, such as wind forecast [32], as well as for event detection (classification) problem [33]. One special case of using a DNN is in deep reinforcement learning where the input is given by the states of an MDP and the output represents the actions of the MDP.

IV. Proposed Method

In this section, we propose the use of Deep Reinforcement Learning (DRL) as an on-line method to perform optimal building resource allocation at different levels of aggregation. The general architecture of our proposed method is depicted in Fig. 1. DRL (RL combined with DNNs of k hidden layers) can learn to act better than the standard RL by automatically extracting patterns, such as those of electricity consumption. Overall, we can represent the DNN method, from a very general perspective, as a black box model with good generalization capabilities over a given input distribution as follows:

$$\xrightarrow[data]{Input} DNN_{(k)} \xrightarrow[Data\ estimation]{Output}. \quad (10)$$

In the remaining of this section we will introduce two DRL methods, namely Deep Q-learning (DQN) and Deep Policy Gradient (DPG).

$$DRL = \begin{cases} \xrightarrow[states]{Input} DNN_{(k)} \xrightarrow[Q(s,a)]{Output} & \text{Deep Q-learning} \\ \xrightarrow[states]{Input} DNN_{(k)} \xrightarrow[p(a|s)]{Output} & \text{Deep Policy Gradient.} \end{cases}$$

In contrast to value-based methods (e.g., DQN), policy-based model free methods (e.g., DPG) directly parameterize the policy $\pi(a|s;\theta)$ and update the parameters θ by performing, typically approximate, gradient ascent on the expected long-term reward [34].

A. Deep Q-Learning (DQN)

Learning in DRL is done as follows. The DNN is trained with a variant of the Q-learning algorithm, using stochastic gradient descent to update its parameters [15]. Firstly, the value-function from the standard RL algorithm is replaced by a deep Q-network with parameters θ, given by the weights and biases of DNN, such that $Q(s,a,\theta) \approx Q^{\pi}(s,a)$. This approximation is used further to define the objective function by mean-squared error in Q-values

$$\mathcal{L}(\theta) = \mathbb{E}\left[\left(r + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}, \theta) - Q(s_t, a_t, \theta)\right)^2\right]. \quad (11)$$

Leading to the following Q-learning gradient

$$\frac{\partial \mathcal{L}(\theta)}{\partial \theta} = \mathbb{E}\left[\left(r + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}, \theta) - Q(s_t, a_t, \theta)\right)\frac{\partial Q(s_t, a_t, \theta)}{\partial \theta}\right]. \quad (12)$$

Usually, this standard Q-learning algorithm used in synergy with neural networks oscillates or diverges, mainly because data are sequential. To overcome this limitation of correlated data and non-stationary distributions, we use an *experience replay* mechanism which randomly samples previous mini-batch of transitions (s_t, a_t, r_t, s_{t+1}) from the dataset $\mathcal{D}$, and therefore smooths the training distribution over many historical data. It is straightforward to integrate the above Deep RL approach into Eq.1. The binary action vector $a_t \in \mathcal{A}$ is augmented to $\max_{a_{t+1}} Q(s_{t+1}, a_{t+1}, \theta)$ and therefore $\sum_{d=1}^{m} a_t P_{d,t}^{-}$ is optimally controlled. Specifically, rather than enforcing the constraints on the time window required by a specific device d and the comfort of end-users considered in $A1$, Eq.8, our idea is to encapsulate them in the reward function, $r_t(\lambda_t^{+}, \lambda_t^{-}, P_{i,d}^{-})$, which is further detailed in Section IV-A.

B. Deep Policy Gradient (DPG)

Recently, it has been shown that policy gradient methods are able to decrease the time needed for convergence in continuous games [34], [35]. From an architectural perspective, the neurons from the output layer of the DNN (with parameters θ) corresponding to DPG, instead of estimating the $Q(s_t, a, \theta)$, $\forall a \in \mathcal{A}$ as in DQN, they estimate the probability to take action a in a specific state s_t, such as $p(a|s_t, \theta)$, $\forall a \in \mathcal{A}$. This offers a clear advantage to DPG over DQN when there is a need to perform multiple actions simultaneously, as all actions can be sampled and executed simultaneously in the game using their own probability.

In the policy gradient context, the approximate optimization problem defined in Eq. (1) or Eq. (7) is an equivalent of maximizing the total expected reward of a parameterized model under a policy π, as follows

$$\text{maximize} \quad \mathbb{E}_{x\sim p(x|\theta)}[\mathcal{R}|\pi]. \quad (13)$$

In the DPG context, the parameterized model is the DNN. Thus, the DNN becomes a probability density function over its inputs (the game states), i.e., $f(x)$, leading Eq. (13) to the following optimization problem

$$\text{maximize} \quad \mathbb{E}_{x\sim p(x|\theta)}[f(x)] \quad (14)$$

As shown in [36] and [37], the unbiased gradient estimation uses $f(x)$ as a score function yielding

$$\begin{aligned}\nabla_\theta \mathbb{E}_x\big[f(x)\big] &= \nabla_\theta \int \mathrm{dx}\ p(x|\theta) f(x) = \int dx \nabla_\theta p(x|\theta) f(x) \\ &= \int \mathrm{dx}\ p(x|\theta) \frac{\nabla_\theta p(x|\theta)}{p(x|\theta)} f(x) \\ &= \int \mathrm{dx}\ p(x|\theta) \nabla_\theta \log\ p(x|\theta) f(x) \\ &= \mathbb{E}_x\big[f(x) \nabla_\theta \log\ p(x|\theta)\big] \end{aligned} \tag{15}$$

where $\frac{\partial}{\partial\theta} = \nabla_\theta$ denote the first-order partial derivative over the output data. Intuitively, to solve the gradient of Eq. (15), first we have to take samples of $x_i \sim p(x|\theta)$ and to compute the estimated gradient, such that $\hat{g}_i^\theta = f(x_i)\nabla_\theta \log\ p(x_i|\theta)$. Moving in the $\hat{g}_i$ direction increases the log-probability of that particular sample x_i proportional with the reward associated with it, $f(x_i)$. In other words, this practically shows how good is that sample. As in policy gradient the reward is available at the end of a game, these samples are collected in a trajectory, i.e., $\tau = (s_0, a_0, r_0, \ldots, s_{T-1}, a_{T-1}, r_{T-1})$. To compute the gradient of a trajectory, we need to calculate and differentiate the density $p(\tau|\theta)$ with respect to θ as follows:

$$p(\tau|\theta) = p(s_0) \prod_{t=0}^{T-1} \big[\pi(a_t|s_t, \theta) p(s_{t+1}|s_t, a_t)\big]. \tag{16}$$

By taking the log-probability of Eq. (16), we obtain

$$\begin{aligned}\log\ p(\tau|\theta) = \log\ p(s_0) + \sum_{t=0}^{T-1} \big[&\log\ \pi(a_t|s_t, \theta) \\ &+ \log\ p(s_{t+1}|s_t, a_t)\big]. \end{aligned} \tag{17}$$

Taking the derivative of Eq. (17) with respect to θ leads to

$$\frac{\partial}{\partial\theta} \log\ p(\tau|\theta) = \frac{\partial}{\partial\theta} \sum_{t=0}^{T-1} \log\ \pi(a_t|s_t, \theta). \tag{18}$$

Finally, we can write the gradient update $\hat{g}_\tau^\theta$ for parameters θ after considering a trajectory τ as

$$\hat{g}_\tau^\theta \propto \mathcal{R}_\tau \frac{\partial}{\partial\theta} \sum_{t=0}^{T-1} \log\ \pi(a_t|s_t, \theta). \tag{19}$$

V. Implementation Details

A. Learning and Validation

To train the DQN and DPG models we have used as a starting point an off-line database. First, we created the built environment game. In this game for each day we did not alter the base load, but for the assumed flexible loads we considered many possibilities (even if they did not happen in reality). We evaluated each possibility. We gave a positive reward to it if it was close to our optimization goal. If not, we assigned to that possibility a negative reward. At the begin of the learning there are a lot of random choices, but in time (many iterations), the reinforcement learning model converges and will learn to choose just the possibilities which are close to the optimization goal. Finally, we did not optimize the "existing" off-line database, but we obtained an alternative optimized version of it, which would be better if this strategy would have been used in reality.

The main advantage of (deep) reinforcement learning is that after the model is trained completely on such off-line database, it can be exploited on-line in a real environment. If this on-line environment has slightly changed from the "training" environment, the reinforcement learning model can learn by default these changes and adapt dynamically its behavior to achieve the best possible performance.

B. Network Architecture

Aiming to have a fair comparison between DQN and DPG, the architecture of the deep neural networks used is similar for both models and it has the following characteristics. Each reinforcement learning state (encoded in the input layer), is given by a time-window of two consecutive time steps. Thus, in the case of the peak reduction problem the input layer has 11 neurons, i.e., time step t, and base load, PV, AC state, EV and dishwasher at $t-1$ and t. This leads to a very large number of states for one building. Furthermore, these continuous states will linearly increase with the number of buildings.

Please note that with the exception of base load and generation which are fixed, the other state components are given by the dynamically adapted values (the ones obtained during the learning process) and not the initial ones measured by the smart meters. For the cost reduction problem, the input layer has an extra neuron which is used to encode the ToU tariff. The number of layers and the number of neurons in each layer was carefully chosen, by performing a small trial and error procedure. Furthermore, the networks have three layers of hidden neurons, each layer having 100 neurons with Rectifier Linear Units (ReLU) as activation function.

The output layer differs for DQN and DPG. For DQN the output layer has 8 neurons, each neuron representing the Q-value of a combined action. Each combined action is a possible combination of the actions of the three flexible devices,[2] i.e., stop air conditioner (a_1), electric vehicle on/off (a_2), dishwasher on/off (a_3). By contrast, the DPG output layer has just three neurons, each neuron representing a device action. More precisely, it gives the probability to perform the action associated with the flexible device for the specific input state. This is a clear advantage of DPG over DQN as it scales linearly with the number of flexible devices.

Hyper-parameters settings: In all experiments performed, the learning rate is set to $\alpha = 10^{-2}$, the discount factor to $\gamma = 0.99$, and $\eta = 0.01$. We train the models for 5000 episodes, where an episode is composed by 20 randomly chosen days. The weights update is performed after every two episodes. The final policy is kept as the output of the learning process.

C. The Reward Vectors for DRL

Regarding the multi-objective optimization problems solved in this paper, an accurate reward function is computed at the end of the day, instead of at each time step of the day. Thus, we

[2]The number of neurons in the output layer of DQN is exponentially correlated with respect to the number of flexible devices.

derived a simple multiple-task joint reward with three reward components.

Component 1: For all $\tau = (s_t, a_t, r_t)$ the reward vectors will be able to control the actions of the three types of flexible consumption, and therefore the total shiftable and scalable load in a household $\sum_{d=1}^{m} a_t P_{d,t}^{-}$, using differentiated rewards

$$r_{a_1} = \begin{cases} -n_{a_1^+} & \text{if } n_{a_1^+} > 10 \\ \zeta_1 & \text{if } n_{a_1^+} \in [1, 10]; \\ \zeta_2 & \text{if } n_{a_1^+} < 1 \end{cases}$$
$$r_{a_2} = \begin{cases} -4|n_{a_2^t} - n_{a_2^+}| & \text{if } n_{a_2^+} \neq n_{a_2^t}, \quad \forall n_{a_2^t} \in \mathbb{N} \\ n_{a_2^+} & \text{if } n_{a_2^+} = n_{a_2^t}, \quad \forall n_{a_2^t} \in \mathbb{N} \end{cases}$$
$$r_{a_3} = \begin{cases} -n_{a_3^+} & \text{if } n_{a_3^+} > 2 \\ \zeta_1 & \text{if } n_{a_3^+} \in [1, 2] \\ \zeta_2 & \text{if } n_{a_3^+} < 1 \end{cases} \tag{20}$$

where $n_{a_1^+}$, $n_{a_2^+}$, and $n_{a_3^+}$ represent how many times the action corresponding with the flexible device is performed, and $n_{a_2^t}$ is the targeted number of loads per day for the electric car. The choice of ζ_1 and ζ_2 coefficients was based on a trial and error procedure. The obtained values are $\zeta_1 = 40$ and $\zeta_2 = -50$.

Component 2: Controlling the total energy consumption defined in Eq. (7) is done as follows

$$r = \begin{cases} -3\zeta_1 - 1 & \text{if } \max(\tilde{P}^-) > \max(P^-) \\ -3\zeta_2 + 4\big[\max(P^-) - \max(\tilde{P}^-)\big] & \text{otherwise.} \end{cases} \tag{21}$$

Further on, shifting the consumption through the time when there is more generation Eq. (6).

$$r = \begin{cases} \frac{\zeta_1}{2} - \big|min(\tilde{P}^-)\big| & \text{if } \tilde{P}^- < 0 \\ -\frac{\zeta_2}{2} & \text{otherwise.} \end{cases} \tag{22}$$

The control of AC under the A2, Eq. (8) is given by

$$r = \begin{cases} \frac{\zeta_1}{8} + 2\big[max(\tilde{P}_{AC}^-) - max(P_{AC}^-)\big] & \text{if } \tilde{P}^- < 0 \\ -\frac{\zeta_2}{10} & \text{otherwise.} \end{cases} \tag{23}$$

Component 3: Controlling the total cost $\mathcal{C}$, defined in Eq. (1).

$$r = \begin{cases} 5\big|\tilde{\mathcal{C}} - \mathcal{C}\big| & \text{if } \tilde{\mathcal{C}} < \mathcal{C} \\ -3\zeta_1 - 1 & \text{otherwise.} \end{cases} \tag{24}$$

The agent must learn multiple tasks consecutively with the goal of optimizing performance across all previously learned tasks. So, we used for solving Eq. (7) the *Component 1* and *2* of the reward, while for Eq. (1) the *Component 1* and *3*.

The joint reward components could be easily generalized to perform an arbitrary number of tasks. However, the range intervals for $n_{a_1^+}$ and $n_{a_3^+}$ considered in Eq. (20) as well as the *positive* and *negative* coefficients (i.e., ζ_1 and ζ_2) used in Eq. (20)–(24) are dependent on the application. Also in Eq. (20) the range of $n_{a_1^+}$ and $n_{a_3^+}$ may be enlarged if comfort limits are relaxed. Algorithm 1 exemplifies on DPG, how DRL can be implemented. We have implemented both methods, DQN and DPG, in Python.

Algorithm 1 Deep Policy Gradient (DPG) - Estimating Eq. (13)

```
initialize model: hyper-parameters (α, γ, ζ)
initialize model: DNN with random weights θ
initialize game: first state s from a random day
for iteration = 1 to arbitrary number do
    sample actions p(a1, a2, a3|θ, s) with DNN
    collect probabilities p(a1, a2, a3|θ, s) in A
    collect hidden neurons values in H from DNN
    collect s in S
    execute actions a1, a2, a3 and move to next state s'
    collect reward r in R from game
    if episode is finished then
        compute discounted rewards R_d from R
        estimate gradients from A*R_d, θ, S, and H (Eq. (19))
        update θ with the estimated gradient
        empty A, R, S, and H
    end if
    if current day ends then
        reset game: first state s' from a random day
    end if
    set s = s'
end for
```

VI. Results and Discussion

In this section, we validate our proposed methods and we analyze their performance on a large real-world database recorded by Pecan Street.Inc. First, the database is described. Then, numerical results are given for both problems, i.e., peak reduction and cost minimization for various number of buildings.

A. Data Set Characteristics

1) Buildings Pattern: To validate our proposed method we used the Pecan Street dataset. The disaggregated customer energy data contains up to 90 million unique electricity consumption records per day, which are used in order to build specific device patterns. Figure 3(a), 3(b), and 3(c) show three different building patterns averaged over the year 2015 with 15 minutes resolution. In these patterns the solar generation uncertainty as well as the people behavior characteristics are notable, e.g., even if all three buildings have an electric vehicle, just in the case of the third building, Fig. 3(c), it is used frequently. In our experiments, we have used the data between 27th October 2012 and 3rd September 2016.

2) Price Data: We use the time-of-use (ToU) tariff provided by the local grid operator Austin Energy for customers who live inside the City of Austin, Texas.[3] The λ^- summer rates are composed by on-peak, mid-peak and off-peak hours and the winter tariff has the mid-peak and off-peak hours components. There is also a difference between weekend and working-days tariff. Additionally, the self-generating customers are receiving an amount that is being paid by the utility for solar generation, called the value of solar tariff (VOST).

[3]http://austinenergy.com/wps/portal/ae/residential/rates/residential-electric-rates-and-line-items (Last visit: 30 October 2016).

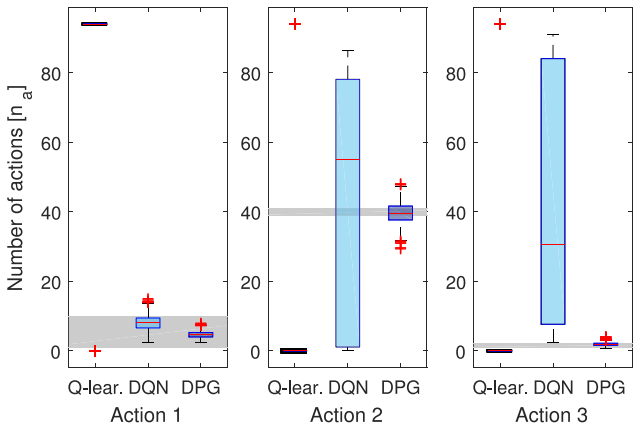


Fig. 2. Learning to satisfy the constraints bounds on one building (Equation (20)). The results are computed using all testing episodes. The gray areas show the bounds for each action. Remarkable, DPG always learns a good behavior, while Q-learning never does.

B. Baseline Reinforcement Learning

Before going into the assessment of the proposed methods, we perform a comparison of their behavior with a widely known baseline reinforcement learning method, i.e., Q-learning. We considered the peak reduction problem. To be able to perform Q-learning on it we discretized the values of the continuous states in two integer numbers per state dimension. This leads to 2048 discrete states. This discretization offered poor generalization capabilities to Q-learning and, in fact, it was not able to learn properly the problem constraints, as reflected in Figure 2. Remarkable, the DPG method learns to stay within the constraints-bounds. Perhaps, a higher granularity would help Q-learning generalize better, but if we would consider for example 3 integer numbers per state dimension this would lead to a number of 177147 (3^{11}) discrete states.

In general, reinforcement learning cannot handle properly continuous states due to the fact that it has to discretize them. Then a matrix with Q-values of the (discretized state, action) pairs is stored in the computer memory. Still, this matrix has an exponential memory footprint with respect to the number of actions and the discretization steps, and it cannot be handled properly by general purpose hardware. For instance, just 11 dimensions for the state space (as in our case) and 10 discretization steps lead to 100 billions discretized states. Thus, the only option is to employ artificial neural networks to store the (state, action) Q-values. This can be done either directly, as in DQN, or indirectly, as it happens in DPG. However, this does not make sense to be considered as even in the case analyzed by us with 2048 discrete states the training time of Q-learning was almost double than the one of DQN and DPG (order of minutes for one building). For these reasons, further on, we analyze just the performance of DPG and DQN.

C. Numerical Results - Peak Reduction Problem

The numerical results in terms of peak reduction at the single building level are showed in Table I and Figure 3 for three different buildings (B_I, B_{II} and B_{III}), over one year with 15 minutes resolution.

TABLE I
DAILY PEAK VALUE AT THE BUILDING LEVEL (B) AVERAGED OVER ONE YEAR WITH 15 MINUTES RESOLUTION VERSUS OPTIMIZED PEAK VALUE USING DQN AND DPG METHODS

	Method	B_I		B_{II}		B_{III}	
		Mean (μ)	St.dev. (Σ)	Mean (μ)	St.dev. (Σ)	Mean (μ)	St.dev. (Σ)
Peak [kW]	-	3.81	1.72	3.77	2.32	4.55	1.52
Optimized peak $[kW]$	DQN	2.72	1.45	2.72	1.21	3.59	1.41
	DPG	2.55	1.36	2.49	1.13	3.12	1.31

TABLE II
DAILY COST MINIMIZATION RESULTS AT THE BUILDING LEVEL AVERAGE OVER ONE YEAR WITH 15 MINUTES RESOLUTION USING DQN AND DPG METHODS

	Method	B_I		B_{II}		B_{III}	
		Mean (μ)	St.dev. (Σ)	Mean (μ)	St.dev. (Σ)	Mean (μ)	St.dev. (Σ)
Peak [kW]	-	3.81	1.72	3.77	2.32	4.55	1.52
Peak reduction $[kW]$	DQN	3.12	1.51	3.48	2.13	3.62	1.34
	DPG	2.97	1.46	2.69	1.14	3.17	1.29
Cost [$/day]	-	2.31	3.09	1.93	2.23	3.13	3.85
Minimized cost $[\$/day]$	DQN	2.19	3.01	1.91	2.18	2.85	3.62
	DPG	2.08	2.78	1.79	2.06	2.73	3.38

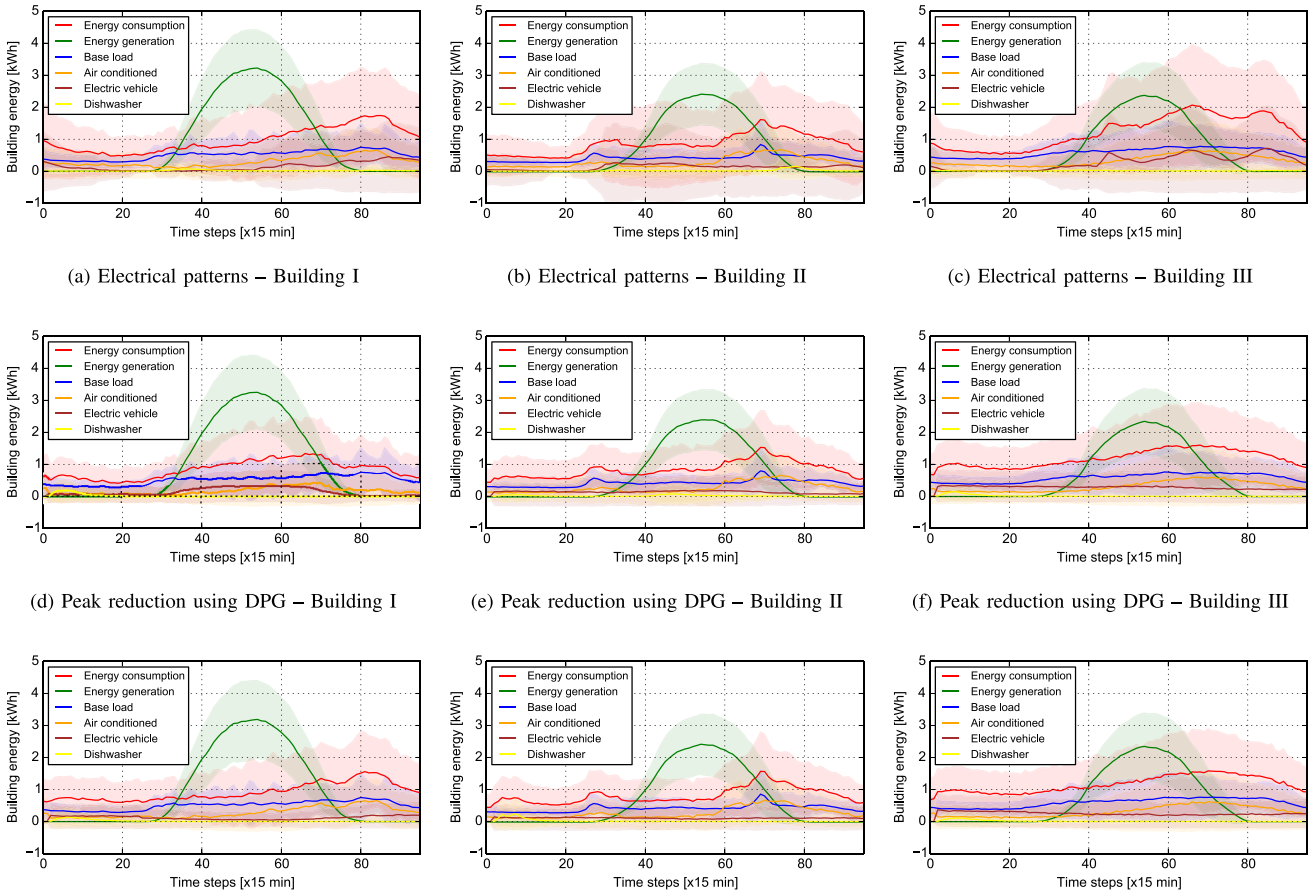
D. Numerical Results - Cost Minimization Problem

The results for the cost minimization problem are summarized in Table II. The difference between the cost minimization solutions obtained for every building correlated with their average electrical patterns (see Figure 3) give as a first indication about the individual capabilities of the end-users to adopt a more conservative behavior.

As it can be observed in Table II and in Figure 3, a secondary advantage of solving the cost minimization problem is its impact on solving of the peak reduction problem also. Therein, the best results in terms of both, peak reduction and cost reduction, are obtained for building B_{III} using DPG.

E. Scalability and Learning Capabilities of DRL

To test whether good estimations occur in practice and at scale, we investigate the performance of our proposed methods, in three cases with different numbers of customers using data from the Pecan Street smart grid test-bed. Specifically, we are investigating and analyzing the corresponding results using DQN and DPG methods for 10, 20 and 48 buildings, respectively. Tables III and IV show that our proposed approach is scalable for both, peak reduction and cost minimization, respectively. More than that, they show that at the aggregated level, when more customers are taking the cost minimization problem into consideration, this solves implicitly also the peak reduction problem. Same as at the building level, DPG is more stable than DQN, and achieves a better performance. Overall, in the case of the cost reduction problem for 48 buildings, DPG reduces the peak with 26.3% and minimizes the cost with 27.4%, while DQN reduces the peak with just 9.6% and minimizes the cost with just 14.1%. To visualize how DPG performs, in Figure 4 we depict the unoptimized and the optimized annualized energy costs for each of the 48 buildings. We can observe that the buildings behave very differently and in some cases DPG is capable to halve the yearly cost, while



(a) Electrical patterns – Building I (b) Electrical patterns – Building II (c) Electrical patterns – Building III

(d) Peak reduction using DPG – Building I (e) Peak reduction using DPG – Building II (f) Peak reduction using DPG – Building III

(g) Cost minimization using DPG – Building I (h) Cost minimization using DPG – Building II (i) Cost minimization using DPG – Building III

Fig. 3. (a), (b) and (c) represent different building electrical patterns averaged over one year (solid line) with 15 minute resolution, followed by their standard deviation (shadow area). Their yearly average optimization results using Deep Policy Gradient method for the peak reduction problem are depicted in (d), (e) and (f) whether the cost minimization results are shown in (g), (h) and (i).

TABLE III
PEAK REDUCTION – DAILY OPTIMIZATION RESULTS AT DIFFERENT LEVELS OF AGGREGATION AVERAGE OVER ONE YEAR WITH 15 MINUTES RESOLUTION USING DQN AND DPG METHODS

	Method	Number of buildings 10		20		48	
		Mean (μ)	St.dev. (Σ)	Mean (μ)	St.dev. (Σ)	Mean (μ)	St.dev. (Σ)
Peak [kW]	-	59.79	6.12	124.72	10.28	281.88	14.32
Optimized peak $[kW]$	DQN	49.67	5.62	106.84	7.49	238.12	12.98
	DPG	41.74	5.08	93.83	7.29	213.01	12.02

TABLE IV
COST REDUCTION – DAILY OPTIMIZATION RESULTS AT DIFFERENT LEVELS OF AGGREGATION AVERAGE OVER ONE YEAR WITH 15 MINUTES RESOLUTION USING DQN AND DPG METHODS

	Method	Number of buildings 10		20		48	
		Mean (μ)	St.dev. (Σ)	Mean (μ)	St.dev. (Σ)	Mean (μ)	St.dev. (Σ)
Peak [kW]	-	59.79	6.12	124.72	10.28	281.88	14.32
Peak reduction $[kW]$	DQN	54.85	5.93	116.72	9.24	254.67	13.21
	DPG	44.91	4.80	92.41	7.74	207.73	11.48
Cost [$/day]	-	57.79	20.90	118.03	30.01	231.27	38.76
Minimized cost $[\$/day]$	DQN	47.71	17.83	93.68	24.18	198.51	32.67
	DPG	44.35	16.01	82.71	21.48	167.70	28.62

in other cases it succeeds to reduce the cost with just a few percentage points.

1) Convergence Capabilities of DPG: The convergence is assessed through many iterations over episodes. For example, the learning capabilities of DPG method in terms of peak reduction and their corresponding reward function for a building are showed in Figure 5. Each episode represents an average value over 20 randomly chosen days. Initially, we may observe that the reward increases fast, while after about 1000 episodes the reward, as expected, increases much slower. Therefore, after approximatively 1000 episodes the average peak value and the optimized average peak value using the Deep Policy Gradient method converge. Still, the long-term reward expectation, as was expressed in Eq. (13), is increasing until approximatively 2500 episodes.

2) Computational time Requirements: Both DRL variants have the advantage of handling naturally much larger continuous state spaces, leading to better performance. In comparison with heuristic methods (e.g., PSO), after DRL learns how to act, it can make decisions (e.g., choosing the optimal control action) in a few milliseconds, while PSO needs to re-run the costly optimization process for every decision.

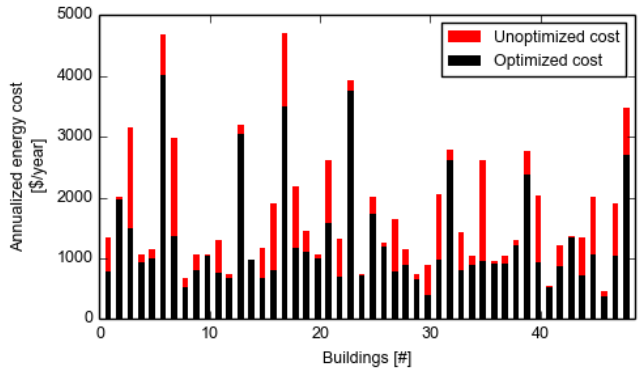


Fig. 4. Yearly savings per buildings when cost optimization is performed at the aggregated level on 48 buildings using Deep Policy Gradient (DPG) method.

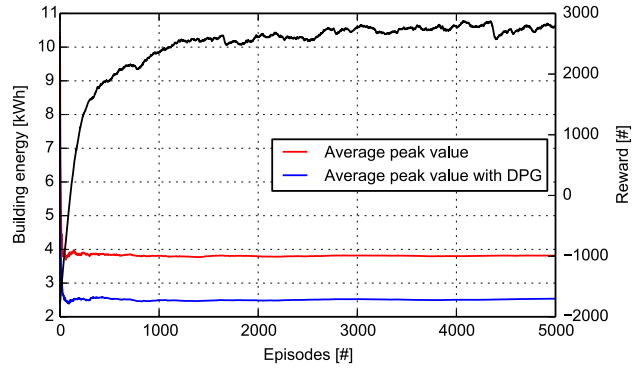


Fig. 5. Learning capabilities of Deep Policy Gradient method in terms of peak reduction and their corresponding reward function for a building (i.e., Fig. 2 Building I). Every episode represents an average value over 20 random days.

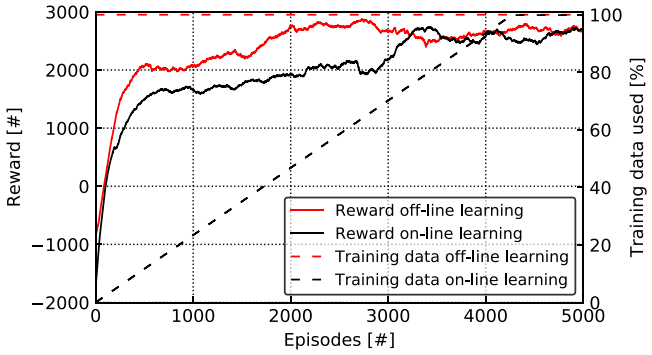


Fig. 6. On-line versus off-line learning capabilities of Deep Policy Gradient method in terms of reward function for a building (i.e., Fig. 2 Building I). Note that the right side y-axis shows the amount of training data served to the DPG. Every episode represents an average value over 20 random days.

3) On-Line Versus Off-Line Learning Performance: One of the most important characteristics of all deep reinforcement learning methods is their ability to learn in an on-line manner. To illustrate this, in Figure 6 it is shown how the DPG performance is continuously improving as more real-time data is collected. This case study compares the DPG learning capabilities in an off-line learning case, where all training data are used from the beginning, with a practical application environment, where increasing amounts of training data are accumulating over time. Note that although the reward function for the case of off-line learning is initially dominating the one for on-line learning, they become equivalent after about 80% of the data has been served, or 3500 episodes. However, further studies could be done in order to compare the performance of *on-line* versus *off-line* methods. In this case, an offline method may be seen as a combination of one prediction method and one optimization method.

TABLE V
SYMBOLS AND NOTATIONS

Symbol	Description
$\mathbb{N}, \mathbb{R}, \mathbb{R}_+$	The set of natural, real, and positive real numbers
$\sum, \prod, \int$	Sum, product, and integral
$p[\cdot], \mathbf{p}[\cdot], P[\cdot]$	Probability value/vector/matrix
$p(a\|b)$	The conditional probability of a given b
$\mathcal{N}(\mu, \sigma^2)$	The normal distribution
$\mathbb{E}[\cdot]$	Expected value operator
$\bigtriangledown f$, $\partial[\cdot]$	The gradient of f; The partial derivative of $[\cdot]$
$\propto$	Proportionality
$f \otimes g$	Composite functions
$\tilde{a}$	Estimated value of a

Notation	Description
t, T	Time slot and time horizon
$\mathbf{D}$	Dataset
$\mathbf{X}$	The training set, $\forall \mathbf{X} \in \mathbf{D}$
$\mathcal{B}$	Set of buildings
i	index of buildings, such that $B_i \in \mathcal{B}, \forall i \in \mathbb{N}$
d	index of electrical devices, $d \in \{1, .., m_i\}$.
P^+	Power generation
P^-	Power consumption
P_d^-	Power consumption per device
λ_t^+	The price value of P^+
λ_t^-	The price value of P^-
$\mathcal{S}$	The set of states, $\forall s \in \mathcal{S}$.
$\mathcal{A}$	The set of actions, $\forall a \in \mathcal{A}$.
$\mathcal{T}$	Transition probability function, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$
$\mathcal{R}$	The reward function, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$
Q	The quality matrix, $Q : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$
π	represents the stochastic policy, $\pi : \mathcal{S} \times \mathcal{A} \times \mathcal{R} \rightarrow \mathbb{R}_+$
θ	represents the set of parameters in Deep Belief Network
k	represents the number of layers in Deep Belief Network
η	represents a coefficient controlling the activation function
τ	represents a trajectory, $\tau = (s_t, a_t, r_t)$
g, $\hat{g}$	represents the gradient and the estimated gradient
n_a	represents the index of the action, $n_a \in \{a_1, a_2, a_3\}$.
ζ_1, ζ_2	represents the coefficients controlling the reward function

VII. CONCLUSION

In this paper, we proposed the use of Deep Reinforcement Learning, as a hybrid method which combines Reinforcement Learning with Deep Learning, with the aim of conceiving an on-line optimization for the scheduling of electricity consuming devices in residential buildings and aggregations of buildings. We have shown that a single agent, empowered with a suitable learning algorithm, can solve many challenging tasks. We proposed two optimization methods, Deep Q-learning and Deep Policy Gradient, to solve the same sequential decision problems at both the building level and the aggregated level. Also, we show the advantages of these methods in solving these tasks in comparison with a well-known Reinforcement Learning method, i.e., Q-learning. At both levels, we showed that Deep Policy Gradient is more suited to perform on-line scheduling of energy resources than Deep Q-learning. We explored and validated our proposed methods using the large Pecan Street database. Both proposed methods are able to successfully perform either the minimization of the

energy cost or the flattening of the net energy profile. For the minimization of the energy cost, a variable electricity price signal is investigated to incentivize customers to shift their consumption to low-price, off-peak periods.

REFERENCES

[1] M. R. Alam, M. St-Hilaire, and T. Kunz, "Computational methods for residential energy cost optimization in smart grids: A survey," *ACM Comput. Surveys*, vol. 49, no. 1, pp. 1–34, Apr. 2016.

[2] A. Barbato and A. Capone, "Optimization models and methods for demand-side management of residential users: A survey," *Energies*, vol. 7, no. 9, pp. 5787–5824, 2014.

[3] E. Loukarakis, C. J. Dent, and J. W. Bialek, "Decentralized multi-period economic dispatch for real-time flexible demand management," *IEEE Trans. Power Syst.*, vol. 31, no. 1, pp. 672–684, Jan. 2016.

[4] A.-H. Mohsenian-Rad and A. Leon-Garcia, "Optimal residential load control with price prediction in real-time electricity pricing environments," *IEEE Trans. Smart Grid*, vol. 1, no. 2, pp. 120–133, Sep. 2010.

[5] N. G. Paterakis, O. Erdinç, I. N. Pappi, A. G. Bakirtzis, and J. P. S. Catalão, "Coordinated operation of a neighborhood of smart households comprising electric vehicles, energy storage and distributed generation," *IEEE Trans. Smart Grid*, vol. 7, no. 6, pp. 2736–2747, Nov. 2016.

[6] J. S. Vardakas, N. Zorba, and C. V. Verikoukis, "A survey on demand response programs in smart grids: Pricing methods and optimization algorithms," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 152–178, 1rst Quart., 2015.

[7] L. A. Hurtado, E. Mocanu, P. H. Nguyen, M. Gibescu, and W. L. Kling, "Comfort-constrained demand flexibility management for building aggregations using a decentralized approach," in *Proc. Int. Conf. Smart Cities Green ICT Syst. (SMARTGREENS)*, Lisbon, Portugal, May 2015, pp. 1–10.

[8] C. J. C. H. Watkins and P. Dayan, "Technical note: Q-learning," *J. Mach. Learn. Res.*, vol. 8, nos. 3–4, pp. 279–292, May 1992.

[9] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, 1st ed. Cambridge, MA, USA: MIT Press, 1998.

[10] D. Ernst, M. Glavic, F. Capitanescu, and L. Wehenkel, "Reinforcement learning versus model predictive control: A comparison on a power system problem," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 2, pp. 517–529, Apr. 2009.

[11] D. O'Neill, M. Levorato, A. Goldsmith, and U. Mitra, "Residential demand response using reinforcement learning," in *Proc. 1st IEEE Int. Conf. Smart Grid Commun. (SmartGridComm)*, Gaithersburg, MD, USA, 2010, pp. 409–414.

[12] B.-G. Kim, Y. Zhang, M. van der Schaar, and J.-W. Lee, "Dynamic pricing and energy consumption scheduling with reinforcement learning," *IEEE Trans. Smart Grid*, vol. 7, no. 5, pp. 2187–2198, Sep. 2016.

[13] F. Ruelens *et al.*, "Demand response of a heterogeneous cluster of electric water heaters using batch reinforcement learning," in *Proc. Power Syst. Comput. Conf. (PSCC)*, Wrocław, Poland, 2014, pp. 1–7.

[14] H. Berlink and A. H. R. Costa, "Batch reinforcement learning for smart home energy management," in *Proc. 24th Int. Conf. Artif. Intell. (IJCAI)*, 2015, pp. 2561–2567.

[15] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[16] V. François-Lavet, D. Taralla, D. Ernst, and R. Fonteneau, "Deep reinforcement learning solutions for energy microgrids management," in *Proc. Eur. Workshop Reinforcement Learn. (EWRL)*, 2016.

[17] E. Mocanu, P. H. Nguyen, W. L. Kling, and M. Gibescu, "Unsupervised energy prediction in a smart grid context using reinforcement cross-building transfer learning," *Energy Build.*, vol. 116, pp. 646–655, Mar. 2016.

[18] J. L. Bode, M. J. Sullivan, and J. H. Eto, "Measuring short-term air conditioner demand reductions for operations and settlement," Lawrence Berkeley Nat. Lab., Berkeley, CA, USA, Rep. LBNL-5330E, 2012.

[19] M. Kraning, E. Chu, J. Lavaei, and S. Boyd, "Dynamic network energy management via proximal message passing," *Found. Trends® Optim.*, vol. 1, no. 2, pp. 73–126, 2014.

[20] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Stat. (AISTATS)*, vol. 15, 2011, pp. 315–323.

[21] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, 2013, p. 3.

[22] M. D. Zeiler *et al.*, "On rectified linear units for speech processing," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Vancouver, BC, Canada, May 2013, pp. 3517–3521.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Santiago, Chile, 2015, pp. 1026–1034.

[24] E. Mocanu, P. H. Nguyen, M. Gibescu, and W. L. Kling, "Comparison of machine learning methods for estimating energy consumption in buildings," in *Proc. 13th Int. Conf. Probabilistic Methods Appl. Power Syst.*, Durham, U.K., 2014, pp. 1–6.

[25] E. Mocanu, P. H. Nguyen, M. Gibescu, and W. L. Kling, "Deep learning for estimating building energy consumption," *Sustain. Energy Grids Netw.*, vol. 6, pp. 91–99, Jun. 2016.

[26] H. Shi, M. Xu, and R. Li, "Deep learning for household load forecasting—A novel pooling deep RNN," *IEEE Trans. Smart Grid*, to be published, doi: 10.1109/TSG.2017.2686012.

[27] E. Mocanu, P. H. Nguyen, M. Gibescu, E. M. Larsen, and P. Pinson, "Demand forecasting at low aggregation levels using factored conditional restricted Boltzmann machine," in *Proc. IEEE Power Syst. Comput. Conf. (PSCC)*, Genoa, Italy, Jun. 2016, pp. 1–7.

[28] S. Ryu, J. Noh, and H. Kim, "Deep neural network based demand side short term load forecasting," *Energies*, vol. 10, no. 1, p. 3, 2016.

[29] D. L. Marino, K. Amarasinghe, and M. Manic, "Building energy load forecasting using deep neural networks," in *Proc. 42nd Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, Florence, Italy, 2016, pp. 7046–7051.

[30] M. Manic, K. Amarasinghe, J. J. Rodriguez-Andina, and C. Rieger, "Intelligent buildings of the future: Cyberaware, deep learning powered, and human interacting," *IEEE Ind. Electron. Mag.*, vol. 10, no. 4, pp. 32–49, Dec. 2016.

[31] N. G. Paterakis, E. Mocanu, M. Gibescu, B. Stappers, and W. van Alst, "Deep learning versus traditional machine learning methods for aggregated energy demand prediction," in *Proc. 7th IEEE Int. Conf. Innov. Smart Grid Technol.*, Turin, Italy, 2017, pp. 1–6.

[32] H.-Z. Wang *et al.*, "Deep learning based ensemble approach for probabilistic wind power forecasting," *Appl. Energy*, vol. 188, pp. 56–70, Feb. 2017.

[33] Y. He, G. J. Mendis, and J. Wei, "Real-time detection of false data injection attacks in smart grid: A deep learning-based intelligent mechanism," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2505–2516, Sep. 2017.

[34] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," in *Proc. 33rd Int. Conf. Mach. Learn.*, vol. 48. New York, NY, USA, Jun. 2016, pp. 1928–1937. [Online]. Available: http://proceedings.mlr.press/v48/mniha16.html

[35] J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *CoRR*, vol. abs/1506.02438, Jun. 2015. [Online]. Available: http://arxiv.org/abs/1506.02438

[36] J. Peters and S. Schaal, "Reinforcement learning of motor skills with policy gradients," *Neural Netw.*, vol. 21, no. 4, pp. 682–697, 2008.

[37] H. Bou-Ammar, E. Eaton, P. Ruvolo, and M. E. Taylor, "Online multi-task learning for policy gradient methods," in *Proc. 31st Int. Conf. Mach. Learn. (ICML)*, Beijing, China, Jun. 2014, pp. 1206–1214.

Elena Mocanu received the B.Sc. degree in mathematics and physics from the Transilvania University of Brasov, Romania, in 2004, the M.Sc. degree in theoretical physics from the University of Bucharest, Romania, in 2011, the M.Sc. degree in operations research from Maastricht University, The Netherlands, in 2013, and the Ph.D. degree in machine learning and smart grids from the Eindhoven University of Technology, The Netherlands, in 2017.

She has been an Assistant Lecturer with the Department of Information Technology, University of Bucharest from 2008 until 2011. She was a Visiting Researcher with the Technical University of Denmark, Kongens Lyngby, Denmark and the University of Texas at Austin, Austin, Texas, USA in 2015 and 2016, respectively. She is currently a Researcher with the Control System Technology Group, Eindhoven University of Technology, where she focused on Artificial Intelligence and Autonomous Systems.

Decebal Constantin Mocanu has been an Assistant Professor in artificial intelligence and machine learning with the Data Mining Group, Department of Mathematics and Computer Science, Eindhoven University of Technology, The Netherlands, since 2017.

He received the B.Eng. degree in computer science from the Polytechnic University of Bucharest, Romania, in 2010 and the M.Sc. degree in artificial intelligence from Maastricht University, The Netherlands, in 2013, and the Ph.D. degree in artificial intelligence and network science from the Eindhoven University of Technology in 2017. From 2001 until 2013, he was a Software Engineer with several companies. He was a Visiting Scholar with the University of Pennsylvania, Philadelphia, PA, USA and a Visiting Researcher with the University of Texas at Austin, Austin, Texas, USA, in 2014 and 2016, respectively. He was a recipient of the Best "Master AI Thesis Award," from Maastricht University, in 2013.

Phuong H. Nguyen (M'06) was born in Hanoi, Vietnam, in 1980. He received the B.E. degree in electrical engineering from the Hanoi University of Technology, Vietnam, in 2002, the M.Eng. degree from the Electrical Engineering Department, Asian Institute of Technology, Thailand, in 2004, and the Ph.D. degree from the Eindhoven University of Technology, The Netherlands, in 2010, where he was a Post-Doctoral Researcher. He is an Assistant Professor with Electrical Energy Systems Group, Eindhoven University of Technology. He has been a Visiting Researcher with the Real-Time Power and Intelligent Systems Laboratory, Clemson University, USA, in 2012 and 2013. His research interests include distributed state estimation, control and operation of the power system, multiagent systems, and their applications in the future power delivery system.

Antonio Liotta (M'97–SM'15) is a Professor of data science and the Founding Director of the Data Science Research Centre, University of Derby, U.K. He is the Director of the Joint Intellisensing Lab (with nodes in the U.K., The Netherlands, Italy, Australia, and China), and a Guest Professor with Shanghai Ocean University, China, and the Eindhoven University of Technology, The Netherlands. His team is at the forefront of research in network and data science, specifically in the context of *Smart Cities*, *Internet of Things*, and *smart sensing*. He studies the complexity of modern systems from the viewpoints of complex networks, machine learning, and artificial intelligence. He is a member of the U.K. Higher Education Academy and serves the Peer Review College of the U.K. Engineering and Physical Sciences Research Council. He is the Editor-in-Chief of the (Springer) *Internet of Things* book series; an Associate Editor of the Journals JNSM, IJNM, JMM, and IF; and an Editorial Board Member of six more journals. He has six patents and over 270 publications to his credit, and has authored the book entitled *Networks for Pervasive Services: Six Ways to Upgrade the Internet*.

Michael E. Webber received the B.S. and B.A. degrees from the University of Texas at Austin, Austin, TX, USA, and the M.S. and Ph.D. degrees in mechanical engineering from Stanford, Stanford, CA, USA. He was the Deputy Director of the Energy Institute, the Co-Director of the Clean Energy Incubator, a Josey Centennial Professor in energy resources, an Author, and a Professor of mechanical engineering. He trains the next generation of energy leaders with the University of Texas at Austin and beyond through research and education at the convergence of engineering, policy, and commercialization. His recent book, entitled *Thirst for Power: Energy, Water, and Human Survival*, addresses the connection between Earth's most valuable resources and offers a hopeful approach toward a sustainable future, is receiving wide praise. His television special Energy at the Movies was in national syndication on PBS stations 2013–2015, and a suite of energy literacy tools titled Energy 101, including videos, online courses, and an interactive ebook, is available globally. He was selected as a fellow of ASME, has authored over 300 publications, holds four patents, and serves on the advisory board for Scientific American. He was honored as an American Fellow of the German Marshall Fund, an AT&T Industrial Ecology Fellow, and on four separate occasions by the University of Texas at Austin for exceptional teaching.

Madeleine Gibescu received the Dipl.Eng. degree in power engineering from University Politehnica, Bucharest, Romania, in 1993 and the M.S.E.E. and Ph.D. degrees from the University of Washington, Seattle, WA, USA, in 1995 and 2003, respectively.

She was a Research Engineer with Clear-Sight Systems, and a Power Systems Engineer with Alstom Grid, WA, USA. Since 2007, she has been an Assistant Professor with Electrical Sustainable Energy Department, Delft University of Technology, Delft, The Netherlands. She is currently an Associate Professor with the Electrical Energy Systems Department, Eindhoven University of Technology, The Netherlands. Her research interest is in the area of smart and sustainable power systems.

J. G. (Han) Slootweg (M'00) received the M.Sc. degree (*cum laude*) in electrical power engineering and the Ph.D. degree from the Delft University of Technology, Delft, The Netherlands, in 1998 and 2003, respectively, and the M.Sc. degree in business administration.

He is currently a Manager of the Innovation Department, Enexis B.V., Hertogenbosch, The Netherlands, one of the largest Distribution Network Operators of the Netherlands. Its spearheads are energy transition (including distributed generation and smart grids), asset condition assessment and increasing workforce productivity through new technologies. He also holds a professorship in smart grids with the Electrical Energy Systems Group, Eindhoven University of Technology, Eindhoven, The Netherlands. He has (co-)authored over 100 papers, covering a broad range of aspects of the electricity supply.