Electrical Power and Energy Systems 152 (2023) 109230

Contents lists available at ScienceDirect

International Journal of Electrical Power and Energy Systems

journal homepage: www.elsevier.com/locate/ijepes

Optimal energy system scheduling using a constraint-aware reinforcement learning algorithm

Hou Shengren [a], Pedro P. Vergara [a,*], Edgar Mauricio Salazar Duque [b], Peter Palensky [a]

[a] Intelligent Electrical Power Grids (IEPG) Group, Delft University of Technology, Delft 2628CD, The Netherlands
[b] Electrical Energy Systems (EES) Group, Eindhoven University of Technology, Eindhoven 5612AE, The Netherlands

ARTICLE INFO

Keywords:
Energy management systems
Distributed energy system
Safe reinforcement learning
Machine learning
Nonlinear programming

ABSTRACT

The massive integration of renewable-based distributed energy resources (DERs) inherently increases the energy system's complexity, especially when it comes to defining its operational schedule. Deep reinforcement learning (DRL) algorithms arise as a promising solution due to their data-driven and model-free features. However, current DRL algorithms fail to enforce rigorous operational constraints (e.g., power balance, ramping up or down constraints) limiting their implementation in real systems. To overcome this, in this paper, a DRL algorithm (namely MIP-DQN) is proposed, capable of *strictly* enforcing all operational constraints in the action space, ensuring the feasibility of the defined schedule in real-time operation. This is done by leveraging recent optimization advances for deep neural networks (DNNs) that allow their representation as a MIP formulation, enabling further consideration of any action space constraints. Comprehensive numerical simulations show that the proposed algorithm outperforms existing state-of-the-art DRL algorithms, obtaining a lower error when compared with the optimal global solution (upper boundary) obtained after solving a mathematical programming formulation with perfect forecast information; while strictly enforcing all operational constraints (even in unseen test days).

1. Introduction

To reduce the impact of the energy sector on the environment, distributed energy resources (DERs) are being integrated into our energy systems. Such DERs, in the form of renewable-based systems (e.g., PV systems and wind turbines) and small-scale energy storage systems (ESSs), provide more flexibility, enabling a more efficient operation. Nevertheless, these DERs also increase the energy system's complexity, especially when it comes to defining its operational schedule. Moreover, due to their weather-dependent nature, renewable-based DERs inherently increase the energy system's levels of uncertainty, requiring scheduling algorithms capable of providing fast and good-quality, but feasible, solutions [1]. In the technical literature, two main approaches are available to deal with the optimal scheduling of energy systems; namely, *model-based* and *model-free* approach. A detailed literature review is presented next.

1.1. Literature review

In general, model-based approaches rely on precise models to build complex mathematical formulations in order to consider the energy system' operational constraints. Depending on how these constraints are modeled, the derived mathematical formulations can be classified as linear, nonlinear programming, or dynamic programming problems [2]. In this regard, in [3], a mixed-integer nonlinear programming (MINLP) formulation is used to determine the optimal operation of an unbalanced three-phase energy system. In order to reduce the complexity of the proposed formulations, linearizations and simplifications are introduced. Similar work has been done in [4]. Nevertheless, the model-based nature of these methods requires considerable precision of the built mathematical models, which limits their performance, especially if uncertainty is to be considered.

Generally, in model-based approaches, uncertainty is modeled either by using a probability distribution function or by leveraging a set of representative scenarios, leading to stochastic or robust mathematical formulations, such as the ones presented in [5–7]. Other approaches, such as the one in [8], leverage a rolling time horizon approach to eliminate the forecast error when defining the DERs optimal energy scheduling. To guarantee the feasibility of the defined schedule under various operational scenarios, in [9], an adjustable two-stage robust optimization framework is proposed, solving simultaneously a day-ahead scheduling and real-time regulation problem of an integrated

* Corresponding author.
E-mail addresses: h.shengren@tudelft.nl (H. Shengren), p.p.vergarabarrios@tudelft.nl (P.P. Vergara), e.m.salazar.duque@tue.nl (E.M. Salazar Duque), p.palensky@tudelft.nl (P. Palensky).

https://doi.org/10.1016/j.ijepes.2023.109230
Received 24 October 2022; Received in revised form 15 February 2023; Accepted 7 May 2023
Available online 31 May 2023

Notation

The notation used throughout this paper is reproduced below for reference.

Sets:

$\mathcal{G}, \mathcal{B}, \mathcal{L}, \mathcal{V}$	Set of (DGs) distributed generators, EESs, Loads and PVs.
$\mathcal{S}, \mathcal{A}$	Set of states, set of actions.
$\mathcal{T}$	Set of time steps

Indexes:

i	DG unit $i \in \mathcal{G}$
j	ESS $j \in \mathcal{B}$
m	PV unit $m \in \mathcal{V}$
k	Load demand $m \in \mathcal{L}$
t	Time-step $t \in \mathcal{T}$

Parameters:

$\theta, \theta^{\text{target}}, \omega$	Parameters for the DNN's Q_θ, $Q_{\theta^{\text{target}}}$ and π_w
a_i, b_i, c_i	Quadratic, linear and constant parameters associated to the ith DG operation cost
Δt	Length discretization of the operational time
γ	Discount factor
$\overline{P}_t^G, \underline{P}_i^G$	Maximum/minimum generation limit of the DG units
RU_i, RD_i	Ramping up/ramping down ability of the DG units
$\overline{P}_j^B$ $\underline{P}_j^B$	Maximum/minimum charging/discharging limit of the ESSs
$\overline{SOC}_j^B$	Maximum SOC of the ESSs
$\underline{SOC}_j^B$	Minimum SOC of the ESSs
Ej^B	Energy capacity of the ESSs
$\overline{P}^C$	Maximum main network export/import limit
β	Electricity sell coefficient
η_B	Energy exchange efficiency for ESSs
σ_1, σ_2	Reward re-scale and constrain penalty coefficients
ρ_t	Electricity price for time slot t
$P_{m,t}^V$	Active power of PV systems
$P_{k,t}^L$	Active power demand

Continuous Variables:

$P_{i,t}^G$	Active power output of DG units
$P_{j,t}^B$	Active power discharge/charge of ESSs
$SOC_{j,t}^B$	State of charge for ESSs
P_t^N	Active power exported/imported to/from the main network
ΔP_t	Active power unbalance

energy system. In [10], a chance-constrained programming model is proposed to schedule an active distribution network incorporating office buildings. Nevertheless, modeling the probability distribution of uncertain data is challenging, while using a large number of scenarios may cause a computational burden. Therefore, although capable of providing good quality solutions, existing model-based approaches are not adequate for handling the increased uncertainty level of renewable-based energy systems, as their performance and efficiency mainly depend on the accuracy of the used models and their approximations. Moreover, the computational complexity of these methods increases dramatically with the system size, imposing scalability and convergence challenges.

To overcome this, model-free approaches have been introduced as an alternative solution. The most promising approach is based on the use of reinforcement learning (RL) [11], modeling the decision-making problem as a Markov Decision Process (MDP). One of the most interesting features of RL algorithms is that they can learn any system's dynamics by interaction, providing good-quality solutions guided by a reward value used as a performance indicator [12]. Recently, deep reinforcement learning (DRL) algorithms have shown good performance when solving MDPs in energy systems tasks [13], ranging from, home energy management [14], microgrid dispatch [15], voltage regulation [16], and electricity network operation [17]. Other applications include, for instance, a standardized DRL approaches for demand response in smart buildings [13], and learning to solve fast optimal power flow problems using DRL algorithms, specifically the proximal policy optimization (PPO) algorithm and imitation learning [18]. In [19], a performance comparison of the soft actor–critic (SAC) algorithm with a rule-based control method on the surrogate simulation model developed by [13], is presented. In [16], the voltage regulation problem of a distribution network is first modeled as a partial-observable MDP, and then multi-agent DRL algorithms are leveraged to execute the optimal solutions. In [20], a DRL approach-based proactive operation framework is proposed to model the stochastic behavior and uncertainty of solar energy for residential buildings. In [21], a DRL algorithm is developed to solve a stochastic energy management problem considering power flow constraints, resulting in an optimal policy that minimizes total operational cost (although operational constraints are disregarded).

Different from the energy-related MDPs presented above, the operational schedule of DERs within an energy system must enforce a rigorous set of operational constraints to ensure a reliable and safe operation, e.g., generation and consumption must always be balanced during real-time operation, ramping-up and down constraints, etc. Nevertheless, current DRL algorithms lack of safety guarantees [22], as these constraints cannot be directly imposed in the algorithm's formulation. Different strategies to indirectly enforce operational constraints have been proposed to overcome this. In [23], a DG unit is set as a slack bus with unlimited generation capacity, avoiding unbalance by the outputs of the generators controlled by DRL agents. In [24], a penalty term is added to the reward function to guide the learning process aiming to reduce operating costs while enforcing power balance. A similar penalty approach has been used to enforce voltage magnitude constraints in case the electricity network operation is considered. For instance, [25] modeled the dispatch of PV inverters as an MDP, and built a decentralized dispatch framework penalizing RL agents when actions lead to voltage violations. In research [26], an on-policy RL algorithm with eligibility traces is developed to dispatch the energy storage system to minimize the cost and regulate voltage magnitudes. A similar work is presented in [27]. In [28], a service assistant restoration problem is modeled as MDP. Then, imitation learning is employed as expert demonstrations enabling a deep deterministic policy gradient (DDPG) agent learn a safe policy for online implementation. In [29], a double auction market-based coordination framework is proposed to schedule the energy trading between multi-energy microgrids. Multi-agent twin delayed deep deterministic algorithm (TD3) is used to solve the formulated problem, while a large penalty is imposed on the reward function to reduce the energy unbalance. In [30], the SAC algorithm is leveraged to control a virtual power plant to provide frequency regulation services, penalizing any frequency deviation. Nevertheless, although these strategies may enforce operational constraints during training, they are either based on nonpractical assumptions or fail to guarantee the feasibility of the defined operating schedule in real-time, especially in cases of large peak consumption or renewable-based generation [31].

Table 1
Summary of research literature for DRL algorithms and constraint enforcing approaches.

Work	Research problem	Constraint enforcing	Advantages	Disadvantages	Open-access
[13]	Residential building energy schedule	Constraints disregarded	Simple	Not realistic	Yes
[14]	Microgrid operation				Yes
[20]	Residential buildings energy schedule				No
[15]	Microgrid operation	Penalty function	Easy to implement	No constraint guarantee	No
[16]	Voltage regulation				Yes
[18]	Optimal power flow				No
[21]	Energy dispatch				No
[24]	Energy dispatch				No
[31]	Optimal energy system scheduling				Yes
[25]	PV-inverter voltage regulation				No
[26]	Battery schedule and voltage regulation				No
[29]	Energy trading between microgrids				No
[28]	Restoration services	Imitation learning and penalty function	Accelerating training speed Improve the performance	No constraint guarantee	No
[23]	Energy Management	Unlimited slack bus	Simple	Not realistic	Yes
[34]	Energy management	Safe layer	Guarantee the feasibility	Performance deterioration Not fully model-free	No
[36]	Energy hub trading	Gaussian process Safe layer			
[37]	Microgrid operation	Action projection			
[38]	Distribution network operation	Constrained policy optimization	Probabilistic guarantee feasibility	No constraint guarantee Higher computation time	No
[39]	EV management				

Strategies based on safe RL have also been proposed to directly enforce operational constraints, exploiting results from different research areas, such as robot manipulation [32,33]. In [34], an action projection layer is implemented, correcting the action defined by the DRL algorithm via a projection operator. Unfortunately, this projection operator degrades the DRL algorithm's performance, as shown in [35]. In [36], safe DDPG is used for real-time automatic control of a smart hub, while a safety net is used to estimate the feasibility of decided actions. A similar strategy is proposed in [37], in which the action proposed by the DRL algorithm is used as starting point to solve a mathematical programming formulation, ensuring constraints compliance. In [38], a constrained policy gradient approach is proposed, updating the parameters of the DNN model in the direction that minimizes the power unbalance. In [39], the same approach is used to solve an EVs coordination problem. This policy optimization approach allows the DRL algorithm to provide a probabilistic notion of safety. Nevertheless, feasibility is paramount in energy systems operation, and it should be certifiable. In this regard, enforcing operational constraints during the online scheduling stage is a critical challenge for DRL algorithms and it must be addressed in order to enable their wide adoption in real systems. A summary of the discussed research literature is presented in Table 1. The openness and free online availability of the algorithms discussed here are also highlighted in Table 1.

1.2. Contributions

To overcome the above-discussed limitations, this paper proposes a DRL algorithm (namely MIP-DQP) to define the optimal schedule of a renewable-based energy system, capable of *strictly* enforcing all the operational constraints in the action space, ensuring the feasibility of the defined scheduled in real-time operation. To do this, we used recent optimization advances for DNNs that allow their representation as a mixed-integer linear (MIP) formulation, enabling further consideration of any action space constraints. Such approaches have been also employed in the context of feature visualization and adversarial machine learning [40]. The performance of the proposed algorithm has been compared with other state-of-the-art DRL algorithms available in the literature, including DDPG, PPO, SAC, and TD3 algorithms [11], to show its effectiveness. A comparison with the optimal global solution is also presented, obtained by solving the energy system scheduling problem as a mathematical programming formulation considering full knowledge of future information (i.e., consumption, dynamic prices, and renewable-based generation). The main contributions of this paper are as follows:

- A value-based DRL algorithm to solve the energy system scheduling problem is proposed, capable of dealing with continuous action spaces. Different from other actor–critic DRL algorithms (e.g., DDPG, PPO, and TD3 [11]), we make use of the action-value function approximated using a DNN, while discarding the policy model learning used during exploration.
- An innovative online execution approach that guarantees that the proposed DRL algorithm *strictly* meets all operational conditions in the action space (e.g., the power balance constraint), even in unseen test data, is also proposed. This is done by leveraging new optimization results from DNNs that allow their representation as a MIP formulation, enabling further consideration of any action space constraints.

The rest of this paper is organized as follows. In Section 2, the optimal energy system scheduling problem is formulated. Then, in Section 3, the formulated problem is modeled as MDP while the proposed MIP-DQN algorithm is illustrated and used to solve the optimal energy system scheduling problem in Section 4. Simulation tests are presented, analyzed and discussed in Section 5, while final conclusions are presented in Section 6.

2. Mathematical programming formulation of the energy systems scheduling problem

The structure of the considered energy system is shown in Fig. 1, including various DERs, such as solar photovoltaic (PV), ESSs, DGs, and

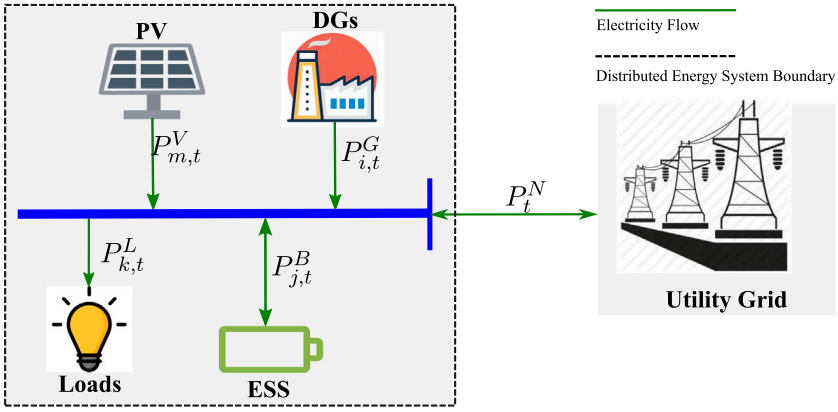

Fig. 1. Illustration of the considered energy system structure composed of various DERs, such as solar photovoltaic (PV), ESSs, DGs, and loads.

loads, while a connection to the utility grid is leveraged to address a demand surplus or shortage problem. For tractable analysis, we assume the day-ahead market where the electricity price of each hour is revealed beforehand. For the energy system in Fig. 1, the optimal energy system scheduling problem can be modeled by the nonlinear programming (NLP) formulation described by (1)–(11). The objective function in (1) aims at minimizing the operating cost for the whole time horizon $\mathcal{T}$, comprising the operating cost of the DG units, as presented in (2), and the cost of buying/selling electricity from/to the main network, as in (3). Given the output power of DG units $P^G_{i,t}$, the operating cost can be estimated by using a quadratic function as in (2). The transaction cost between the energy system and the network is settled according to Time-of-Use (ToU) prices, in which it is assumed that selling prices are lower than the purchasing prices. In (3), ρ_t is the ToU price at time slot t, while P^N_t refers to the exported/imported power transaction to/from the network.

$$\min_{P^G_{i,t},P^B_{j,t}} \left\{ \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{G}} \left[C^G_{i,t}(\cdot) + C^E_t(\cdot) \right] \Delta t \right\}, \tag{1}$$

$$C^G_{i,t} = a_i \left(P^G_{i,t} \right)^2 + b_i P^G_{i,t} + c_i, \quad \forall i \in \mathcal{G}. \tag{2}$$

$$C^E_t = \begin{cases} \rho_t P^N_t & P^N_t > 0, \\ \beta \rho_t P^N_t & P^N_t < 0. \end{cases} \tag{3}$$

Subject to:

$$\sum_{i\in\mathcal{G}} P^G_{i,t} + \sum_{m\in\mathcal{V}} P^V_{m,t} + P^N_t + \sum_{j\in\mathcal{B}} P^B_{j,t} = \sum_{k\in\mathcal{L}} P^L_{k,t} \qquad \forall t \in \mathcal{T} \tag{4}$$

$$\underline{P}^G_i \le P^G_{i,t} \le \overline{P}^G_i \qquad \forall i \in \mathcal{G}, \forall t \in \mathcal{T} \tag{5}$$

$$P^G_{i,t} - P^G_{i,t-1} \le RU_i \qquad \forall i \in \mathcal{G}, \forall t \in \mathcal{T} \tag{6}$$

$$P^G_{i,t} - P^G_{i,t+1} \le RD_i \qquad \forall i \in \mathcal{G}, \forall t \in \mathcal{T} \tag{7}$$

$$-\underline{P}^B_j \le P^B_{j,t} \le \overline{P}^B_j \qquad \forall j \in \mathcal{B}, \forall t \in \mathcal{T} \tag{8}$$

$$SOC^B_{j,t} = SOC^B_{j,t-1} + \eta_B P^B_{j,t} \Delta t / E^B_j \qquad \forall j \in \mathcal{B}, \forall t \in \mathcal{T} \tag{9}$$

$$\underline{SOC}^B_j \le SOC^B_{j,t} \le \overline{SOC}^B_j \qquad \forall j \in \mathcal{B}, \forall t \in \mathcal{T} \tag{10}$$

$$-\overline{P}^C \le P^N_t \le \overline{P}^C \qquad \forall t \in \mathcal{T} \tag{11}$$

Expression (4) defines the power balance constraint. Expression (5) defines the DG units generation power limits while (6) and (7) enforce the DG unit's ramping up and down constraints, respectively. Energy storage systems (ESSs) are modeled using (8)–(10). In this model, the operation cost of ESSs is not considered, while ESSs are allowed to schedule their discharge and charge power in advance. Expression (8) defines the charging and discharging power limits, while expression (9) models the state of charge (SOC) as a function of the charging and discharging power. Expression in (10) limits the energy stored in the ESSs, avoiding the impacts caused by over-charging and over-discharging. Finally, the main network export/import power limit is modeled by the expression in (11). Notice that in order to solve the mathematical formulation described by (1)–(11), *full knowledge* of future information (e.g., renewable-based generation, consumption and dynamic prices) is required, for instance, provided via a forecasting algorithm. The proposed DRL algorithm is able to provide good-quality solutions with only current information, as shown later. Next, the MDP formulation of the optimal scheduling problem is presented.

3. MDP formulation & value-based DRL

The above-presented decision-making problem can be modeled as a finite MDP, represented by a 5-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ represents the set of system states, $\mathcal{A}$ the set of actions, $\mathcal{P}$ the state transition probability function, $\mathcal{R}$ the reward function, and γ a discount factor. In this formulation, the energy system operator can be modeled as an RL agent. The state information provides an important basis for the operator to dispatch units. We define a state at time t as $s_t = (P^V_t, P^L_t, P^G_{t-1}, SOC_t), \quad s_t \in \mathcal{S}$, while the actions, defining the scheduling of the DG units and the ESSs, as $a_t = (P^G_{i,t}, P^B_t), \quad a_t \in \mathcal{A}$. Notice that the RL agent does not directly control the transaction between the energy system and the main network (i.e., P^N_t). Instead, after any action is executed, power is exported/imported from the main network to maintain the power balance. Nevertheless, a maximum power capacity constraint exists and must be enforced i.e., (11). Notice that if the maximum export/import limits are defined to be a low value (as done in this paper), in most cases, the power balance constraint will not be automatically met.

Given the state s_t and action a_t at time step t, the energy system transit to the next state s_{t+1} defined by the next transition probability function

$$p(S_{t+1}, R_t | S_t, A_t) = \Pr\left\{ S_{t+1} = s_{t+1}, R_t = r_t \mid S_t = s_t, A_t = a_t \right\}, \tag{12}$$

which models the energy system's dynamics. In model-based algorithms, the uncertainty is predicted by a determined value or sampling from a prior probability distribution. In contrast, DRL algorithms are a model-free approach, capable of learning such dynamics from interactions. To guide learning, a reward r_t must be provided by the environment in order for the RL agent to quantify the goodness of any action taken. In the energy system scheduling problem, the reward function $\mathcal{R}(s_t, a_t)$ should guide the RL agent to take actions that minimize the total operating cost, while enforcing the power balance constraint. This can be done by using the reward function

$$\mathcal{R}_t(s_t, a_t) = r_t = -\sigma_1 \left[\sum_{i\in\mathcal{G}} \left(C^G_{i,t} + C^E_t \right) \right] - \sigma_2 \Delta P_t, \forall t \in \mathcal{T}, \tag{13}$$

in which ΔP_t corresponds to the power unbalance at time-step t, defined as,

$$\Delta P_t = \left| \sum_{i\in\mathcal{G}} P^G_{i,t} + \sum_{m\in\mathcal{V}} P^V_{m,t} + P^N_t + \sum_{j\in\mathcal{B}} P^B_{j,t} - \sum_{k\in\mathcal{L}} P^L_{k,t} \right|. \tag{14}$$

In (13), σ_1 and σ_2 are used to control the order of magnitude and the trade-off between the operating cost minimization and the penalty incurred in case of power unbalance. The procedure used to solve the proposed MDP using value-based RL algorithms is presented next.

3.1. DRL value-based algorithms

Define $Q_\pi(S_t, A_t)$ as the action-value function that estimates the *expected cumulative reward* given that action a_t is taken at state s_t and following policy $\pi(\cdot)$ after that. The action-value function $Q_\pi(S_t, A_t)$ can be expressed recursively as [12],

$$Q_\pi(S_t, A_t) = \mathbb{E}_\pi \left[r_t + \gamma Q_\pi(s_{t+1}, a_{t+1}) | S_t = s_t, A_t = a_t \right]. \tag{15}$$

Bellman's principle of optimality states that the optimal action-value function for an MDP has the recursive expression

$$Q^*_\pi(S_t, A_t) = \mathbb{E}_\pi \left[r_t + \gamma \max_{a_{t+1}\in\mathcal{A}} Q^*_\pi(s_{t+1}, a_{t+1}) | S_t = s_t, A_t = a_t \right], \tag{16}$$

which solution can be obtained by using a Temporal Difference (TD) algorithm [41], which solves the following update rule iteratively.

$$\hat{Q}(S_t, A_t) \doteq \hat{Q}(S_t, A_t) + \alpha \left[r_t + \gamma \max_{a_{t+1}\in\mathcal{A}} \hat{Q}(s_{t+1}, a_{t+1}) - \hat{Q}(S_t, A_t) \right], \tag{17}$$

in which $\hat{Q}(\cdot)$ corresponds to a function approximator used to represent $Q^*_\pi(\cdot)$ and $\alpha \in (0, 1]$ is a learning rate. Once a good quality representation of $Q^*_\pi(\cdot)$ is obtained via $\hat{Q}(\cdot)$, at time step t and state s_t, optimal actions a_t can be sampled from the optimal policy, i.e., $a_t \sim \pi^*(s_t)$, obtained as

$$\pi^*(S_t) = \max_{a\in\mathcal{A}} \hat{Q}(S_t = s_t, a). \tag{18}$$

For continuous state and action spaces, the optimal action-value function $Q^*_\pi(\cdot)$ can be approximated using a DNN i.e., $\hat{Q}(\cdot) = Q_\theta(\cdot)$ with parameters θ, leading to an algorithm known as deep Q-networks (DQNs) [42]. In this case, the iterative procedure shown in (17) can be seen as a regression problem whose objective is to estimate the DNN's parameters θ via stochastic gradient ascent. In DQNs, the Q_θ is updated using the value $r_t + \gamma \max_{a\in\mathcal{A}} Q_{\theta^{target}}(s_t, a)$, where $Q_{\theta^{\text{target}}}$ is a *target* Q-function.[1] Under this value definition, parameters θ can be obtained minimizing a loss function over mini-batches B of past data $\left\{(s_t, a_t, r_t, s_{t+1})\right\}_{i=1}^{|B|}$. In this case, the loss definition used to train the DQN is based on the mean squared Bellman error, defined as[2]

$$\min_\theta \sum_{i=1}^{|B|} \left(r_{t,i} + \gamma Q_{\theta^{\text{target}}} \left(s_{t+1,i}, \arg\max_a Q_\theta \left(s_{t+1,i}, a \right) \right) - Q_\theta \left(s_{t+1,i}, a_{t,i} \right) \right)^2. \tag{19}$$

Notice that in continuous action spaces, the procedure used in (18) to sample actions from the action-value function Q_θ is not feasible since an exhaustive action enumeration (i.e., the Max-Q problem) is not possible. Moreover, in (18) actions constraints are completely disregarded. To overcome this, we combine value-based DRL algorithms with mixed-integer programming, as explained next.

[1] i.e., a copy of model Q_θ which parameters are updated less frequently. This procedure helps to stabilize learning within the DRL algorithm. For a more detailed explanation, see [43].

[2] For a more detailed derivation of the loss function in (19), see [43].

Algorithm 1: Training procedure for MIP-DQN

Define the maximum training epochs T, episode length L.
Initialize parameters of functions Q_θ, $Q_{\theta^{\text{target}}}$, and π_ω; Initialize reply buffer R. ;
for $t = 1$ **to** T **do**
  Sample an initial state s_0 from the initial distribution
  for $l = 1$ **to** L **do**
    Sample an action with exploration noise $a_t \sim \pi_\omega(s_t) + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma)$ and observe reward r_t and new state s_{t+1}. ;
    Store transition tuple (s_t, a_t, r_t, s_{t+1}) in R.;
  Sample a random mini-batch of $|B|$ transitions (s_t, a_t, r_t, s_{t+1}) from R.;
  Update the Q-function parameters by using (19).;
  Update the execution policy function parameters by using $\omega \leftarrow \omega + \nabla_\omega \frac{1}{|B|} \sum_{s\in B} Q_\theta \left(s, \pi_\omega(s) \right)$.
  Update the target-Q function parameters:
  $$\theta^{\text{target}} \leftarrow \tau\theta + (1-\tau)\theta^{\text{target}}$$

4. Proposed MIP-DQN algorithm

The proposed DRL algorithm is named MIP-DQN and is defined through two main procedures: training and deployment (or online execution). The main objective of the training procedure is to estimate the parameters θ of the DNN used to approximate the action-value function Q_θ; whereas during deployment, the obtained function Q_θ is used to take actions to directly operate assets within the energy system. Both procedures are explained in detail below.

4.1. Training procedure

The training process developed for the MIP-DQN algorithm is described in Algorithm 1. This process starts by randomly initializing the parameters of the DNN functions Q_θ, $Q_{\theta^{\text{target}}}$. Then, interactions with a model of the energy system take place. In traditional valued-based RL algorithms, exploration is done by sampling actions from the current estimate of the action-value function Q_θ. However, and as explained before, sampling actions from Q_θ following (18) is not a feasible procedure in continuous action spaces. Instead, we propose to use a parameterized deterministic optimal policy π_ω, which is also approximated using a DNN model and randomly initialized. Similar to other works [43,44], the policy function π_ω, the action-value functions Q_θ and $Q_{\theta^{\text{target}}}$, will be jointly approximated.

Within one epoch, for each time step t, a transition tuple of the form (s_t, a_t, r_t, s_{t+1}) is collected and store in a replay buffer R. Then, a subset B of these samples is selected and used to update the parameters of functions Q_θ, $Q_{\theta^{\text{target}}}$ and π_ω as shown in Algorithm 1. This procedure is iteratively done until a maximum number of epochs is reached.

Different from other DRL algorithms, such as DDPG and PPO, after training, we make use of the action-value function Q_θ and discard the approximated policy π_ω. Moreover, it is critical to notice that the power balance constraint is only enforced via the penalty added to the reward function in (13). Thus, it is expected that at the end of the training procedure, such equality constraint is not *strictly* met. The procedure used to enforce constraints is developed for the deployment or online execution, as explained next.

4.2. Deployment (online execution) procedure

After convergence of the training procedure, the action-value function Q_θ, with fixed parameters θ, can be used to take actions to control different energy resources. To do this, the problem stated in (18) must be solved. In this case, as function Q_θ represents a DNN, in

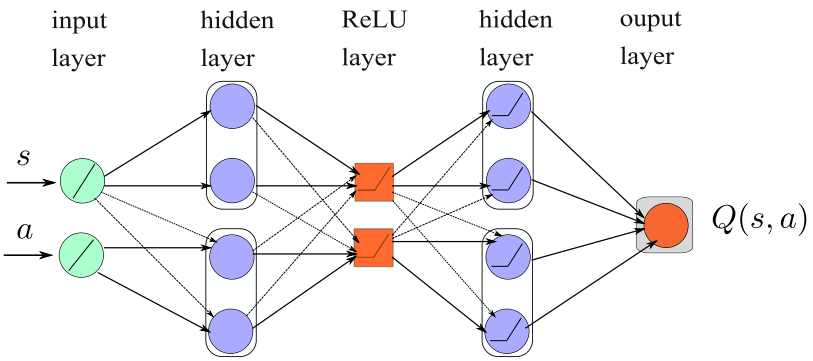

Fig. 2. Layer structure of the DNN used to approximate the action-value function $Q(s,a)$. We denoted this DNN model as $Q_\theta(s,a)$ in Algorithm 1.

order to solve (18), we leverage recent optimization results for DNNs. Thus, proposing a transformation of the DNN model Q_θ into a MIP formulation.

4.2.1. MIP for deep neural networks

Let the DNN $Q_\theta(s,a)$ in Fig. 2 consists of $K+1$ layers, listed from 0 to K. Layer 0 is the input of the DNN, while the last layer, K refers to the outputs of the DNN. Each layer $k \in \{0,1,\dots,K\}$ have U_k units, which is denoted by $u_{j,k}$, the j_{th} unit of the layer k. Let x^k refers to the output vector of layer k, then x_j^k is the output of unit $u_{j,k}$, $(j = 1,2,\dots,U_k)$. As layer 0 is the input of the DNN, then x_j^0 is j_{th} input value for the DNN. For each layer $k \le 1$, the unit $u_{j,k}$ computes the output vector x^k below:

$$x^k = h\left(W^{k-1}x^{k-1} + b^{k-1}\right) \tag{20}$$

where W^{k-1} and b^{k-1} are matrices of weights and biases that compose the set of parameters θ i.e., $\theta = \{W, b\}$ and $h(\cdot)$ is the activation function, which in this case corresponds to the ReLU function, described as: for a real vector y, $\mathrm{ReLU}(y) := \max\{0, y\}$.

Based on the above definitions, the DNN of Fig. 2, with fixed parameters θ, can be modeled as a valid MIP problem by modeling the ReLU function using binary constraints. Thus, using a binary activation variable z_j^k for each unit $u_{j,k}$, the MIP formulation of a DNN can be expressed as [40]:

$$\min_{x_j^k, s_j^k, z_j^k, \forall k} \left\{ \sum_{k=0}^{K} \sum_{j=1}^{l_k} c_j^k x_j^k + \sum_{k=1}^{K} \sum_{j=1}^{l_k} d_j^k z_j^k \right\} \tag{21}$$

Subject to:

$$\left.\begin{aligned} \textstyle\sum_{i=1}^{l_{k-1}} w_{ij}^{k-1} x_i^{k-1} + b_j^{k-1} &= x_j^k - s_j^k \\ x_j^k, s_j^k &\ge 0 \\ z_j^k &\in \{0,1\} \\ z_j^k = 1 &\rightarrow x_j^k \le 0 \\ z_j^k = 0 &\rightarrow s_j^k \le 0 \end{aligned}\right\} \quad \forall k, \forall j, \tag{22}$$

$$lb_j^0 \le x_j^0 \le ub_j^0, \quad j \in I_0, \tag{23}$$

$$\left.\begin{aligned} lb_j^k \le x_j^k \le ub_j^k \\ \overline{lb}_j^k \le s_j^k \le \overline{ub}_j^k \end{aligned}\right\} \forall k, \forall j. \tag{24}$$

In the above formulation, weights $w_{i,j}^{k-1}$ and biases b_j^k are fixed (constant) parameters; while the same holds for the objective function costs c_j^k and d_j^k. The ReLU function output for each unit is defined by (22), while (23) and (24) define lower and upper bounds for the x and s variables: for the input layer ($k = 0$), these bounds have physical meaning (same limits of the Q_θ inputs i.e., s and a), while for $k \ge 1$, these bounds can be defined based on the fixed parameters θ [45]. Finally, notice that in order for the MIP formulation to be equivalent to the DNN, ReLU activation functions must be used, as explained in [40].

4.2.2. Enforcing constraints in online execution

For an arbitrary state s_t, the optimal action a_t can be obtained by solving the MIP in (21)–(24) derived from Q_θ. In this case, as the decision variables are the actions a_t (see (18)), the power balance constraint in (4) as well as the ramp-up and ramp-down constraints in (6) and (7), respectively; can also be added to the MIP formulation described by (21)–(24). As a result, the optimal actions obtained by solving this MIP *strictly* enforce all operational constraints in the action space. This problem can be represented as,

$$\begin{aligned} \max_{a \in \mathcal{A}, x_j^k, s_j^k, z_j^k, \forall k} \quad & \{(21)\} \\ \text{s.t.} \quad & (22)\text{–}(24),\ (4),\ (6),\ (7). \end{aligned} \tag{25}$$

To better understand the MIP formulation stated in (25), Fig. 3 shows a re-interpretation of the power balance constraint in (4) as a hyperplane that define the feasibility region (for a three dimensional space) of the action space. Notice that such hyperplane may have different parameters for different time steps. Thus, if the hyperplane that enforces the power balance constraint is added to the MIP formulation that represents the DNN Q_θ, the solution of such mathematical problem will ensure minimum operating cost (via the maximization of Q_θ) and enforce all action space constraints, as exemplified in Fig. 4. In this case, this re-interpretation of the DNN as a MIP formulation offers enough flexibility to enforce equality constraints (as well as other constraints over the action space) for the energy system scheduling problem, such as the power balance. Algorithm 2 shows the step-by-step procedure used during the online execution of the proposed MIP-DQN algorithm.

Algorithm 2: Online Execution for the MIP-DQN Algorithm

Extract trained parameters θ from Q_θ;
Formulate the Q-function network Q_θ as a MIP formulation according to (21)–(24). Add all action space constraints i.e., (4), (6) and (7).
Extract initial state s_0 based on real-time data;
for $t = 1$ **to** T **do**
  For state s_t, get optimal action by solving (25) using commercial MIP solvers;

5. Simulation results and discussions

In this section, simulation results and discussions are presented. A comparison with DRL algorithms available in the literature, including PPO, SAC, DDPG and TD3 algorithms, is also presented.

5.1. Case study and simulations setup

To test the developed MIP-DQN algorithm, an energy system consisting of three DG units and an ESS is defined. The DG unit's parameters are shown in Table 2, while for the ESS, the charging/discharging

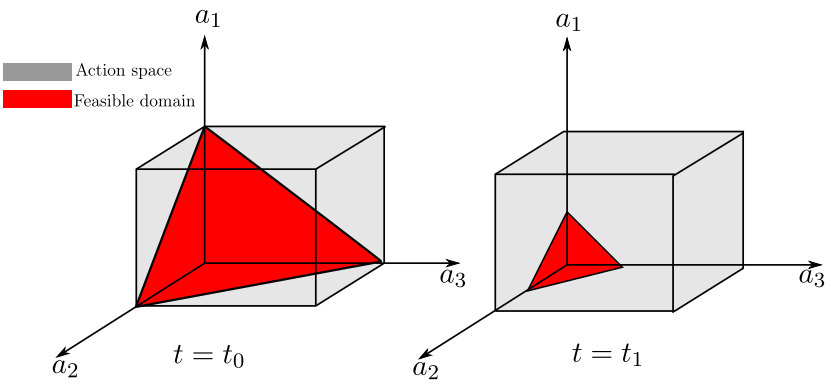

Fig. 3. Action space (grey) and feasible action space (red) illustration. Actions a_1, a_2, a_3 refer to generic actions in a three dimension action space $\mathcal{A}$. For each time step t, the power balance constraint in (4) can be seen as the hyperplane $a_1 + a_2 + a_3 = d$ that defines the feasible actions space.

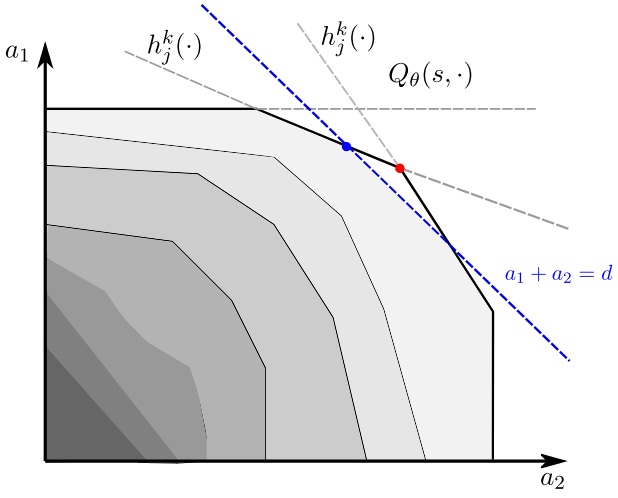

Fig. 4. Visualization of the constraint space whose boundaries are formed by the hyperplanes $h_j^k(\cdot)$ defined by the ReLU activation functions derived from the deconstructed DNN $Q_\theta(s,\cdot)$ as a MIP formulation, for an specific state s and actions a_1 and a_2. The grey are shows the increasing value (from darker to lighter) of ∇Q_θ. The red point exemplifies the optimal solution of $\max_{a\in\mathcal{A}} Q_\theta(s,\cdot)$ if constraint $a_1 + a_2 = d$ is disregarded. If such a constraint is added to the MIP formulation, the solution represented with the blue point will be reached.

Table 2
DG units information.

Units	a [\$/kW^2]	b [\$/kW]	c [\$]	$\underline{P}^G$ [kW]	$\overline{P}^G$ [kW]	RU [kW]	RD [kW]
DG_1	0.0034	3	30	10	150	100	100
DG_2	0.001	10	40	50	375	100	100
DG_3	0.001	15	70	100	500	200	200

limits, nominal capacity, and energy efficiency (η_B) are set to 100 kW, 500 kW, and 0.90, respectively. We assume that the network's maximum export/import limit is defined as 30 kW. To encourage the use of renewable energies, we set selling prices as half of the current electricity prices, i.e., $\beta = 0.5$.

One-year demand consumption and PV generation data are used as the original data-set, sampled in hour resolution. Fig. 5 shows the mean and standard deviation of the demand consumption and PV generation during summer and winter for a period of 24 h, defined as the length of one episode ($T = 24$). The original dataset is divided into two additional datasets: training and testing. The training dataset contains the first three weeks of each month, while the testing dataset contains the remaining data. This allows the DRL algorithm to learn any seasonal and weekly behavior available in the PV generation and demand consumption data [31]. During training, the EES's initial SOC was randomly set. To implement our MIP-DQN algorithm, PyTorch and OMLT (see [45]) package has been used. Default settings were used for all the implemented DRL algorithms, as shown in Table 3. All implemented algorithms are openly available in [46]. Hyper-parameters σ_1 and σ_2 are defined as 0.01 and 20, respectively, as default values. Each test is run with five random seeds to eliminate randomness from code implementation.

5.2. Validation and algorithms for comparison

In the research literature, DRL algorithms are usually compared with simple rule-based or MPC-based algorithms (considering the impacts of any forecasting error) [47]. Nevertheless, this procedure does not allow us to estimate the optimality gap between current DRL algorithms and the optimal global solution with a perfect forecast of the stochastic variables (i.e., generation and demand consumption). In this case, this optimal global solution with full knowledge should be regarded as an upper boundary, as none algorithm would perform better. Based on this, to validate and fairly compare the performance of the proposed MIP-DQN algorithm, besides comparing the optimal DERs schedule defined by several state-of-the-art DRL algorithms (DDPG, PPO, TD3), we compared with the optimal global solution obtained considering perfect forecast for the next 24 h. In this case, the optimal global solution is found by solving the nonlinear mathematical programming formulation in Section 2, implemented using Pyomo [48]. Notice that different from the optimal global solution, all the tested DRL algorithms are able to make decisions only using current information. Finally, to evaluate the DRL algorithms' performance, the total operating cost, as in (1), and the power unbalance, as in (14), are used as metrics.

5.3. Performance on the training set

Fig. 6 shows the average reward, operating cost, and power unbalance for the developed MIP-DQN algorithm and other DRL algorithms during the training process. As can be seen in Fig. 6, the average reward increases rapidly after 100 episodes of training, while the operating cost and the power unbalance significantly decrease. This behavior during training is typical of DRL algorithms as the DNN's parameters are randomly initialized, leading initially to random actions causing high power unbalance. Throughout the training, and due to the introduction of the penalty terms used in the reward definition in (13), the DNN's parameters are updated, leading to higher quality actions, reducing

Table 3
Parameters for DRL algorithms.

Algorithm	Batch size $\|B\|$	Learning rate	Buffer size R	γ	Network dimension	Optimizer
DDPG	256	1e−4	5e4	0.995	(64, 64, 64)	Adam
SAC	256	1e−4	5e4	0.995	(64, 64, 64)	Adam
TD3	256	1e−4	5e4	0.995	(64, 64, 64)	Adam
PPO	256	1e−4	–	0.995	(64, 64, 64)	Adam
MIP-DQN	256	1e−4	5e4	0.995	(64, 64, 64)	Adam

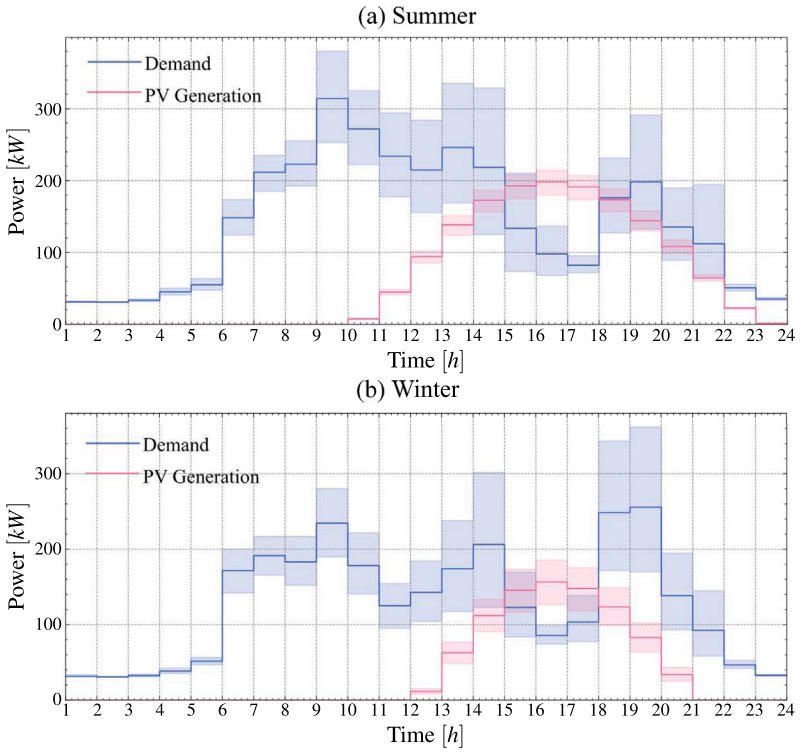

Fig. 5. Mean and standard deviation of the demand consumption and PV generation.

power unbalance, and showing a lower operating cost. All algorithms converged before 400 episodes. After the last training episode, the power unbalance (presented by the average with 95% confident interval) of DDPG, SAC, PPO, and TD3 are 64.8 ± 99 kW, 807 ± 121 kW, 65 ± 18 kW, 304 ± 104 kW, respectively; while a power unbalance of 12 ± 15 kW was observed for the proposed MIP-DQN algorithm. This result shows how the proposed MIP-DQN algorithm outperformed other DRL algorithms during the training process. Nevertheless, and as expected, none of the tested DRL algorithms (including the proposed MIP-DQN) can strictly enforce the power balance; if such algorithms are used in real-time operation, they might lead to unfeasible operation. Next, we show how our proposed algorithm can overcome this during online execution, even in unseen data.

5.4. Performance on the test set

After training, the DNN's parameters of all the DRL algorithms are fixed as shown in Algorithm 2. A performance comparison is now made on the test set. Recall that the data on the test set is not used during training; therefore, it has not been seen by any of the DRL algorithms. To compare results on the test set, Fig. 7 shows the cumulative operating cost and power unbalance (which can be seen as a cumulative error) for 10 different days using the proposed MIP-DQN algorithm, as well as other DRL algorithms. The optimal global solution obtained by solving the NLP formulation and considering the perfect forecast is also presented. As can be seen in Fig. 7, during online operation and for all 10 test days, the proposed MIP-DQN algorithm strictly meets the power balance constraint, while other DRL algorithms fail to deal with such equality constraint. Notice in Fig. 7 how DRL algorithms such as DDPG and TD3 reach a cumulative power unbalance near 0.14 MW at the end of the test period. As a result of such high unbalances, an operating cost of 53.3% higher than the optimal global solution is also observed. In contrast, the proposed MIP-DQN algorithm achieves an operating cost of 94 $k$$, i.e., 17.6% higher than the optimal solution.

To test the performance with a higher number of test days, Table 4 presents the average cumulative error (with respect to the solution obtained by solving the NLP formulation with perfect forecast), the average power unbalances, and total average computational time (over 30 test days) of the proposed MIP-DQN algorithm as well as other DRL algorithms. As can be seen, the proposed MIP-DQN algorithm has the lowest average error, 13.7%; while strictly meeting the power balance (and other) constraint. In contrast, algorithms such as PPO showed poor performance reaching an error of 52.4%. As expected, the total computational time required to execute the proposed MIP-DQN algorithm is higher than other DRL algorithms. This increase in the computational time is a result of the MIP formulation required to be solved in order to enforce the equality constraint (see (25)). Nevertheless, for this case, the proposed MIP-DQN algorithm can still be used for real-time operation as it only requires less than 20 s for execution. In this case, it is important to highlight that the computation time of the proposed MIP-DQN algorithm is impacted by the size of formulated MIP problem, which is only determined by the size of the used Q network (layers, units of each layer, etc.) and not by the size of the energy system (microgrid) considered. Previous research

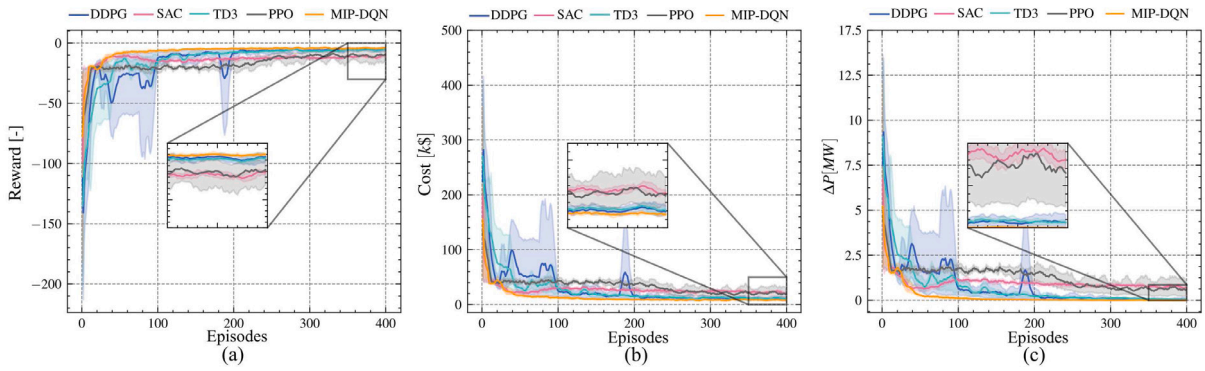

Fig. 6. Mean and 95% confident interval for the reward, operating cost and power unbalance for the developed MIP-DQN algorithm, as well as for other DRL algorithms, during training. As expected, none of these DRL algorithms are able to enforce the power balance constraint.

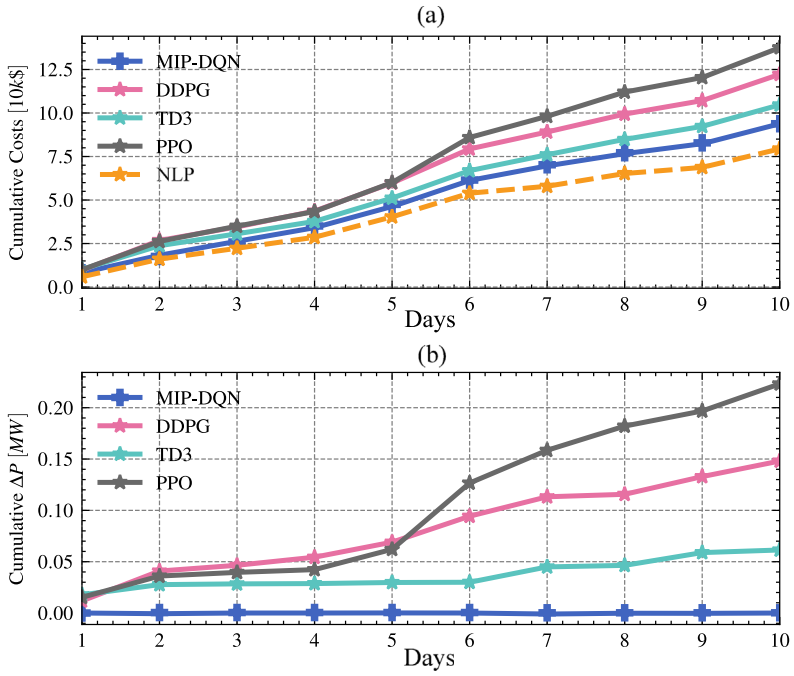

Fig. 7. Cumulative costs and power unbalance for 10 days in the test set. The proposed MIP-DQN algorithm is able to strictly meet the power balance constraint while other DRL algorithms fail to do so.

Table 4
Performance comparison of different DRL algorithms in a new test set of 30 days.

Algorithms	Error	ΔP [MW]	Computational time [s]
MIP-DQN	13.7 ± 0.3%	**0.0**	17
DDPG	47.3 ± 1.9%	0.14 ± 0.021	4.3
TD3	31.5 ± 0.7%	0.06 ± 0.011	4.9
PPO	52.4 ± 0.3%	0.15 ± 0.007	4.3

has shown that (small) neural networks can generalize well in real environments [19,28], supporting the applicability of DRL models in real systems.

5.5. Dispatch decisions comparison

Until now, the general performance of the proposed MIP-DQN algorithm has been presented, highlighting its capability of strictly enforcing the power balance constraint, even in unseen operational days. Next, a comparison in terms of the scheduling of the DG units and the ESSs is introduced. To do this, Fig. 8 displays the output power of all the DG units, ESSs and the imported/exported power from the network for: the proposed MIP-DQN algorithm (Fig. 8*b*), and the optimal solution obtained after solving the NLP formulation considering perfect forecast (Fig. 8*c*). Notice in Fig. 8 that when the electricity price is high, and the net power is low, the proposed MIP-DQN algorithm dispatches the ESSs in charging mode, and a similar dispatch decision is observed in the optimal global solution. Notice also that, when compared with the optimal solution, the proposed MIP-DQN algorithm dispatched 3_{th} DG during the peak hour, which can be considered a sub-optimal decision as the operating cost of such DG is higher than the others. This difference in this dispatch decision can be due to the estimated Q-function, which might not be good enough to represent the true action-value function. In this sense, as the proposed MIP-DQN algorithm chooses actions that maximize its Q-value estimation, the largest Q-value might not represent the

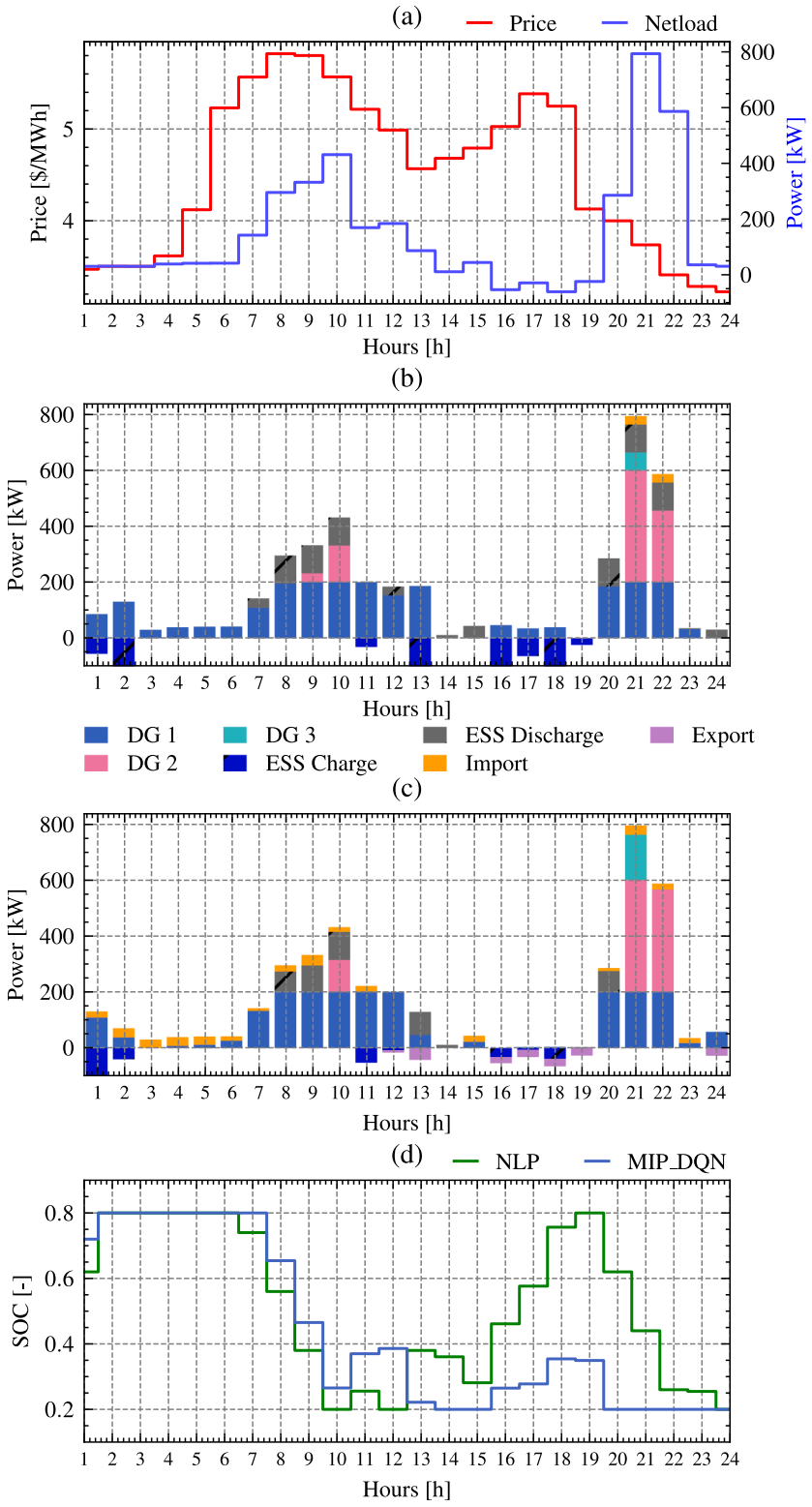

Fig. 8. Operational schedule of all DG units and ESSs defined by the proposed MIP-DQN algorithm and the optimal global solution obtained by solving the NLP formulation considering perfect forecast.

best action for this specific state–action pair. Nevertheless, even in executing a sub-optimal decision, the proposed MIP-DQN algorithm is able to meet the power balance constraint, guaranteeing operational feasibility. Finally, although differences in the dispatch decisions made by the proposed MIP-DQN algorithm and the optimal solution can be observed, it is important to highlight that the optimal global solution is obtained considering the perfect forecast of the future generation and demand consumption for the next 24 h, while the proposed MIP-DQN algorithm provides dispatch decisions in an hourly basis, without knowledge of the future values of the stochastic variables.

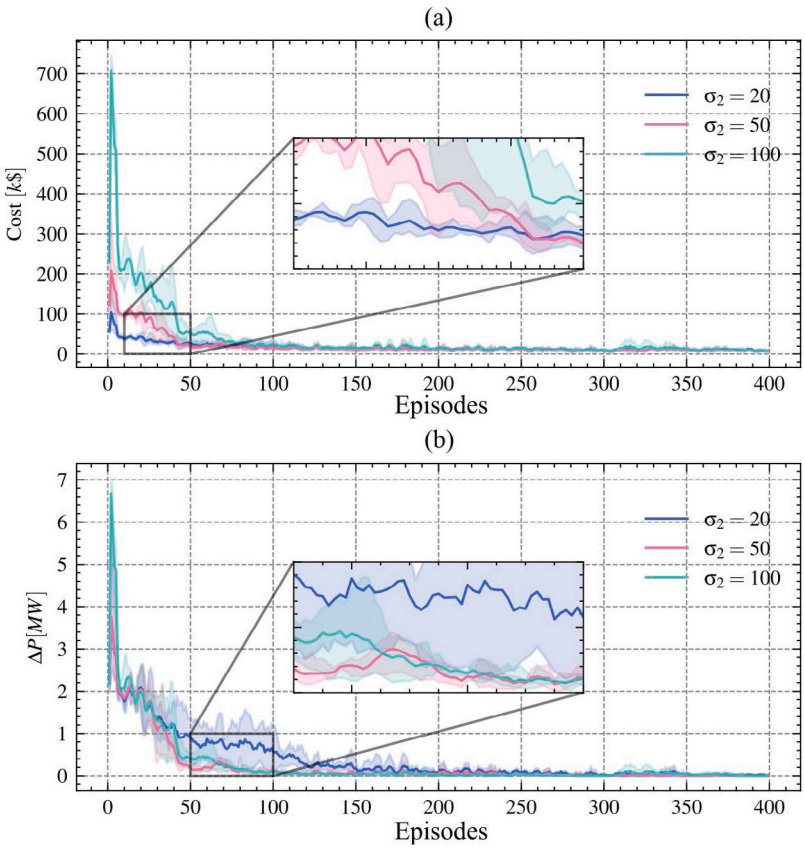

Fig. 9. Average reward, operating cost, and power unbalance of the proposed MIP-DQN algorithm for different values of σ_2.

5.6. Sensitivity analysis

To better understand the impact of hyperparameter σ_2 in the reward function in (13), Fig. 9 shows the average operating cost and power unbalance (during training) for the proposed MIP-DQN algorithm for $\sigma_2 = 20, 50, 100$. As can be seen in Fig. 9, and as expected, higher values of σ_2 accelerate the convergence of the proposed MIP-DQN algorithm to rapidly reduce power unbalance, while having no apparent impact on the convergence of the operating cost. On the other hand, lower values of σ_2 seem to accelerate the convergence of the operating cost leaving behind the convergence of the power unbalance. In general, for the test performed, it was observed that the proposed MIP-DQN algorithm could converge in less than 200 episodes.

5.7. Comparison with safe DDPG algorithm

A comparison with current safe DRL algorithms is also performed. In this case, the proposed MIP-DQN algorithm is compared with a Safe DDPG algorithm, as presented in [49]. Fig. 10 shows the average reward (Fig. 10*a*), operating cost (Fig. 10*b*), and power unbalance (Fig. 10*c*) for the two algorithms being compared. In this case, and as expected, both algorithms fail to enforce the power unbalance constraint strictly during training. At the beginning of the training stage, the Safe DDPG algorithm shows a lower operating cost and power unbalance, and higher reward, when compared to the MIP-DQN algorithm. This is mainly due to the trained linear safe layer of the Safe DDPG, which projects the exploration action to a safer one, while the MIP-DQN algorithm is free to explore the action space regardless of the feasibility of the decided action. Nevertheless, along with the training, the Safe DDPG algorithm fails to learn to reduce further or eliminate power unbalance, while our proposed MIP-DQN algorithm reduces the unbalance sharply. This behavior is mainly due to the reward shaping of the MIP-DQN algorithm, which can learn to avoid the penalty due to the power unbalance during the training. It is important to highlight that the performance of the Safe DDPG algorithm depends on the quality of the trained safe layer that project the original action of the DDPG algorithm to a feasible one. In this case, as the safe layer is a linear function, its generalization capabilities may not be enough to learn the complex nonlinear energy system dynamic. Thus, even after projection, the action cannot fully meet the power unbalance constraint. Moreover, as the safe layer modified the action during exploration, it also harms the performance of the trained RL algorithm as shown in Fig. 10. Compared to the Safe DDPG algorithm, the proposed MIP-DQN algorithm learns to eliminate the unbalance in a small value after training and guarantees the feasibility during the execution (Fig. 7).

5.8. Larger case study

To test the performance of the proposed MIP-DQN algorithm on an energy system with multiple ESSs, an environment with three ESSs and three DG generators is designed. For this new environment, Fig. 11 shows the average operating cost and power unbalance of the proposed MIP-DQN algorithm as well as other state-of-the-art DRL algorithms, during the training process. As can be seen in Fig. 11, the operating cost and power unbalance are significantly reduced. In this case, all tested DRL algorithms converged at around 400 episodes. The power unbalances (presented by the average with 95% confident interval) of the DDPG, SAC, PPO and TD3 algorithms are 97 ± 125 kW, 533 ± 208 kW, 45 ± 19 kW, 462 ± 98 kW, respectively. In contrast, a power unbalance of 17 ± 22 kW was observed for the proposed MIP-DQN algorithm. Similar to the results presented in Section 5.3 for the smaller case study (see Fig. 6), none of the tested DRL algorithms can strictly enforce the power balance during training. Most of the observed power balance violations happen during peak load days, consistent with previous results [31]. Nevertheless, the proposed MIP-DQN algorithm is able to

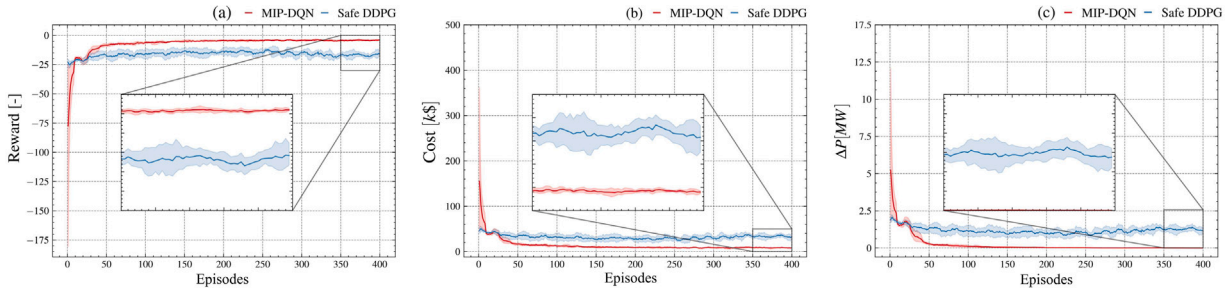

Fig. 10. Mean and 95% confident interval for the reward, operating cost and power unbalance for the developed MIP-DQN and Safe DDPG algorithms.

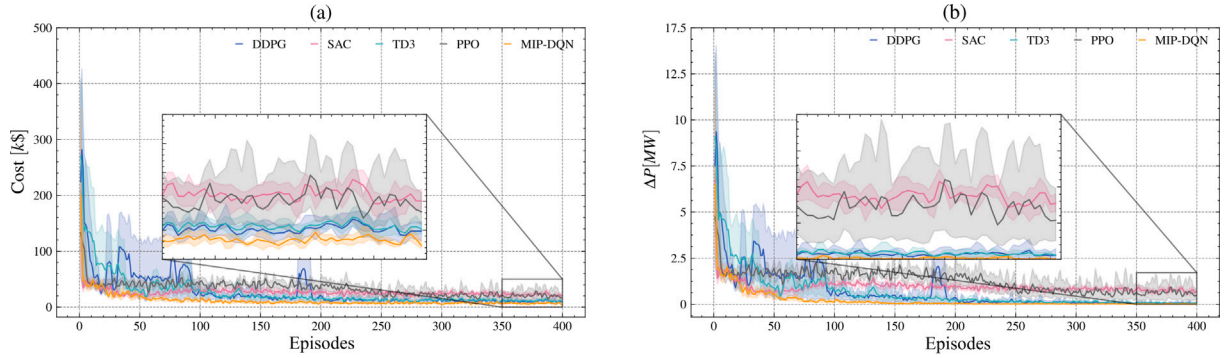

Fig. 11. Mean and 95% confident interval for the operating cost and power unbalance for the developed MIP-DQN algorithm, as well as for other DRL algorithms, during training.

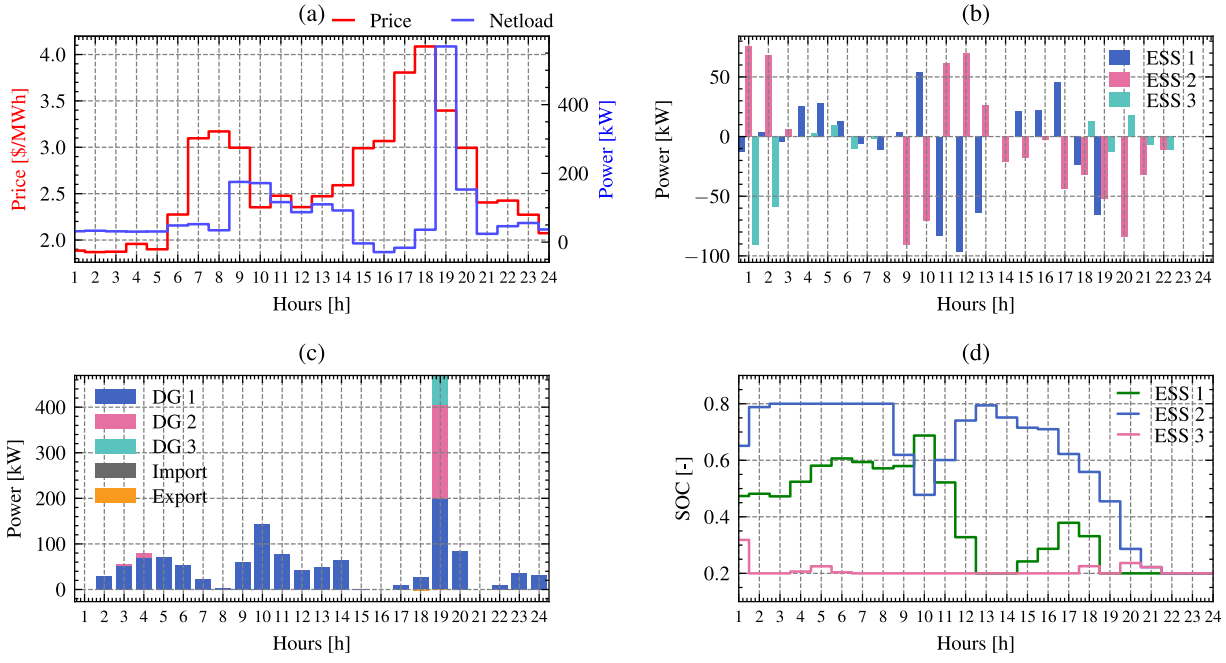

Fig. 12. Operational schedule of all ESSs and DG units defined by the proposed MIP-DQN algorithm for a larger case study composed of three ESSs and three DG units.

enforce power unbalance during the online execution, even on peak load days, as shown next. Additionally, compared to the result of simulations in Section 5.3, no performance degeneration is observed, proving the scalability of the proposed MIP-DQN algorithm.

Fig. 12 shows the scheduling decisions from the MIP-DQN algorithm for all three ESSs (Fig. 12*b*) and DG generators (Fig. 12*c*), and corresponding SOC changes (Fig. 12*d*) in a typical day with extreme peak load. Notice that the power balance is strictly enforced during the peak load day. For instance, at 19 h, the load is extremely high, and the MIP-DQN algorithm dispatches all the ESSs in discharging mode. This avoided importing electricity from the main grid as the electricity price was high at that particular time. These results showed that the proposed MIP-DQN algorithm learned to schedule feasible decisions for multiple ESSs in extreme peak situations. Notice also that, at hours 3 and 4, the proposed MIP-DQN algorithm dispatches the 2_{th} DG, instead of fully using the 1_{th} DG, which can be considered as a sub-optimal decision because the operating cost of 2_{th} DG is higher than that of 1_{th} DG. A similar result was observed in Fig. 8. Nevertheless, even in executing a sub-optimal decision, the proposed MIP-DQN algorithm is able to meet the power balance constraint, guaranteeing operational

feasibility. Thus, the proposed MIP-DQN algorithm can provide feasible dispatch decisions hourly for multiple ESSs, displaying prominent scalability features.

5.9. Discussion

The penetration of renewable-based DERs energies significantly increases the uncertainty and complexity of the operation of energy systems. Existing model-based approaches may not perform well when defining the operational schedule of DERs in real time due to their poor accuracy and high computational time requirements. Due to this, current efforts are put into leveraging RL algorithms' model-free and data-driven nature. After offline training, RL algorithms can provide near-optimal solutions in real-time. Nevertheless, the most critical challenge to enabling RL algorithms deployment in real energy systems scheduling frameworks is their lack of constraint enforcing guarantee. Even though several safe RL algorithms have tackled this problem, these approaches fail to meet the required security levels of energy systems operation [50]. In general, model-based optimization approaches can guarantee the feasibility of the defined DERs schedule by setting hard constraints in the mathematical formulation, which is impossible to do in current RL algorithms.

To overcome the problem mentioned above, inspired by recent advances in deep learning and optimization research areas, we first bring constraint enforcement in RL algorithms combining deep learning and optimization theory. We developed a DRL algorithm, namely MIP-DQN, that can theoretically guarantee the feasibility of the decided solution and get the optimal solution during the online scheduling stage. To do this, we redesigned the training and online-scheduling procedure. The proposed MIP-DQN algorithm uses a trained Q-network to approximate the state–action values function. Exploration and exploitation are executed based on a trained policy network to update the Q-network parameters. After training, the Q-network is assumed to approximate the optimal Q-values. Then, the trained Q-network is extracted and formulated as MIP formulation, which can be used to impose hard constraints in the action space, ensuring the feasibility of the defined schedule. In this case, the power balance constraint is used as an example to show the effectiveness of the proposed approach. Results showed that MIP-DQN strictly meets the power balance constraint, showing a lower error when compared with other DRL algorithms and the optimal global solution.

The essence of the proposed MIP-DQN algorithm is using a trained Q-network as a surrogate function for the optimal operational decisions. As above-mentioned, the optimality is defined by the Q-network modeled as a MIP formulation. Thus, the approximation quality of the Q network determines the proposed algorithm's performance. In Fig. 8, we showed that the proposed MIP-DQN could be considered a good quality operational schedule, albeit sub-optimal. Thus, efforts to reduce the error when compared with the optimal global solution must be centered on increasing the quality of the approximation of the Q-values via the used deep neural network. Additionally, the proposed MIP-DQN algorithm still needs to integrate a penalty term into the reward function to explore the right direction during the training process. This introduces extra hyperparameters that also impact the approximation performance of the obtained Q-function. An alternative exploration approach that can be used is to model the DNN as a MIP formulation in each iteration step; nevertheless, this would imply higher training time.

6. Conclusion

This paper proposed a value-based DRL algorithm, namely MIP-DQN, to define the optimal dispatch decisions of multiple distributed energy resources within a renewable-based energy system. The proposed DRL algorithm was developed for continuous action (and state) spaces with the main feature of strictly enforcing all operational constraints in the action space during online execution, ensuring the feasibility of the defined schedule. This is done by re-formulating the deep neural network (DNN), used to approximate the action-value Q-function, as a mixed-integer programming (MIP) formulation enabling to further consider any action space constraint. Results showed that the proposed MIP-DQN algorithm obtained near-optimal solutions, with an error of 13.7% when compared with the optimal solution obtained with a perfect forecast of the stochastic variables. A comparison with other DRL algorithms was also presented, observing higher errors than the proposed algorithm while failing to meet the power balance constraint on unseen test days. Future work directions include implementing *plug-and-play* features, considering DERs' uncertain availability.

CRediT authorship contribution statement

Hou Shengren: Conceptualization, Methodology, Software, Validation, Writing – original draft. **Pedro P. Vergara:** Writing – review & editing, Supervision. **Edgar Mauricio Salazar Duque:** Software, Visualization. **Peter Palensky:** Funding acquisition.

Declaration of competing interest

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

Data availability

We have shared the data and code freely online. Check the paper for the link to the source.

Appendix

A sketch of a mathematical proof that ensures that the proposed MIP-DQN model provides the optimal solution while strictly enforcing linear constraints is presented below. To do this, we first assume the feasibility to the problem presented in Section 2 and also present (and adapt notation to match this paper) the Corollary 19, from [51] as,

Corollary 1. *If the input (s, a) of the Q-network is a polytope and the DNN is a rectifier network (i.e., ReLU activation functions are used), then the mapping from input (s, a) to the output $Q(s, a)$ of such a Q-network is mixed-integer representable.*

The proof of Corollary 19 is available in [51]. Note that this corollary implies that for any rectifier DNN, a mixed-integer formulation exists as long as the input is bounded. The Q-network used in the proposed MIP-DQN algorithm is a DNN with a rectifier activation function while the input (s, a) are bounded as these correspond to the state and action variables as presented in Section 3. We denote the optimal solution to this MIP formulation as (s^*, a^*) whose optimal objective function value is $Q(s^*, a^*)$.

Now, the extended MIP formulation obtained by adding on top of the MIP representation of the $Q(s, a)$ an equality constraint (in this case, (4)) is also a feasible MIP representation. This is a consequence of the fact that such a mixed-integer representation of $Q(s, a)$ is composed of a set of linear regions whose unions form a bounded polyhedron (or polytope) (see Theorem 20 in [51]), which we denote this here as $\mathcal{S}$ (see a representation in Fig. 4). The addition of (4) to $\mathcal{S}$, which is also a linear constraint, does not modify its nature of a bounded polyhedron (or polytope).

By exhaustion, two cases are distinguished: In the first case, the extended bounded polyhedron $\mathcal{S}' = \mathcal{S} \cup (4)$ is empty, rendering the solution of the MIP unfeasible, i.e., equality constraint in (4) cannot be met. This is not possible as we assumed feasibility for the optimization

problem. In the second case, S' is not empty, in which an optimal solution exits and is feasible. If this is the case, and denoting such optimal solution as (s', a'), such solution meets the following condition: $Q(s', a') \leq Q(s^*, a^*)$. This condition simply implies that (s', a'), by meeting the equality constraint in (4), will at least have a q-value that is in the limit the same as the optimal solution $Q(s^*, a^*)$. This proves the fact that by solving the extended MIP formulation, a feasible and optimal solution that meets the equality constraint (4) is obtained. Nevertheless, it is important to highlight that optimality here relates to the good quality solution provided by the trained Q-network.

References

[1] Zia MF, Elbouchikhi E, Benbouzid M. Microgrids energy management systems: A critical review on methods, solutions, and prospects. Appl Energy 2018;222:1033–55.

[2] de Souza ACZ, Castilla M. Microgrids design and implementation. Springer; 2019.

[3] Vergara PP, López JC, Rider MJ, da Silva LCP. Optimal operation of unbalanced three-phase islanded droop-based microgrids. IEEE Trans Smart Grid 2019;10(1):928–40. http://dx.doi.org/10.1109/TSG.2017.2756021.

[4] Giraldo JS, Castrillon JA, López JC, Rider MJ, Castro CA. Microgrids energy management using robust convex programming. IEEE Trans Smart Grid 2019;10(4):4520–30. http://dx.doi.org/10.1109/TSG.2018.2863049.

[5] Yousefi M, Hajizadeh A, Soltani MN. A comparison study on stochastic modeling methods for home energy management systems. IEEE Trans Ind Inf 2019;15(8):4799–808. http://dx.doi.org/10.1109/TII.2019.2908431.

[6] Yousefi M, Hajizadeh A, Soltani MN, Hredzak B. Predictive home energy management system with photovoltaic array, heat pump, and plug-in electric vehicle. IEEE Trans Ind Inf 2021;17(1):430–40. http://dx.doi.org/10.1109/TII.2020.2971530.

[7] Vergara PP, López JC, Rider MJ, Shaker HR, da Silva LC, Jørgensen BN. A stochastic programming model for the optimal operation of unbalanced three-phase islanded microgrids. Int J Electr Power Energy Syst 2020;115:105446.

[8] Arroyo J, Manna C, Spiessens F, Helsen L. Reinforced model predictive control (RL-MPC) for building energy management. Appl Energy 2022;309:118346.

[9] Chen L, Tang H, Wu J, Li C, Wang Y. A robust optimization framework for energy management of CCHP users with integrated demand response in electricity market. Int J Electr Power Energy Syst 2022;141:108181.

[10] Su S, Li Z, Jin X, Yamashita K, Xia M, Chen Q. Energy management for active distribution network incorporating office buildings based on chance-constrained programming. Int J Electr Power Energy Syst 2022;134:107360.

[11] Chen X, Qu G, Tang Y, Low S, Li N. Reinforcement learning for decision-making and control in power systems: Tutorial, review, and vision. 2021, arXiv preprint arXiv:2102.01168.

[12] Sutton RS, Barto AG. Reinforcement learning: An introduction. MIT Press; 2018.

[13] Vázquez-Canteli JR, Dey S, Henze G, Nagy Z. CityLearn: Standardizing research in multi-agent reinforcement learning for demand response and urban energy management. 2020, arXiv preprint arXiv:2012.10504.

[14] Nakabi TA, Toivanen P. Deep reinforcement learning for energy management in a microgrid with flexible demand. Sustain Energy Grids Netw 2021;25:100413.

[15] Ji Y, Wang J, Xu J, Fang X, Zhang H. Real-time energy management of a microgrid using deep reinforcement learning. Energies 2019;12(12):2291.

[16] Wang J, Xu W, Gu Y, Song W, Green TC. Multi-agent reinforcement learning for active voltage control on power distribution networks. Adv Neural Inf Process Syst 2021;34:3271–84.

[17] Kelly A, O'Sullivan A, de Mars P, Marot A. Reinforcement learning for electricity network operation. 2020, arXiv preprint arXiv:2003.07339.

[18] Zhou Y, Zhang B, Xu C, Lan T, Diao R, Shi D, et al. A data-driven method for fast ac optimal power flow solutions via deep reinforcement learning. J Mod Power Syst Clean Energy 2020;8(6):1128–39.

[19] Pinto G, Deltetto D, Capozzoli A. Data-driven district energy management with surrogate models and deep reinforcement learning. Appl Energy 2021;304:117642.

[20] Heidari A, Maréchal F, Khovalyg D. Reinforcement learning for proactive operation of residential energy systems by learning stochastic occupant behavior and fluctuating solar energy: Balancing comfort, hygiene and energy use. Appl Energy 2022;318:119206.

[21] Liu L, Zhu J, Chen J, Ye H. Deep reinforcement learning for stochastic dynamic microgrid energy management. In: 2021 IEEE 4th international electrical and energy conference. IEEE; 2021, p. 1–6.

[22] Massiani P-F, Heim S, Solowjow F, Trimpe S. Safe value functions. IEEE Trans Automat Control 2022;1–16. http://dx.doi.org/10.1109/TAC.2022.3200948.

[23] Zhou S, Hu Z, Gu W, Jiang M, Chen M, Hong Q, et al. Combined heat and power system intelligent economic dispatch: A deep reinforcement learning approach. Int J Electr Power Energy Syst 2020;120:106016.

[24] Ji Y, Wang J, Xu J, Li D. Data-driven online energy scheduling of a microgrid based on deep reinforcement learning. Energies 2021;14(8):2120.

[25] Vergara PP, Salazar M, Giraldo JS, Palensky P. Optimal dispatch of PV inverters in unbalanced distribution systems using reinforcement learning. Int J Electr Power Energy Syst 2022;136:107628.

[26] Salazar Duque EM, Giraldo JS, Vergara PP, Nguyen P, van der Molen A, Slootweg H. Community energy storage operation via reinforcement learning with eligibility traces. Electr Power Syst Res 2022;212:108515. http://dx.doi.org/10.1016/j.epsr.2022.108515.

[27] Liu W, Zhuang P, Liang H, Peng J, Huang Z. Distributed economic dispatch in microgrids based on cooperative reinforcement learning. IEEE Trans Neural Netw Learn Syst 2018;29(6):2192–203.

[28] Du Y, Wu D. Deep reinforcement learning from demonstrations to assist service restoration in islanded microgrids. IEEE Trans Sustain Energy 2022;13(2):1062–72.

[29] Qiu D, Chen T, Strbac G, Bu S. Coordination for multi-energy microgrids using multi-agent reinforcement learning. IEEE Trans Ind Inf 2022.

[30] Yi Z, Xu Y, Wang X, Gu W, Sun H, Wu Q, et al. An improved two-stage deep reinforcement learning approach for regulation service disaggregation in a virtual power plant. IEEE Trans Smart Grid 2022.

[31] Shengren H, Salazar EM, Vergara PP, Palensky P. Performance comparison of deep RL algorithms for energy systems optimal scheduling. 2022, arXiv preprint arXiv:2208.00728.

[32] Hu B, Li J. Shifting deep reinforcement learning algorithm toward training directly in transient real-world environment: A case study in powertrain control. IEEE Trans Ind Inf 2021;17(12):8198–206. http://dx.doi.org/10.1109/TII.2021.3063489.

[33] Garcıa J, Fernández F. A comprehensive survey on safe reinforcement learning. J Mach Learn Res 2015;16(1):1437–80.

[34] Eichelbeck M, Markgraf H, Althoff M. Contingency-constrained economic dispatch with safe reinforcement learning. 2022, arXiv preprint arXiv:2205.06212.

[35] Gros S, Zanon M, Bemporad A. Safe reinforcement learning via projection on a safe set: How to achieve optimality? IFAC-PapersOnLine 2020;53(2):8076–81.

[36] Qiu D, Dong Z, Zhang X, Wang Y, Strbac G. Safe reinforcement learning for real-time automatic control in a smart energy-hub. Appl Energy 2022;309:118403.

[37] Park H, Min D, Ryu J-h, Choi DG. DIP-QL: A novel reinforcement learning method for constrained industrial systems. IEEE Trans Ind Inf 2022.

[38] Li H, He H. Learning to operate distribution networks with safe deep reinforcement learning. IEEE Trans Smart Grid 2022;13(3):1860–72. http://dx.doi.org/10.1109/TSG.2022.3142961.

[39] Li H, Wan Z, He H. Constrained EV charging scheduling based on safe deep reinforcement learning. IEEE Trans Smart Grid 2019;11(3):2427–39.

[40] Fischetti M, Jo J. Deep neural networks and mixed integer linear optimization. In: Constraints, vol. 23. 2018, p. 296–309.

[41] Watkins CJ, Dayan P. Q-learning. Mach Learn 1992;8(3):279–92.

[42] Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, et al. Human-level control through deep reinforcement learning. Nature 2015;518(7540):529–33.

[43] Ryu M, Chow Y, Anderson R, Tjandraatmadja C, Boutilier C. CAQL: Continuous action Q-learning. In: International conference on learning representations. 2020.

[44] Lim S, Joseph A, Le L, Pan Y, White M. Actor-expert: A framework for using q-learning in continuous action spaces. 2018, arXiv preprint arXiv:1810.09103.

[45] Ceccon F, Jalving J, Haddad J, Thebelt A, Tsay C, Laird CD, et al. OMLT: Optimization & machine learning toolkit. 2022, arXiv preprint arXiv:2202.02414.

[46] Shengren H, Vergara P. 2022. https://github.com/ShengrenHou/Energy-management-MIP-Deep-Reinforcement-Learning.

[47] Guo C, Wang X, Zheng Y, Zhang F. Optimal energy management of multi-microgrids connected to distribution system based on deep reinforcement learning. Int J Electr Power Energy Syst 2021;131:107048. http://dx.doi.org/10.1016/j.ijepes.2021.107048, URL: https://www.sciencedirect.com/science/article/pii/S0142061521002878.

[48] Hart WE, Laird CD, Watson J-P, Woodruff DL, Hackebeil GA, Nicholson BL, et al. Pyomo-optimization modeling in Python, vol. 67. Springer; 2017.

[49] Dalal G, Dvijotham K, Vecerik M, Hester T, Paduraru C, Tassa Y. Safe exploration in continuous action spaces. 2018, arXiv preprint arXiv:1801.08757.

[50] Ray A, Achiam J, Amodei D. Benchmarking safe exploration in deep reinforcement learning, 7. 2019, p. 1, arXiv preprint arXiv:1910.01708.

[51] Serra T, Tjandraatmadja C, Ramalingam S. Bounding and counting linear regions of deep neural networks. In: International conference on machine learning. PMLR; 2018, p. 4558–66.