Machine Learning with Applications 24 (2026) 100864

Contents lists available at ScienceDirect

Machine Learning with Applications

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

AI-powered predictive control for hybrid renewable microgrids: Integrating photovoltaic generation and battery storage in smart homes and industrial applications

Sebastián López Flórez [a,b,*], Guillermo Hernández [a], Javier Prieto [a], Fernando de la Prieta [a]

[a] BISITE Digital Innovation Hub, University of Salamanca, Edificio Multiusos I+D+i, Calle Espejo 2, 37007 Salamanca, Spain
[b] Universidad Tecnológica de Pereira, Cra. 27 N 10-02, Pereira, Risaralda, Colombia

ARTICLE INFO

Keywords:
Energy management
Energy efficiency optimization
Risk planning
Reinforcement learning

ABSTRACT

The integration of renewable energy sources into microgrids remains challenging due to generation intermittency, storage inefficiencies, and progressive battery degradation. In this work, our main contribution is a standardized empirical benchmark of deep reinforcement learning for microgrid energy management across multiple operating regimes under a consistent evaluation protocol. To make these evaluations rigorous and comparable, we define a common state/action/reward specification that explicitly encodes battery-health proxies, dynamic pricing, and operational constraints, and we adapt a representative set of RL families accordingly: value-based methods (DoubleDQN, NoisyNet-DQN, PER-DQN, C51), policy-gradient approaches (PG), and actor–critic algorithms (PPO, A2C, SAC, DDPG, A3C-Energy). We benchmark the resulting agents on real-world driven datasets and environments (GEFCom2014, StoreNet, and a CityLearn-inspired setting), assessing performance under a consistent evaluation pipeline. Empirically, the reported results indicate that the adapted agents tend to reduce daily operating cost, smooth charge–discharge cycling, and improve battery-health proxies under realistic operating conditions, with gains that vary by dataset and operating regime. In particular, DQN-based variants exhibit consistent gains over standard exploration schemes, while methods such as PPO and actor–critic configurations maintain competitive and more stable behavior under highly volatile price signals and renewable fluctuations. Overall, the study provides scenario-conditional evidence on which RL families tend to perform well or poorly under the tested conditions, rather than a universal ranking across datasets.

1. Introduction

The energy industry seeks to enhance efficiency and profitability while reducing emissions and promoting renewable energy sources to facilitate a sustainable transition. However, significant challenges arise when integrating these new sources into existing systems. Energy losses during storage, the intermittent nature of solar energy, and transmission line losses complicate the overall energy landscape. Batteries that undergo repeated cycles of charging and discharging experience degradation, which reduces their lifespan and efficiency. This degradation increases operational costs and limits the widespread adoption of energy storage systems. As a result, these factors hinder the overall optimization of energy systems and present challenges for microgrids, particularly when they need to operate independently or in conjunction with the main energy distribution grid. It is essential to develop energy management systems (EMS) focused on efficient energy distribution to address these issues. These systems aim to improve battery lifespan while effectively managing the state of charge (SOC) of supercapacitors. EMS solutions tackle real-time energy management challenges in hybrid battery-supercapacitor energy storage systems (BS-HESS) (Gassi & Baysal, 2023; Nambisan & Khanra, 2024).

The key to their effectiveness lies in implementing penalty-based strategies considering energy trading costs, battery energy storage system (BESS) operating expenses, penalties for exceeding capacity limitations or failing to store excess renewable energy, and rewards for maintaining optimal SOC levels. By balancing these variables through modeling equations, these systems aim to improve overall efficiency, extend battery life, ensure grid stability, and optimize operating costs, marking a significant step toward realizing next-generation smart grids (Wu & Wang, 2018). The objective of the Microgrid energy management strategy is to optimize the use of distributed energy while minimizing long-term operating costs through electricity trading on the distribution

* Corresponding author at: BISITE Digital Innovation Hub, University of Salamanca, Edificio Multiusos I+D+i, Calle Espejo 2, 37007 Salamanca, Spain.
E-mail addresses: sebastianlopezflorez@usal.es (S.L. Flórez), guillehg@usal.es (G. Hernández), javierp@usal.es (J. Prieto), fer@usal.es (F. de la Prieta).

https://doi.org/10.1016/j.mlwa.2026.100864
Received 30 September 2025; Received in revised form 28 January 2026; Accepted 8 February 2026
Available online 23 February 2026

Nomenclature

α	Deviation from ideal p–n junction	β	Action space boundaries
η	Efficiency	γ	Discount factor
λ	Local Tip Speed Ratio	π	Policy
ρ	Density (kg/m^3)	τ	Trajectory
a_{ang}	Angular induction factor	$\mathcal{A}$	Set of actions
a_{axial}	Axial induction factor	$\mathcal{S}$	Set of states
C_n	Battery nominal capacity	$\mathcal{T}$	Maximum episode length
C_p	Specific heat (J/kg K)	E_r	Grid energy (kWh)
COP	Coefficient of Performance	E_{ES}	Stored energy (kWh)
E	Energy (J or Wh)	E_{pv}	Renewable energy (kWh)
F	Correction factor	E_{th}	Thermal energy (kWh)
I	Current (A)	R	Reward function
k	Boltzmann's constant	P	State transition probability
$\dot{m}$	Mass flow rate (kg/s)	L	Non-shiftable loads (kWh)
P	Power (W)	K	Temporal features set
Q	Thermal energy (J)	H	State of charge
S	Solar irradiation	C	Energy cost ($/kWh)
SoC	State of Charge (%)	v	Speed (m/s)
T	Temperature (K or °C)	W	Compressor power (W)
V	Voltage (V)	A	Area (m^2)
		C	Condenser capacity (J)

grid (Affi et al., 2021). To facilitate this, the IEC 61970 standard defines the communication protocols and data models necessary for efficient information exchange between the EMS and various electrical grid or microgrid components. By providing a standardized communication framework, IEC 61970 ensures that the EMS can effectively optimize energy distribution across the grid or microgrid (IEC61970, 2003).

Traditional optimization methods, such as mixed-integer linear, dynamic, and stochastic linear programming (Anderson & Fok, 2000; Cardoso et al., 2013; Sobu & Wu, 2012), are well established for sequential decision problems, but they typically solve each instance from scratch. Conventional EMS controllers such as PID, MPC, and dynamic programming (DP) face practical limitations in PV–BESS microgrids. PID-type strategies are lightweight but require careful tuning and typically struggle under nonstationary conditions (volatile prices, uncertain PV/load) and multi-objective goals such as cost–degradation trade-offs. MPC can explicitly handle constraints, yet its performance depends on an accurate model and repeated online optimization, which can be computationally demanding and brittle under modeling errors. DP offers optimality in principle, but it relies on a known transition model and suffers from the curse of dimensionality as state and uncertainty grow. In contrast, reinforcement learning learns a control policy directly from historical interaction data, so once trained, it can produce fast real-time decisions without re-solving an optimization problem at every step, while naturally incorporating complex objectives and penalty terms under uncertainty (Liu et al., 2023). This restart requirement imposes substantial computational overhead and undermines real-time applicability (Zhang et al., 2019), a critical limitation in microgrid energy management where operating conditions shift rapidly. Heuristic approaches such as genetic algorithms (Chen et al., 2011) and particle swarm optimization (Liu et al., 2018) offer simplicity and flexibility, yet they often lack robustness and generalization, yielding suboptimal behavior in complex, nonstationary energy systems. These limitations motivate the adoption of reinforcement learning (RL) for adaptive control.

Within this context, the literature reveals several persistent gaps. First, a broadly agreed-upon evaluation protocol for BESS control is still missing, which can hinder fair and reproducible comparisons across methods when experimental setups differ. Second, although some studies include degradation costs, these are rarely computed explicitly or elevated to a primary objective; as a result, models neglect a key driver of long-term storage performance. Third, reward and penalty formulations are frequently under-specified: a coherent set of equations that accounts for losses, gains, and constraint violations is often missing, complicating both implementation and analysis. Finally, while reinforcement learning appears in multiple works, replication is hindered by opaque formulations and the absence of standardized datasets, making it difficult to assess whether a reported policy is genuinely a state-of-the-art estimator under diverse operating conditions.

In the context of battery management, the EMS utilizes real-time data, such as SoC and energy demand, to make decisions that optimize energy flow and extend the lifespan of energy storage devices. Cao et al. (2020) proposes a BESS operation strategy based on DRL using a NoisyNet-DDQN, combined with a hybrid CNN+LSTM model to forecast next-day hourly prices. The EMS is formulated as an MDP where the agent observes the SoC, predicted prices, and operational limits, but acts through a reduced set of discrete actions, limiting its ability to produce fine-grained or physically coherent control decisions. Ref. Liu et al. (2023) handles wind power uncertainty through a two-step scheme: an LSTM-LUBE model generates wind power intervals, which are then used in an Expected Risk Minimization (ERM) problem solved via Deep Q-Learning. The agent selects among 11 discrete charge/discharge actions and models aging through a coefficient k linked to depth-of-discharge and cycle count. In Shang et al. (2020), a framework is proposed that combines Q-learning with Monte Carlo Tree Search (MCTS) to enhance exploration and reduce the computational complexity of BESS dispatch. MCTS builds a decision tree constrained by fuzzy rules that guide the search toward plausible actions, effectively reducing the action space and improving learning efficiency. However, across these lines of work, control actions remain discretized, a fundamental limitation of DQN-based methods rather than continuous or ramp-rate-aware, leading to coarse and often non-physical dispatch decisions (Sage et al., 2024). Furthermore, battery aging is typically encoded as a simple reward penalty or exogenous cost rather than through explicit SoH, thermal, or C-rate dynamics, despite well-established evidence that degradation is nonlinear and strongly dependent on temperature, C-rate, depth-of-discharge and, the operative SoC window (Terkes et al., 2024). Even when nonlinear, non-convex aging mechanisms are acknowledged, solutions often augment Q-learning with MCTS and fuzzy rules while still relying on discrete action spaces, leaving key operational constraints—such as SoC bounds coupled with ramprate, C-rate, and thermal limits unenforced during learning (Shang et al., 2020). The safe RL literature likewise highlights

Table 1
Structural comparison of key related works.
Source: Updated with Ali et al. (2024).

	Cao 2020	Liu 2023	Ali 2024
Task	BESS arbitrage	Wind EMS	Dairy farm PV/Wind + BESS
Algorithm	NN–DDQN	DQN (dynamic ϵ, PER)	Q-learning (tabular)
Actions	Discrete(5)	Discrete(11)	Discrete(3)
Forecasting	Day-ahead price CNN–LSTM	Wind intervals (LSTM–LUBE)	–
Degradation	Rainflow (pen.)	Linear penalty	–
Efficiency	Nonlinear (SoC/C-rate)	Fixed (90%/90%)	Fixed
Constraints	SoC, P	SoC, P; exclusivity $P_{\text{dis}}P_{\text{ch}} = 0$	SoC, P (Powerwall 13.5 kWh/5 kW)
Validation	1 market (UK; 2015/2016)	1 dataset (TR; 2018, 8016 h)	FI (train/test), IE (test); simulation
Baselines	MILP, DQN, DD-DQN	DQN variants (no/point/interval)	MSC, TOU
Objectives	Cost	Cost (purchase+wear+penalties)	Cost; peak demand

the necessity of CMDP-style constraint handling to ensure feasibility, an element generally absent from these DQN-based formulations. In the residential domain (Su et al., 2025).

Discussion of Table 1. From the comparison, a clear methodological pattern emerges: action parameterization is discrete, which simplifies value-based learning but precludes continuous, ramp rate/C rate aware control and thus lowers the physical fidelity of dispatch; battery degradation is injected as a reward-cost term, with no explicit SoH state or thermo-electrochemical coupling, and validation concentrates on single-market/dataset simulation with classic baselines (MILP, DQN variants). This combination DQN/DQL with discrete actions + coupled forecasting + degradation as penalty defines the immediate SOTA in EMS and explains the gaps in physical realism, robustness, and policy transferability (Ali et al., 2024; Cao et al., 2020; Liu et al., 2023).

In the residential domain, Liu et al. (2020) employs DQN and DDQN within an HEMS to control schedulable loads and the home battery. The model incorporates penalties for violating user preferences and aims to maximize PV self-consumption, using real-world data. However, the action space for the BESS remains discrete and binary, and the absence of degradation modeling allows the agent to learn policies that increase cycling without accounting for their impact on battery lifetime. Finally, Slama and Mahmoud (2023) introduces an IHEM that applies Q-learning with a neural network to optimize load scheduling and battery operation over discrete time intervals. The problem is formulated as an MDP in which the agent selects among categorical actions representing combinations of appliance operation and BESS states. The approach achieves nearly a 20% cost reduction compared to classical Integer Linear Programming (ILP) methods, demonstrating greater adaptability to variations in demand and renewable generation. However, the model lacks explicit representation of battery aging and omits key dynamic constraints, such as ramp limits, charge/discharge rates, and thermal effects, which limit the transferability of the learned policy to real-world scenarios where adherence to physical operating constraints is essential.

Recent research in battery technologies and energy management systems has led to significant developments that enhance safety, efficiency, and sustainability. A study discussed in Zhao et al. (2024) applies the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm to microgrid EMS, formulating it as a partially observable Markov decision process (POMDP) to capture nonlinear dynamics and battery losses. The TD3-based controller enhances control stability and adaptability, achieving lower operating costs and higher energy efficiency in both residential and large-scale microgrid scenarios, demonstrating the efficacy of continuous-action DRL for realistic microgrid management. In Lin et al. (2023), a method is introduced for predicting the long-term degradation trajectory of Lithium-ion (Li-ion) batteries using deep learning techniques. The proposed approach leverages a transferred CNN model trained on a synthetic dataset generated via a polynomial function. This method enables early-stage prediction of battery degradation with minimal data from the initial cycling period, offering a reliable solution for BESS maintenance and optimization. Additionally,

The Table 2 highlights a clear shift from value-based DQN toward actor–critic control with continuous actions. These methods improve physical fidelity via smoother set-points and stronger optimization dynamics; yet they still lack explicit, enforceable physical guarantees and SoH-aware modeling. Closing this gap will require safe-RL layers and degradation-coupled state representations to ensure robustness and transferability on real assets (Li et al., 2025; Wei et al., 2023; Zhao et al., 2024).

Motivated by these identified research gaps, our proposal aims to leverage the scenario-robust performance of RL by adapting and testing various architectural frameworks to tailor RL methodologies to the unique challenges of BESS operation under diverse price and renewable conditions. Data-driven learning has also been adopted broadly across engineering domains through ensemble and optimization-based pipelines (Çiftçioğlu et al., 2025; Shafighfard et al., 2025, 2024).

The main contributions of this work are:

- **Development of an adaptive RL-based control framework:** We formulate a unified EMS environment with a common state /action/reward specification to enable reproducible comparisons of RL controllers under consistent training and evaluation protocols across multiple datasets and operating regimes (price volatility, renewable intermittency, and demand variability).
- **Unified cost and penalty formulation:** We develop a clear, modular cost–penalty formulation that combines energy trading costs, battery usage/degradation considerations, and operational constraints into a single reward signal. Our intent is not to present a one-size-fits-all objective, but to provide a consistent baseline and then test alternative term combinations/weights to determine which formulation best matches the needs of the different use cases considered.
- **Comprehensive empirical validation and statistical evaluation:** Through consistent benchmarking across heterogeneous datasets, we characterize when value-based methods tend to be competitive in cost-oriented objectives and when policy-gradient/actor–critic methods tend to be more stable under volatile prices and renewable fluctuations, highlighting regime-dependent strengths and failure modes.

Accordingly, this study asks whether an adaptive RL framework with standardized penalty modeling can outperform conventional EMS optimization in terms of cost efficiency and SOC stability, while accounting for battery degradation and market trading dynamics across real-world, high-resolution datasets.

This paper presents a comprehensive RL framework for microgrid optimization, structured in three main sections: Section 2 formulates the energy management problem as an MDP with a state space that explicitly accounts for battery degradation. Section 3 evaluates multiple RL algorithms across two real-world datasets. In Database 1, the baseline DQN achieves the highest economic performance in terms of daily net benefit (69.72 ± 2.86 USD/day), exceeding literature baselines such as MGO (69.42 USD/day) and DQN from Hau et al. (2020) (53.45 USD/day), while PER, C51, and PPO obtain lower net benefits ($59.22 \pm$

Table 2
Condensed comparison of recent continuous-action DRL methods.

	Zhao 2024	Wei 2023	Li 2025
Task	Microgrid EMS	Smart home EMS	Wind–BESS dispatch
Algorithm	TD3	PPO with mixed output policy	SAC
Actions	Continuous	Discrete + Continuous	Continuous
Forecasting	–	–	LSTM wind forecast
Degradation	Nonlinear losses	SoC/DoD cost terms (poly fit)	Simple penalties (no SoH)
Constraints	SoC, Power	SoC, Power	SoC, Power; no ch/dis simult.
Validation	Simulation	Simulation	Real data + simulation
Objectives	Cost, efficiency	Cost, stability+ ESS degradation	Forecast-error penalties; ESS performance

1.36, 51.02 ± 0.83, and 49.27 ± 1.41 USD/day, respectively). Complementarily, robust aggregate statistics of normalized performance indicate that C51 attains the highest central-tendency estimates (Median 50.41 ± 0.49, IQM 50.31 ± 0.55), followed closely by PPO and PER, whereas the rule-based controller is consistently worst. In Database 2, policy-gradient methods achieve the best relative economic outcome, with PG reaching the highest net benefit (€8,321.07; €22.74/day), slightly above DQN (€7,809.31; €21.34/day). Section 4 concludes that value-based methods provide strong economic performance and stability under the proposed protocol, while standardized evaluation (including uncertainty-aware reporting) remains essential for reliable industrial deployment.

2. Dynamic BESS control framework

The model of the solar microgrid system includes a series of photovoltaic solar panels interconnected with a BESS. The key components of the solar microgrid system include a DC/AC converter, which ensures compatibility of the generated energy with residential and industrial loads. Interaction with the main electrical grid is managed through a Real-Time Pricing (RTP) mechanism, dynamically adjusting costs to reflect current demand and supply conditions. The intelligence center of the EMS optimally controls the flow and distribution of electricity, adjusting operations based on multiple real-time data inputs such as solar radiation, ambient temperature, and fluctuating energy prices.

2.1. State space

The operational efficacy of BESS within microgrids is fundamentally tied to their ability to regulate charge/discharge cycles, a process governed by dynamic control mechanisms that balance energy flow while maintaining system stability. At the core of this regulation lies the state of charge (SoC), an internal state variable that evolves in response to both controllable actions (charge/discharge commands) and uncontrollable external conditions (load demand, renewable generation, and market prices).

To formalize this interplay, we define the microgrid's operational state at time t through a state space $\mathcal{S}$, which incorporates five key variables capturing both the stochastic environment and controllable energy dynamics. These variables electrical load demand (P_t^{Load}), photovoltaic generation (P_t^{PV}), real-time electricity price (P_t^{Price}), its 24-hour rolling average (P_t^{Asp}) are calculated at each time step (Liu et al., 2023), and the battery's state of charge (SOC_t)fully characterize the system's evolution under both controllable actions and external stochastic conditions:

$$s_t \in \mathcal{S} = \left[P_t^{\text{Load}},\ P_t^{\text{PV}},\ P_t^{\text{Price}},\ P_t^{\text{Asp}},\ \text{SOC}_t \right]^{\top}. \tag{1}$$

The auxiliary price signal P_t^{Asp} is defined as a 24-hour rolling average:

$$P_t^{\text{Asp}} = \frac{1}{H} \sum_{k=0}^{H-1} P_{t-k}^{\text{Price}}, \qquad H = \frac{24}{\Delta t}, \tag{2}$$

where Δt denotes the control interval in hours.

The SoC dynamics are modeled as

$$\text{SOC}_{t+1} = \text{SOC}_t + \frac{P_{\text{rated}} \cdot \Delta t}{E_{\text{rated}}} a_t \left[\eta_t^c \, \mathbb{1}(a_t \geq 0) + (\eta_t^d)^{-1} \, \mathbb{1}(a_t < 0) \right], \tag{3}$$

where $a_t \in [-1, 1]$ is the normalized charge/discharge command, P_{rated} and E_{rated} denotes the rated power and energy of the BESS, and η_t^c, η_t^d are the charge and discharge efficiencies at time t.

Following Jeong et al. (2023), these efficiencies are computed at the *cell level*. Let v_t^{oc} be the open-circuit voltage of a single cell at SOC_t, and $P_{t,\text{cell}}^c$, $P_{t,\text{cell}}^d \geq 0$ denote the charging and discharging powers per cell, respectively. Let $i_{t,\text{cell}}^c$ and $i_{t,\text{cell}}^d$ be the corresponding cell currents obtained from the equivalent-circuit model. The charge and discharge efficiencies are then given by

$$\eta_t^c = \frac{v_t^{oc} \, i_{t,\text{cell}}^c}{P_{t,\text{cell}}^c}, \qquad \eta_t^d = \frac{P_{t,\text{cell}}^d}{v_t^{oc} \, i_{t,\text{cell}}^d}, \tag{4}$$

which correspond directly to Eqs. (10a)–(10b) in Jeong et al. (2023).

The rated power P_{rated} and capacity E_{rated} specify the maximum power throughput and total usable energy of the BESS, while the dispatch interval Δt determines how often the SoC is updated. Together with the nonlinear efficiencies η_t^c and η_t^d, this formulation provides a physically consistent description of the charge/discharge process, enabling the RL controller to account for realistic conversion losses and battery behavior (Chen et al., 2011; Liu et al., 2023).

2.2. Action space

The action space (a_t) is defined by eleven discrete levels representing charging and discharging operations for the BESS. Positive values correspond to charging states, while negative values indicate discharging. The available actions are evenly distributed between −1 and 1, including zero as a neutral state:

$$a_t \in A = \{-1, -0.8, \ldots, 0, \ldots, 0.8, 1\} \tag{5}$$

2.3. Reward function

The equations used in this paper are largely derived from the paper (Liu et al., 2023), where the reward function is designed to incentivize the agent to minimize the operating costs of the microgrid by meeting the system constraints. Key components of the reward function include:

1. **Net Energy Trading Cost**: This term captures the net cost (or revenue) of energy exchanged with the main grid through the BESS in the interval $[t, t+1]$ and is defined as.

$$Cost_t^{\text{trade}} = P_t^{\text{Price}} \left(SOC_{t+1} - SOC_t \right) E_{\text{rated}}, \tag{6}$$

where P_t^{Price} is the real-time electricity price and E_{rated} is the nominal energy capacity of the battery. Positive values of $Cost_t^{\text{trade}}$ correspond to net energy imports (charging, $SOC_{t+1} > SOC_t$), whereas negative values indicate net exports (discharging, $SOC_{t+1} < SOC_t$), i.e., revenue from selling energy.

2. **BESS Operation Cost**: This component quantifies the operational expenditures of the BESS, including all costs related to charge/discharge cycling. It incorporates efficiency losses, cycle-based degradation effects, and maintenance requirements necessary to maintain system performance throughout the BESS lifecycle.

$$Cost_t^{op} = k \times |SOC_{t+1} - SOC_t| \times E_{rated} \tag{7}$$

$$k = \frac{C_i}{\eta_d \times E_{rated} \times \delta \times N_c} \tag{8}$$

The operational cost at any time t is given by $C_{\text{op},t} = k \cdot E_{\text{rated}} \cdot |\text{SOC}_{t+1} - \text{SOC}_t|$, where k is a coefficient reflecting the cost associated with the change in the SOC, $|\text{SOC}_{t+1} - \text{SOC}_t|$ represents the absolute change in SOC between consecutive time intervals, and E_{rated} is the rated energy capacity of the BESS. In addition, the total cost of the BESS considers the initial investment K, where C_i is the initial investment of the BESS, η_d is the discharging efficiency, E_{rated} is the rated capacity of the BESS, δ denotes the depth of discharge (DoD), and N_c represents the life cycle at the rated DoD.

3. **Battery degradation penalty**:
To discourage operation in harmful SOC regions and account for the long-term degradation of the BESS, we replace the constant capacity violation term with a degradation-aware penalty inspired by Kim and Shin (2023). The penalty associated with the BESS at time t is defined as where Pnt_t^{BESS} is a piecewise function of the next SOC value:

$$Pnt_t^{BESS} = \begin{cases} \varphi_1\, c_{d,k}\, C_B, & SOC_{\min} < SOC_{t+1} \le 0.4, \\ 0, & 0.4 < SOC_{t+1} \le 0.6, \\ 0, & 0.6 < SOC_{t+1} \le 0.8, \\ \varphi_2\, c_{d,k}\, C_B, & 0.8 < SOC_{t+1} \le SOC_{\max}, \end{cases}$$

with C_B [€/kWh] denoting the investment cost of the BESS, $c_{d,k}$ the degradation coefficient at episode k (computed from the depth-of-discharge and partial-cycling indices), and φ_1, φ_2 tuning parameters that control the penalty severity in low and high SOC regions, respectively. The SOC is saturated to the admissible range $[SOC_{\min}, SOC_{\max}]$ in the state update, so explicit hard penalties for exceeding the physical limits are no longer required.
Following the battery aging model in Kim and Shin (2023), the degradation coefficient $c_{d,k}$ used in our BESS penalty is computed from the DOD and a partial cycling (ρ_k) index. Let $\{SOC_t\}_{t=0}^{T_k}$ denote the SOC trajectory during episode k. First, the DOD is defined as

$$D_k = \max_t SOC_t - \min_t SOC_t, \tag{9}$$

The SOC range is then partitioned into four bands $x \in \{A, B, C, D\}$, i.e., $[0.8, 1.0]$, $[0.6, 0.8)$, $[0.4, 0.6)$, and $[0.0, 0.4)$ in p.u. In our discrete-time implementation, the fraction of time spent in each band is computed as

$$\rho_{A,k} = \frac{1}{T_k + 1} \sum_{t=0}^{T_k} \mathbf{1}\{0.8 \le SOC_t \le 1.0\},$$

and analogously for $\rho_{B,k}, \rho_{C,k}, \rho_{D,k}$ using the corresponding SOC intervals. The overall PC index is then given by

$$\rho_k = a\,\rho_{A,k} + b\,\rho_{B,k} + c\,\rho_{C,k} + d\,\rho_{D,k}, \tag{10}$$

Finally, the degradation coefficient is obtained as

$$c_{d,k} = \frac{\rho_k}{D_k}, \tag{11}$$

In accordance with Equation (10) from Kim and Shin (2023), the state of charge values are stored in a history array, from which D_k is calculated as $\max_t SOC_t - \min_t SOC_t$. Meanwhile, the fractions $\rho_{A,k}, \rho_{B,k}, \rho_{C,k}, \rho_{D,k}$ are determined as empirical time averages over the four SOC bands. We emphasize that this degradation formulation is a simplified proxy based on SOC-band exposure and cycle/DoD-related coefficients, rather than a full electrochemical aging or state-of-health (SoH) evolution model. Accordingly, the absolute magnitude of degradation-related costs and, potentially, relative method rankings may vary under alternative aging models or parameterizations.

4. **Penalty for Not Storing Excess PV Energy**: If there is spare BESS capacity but the agent fails to store surplus wind energy, a penalty is imposed:

$$Pnt_{\text{PV}_t} = \begin{cases} 0 & \text{if } P_t^{PV} \le (P_t^{Load} + a_t \times P_{rated}) \\ \exp(2.5 \times (1 - SOC_{t+1})) - 1 & \text{if } P_t^{PV} > (P_t^{Load} + a_t \times P_{rated}) \end{cases}$$

The penalty for not storing excess PV energy is denoted as Pnt_{PV_t} and is adapted from the proposal in Liu et al. (2023). As in the original formulation, the term is an exponential function of the remaining BESS capacity: when spare capacity is high, the penalty is large, and as the BESS capacity decreases, this penalty is gradually reduced. In this way, the function encourages the system to prioritize storing excess renewable generation whenever possible.

5. **Reward for Maintaining Optimal SOC**: A reward or penalty is given based on how close the SOC is to the target SOC for emergency reserves:

$$Pnt_{\text{res}_t} = M \times (\text{SOC}_{t+1} - \text{SOC}_t) \times E_{\text{rated}} \times P_{\text{ravg}_t}$$

where M is a piecewise multiplier that controls the reward/penalty based on the deviation from the target SOC.

$$M = \begin{cases} 1 + m_{bef}^{upp} \times \exp\left(\ln\left(\frac{m_{bef}^{low}-1}{m_{bef}^{upp}}\right) \times \frac{SOC_t}{SOC_{target}}\right) & \text{if } SOC_t \le SOC_{target} \\ \exp\left(\ln\left(m_{aft}^{ch}\right) \times \frac{SOC_t - SOC_{target}}{1 - SOC_{target}}\right) & \text{if } SOC_t > SOC_{target} \;\&\; a_t \ge 0 \\ m_{aft}^{dis} \times \exp\left(\ln\left(\frac{1}{m_{aft}^{dis}}\right) \times \frac{SOC_t - SOC_{target}}{1 - SOC_{target}}\right) & \text{if } SOC_t > SOC_{target} \;\&\; a_t < 0 \end{cases}$$

The parameters $m_{\text{bef}}^{\text{upp}}$, $m_{\text{bef}}^{\text{low}}$, $m_{\text{aft}}^{\text{ch}}$, and $m_{\text{aft}}^{\text{dis}}$ control the penalty and reward system for managing the SOC in the BESS. $m_{\text{bef}}^{\text{upp}}$ and $m_{\text{bef}}^{\text{low}}$ adjust the penalties when SOC is below the target, encouraging charging. $m_{\text{aft}}^{\text{ch}}$ rewards storing excess energy when SOC is above the target during charging, while $m_{\text{aft}}^{\text{dis}}$ controls penalties for discharging when SOC exceeds the target, preventing over-discharging and ensuring optimal energy reserves (Aih, 2020). These parameters were tuned empirically, replicating the functional behavior of the multiplier curves described in Hau et al. (2020), to balance energy trading efficiency and battery health.

The reward function developed in this work is a multi-objective scalar, R_3, which balances several key operational objectives by means of weighting coefficients $\lambda_{\bullet}$. It is defined as

$$r(s_t, a_t) = -\Big(\lambda_{\text{trade}}\, Cost_t^{\text{trade}} + \lambda_{\text{op}}\, Cost_t^{\text{op}} + \lambda_{\text{deg}}\, Pnt_t^{\text{BESS}} + \lambda_{\text{PV}}\, Pnt_t^{\text{PV}} + \lambda_{\text{res}}\, Pnt_t^{\text{res}}\Big), \tag{12}$$

where $Cost_t^{\text{trade}} = P_t^{\text{Price}}(SOC_{t+1} - SOC_t)\, E_{\text{rated}}$ is the net energy trading cost with the main grid, $Cost_t^{\text{op}}$ represents the operational cost of the BESS (including wear-related degradation), Pnt_t^{BESS} is the degradation-aware penalty associated with harmful SOC operation, Pnt_t^{PV} penalizes curtailment of excess renewable generation, and Pnt_t^{res} implements the reserve-related SOC shaping term.

The weighting factors λ_{trade}, λ_{op}, λ_{deg}, λ_{PV} and λ_{res} control the relative importance of economic performance, battery degradation, renewable utilization, and SOC reserve management in the overall

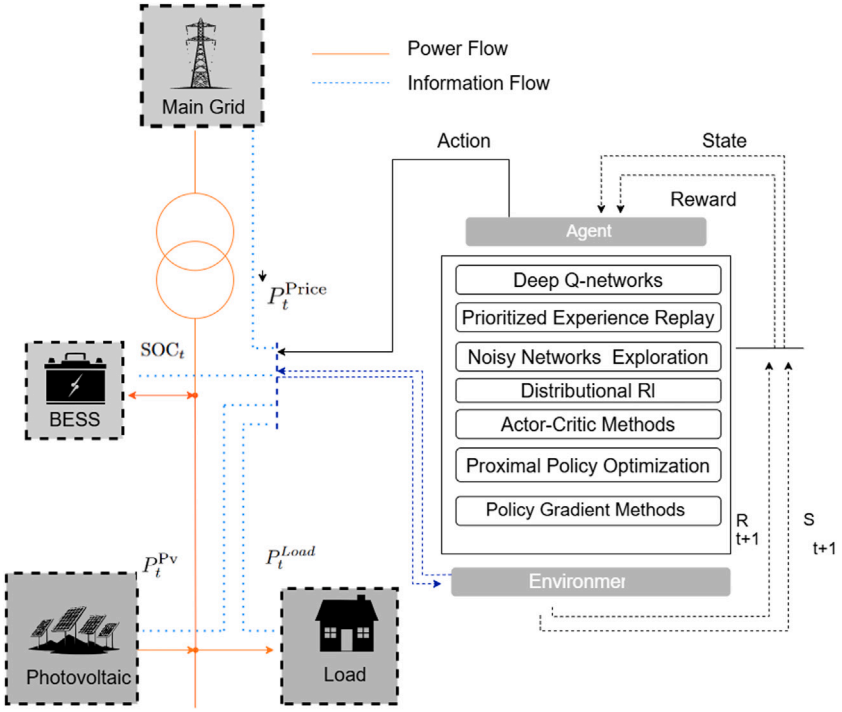

Fig. 1. Reinforcement learning for optimal scheduling of BESS in a PV-Integrated system.

objective. In the implementation, these correspond to the hyperparameters λ_{econ}, λ_{wear} and λ_{deg} used to scale the net trading term, the wear-related cost, and the degradation penalty, respectively.

Qualitatively, the first two terms ($Cost_t^{trade}$ and $Cost_t^{op}$) encourage the agent to minimize the economic operating cost and battery throughput-related wear, whereas the remaining terms ($P_{nt,t}^{\text{BESS}}$, $P_{nt,t}^{\text{PV}}$, and $P_{nt,t}^{\text{res}}$) enforce safe and efficient SoC behavior by discouraging operation in degradation-prone SOC regions, promoting storage of excess PV generation (reducing curtailment), and maintaining sufficient reserve energy for reliability.

To maintain the BESS's efficiency and prolong its operational life, it is critical to avoid scenarios of overcharging (SOC exceeding SOC_{max}) or excessive discharging (SOC falling below SOC_{min}). Overcharging can result in thermal stress and degradation of the storage medium, while deep discharging may cause irreversible capacity loss or lower the efficiency of energy cycles.

$$\text{SOC}_{\text{min}} < \text{SOC}_t < \text{SOC}_{\text{max}}$$

where SOC_{max} and SOC_{min} represent the maximum and minimum energy storage capacities, respectively. These limits ensure that the BESS operates within safe and efficient ranges, prolonging its operational lifespan and maintaining system reliability.

3. Reinforcement learning approach

In this work, energy transaction decisions are modeled as an MDP with a continuous state space S, a discrete action space a, a reward function r, and a discount factor $\gamma \in [0, 1]$. The state at each time step t is defined as S_t, where P_{r_t} represents the real-time electricity price, P_{ravg_t} is the average electricity price over a period, and SOC_t denotes the state of charge of the BESS. The action a_t determines whether to charge or discharge the BESS based on the observed state, while the reward function $r_t = u(S_t|a_t)$ evaluates the immediate benefit of an action to optimize energy costs and storage utilization. As illustrated in Fig. 1, the learning process follows a trial-and-error approach, where the agent receives a reward upon taking an action and updates its state accordingly, progressively refining its decision-making strategy.

In this section, we provide a detailed description of the DRL techniques applied to energy management. These methods are designed to optimize decision-making processes in microgrids.

Exploration–Exploitation Strategy

In most DRL algorithms applied in this context, a balance between exploration (discovering new strategies) and exploitation (utilizing known strategies) is achieved through an ϵ-greedy policy. During the training phase, at each iteration k, the agent chooses a random action a_t with probability ϵ, while with probability $1 - \epsilon$, it selects the action that maximizes the expected reward based on the current policy:

$$\pi(a_t|s_t) = \begin{cases} \text{random action} & \text{with probability } \epsilon \\ \arg\max_a Q(s_t, a) & \text{with probability } 1 - \epsilon \end{cases}$$

The value of ϵ decays over time according to:

$$\epsilon_k = \epsilon_{\text{min}} + (\epsilon_{\text{max}} - \epsilon_{\text{min}}) \cdot \exp^{(-\lambda \cdot k)}$$

where λ controls the rate of decay, and ϵ_{max} and ϵ_{min} represent the initial and final exploration rates, respectively. This strategy encourages the agent to explore more in the early stages of training and gradually shift toward exploiting the learned policy as the training progresses.

3.1. Benchmark DQN

DQN uses a neural network $Q_\theta(s, a)$ with parameters θ to approximate the action–value function $Q(s, a)$, which estimates the expected cumulative reward of taking action a in state s. The network is trained using the standard Bellman update:

$$Q(s_t, a_t) \leftarrow r_t + \gamma \max_{a'} Q_{\theta'}(s_{t+1}, a'),$$

where $\gamma \in [0, 1]$ is the discount factor and $Q_{\theta'}$ is a periodically updated target network.

Double DQN configuration. We also enable a *Double DQN* variant by replacing the target with

$$y_t^{\text{DDQN}} = r_t + \gamma\, Q_{\theta'}\big(s_{t+1}, \arg\max_{a'} Q_\theta(s_{t+1}, a')\big),$$

using the online network Q_θ for action selection and the target network $Q_{\theta'}$ for evaluation. The rest of the training loop (experience replay, target updates, optimizer) remains unchanged.

Algorithm 1 Modified Deep Q-Learning with Strategic Exploration

Input:
  Main Q-network parameters θ, target Q-network parameters θ_{target}
  Replay buffer capacity N, minibatch size B, exploration parameters (ϵ_{start}, ϵ_{stop}, λ)
Initialize:
  Set $\theta_{\text{target}} \leftarrow \theta$
  Replay buffer $\mathcal{D} \leftarrow \emptyset$, $\epsilon \leftarrow \epsilon_{\text{start}}$
  Decay rate λ, total steps $T_{\max}$, exploration steps T_{explore}
for episode = 1 to M **do**
  Reset environment, observe initial state s_0
  for $t = 1$ to T_{episode} **do**
    Strategic Action Selection:
    if with probability ϵ **or** $t < T_{\text{explore}}$ **then**
      Exploration:
        Sample random action $a_t \neq a^*$ where $a^* = \arg\max_a Q(s_t, a; \theta)$ ▷ Avoids greedy action
    else
      Exploitation:
        $a_t \leftarrow \arg\max_a Q(s_t, a; \theta)$
    end if
    Execute a_t, observe (r_t, s_{t+1}, d_t)
    Store transition $(s_t, a_t, r_t, s_{t+1}, d_t)$ in $\mathcal{D}$
    if $|D| \geq B$ **then**
      Sample minibatch $\mathcal{B} \sim \mathcal{D}$
      Compute targets:

$$y_j = \begin{cases} r_j & \text{if } d_j \\ r_j + \gamma \max_{a'} Q(s'_j, a'; \theta_{\text{target}}) & \text{otherwise} \end{cases}$$

      Update θ via SGD on $\frac{1}{B}\sum_j (y_j - Q(s_j, a_j; \theta))^2$
      if $t \mod C == 0$ **then**
        Hard update: $\theta_{\text{target}} \leftarrow \theta$
      end if
    end if
    Decay exploration: $\epsilon \leftarrow \epsilon_{\text{stop}} + (\epsilon_{\text{start}} - \epsilon_{\text{stop}})e^{-\lambda t}$
  end for
end for

3.1.1. Experience replay

A key component of the DQN algorithm is **experience replay**, which stabilizes learning by breaking the correlation between consecutive experiences. At each time step t, the agent's experience is represented as a tuple:

$$e_t = (s_t, a_t, r_{t+1}, s_{t+1})$$

These experiences are stored in a **replay memory** buffer $\mathcal{D}$ with a finite capacity N:

$$\mathcal{D} = \left(e_1, e_2, \ldots, e_N\right)$$

Older experiences are removed as new experiences are added to maintain the buffer size. During training, random mini-batches of experiences are sampled from $\mathcal{D}$ to update the network weights, minimizing the mean squared error (MSE) between the predicted Q-values and the target values from the Bellman equation:

$$L(\theta) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\left[\left(r + \gamma \max_{a'} Q_{\theta'}(s', a') - Q_\theta(s, a)\right)^2\right]$$

The parameters θ' of the **target network** are updated periodically to improve training stability.

Subsequently, the DQL (Deep Q-Learning) algorithm is evaluated as shown below:

To complement value-based DQN baselines and mitigate potential instability and limited generalization under multi-objective rewards and varying operating regimes, we also evaluate actor–critic methods that learn both a policy and a value function. Accordingly, Section 3.2 introduces AC as an alternative control strategy.

3.2. Actor-critic methods

Traditional DRL methods often face critical limitations in real-world energy management applications, including high sample complexity, unstable convergence, and sensitivity to hyperparameters. We introduce Algorithm A3C: Actor-Critic, a policy optimization framework to address these issues. In this work, we adopt a synchronous variant of the Advantage Actor–Critic algorithm, where multiple environments (or agents) are advanced in lock-step and their trajectories are aggregated into batches to perform a single gradient update on the shared actor–critic networks, rather than using fully asynchronous workers. Before deployment, the reinforcement learning agent is trained offline using historical data or within a simulated environment. In this study, offline training is employed due to the difficulty of accurately modeling real-time photovoltaic output and load demand, which are highly uncertain (Rezaeimozafar et al., 2024).

In real-world energy management systems, dynamic pricing schemes, fluctuating demand patterns, and intermittent renewable generation introduce significant uncertainties. To address these challenges, our algorithm leverages two synergistic neural networks:

- **Action Function (Policy Network, π_θ)**: This network outputs optimal control actions a_t given a system state s_t, parameterized by θ. By learning a stochastic policy, it balances exploration (e.g., trying new energy dispatch strategies) and exploitation (e.g., consistently selecting cost-efficient actions) under varying energy prices and load conditions.
- **Evaluation Function (Value Network, V_ϕ)**: This network estimates the expected long-term return of each state (or state–action pair), parameterized by ϕ. It provides a more stable learning signal than pure policy gradients, thus mitigating the variance often encountered in complex energy environments.

Both networks are implemented as deep feed-forward neural architectures that share a common MLP backbone. The input layer receives the state vector $s_t \in \mathbb{R}^d$. This backbone consists of two fully connected layers with 256 neurons each, followed by ReLU activation functions. From this shared representation, two task-specific heads are derived: an *actor head*, composed of a fully connected layer with 256 ReLU units followed by a linear layer with $|A|$ outputs (one logit per discrete action), which is passed through a softmax operation to obtain the categorical policy $\pi_\theta(a_t \mid s_t)$; and a *critic head*, composed of a fully connected layer with 256 ReLU units followed by a linear layer with a single output, yielding the scalar value estimate $V_\phi(s_t)$. All parameters (θ, ϕ) are trained jointly using the Adam optimizer with a learning rate of 3×10^{-4}, and we employ gradient-norm clipping with a threshold of 0.5 together with an entropy regularization term (coefficient 0.01) in the actor loss to encourage sufficient exploration and stabilize training.

These components engage in a co-evolutionary training process where the discrete-action policy network $\pi_\theta(a|s)$ refines its action selection probabilities using value estimates from $V_\phi(s)$, while the value network simultaneously improves its state-value predictions based on the policy's exploration trajectories in the discrete action space. This mutual reinforcement proves particularly valuable in discrete decision-making scenarios like energy system operation, where agents must select among finite control actions (e.g., discrete setpoint adjustments or on/off commands) while responding to real-time price signals and demand fluctuations.

These components engage in a co-evolutionary training process where the discrete-action policy network $\pi_\theta(a \mid s)$ refines its action selection probabilities using value estimates from $V_\phi(s)$, while the value network simultaneously improves its state-value predictions based on the policy's exploration trajectories in the discrete action space. This mutual reinforcement proves particularly valuable in discrete decision-making scenarios like energy system operation, where agents must select among finite control actions (e.g., discrete setpoint adjustments

Table 3
Evaluation map used to distinguish within-dataset temporal robustness from cross-context transfer.

Setting	Evaluation split/test	Interpretation
GEFCom2014	Temporal hold-out (75% train/25% test)	Within-dataset robustness (temporal shift)
Spanish PV–battery dataset	Temporal split for training/validation	Within-dataset robustness (temporal variability)
StoreNet H4	Cross-context transfer test after training on Spanish data	Out-of-domain transfer (distinct household regime)
Urban/CityLearn (3 years)	Year 3 held out (Year 1 train, Year 2 tune/validate)	Within-dataset robustness (out-of-sample year)

or on/off commands) while responding to real-time price signals and demand fluctuations.

To train this actor–critic pair in a stable manner, we employ an n-step Advantage Actor–Critic objective. Given a trajectory segment $(s_t, a_t, r_t, \ldots, s_{t+n})$ generated by interacting with the environment under the stochastic policy $\pi_\theta(a \mid s)$, we define the n-step bootstrapped return as

$$R_t = \sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^n V_\phi(s_{t+n}), \tag{13}$$

where $0 \le \gamma \le 1$ is the discount factor and V_ϕ is the value network. The corresponding advantage estimate is then given by

$$A_t = R_t - V_\phi(s_t). \tag{14}$$

The critic parameters ϕ are updated by minimizing the squared error between the bootstrapped return and the current value estimate:

$$L_v(\phi) = \mathbb{E}\left[(R_t - V_\phi(s_t))^2\right]. \tag{15}$$

This objective drives the value network to provide accurate estimates of the expected return for each state, reducing the variance of the training signal used by the policy.

The actor parameters θ are updated to maximize the expected advantage-weighted log-probability of the actions, with an entropy regularization term that encourages exploration:

$$J(\theta) = \mathbb{E}\Big[\log \pi_\theta(a_t \mid s_t)\, A_t + \beta\, H\big(\pi_\theta(\cdot \mid s_t)\big)\Big], \tag{16}$$

where $H(\pi_\theta(\cdot \mid s_t))$ denotes the entropy of the categorical policy over discrete actions and β is the entropy coefficient. The resulting policy gradient takes the form

$$\nabla_\theta J(\theta) = \mathbb{E}\Big[\nabla_\theta \log \pi_\theta(a_t \mid s_t)\, A_t + \beta\, \nabla_\theta H\big(\pi_\theta(\cdot \mid s_t)\big)\Big], \tag{17}$$

which we approximate using mini-batches of trajectory segments collected from the discrete action space.

In practice, these expectations are estimated from batched rollouts generated by multiple parallel environments, and the gradients are applied using the Adam optimizer described above. Unlike deterministic actor–critic methods such as DDPG, our A3C-Energy formulation does not employ target networks or soft updates, nor does it rely on a deterministic policy $\mu(s)$; instead, exploration arises naturally from sampling actions from the stochastic policy $\pi_\theta(a \mid s)$ and from the entropy regularization term in (16). This design is fully consistent with the discrete action space considered in our energy storage management problem and leads to stable on-policy learning under real-time pricing and demand fluctuations.

Algorithm 2 Synchronous Advantage Actor–Critic (A3C-Energy)

Input: environment $\mathcal{E}$, number of parallel environments N, discount factor γ, rollout length T
Output: optimal actor parameters θ and critic parameters ϕ
Initialize shared actor network $\pi_\theta(a \mid s)$ and critic network $V_\phi(s)$
Initialize optimizer (Adam) for (θ, ϕ)
while not converged **do**
  Reset N parallel environments and obtain initial states $\{s_0^i\}_{i=1}^N$
  for $t = 0$ to $T-1$ **do**
    for $i = 1$ to N **do**
      Sample action $a_t^i \sim \pi_\theta(\cdot \mid s_t^i)$ ▷ stochastic policy, no explicit noise χ_t
      Execute a_t^i in environment i
      Observe reward r_t^i, next state s_{t+1}^i, and terminal flag done_t^i
    end for
  end for
  for $i = 1$ to N **do**
    if done_T^i is true **then**
      $R_T^i \leftarrow 0$
    else
      $R_T^i \leftarrow V_\phi(s_T^i)$ ▷ bootstrap from critic
    end if
    for $t = T-1$ down to 0 **do**
      $R_t^i \leftarrow r_t^i + \gamma R_{t+1}^i$ ▷ n-step return
      $A_t^i \leftarrow R_t^i - V_\phi(s_t^i)$ ▷ advantage estimate
    end for
  end for
  Compute critic loss:
    $L_v(\phi) = \frac{1}{NT} \sum_{i=1}^N \sum_{t=0}^{T-1} \left(R_t^i - V_\phi(s_t^i)\right)^2$
  Compute actor objective with entropy regularization:
    $J(\theta) = \frac{1}{NT} \sum_{i=1}^N \sum_{t=0}^{T-1} \left[\log \pi_\theta(a_t^i \mid s_t^i) A_t^i + \beta\, H\big(\pi_\theta(\cdot \mid s_t^i)\big)\right]$
  Perform one gradient step on θ and ϕ using Adam to maximize $J(\theta)$ and minimize $L_v(\phi)$
end while

4. Initial data and proposed variations

4.1. Data and price profile

For this study, two datasets were used (see Table 3):

Case Study 1: Case Study 1 (GEFCom2014): Temporal hold-out evaluation

The first case study is built upon the Global Energy Forecasting Competition 2014 (GEFCom2014), which provides multi-year time series for electric load, electricity prices, wind power, and solar power (Hong et al., 2016). In this work, GEFCom2014 is not used to participate in the original forecasting challenge, but rather to reproduce the microgrid benchmark proposed by Aih (2020) and Hau et al. (2020). Specifically, the consumer load, PV generation, and the dynamic electricity price Series are extracted from the competition tracks and used to drive a grid-connected PV–BESS system under real-time pricing conditions.

Following the original study, the GEFCom2014-derived time series are processed through a training–validation pipeline tailored to reinforcement learning–based energy transactions without future knowledge. First, the raw hourly load, PV generation, and dynamic price profiles are normalized, augmented with calendar features (hour-of-day, day-of-week, and month/season), and grouped into 24-hour trajectories that define one-day episodes of microgrid operation. These daily episodes are then ordered chronologically and split into two strictly disjoint subsets: a *training* horizon, containing the first 75% of the days, used to train the RL agent and tune its hyperparameters, and a *testing* horizon, containing the remaining 25% of days, reserved exclusively for out-of-sample validation and performance comparison against baseline controllers. During both phases, the agent interacts with each episode step-by-step and only observes present load, PV, and price information, ensuring that no oracle knowledge of future trajectories leaks into the decision process.

Case Study 2: Spanish PV–Battery Dataset and StoreNet H4 for Cross-Context Evaluation

This case study integrates an independent photovoltaic battery dataset from the study *"Profitability of alternative battery operation strategies in photovoltaic self-consumption systems under the current regulatory framework and electricity prices in Spain"* (Durán Gómez et al., 2023). The dataset provides detailed operational records of residential PV systems equipped with battery storage, including consumption profiles, PV generation, and day-ahead electricity prices.

In this work, the Spanish dataset is used for both *training* and *validation* of the reinforcement learning agent. The rich price variability exposes the model to a broad spectrum of decision-making contexts, enabling it to learn flexible charging and discharging behaviors under nonstationary market conditions. A separate validation split is employed to monitor convergence and guide hyperparameter selection while maintaining temporal consistency within the dataset.

After training and validation on the Spanish PV–battery data, the learned policy is evaluated on a distinct residential environment: House 4 (H4) from the StoreNet project. H4 represents a demanding scenario with approximately 8000 kWh of annual consumption, only 1900 kWh of PV generation, and close to 6000 kWh of yearly grid imports. This high-stress configuration provides an ideal comparison environment for assessing the robustness and adaptability of the controller. After training and validation on the Spanish PV–battery data, the learned policy is evaluated on a distinct residential environment: H4 from the StoreNet project Z-score normalization applied to both datasets for consistent scaling. The original StoreNet measurements are recorded at 1-minute resolution, and in our pipeline, they are aggregated to an hourly timescale to match the control step of the RL agent.

Case Study 3: Urban Energy Storage Optimization Project

The third case study relies on a consolidated urban energy dataset inspired by the structure and methodology of recent CityLearn-based research. The dataset contains three consecutive years of realistic building-level operational data, sampled every 15 min. Each record includes electric load, rooftop PV generation, battery state of charge, electricity market price, carbon intensity, and local microclimate observations, forming a homogeneous and well-defined feature set across the entire temporal span.

To ensure methodological rigor and fair assessment of reinforcement learning performance, the dataset is partitioned into three sequential segments with distinct purposes:

- **Training set (Year 1):** The first full year of data is used to train the RL agent. This segment exposes the policy to a broad variety of daily and seasonal patterns, including winter peaks, summer PV surpluses, and price variability. Training exclusively on Year 1 ensures that the agent learns from a stable and internally consistent baseline environment.
- **Validation set (Year 2):** The second year is reserved for hyperparameter tuning, monitoring convergence behavior, and mitigating overfitting. The validation period includes natural shifts in consumption, weather, and market conditions relative to the training year, enabling the identification of policies that remain effective under these shifts, rather than fitting only the specific patterns of the training year.
- **Comparison set (Year 3):** The third year is held out entirely and used only for final performance comparison across RL algorithms. This segment functions as a strictly out-of-sample evaluation period, replicating realistic deployment conditions in which controllers must handle unseen seasonal conditions and operational variability.

4.2. Proposed variations

The standard DQN family is retained, and a small set of domain-specific adaptations is introduced to better fit the microgrid EMS setting. Because most of the evaluated RL methods were originally developed for general-purpose control problems, we standardize the state/action/reward interface so that each algorithm interacts with the environment under the same operational definitions. Concretely, the state vector includes real-time power signals, aggregated/average power statistics, and the battery SOC, while the action space encodes discrete charge/discharge decisions. The reward is reshaped to capture the immediate operational effect of each action, providing consistent learning feedback across methods. To keep the implementation lightweight and computationally efficient, the Q-network uses a simple fully connected architecture: the state input is passed through two ELU-activated dense layers (40 and 160 units) followed by a linear output layer with dimension equal to the number of actions. These programmatic adjustments ensure that each DQN-based variant can update actions and rewards coherently under microgrid constraints, improving decision-making stability and efficiency in realistic EMS operation.

In evaluating the C51 algorithm, the following hyperparameters were used as recommended in prior work (Chen & Gao, 2023): discount factor ($\gamma = 0.99$), learning rate ($\alpha = 0.001$), and value support range ($V_{\min}, V_{\max} = \pm 20$). A high discount factor emphasizes long-term returns, which can improve future reward estimation but may slow convergence. The learning rate was set to balance stable parameter updates with an adequate learning speed (smaller α typically improves stability at the cost of slower learning). Finally, the bounded value range constrains the distributional value estimates to avoid extreme outputs while still covering the expected reward variability in our setting.

The balance between **exploration** and **exploitation** is a fundamental challenge. This algorithm modifies the conventional ϵ-greedy policy by ensuring that during exploration, the agent avoids selecting the currently known best action. The exploration probability follows an **exponential decay** function over time, gradually shifting the focus from exploration to exploitation (Cai et al., 2025; Neves et al., 2024).

Exploration Probability Decay: To control the transition from exploration to exploitation, the exploration probability ϵ_t is defined as:

$$\epsilon_t = \epsilon_{\text{stop}} + (\epsilon_{\text{start}} - \epsilon_{\text{stop}})e^{-\lambda t} \tag{18}$$

where ϵ_{start} is the initial exploration probability, ϵ_{stop} is the minimum exploration probability, λ is the **decay rate**, which controls how quickly the exploration probability decreases, and t is the current time step (or decay step).

This ensures that initially, the agent explores frequently but gradually shifts toward exploitation as learning progresses.

Action Selection Strategy: At each time step, the algorithm generates a random number:

$$r \sim U(0,1) \tag{19}$$

The action selection follows an ϵ-greedy strategy:

$$a_t = \begin{cases} \text{Random action from } \mathcal{A} \setminus \arg\max Q(s_t, a; \theta), & \text{if } r < \epsilon_t \quad \text{(Exploration)} \\ \arg\max Q(s_t, a; \theta), & \text{otherwise} \quad \text{(Exploitation)} \end{cases} \tag{20}$$

where $Q(s_t, a; \theta)$ is the action-value function estimated by a DQN with parameters θ, $\mathcal{A}$ is the set of all possible actions, and $\arg\max Q(s_t, a; \theta)$ is the action with the highest Q-value in the current state s_t.

Exploration Strategy Modification: Unlike the conventional ϵ-greedy strategy, which selects any random action during exploration, this modified approach avoids selecting the current known best action. Instead, it selects from the remaining suboptimal actions:

$$a_{\text{explore}} \sim \mathcal{A} \setminus \arg\max Q(s_t, a; \theta) \tag{21}$$

This ensures that if the agent explores, it selects an alternative action rather than the action that appears optimal so far. Encouraging the

exploration of different strategies helps prevent premature convergence to suboptimal policies. The algorithm 3 can be seen in.

Algorithm 3 Algorithm Implementation in Code Exploration and Exploitation

1: The implementation follows these steps:
2: Compute the exploration probability ϵ_t.
3: Generate a random number $r \sim U(0,1)$ to decide between exploration or exploitation.
4: If exploring, randomly select an action from $A \setminus \{a^*\}$.
5: If exploiting, select $a^* = \arg\max Q(s,a;\theta)$.

This ensures that if the agent explores, it selects an alternative action rather than the action that appears optimal so far. Encouraging the exploration of different strategies helps prevent premature convergence to suboptimal policies. The algorithm 3 can be seen in.

DQN employs **experience replay** to break temporal correlations. Unless otherwise stated, the baseline uses *uniform (random) sampling*. A PER variant is available as an option; PER results are reported separately and are not part of the baseline.

5. Applications in microgrid optimization

5.1. Action definition in battery energy management

In managing energy for lithium-ion batteries, it is crucial to calculate the appropriate amount of energy to either charge or discharge using a reinforcement learning (RL) algorithm. This algorithm determines the agent's action at time t by integrating essential factors into the utility function (Aih, 2020).

- Set a baseline level of contingency reserves—or a target SOC —in the BESS to guarantee support for critical operations. The utility function is defined in segments based on whether the SOC falls below, meets, or exceeds this target threshold.
- When the SOC is below the designated target, the agent prioritizes charging the energy storage system, largely ignoring energy price signals unless they are significantly high—allowing limited discharging in such cases.
- Once the SOC exceeds the target level, the agent adapts its strategy: it charges during periods of low prices and discharges when prices are favorable. As the SOC approaches the target, the agent gradually reverses this behavior to maintain reserves.
- A negative exponential factor M is embedded within the utility function to model this behavior. The associated parameters can be fine-tuned to shape the agent's response at different SOC levels.
- Two additional penalty terms are incorporated to guide the agent's actions:
  - $PntBESSt$: Applies a penalty for decisions that breach BESS operational constraints, supporting system reliability and longevity.
  - $PntPVt$: Penalizes failure to store excess photovoltaic energy when storage capacity is available, promoting maximum utilization of free renewable energy.

The overall utility function R combines these components to guide the agent's decision-making for the BESS. This ensures necessary reserves, efficient storage use, and an extended lifespan of the BESS (Flórez et al., 2023), aligning with the parameters of the investigation (Liu et al., 2023).

The parameters in Table 4 were selected to reflect a realistic grid-scale lithium–ion BESS and to make the trade-offs in the reward function explicit. The power and energy ratings ($P_{\text{rated}} = 1000$ kW, $E_{\text{rated}} = 5000$ kWh) correspond to a 1 MW/5 MWh system, a configuration that allows several hours of continuous operation at rated power. The investment cost $C_i = 171$ \$/kWh lies within the range reported for utility-scale Li-ion systems and is used to convert energy throughput into an equivalent degradation cost. The reference depth of discharge $\delta = 0.8$ and the life cycle value $N_c = 4996$ model a battery whose lifetime is specified at 80% DoD, which is common in commercial datasheets; these parameters enter the wear coefficient so that deeper and more frequent cycles are penalized in accordance with the expected lifetime. The nominal charge and discharge efficiencies $\eta_c = \eta_d = 0.95$ represent typical round-trip efficiencies for modern Li-ion BESS. The target SOC $SOC_{\text{target}} = 0.5$ defines a symmetric emergency reserve, leaving headroom both for absorbing surplus PV and for supplying power during demand peaks. The degradation-aware SOC penalty Pnt_t^{BESS} is further shaped by the coefficients $\varphi_1 = 10^{-3}$ and $\varphi_2 = 10^{-3}$, which scale the contribution of the degradation factor $c_{d,k}$ and the investment cost C_B in low-SOC ($SOC < 0.4$) and high-SOC ($SOC > 0.8$) regions, respectively. Choosing φ_1 and φ_2 in this small range ensures that harmful operation in extreme SOC bands is consistently penalized, while keeping Pnt_t^{BESS} numerically comparable to the economic and wear-related terms so that no single component dominates the learning signal. Finally, the weighting coefficients $\lambda_{\text{econ}} = 0.6$, $\lambda_{\text{wear}} = 1.0$ and $\lambda_{\text{deg}} = 0.4$ encode the chosen compromise between economic performance and battery health: the economic term is given substantial, but not dominant, importance, whereas the wear-related cost and the degradation-aware SOC penalty $P_{nt,t}^{\text{BESS}}$ (as defined in Section 2.3, Eq.) are weighted strongly enough to discourage aggressive cycling and prolonged operation in a harmful SOC region.

Table 4
ESS parameters description.

Symbol	Quantity	Value
P_{rated}	Power rating of BESS	1000 kW
E_{rated}	Energy capacity of BESS	5000 kWh
C_i	Initial investment cost	171 \$/kWh
δ	Reference Depth of Discharge (DOD)	0.8
N_c	Life cycle at rated DOD	4996
SOC_{target}	Emergency reserve	0.5
φ_1	Penalty weight for low SOC region ($SOC < 0.4$)	10^{-3}
φ_2	Penalty weight for high SOC region ($SOC > 0.8$)	10^{-3}
λ_{econ}	Weight of economic term (trading + reserve)	0.6
λ_{wear}	Weight of wear-related operating cost	1.0
λ_{deg}	Weight of degradation-aware SOC penalty	0.4

Table 5
Sensitivity of net benefit to reward-weight configurations (evaluation-only; mean ± std over seeds).

Setting $(\lambda_{\text{econ}}, \lambda_{\text{wear}}, \lambda_{\text{deg}})$	Net benefit (mean ± std)
Battery-protective/degradation-realistic $(0.6, 1.00, 0.4)$	50.31 ± 0.55
More economic emphasis $(0.8, 1.00, 0.5)$	49.75 ± 2.15
More conservative $(0.4, 1.00, 0.4)$	40.64 ± 0.84
Lower SOC/degradation penalty $(0.6, 1.00, 0.3)$	49.90 ± 0.72
Strong battery-protection $(0.5, 1.00, 0.7)$	31.23 ± 0.85
Higher wear regime $(0.6, 1.25, 0.50)$	47.23 ± 0.60

To quantify how sensitive the conclusions are to the reward trade-offs, Table 5 reports a local perturbation study around a battery-protective nominal configuration. The baseline $(\lambda_{\text{econ}}, \lambda_{\text{wear}}, \lambda_{\text{deg}}) = (0.6, 1.0, 0.4)$ was selected to keep the economic incentive active while assigning substantial weight to cycling-related wear and SOC-driven degradation penalties, thereby discouraging aggressive arbitrage and prolonged operation in harmful SOC regions. We then probe two directions that preserve the same design intent. First, we vary the economic weight to represent more profit-seeking versus more conservative behavior, using $(0.4, 1.0, 0.4)$ as a stricter battery-first setting and $(0.8, 1.0, 0.5)$ as a more economically driven setting while still retaining a non-negligible degradation term. Second, we vary the degradation

emphasis to test how strongly SOC-protection influences policy behavior, comparing a lighter SOC penalty $(0.6, 1.0, 0.3)$ against a strongly conservative setting $(0.5, 1.0, 0.7)$ that prioritizes battery preservation. Finally, we include a higher-wear regime $(0.6, 1.25, 0.50)$ to stress the sensitivity of outcomes to cycling costs under the same market conditions. Across all settings, we report relative changes in decomposed market cost, degradation-related cost, total cost, and the probability of critical SOC excursions to reveal whether performance trends and safety proxies remain stable under plausible re-weightings of the objective.

Evaluation metrics

To comprehensively evaluate the effectiveness of the proposed approach, this work defines three complementary categories of performance metrics. Cost-Based Metrics quantify the total and net economic benefits achieved by integrating an ESS, highlighting cost savings, daily averages, and battery use efficiency. Normalized performance metrics enable a fair comparison against baseline scenarios without storage, highlighting relative reductions in electricity costs and CO_2 emissions to assess the sustainability of the solution. Finally, Algorithm Performance Metrics focus on the computational aspects of the learning process, capturing learning efficiency, real-time decision speed, robustness under unseen conditions, and the operational costs associated with battery usage. Together, these metrics provide a holistic view of economic viability, environmental impact, and algorithmic efficiency, ensuring that the proposed solution is evaluated from multiple practical and technical perspectives.

Cost-based metrics

$$\mathrm{TC}_{\mathrm{WOESS}} = \sum_{t=1}^{T} P_t\,(L_t - G_t),$$

$$\mathrm{TC}_{\mathrm{WESS}} = \sum_{t=1}^{T} P_t\,(L_t - G_t + \Delta E_t) \;+\; \sum_{t=1}^{T} P_{\mathrm{wear}}\,|\Delta E_t|,$$

$$\Delta E_t = (SOC_t - SOC_{t-1})\,P_{\mathrm{rated}},$$

$$\mathrm{NCB} = \mathrm{TC}_{\mathrm{WOESS}} - \mathrm{TC}_{\mathrm{WESS}},$$

$$\mathrm{DAC} = \frac{\mathrm{TC}}{T/24}, \qquad \mathrm{DNB} = \frac{\mathrm{NCB}}{T/24}, \qquad \mathrm{BUE} = \frac{\mathrm{NCB}}{\sum_{t=1}^{T} |\Delta E_t|}.$$

Normalized performance metrics

$$\hat{C} = \frac{C_{\mathrm{entry}}}{C_{\text{no battery}}}, \qquad \hat{G} = \frac{G_{\mathrm{entry}}}{G_{\text{no battery}}}, \qquad \text{Overall Score} = \hat{C} + \hat{G}.$$

Algorithm performance metrics

$$\eta = \frac{1}{|D_{\mathrm{train}}|}, \qquad T_{\mathrm{online}} = \sum_{t=0}^{T} \mathrm{DecisionTime}(t),$$

$$\Delta C = \frac{C_{\mathrm{test}} - C_{\mathrm{train}}}{C_{\mathrm{train}}}, \qquad OC = \sum_{t=0}^{T} \rho_{\mathrm{wear}}\,|E_{\mathrm{charge}}(t) - E_{\mathrm{discharge}}(t)|.$$

where P_t is the electricity price at time step t (USD/kWh), L_t is the electrical load energy during interval t (kWh), G_t is the PV energy generated during interval t (kWh), and ΔE_t is the net BESS energy exchanged during interval t (kWh; $\Delta E_t > 0$ for charging and $\Delta E_t < 0$ for discharging). P_{wear} denotes the battery wear-cost coefficient (USD/kWh of BESS energy throughput), T is the number of time steps in the simulation horizon, and $SOC_t \in [0, 1]$ is the state of charge. The BESS rated energy capacity is denoted by E_{rated} (kWh), such that $\Delta E_t = (SOC_t - SOC_{t-1})\,E_{\mathrm{rated}}$. Therefore, $TCWOESS$, $TCWESS$, and NCB represent total costs/savings accumulated over the full simulation horizon; if needed, normalized costs can be obtained by dividing by the corresponding duration (see Table 6).

Table 6
Metric interpretation guidelines.

Metric	Ideal value	Interpretation
$\mathrm{TC}_{\mathrm{WESS}}$	Minimize	Lower total cost with battery
NCB	Maximize	Greater savings from ESS
$\hat{C}$	<1	Cost-effective operation
$\hat{G}$	<1	Environmentally sustainable
Overall score	<2	Better than baseline
η	Higher	More data-efficient learning
T_{online}	Lower	Faster real-time decisions

To measure the performance of the proposed algorithms, metrics commonly used in the state of the art are employed, such as Average Return, Reward Variance, Interquartile Mean (IQM), Optimality Gap (OG), and Probability of Improvement (PoI), which enable a robust, consistent, and statistically informed evaluation of the solutions obtained (Agarwal et al., 2021).

6. Results

In this section, we analyze the performance of various energy management algorithms for microgrid operation, using simulations to evaluate the capacity of each method to maintain essential energy reserves and optimize economic returns. The primary metrics for assessment are the following.

- The probability of SOC being below the critical threshold, which reflects the risk of insufficient energy reserves for mission-critical operations.
- The daily average monetary benefit, which captures the financial efficacy of the algorithm under varying operational conditions.

6.1. Policy optimization benchmark (Database 1 and 2)

We conduct a two-pronged empirical validation. Case Study 1 builds a reproducible benchmark from GEFCom2014 time series (load, PV, price) to emulate a grid-connected PV–BESS. Case Study 2 leverages a residential Spanish PV–battery dataset and evaluates scenario-dependent robustness through testing on StoreNet H4 under a consistent state/action/reward interface. Across both settings, we benchmark canonical RL baselines against DQN variants and policy-gradient methods. The primary endpoints are: the probability of SOC excursions below a critical threshold, and the average daily reward over the training horizon.

6.1.1. Case Study 1 (GEFCom2014): Temporal hold-out evaluation

In Fig. 2, all DQN-style agents exhibit a sharp recovery from an initial exploration-induced collapse toward a stable plateau, but they differ in how quickly they escape the negative-reward regime and how consistently they do so across seeds. PER-DQN reaches competitive returns earlier in this setting, which is qualitatively consistent with the objective of prioritized replay to sample high-error transitions more frequently, while acknowledging that the plotted results alone do not isolate the mechanism. NoisyNet-DQN also accelerates early improvement, although its larger initial dispersion reflects noise-driven exploration that can yield both very poor and very good early trajectories depending on the seed. DoubleDQN progresses more conservatively yet with tighter run-to-run variability, aligning with reduced overestimation bias and more stable target estimates. C51 extends DQN with a distributional value head, learning a categorical approximation of the return distribution on a fixed support; because the update relies on a projected Bellman operator, performance can be sensitive to reward

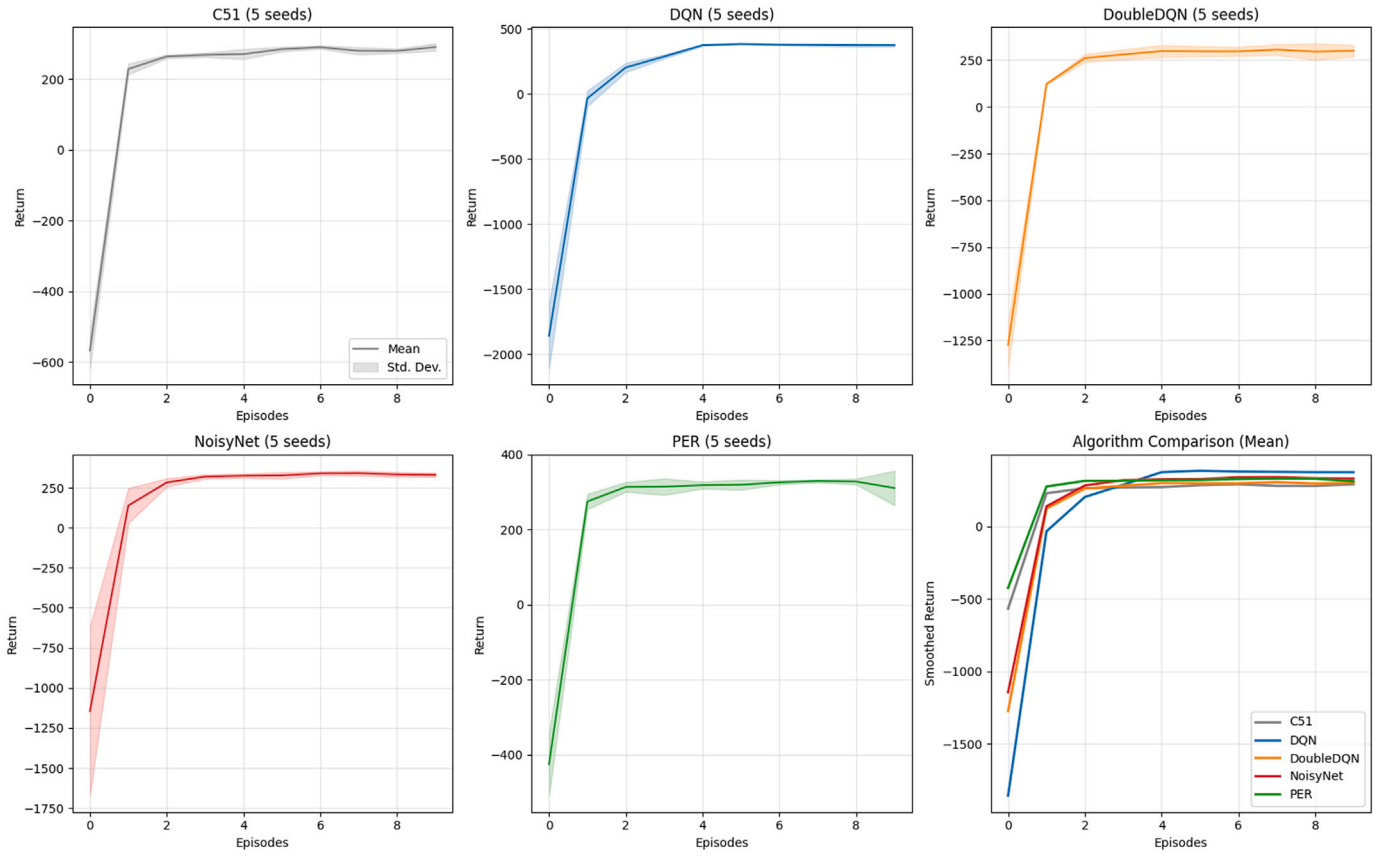

Fig. 2. Performance comparison of DQN variants (DoubleDQN, NoisyNet-DQN, PER-DQN, and DQN2013) over training time and rewards.

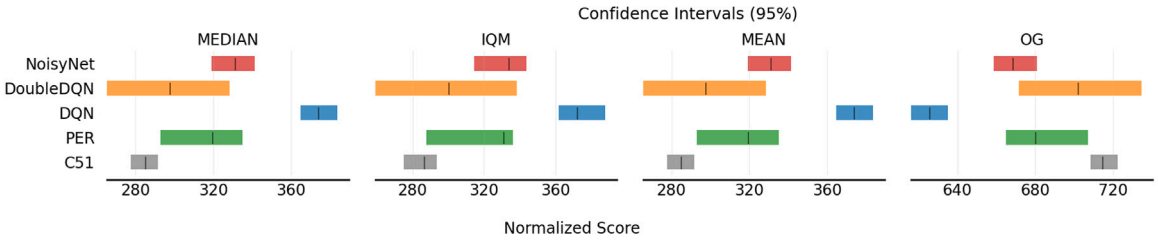

Fig. 3. Aggregate normalized performance metrics with 95% bootstrap confidence intervals: Median, interquartile mean (IQM), Mean, and Optimality Gap (OG) for the evaluated DQN variants (5 seeds).

scaling and the chosen support range. In terms of final-window performance, the interval estimates in Fig. 3 indicate that the baseline DQN attains the highest central-tendency scores (Median/IQM/Mean) among the compared methods, with PER and NoisyNet forming a second tier whose confidence intervals partially overlap, suggesting comparable steady-state performance at the current seed count. DoubleDQN and C51 yield lower central-tendency estimates in this evaluation setting. The optimality-gap (OG) ranking is consistent with these summaries (lower OG corresponds to a smaller shortfall from the target γ), reinforcing that the main separations here are in the achieved final-window return level and its uncertainty, rather than implying large, definitive differences when intervals overlap.

Fig. 4 reports mean episodic return over 100 episodes with five random seeds; the shaded bands denote the cross-seed standard deviation. All three on-policy methods improve from an initially negative-return regime to a positive-return plateau, but they differ in the magnitude of the early transient and in the rate at which variability collapses. AC exhibits the fastest recovery and the earliest stabilization: returns rise sharply within the first ≈10–20 episodes, followed by a slower improvement phase that asymptotically approaches a plateau near the top of the observed range. PG follows a similar two-phase pattern but converges to a slightly lower plateau and retains a modestly larger dispersion during the early-to-mid training window. In contrast, PPO shows a substantially larger initial return collapse and wider early dispersion, indicating higher sensitivity to initialization and sampling noise in the first episodes; although PPO improves steadily, it approaches a lower plateau within the evaluated budget, with the mean curve remaining below AC/PG across most of training.

Fig. 5 reports robust aggregate statistics of normalized performance with 95% bootstrap confidence intervals. We include Median and IQM to reduce sensitivity to outliers and to obtain more stable estimates under a small number of runs, and we report Optimality Gap as a complementary summary to the mean. Within this evaluation, AC attains the highest central-tendency estimates (Median/IQM/Mean) and the lowest OG, PG is intermediate, and PPO is lower on central-tendency metrics with higher OG.

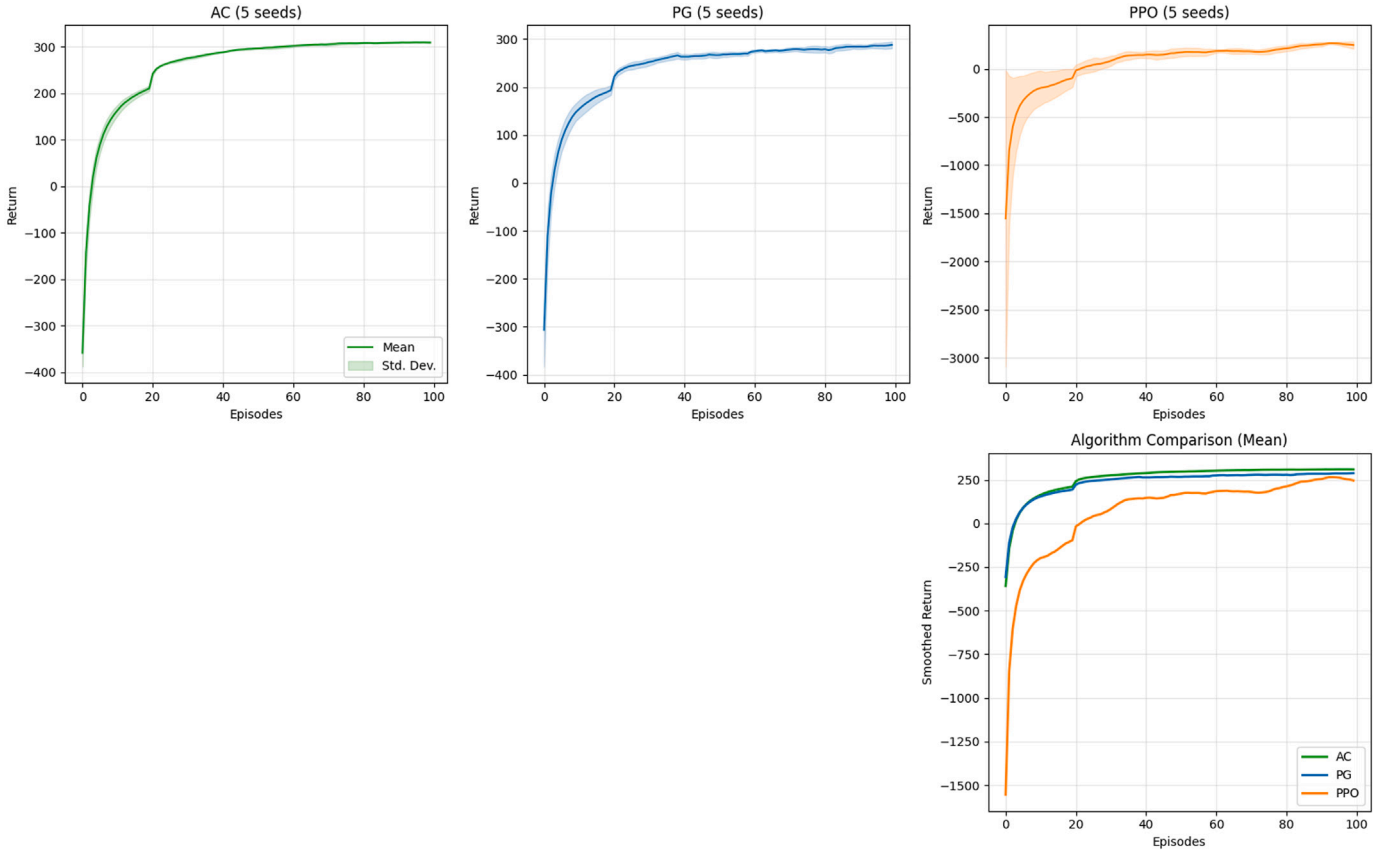

Fig. 4. Learning curves of policy-gradient methods (AC, PG, and PPO) over 100 episodes (5 seeds). Each panel reports the mean episodic return, and the shaded area denotes the standard deviation across seeds.

Table 7
Aggregate performance estimates with 95% confidence intervals (reported as estimate ± half-width) for each metric. Best performance is highlighted in bold. Execution time is reported as estimate ± half-width (95% CI) in seconds per episode (s/episode).

Algorithm	Median	IQM	Mean	OG	Exec time
C51	**50.41 ± 0.49**	**50.31 ± 0.55**	**50.41 ± 0.49**	**949.59 ± 0.49**	**63.85 ± 0.60**
PPO	50.11 ± 1.92	49.81 ± 2.27	50.11 ± 1.92	949.89 ± 1.92	2.26 ± 0.03
NoisyNet	46.58 ± 1.48	46.93 ± 1.88	46.58 ± 1.48	953.42 ± 1.48	14.30 ± 0.41
AC	44.70 ± 2.57	44.54 ± 3.18	44.70 ± 2.57	955.30 ± 2.57	2.82 ± 0.38
PER	49.60 ± 1.54	49.89 ± 1.95	49.60 ± 1.54	950.40 ± 1.54	6.07 ± 0.29
DoubleDQN	43.34 ± 3.43	44.93 ± 4.22	43.34 ± 3.43	956.66 ± 3.43	4.50 ± 0.07
DQN	46.09 ± 2.01	46.66 ± 2.61	46.09 ± 2.01	953.91 ± 2.01	4.07 ± 0.11
PG	39.02 ± 3.31	40.03 ± 4.09	39.02 ± 3.31	960.98 ± 3.31	1.83 ± 0.14
Rule-based	25.30 ± 2.55	25.20 ± 2.18	25.10 ± 1.60	974.90 ± 1.60	–

Table 8
Aggregate performance estimates with 95% confidence intervals (estimate ± half-width). Execution time is reported in seconds per episode (s/episode). Peak GPU memory (VRAM) is reported in GB as estimate ± half-width (95% CI). *VRAM values shown here are illustrative placeholders and must be replaced by measured values on the target hardware.*

Algorithm	Median	IQM	Mean	OG	Exec time	Peak VRAM (GB)
C51	**50.41 ± 0.49**	**50.31 ± 0.55**	**50.41 ± 0.49**	**949.59 ± 0.49**	**63.85 ± 0.60**	**1.90 ± 0.10**
PPO	50.11 ± 1.92	49.81 ± 2.27	50.11 ± 1.92	949.89 ± 1.92	2.26 ± 0.03	3.10 ± 0.20
NoisyNet	46.58 ± 1.48	46.93 ± 1.88	46.58 ± 1.48	953.42 ± 1.48	14.30 ± 0.41	1.60 ± 0.10
AC	44.70 ± 2.57	44.54 ± 3.18	44.70 ± 2.57	955.30 ± 2.57	2.82 ± 0.38	2.60 ± 0.20
PER	49.60 ± 1.54	49.89 ± 1.95	49.60 ± 1.54	950.40 ± 1.54	6.07 ± 0.29	2.10 ± 0.10
DoubleDQN	43.34 ± 3.43	44.93 ± 4.22	43.34 ± 3.43	956.66 ± 3.43	4.50 ± 0.07	1.40 ± 0.10
DQN	46.09 ± 2.01	46.66 ± 2.61	46.09 ± 2.01	953.91 ± 2.01	4.07 ± 0.11	1.30 ± 0.10
PG	39.02 ± 3.31	40.03 ± 4.09	39.02 ± 3.31	960.98 ± 3.31	1.83 ± 0.14	2.20 ± 0.20
Rule-based	25.30 ± 2.55	25.20 ± 2.18	25.10 ± 1.60	974.90 ± 1.60	–	–

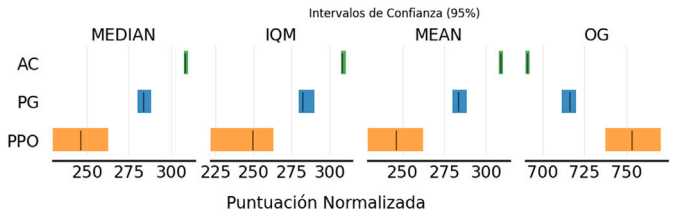

Fig. 5. Aggregate normalized performance with 95% bootstrap confidence intervals (5 seeds): Median, interquartile mean (IQM), Mean, and Optimality Gap (OG) for AC, PG, and PPO. Higher Median/IQM/Mean and lower OG indicate better performance.

Table 8 provides a terminal comparison across algorithms using robust location estimators (Median and IQM), the Mean, and the Optimality Gap (OG), each reported with 95% confidence intervals. Based on point estimates, **C51** achieves the highest central-tendency performance (Median 50.41 ± 0.49, IQM 50.31 ± 0.55), followed closely by **PPO** (Median 50.11 ± 1.92, IQM 49.81 ± 2.27) and **PER** (Median 49.60 ± 1.54, IQM 49.89 ± 1.95). Relative to the baseline **DQN** (Median 46.09 ± 2.01, IQM 46.66 ± 2.61), these correspond to modest improvements of approximately +9.4% (C51), +8.7% (PPO), and +7.6% (PER) in Median, and +7.8% (C51), +6.8% (PPO), and +6.9% (PER) in IQM. Although C51, PPO, and PER differ in point estimates, the between-method gaps are of the same order as the corresponding uncertainty bands, suggesting that their terminal performance is effectively similar under the current evaluation protocol. NoisyNet and DQN form an intermediate band (Median $\approx$ 46–47), while DoubleDQN and AC yield lower central-tendency estimates, and PG attains the lowest scores with the largest uncertainty. The OG column is consistent with these summaries (lower is better): compared to DQN (OG 953.91 ± 2.01), C51 reduces OG by 4.32 points (about 0.45%), with PPO and PER showing similarly small reductions, reinforcing that the main differences captured by the table are the achieved terminal return level and its run-to-run uncertainty.

6.1.2. Case Study 2: Cross-context transfer evaluation on StoreNet H4

In Case Study 2 (Figs. 6–7), The main qualitative differences emerge in *time-to-recovery* from the negative-return regime and *cross-seed dispersion* during this early transient. PER-DQN and C51 exit the high-penalty regime within the first few episodes and reach near-plateau returns sooner, which is consistent with their respective design goals: prioritized replay increases the sampling frequency of high-error transitions, while the categorical distributional head in C51 learns a return distribution that can provide a richer learning signal than expectation-only value regression, In contrast, DoubleDQN and NoisyNet exhibit a slower early recovery and lower early central tendency in returns; this is compatible with more conservative target evaluation in DoubleDQN and exploration-driven variability under parametric noise in NoisyNet, both of which can materially affect the transient without necessarily changing the long-run ceiling under a fixed budget

Fig. 7 complements the learning curves by summarizing *final-window* performance using robust aggregates (Median and IQM), the Mean, and Optimality Gap (OG), each with 95% stratified-bootstrap confidence intervals, as recommended for few-run RL evaluation. In this setting, the interval estimates suggest that DQN and PER achieve the strongest central-tendency performance (Median/IQM/Mean). C51 is close but exhibits greater uncertainty relative to the top group, consistent with sensitivity to reward scaling and support calibration inherent to categorical distributional updates DoubleDQN and NoisyNet yield lower central-tendency estimates and larger OG values (i.e., a larger shortfall from the target threshold γ), reinforcing that the dominant separations in Case Study 2 are in the attained final-window return level and its uncertainty, rather than in unequivocal, large margins when intervals overlap.

In Fig. 8, AC and PG reach their high-return regime quickly and then improve only marginally, with relatively tight across-seed dispersion after the first few dozen episodes. PPO, by contrast, exhibits a much more severe early transient large negative returns and wide seed-to-seed spread and a slower recovery, but continues to improve throughout training and approaches comparable end-of-training returns by the final episodes; this pattern is consistent with PPO's clipped surrogate objective producing more conservative updates that can trade early speed for stability depending on the rollout and hyperparameter regime.

Fig. 9 summarizes the same runs over the final evaluation window: the point estimates for Median/IQM/Mean are broadly comparable across methods, but PPO shows substantially wider 95% stratified-bootstrap intervals, indicating that its final performance is less consistent across seeds under the current setting. OG intervals mirror this picture (lower is better relative to the target threshold γ): AC/PG exhibit tighter gaps, whereas PPO's wider OG interval suggests a higher probability of underperforming the target in some runs despite competitive point estimates.

Table 9 reports the *Case Study 2* terminal *Net Benefit* using robust location summaries (Median and IQM), the Mean, and the Optimality Gap (OG), each with 95% confidence intervals (estimate $\pm$ half-width). Interpreting higher Net Benefit as better, the point estimates indicate that the best terminal performance is achieved by the on-policy methods: **AC** attains the highest Median/Mean (7960.92 ± 891.02), corresponding to an absolute gain of +449.59 (about +5.99%) over the **DQN** baseline (Median 7511.33 ± 149.32). **PPO** provides a comparable terminal level (Median 7780.94 ± 1021.17, +269.61, +3.59% vs. DQN) and achieves the highest IQM (7764.23 ± 1345.48), i.e., +250.89 (about +3.34%) relative to the DQN IQM (7513.34 ± 193.46), suggesting strong typical performance in the middle portion of runs.

Within the value-based family, the **DQN** baseline attains the highest *point estimates* under the Net Benefit metric in this experiment, while the evaluated extensions yield lower point estimates (e.g., C51 Median 7258.17 ± 18.70, −3.37%; NoisyNet 7224.93 ± 332.80, −3.81%; DoubleDQN 7155.39 ± 97.46, −4.74%; PER 7065.09 ± 328.35, −5.94%; PG 6808.30 ± 683.67, −9.36% relative to DQN). The confidence half-widths further indicate a repeatability trade-off: **C51** (and DQN) exhibit tight uncertainty bounds, whereas **AC/PPO** achieve higher point estimates but with wider intervals, implying greater sensitivity to seed and/or operating conditions in this setting.

6.2. Evaluation against state-of-the-art methods

This subsection details a comprehensive experimental evaluation designed to benchmark the proposed state-of-the-art baselines within the EMS optimization domain. To ensure a rigorous comparison, we maintain identical experimental conditions across multiple datasets—fixing the environment specifications, training budgets, and evaluation protocols. We define the training budget in terms of environment interaction steps, but we do not enforce a single fixed budget across all RL families because their convergence speed and sample efficiency differ markedly. In our setting, value-based methods typically reach a stable regime with fewer episodes, whereas actor-based and actor–critic approaches often require longer training to avoid premature conclusions from under-trained policies. For this reason, we allocate family-specific training horizons chosen to reach stable learning plateaus and report learning curves to make these differences explicit, so that performance comparisons reflect both achievable steady-state behavior and the underlying convergence dynamics under the same environment specification and evaluation protocol. Performance is quantified using a multi-faceted set of criteria: terminal objective values (net benefit/cost) and learning dynamics. This standardized framework facilitates a dual assessment of temporal hold-out robustness and cross-context transfer performance.

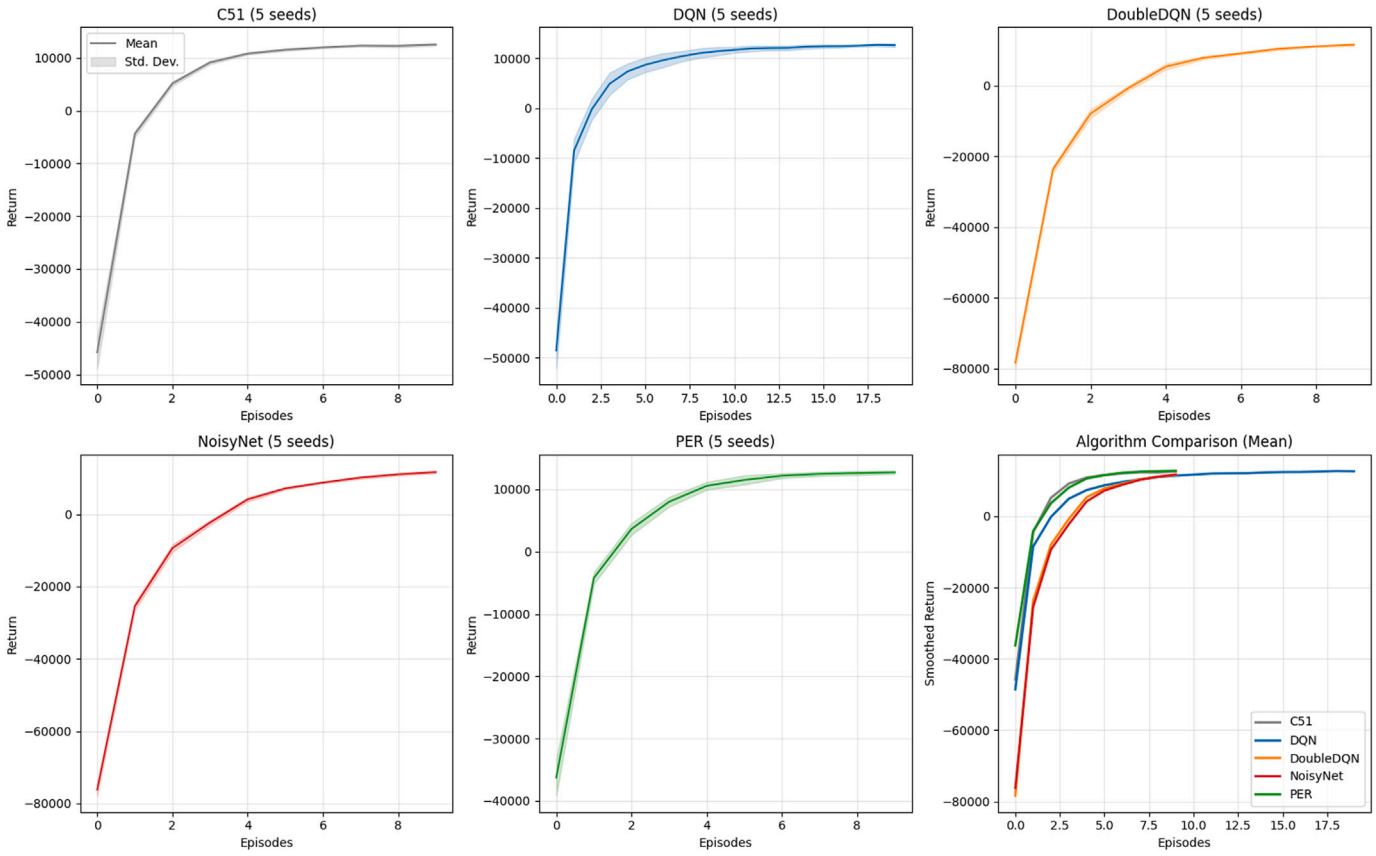

Fig. 6. Case Study 2: Learning curves for DQN and its variants (C51, DoubleDQN, NoisyNet-DQN, and PER-DQN) over training episodes (5 seeds). Solid lines report the mean return and shaded regions denote the standard deviation across seeds; the combined panel overlays mean returns to highlight relative convergence behavior.

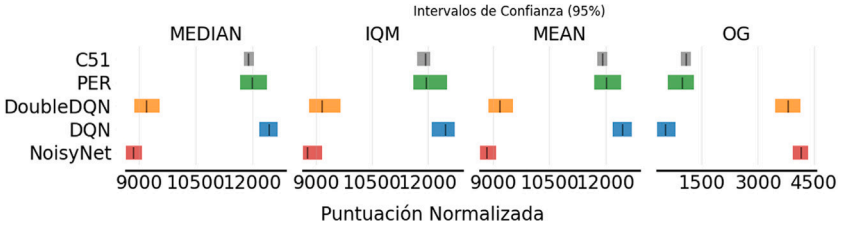

Fig. 7. Case Study 2: Aggregate normalized performance with 95% stratified-bootstrap confidence intervals (Median, IQM, Mean, and Optimality Gap) computed from per-seed averages over the final evaluation window (5 seeds). Higher Median/IQM/Mean and lower OG indicate better performance.

Table 9
Aggregate *Net Benefit* estimates with 95% confidence intervals (reported as estimate ± half-width) for each metric. Best values are highlighted in bold (higher is better for Median/IQM/Mean; lower is better for OG).

Algorithm	Median	IQM	Mean	OG
AC	**7960.92 ± 891.02**	7697.95 ± 1168.00	**7960.92 ± 891.02**	**5039.08 ± 891.02**
PPO	7780.94 ± 1021.17	**7764.23 ± 1345.48**	7780.94 ± 1021.17	5219.06 ± 1021.17
DQN	7511.33 ± 149.32	7513.34 ± 193.46	7511.33 ± 149.32	5488.67 ± 149.32
C51	7258.17 ± 18.70	7265.22 ± 22.35	7258.17 ± 18.70	5741.83 ± 18.70
NoisyNet	7224.93 ± 332.80	7202.95 ± 444.79	7224.93 ± 332.80	5775.07 ± 332.80
DoubDQN	7155.39 ± 97.46	7181.03 ± 115.67	7155.39 ± 97.46	5844.61 ± 97.46
PER	7065.09 ± 328.35	7096.68 ± 421.58	7065.09 ± 328.35	5934.91 ± 328.35
PG	6808.30 ± 683.67	7085.77 ± 862.84	6808.30 ± 683.67	6191.70 ± 683.67
Rule-based	108.70 ± 37.52	113.87 ± 29.70	123.40 ± 24.25	12 876.60 ± 24.25

Table 10 reports the *daily net benefit* (USD/day). Under the evaluated protocol, **DQN** achieves the highest terminal net benefit (69.72 ± 2.86 USD/day). Importantly, this advantage is not marginal: the 95% interval for DQN ([66.86, 72.58]) lies entirely above the corresponding intervals of **PER** (59.22 ± 1.36, i.e., [57.86, 60.58]), **C51** (51.02 ± 0.83, i.e., [50.19, 51.85]), and **PPO** (49.27 ± 1.41, i.e., [47.86, 50.68]), indicating a clear separation at the reported confidence level. In relative terms, PER, C51, and PPO underperform the DQN net benefit by

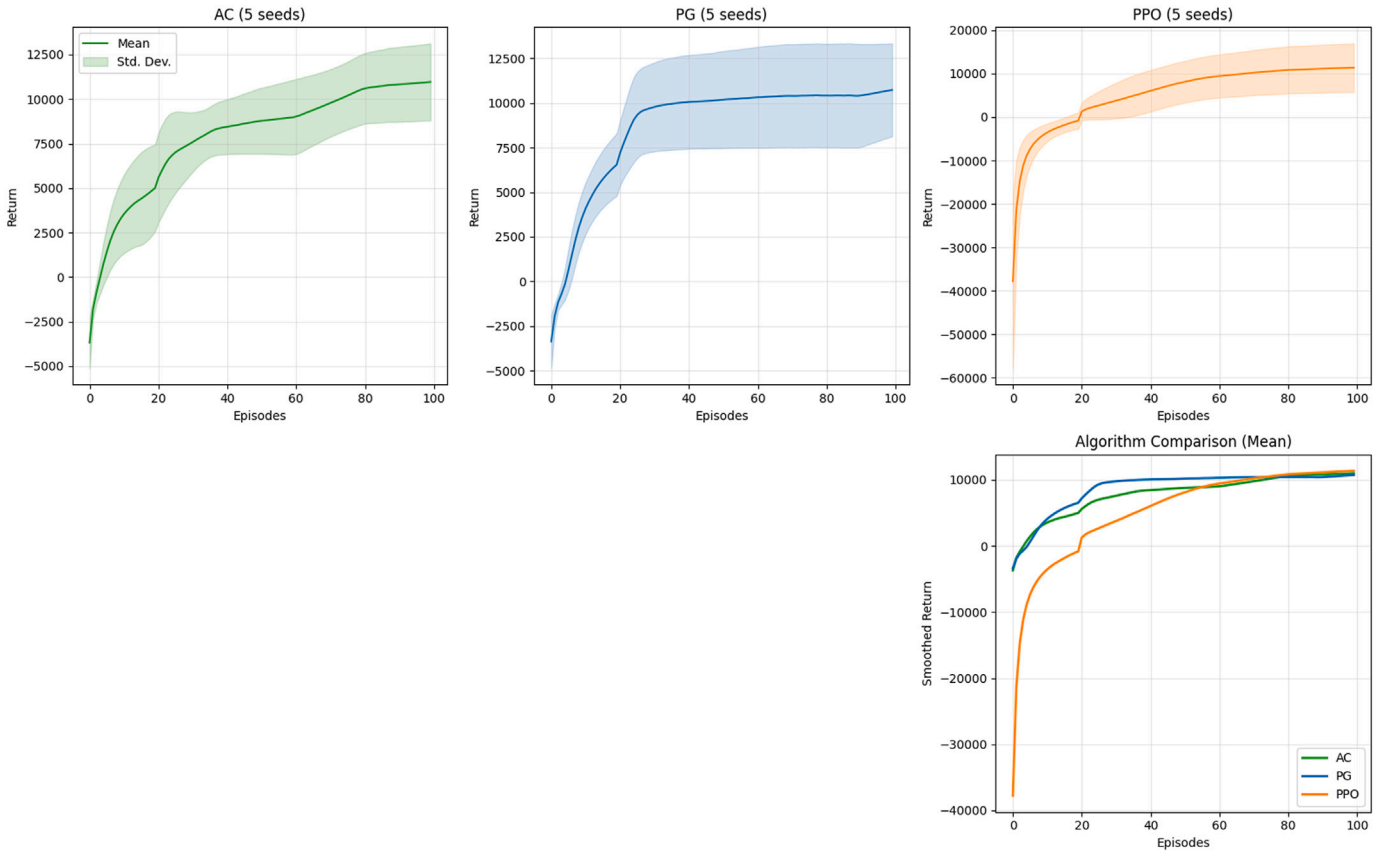

Fig. 8. Learning curves (mean ± std. over 5 seeds) for AC, PG, and PPO across 100 episodes.

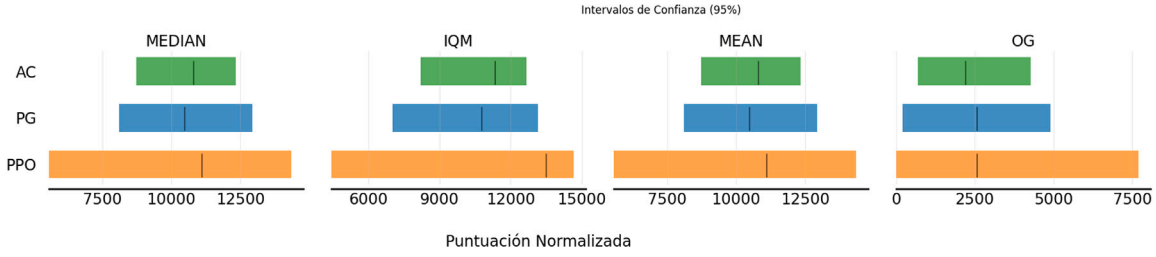

Fig. 9. Final-window aggregate metrics (Median, IQM, Mean, and Optimality Gap) with 95% stratified-bootstrap confidence intervals (5 seeds) for AC, PG, and PPO.

Table 10
Daily net benefit comparison on Database 1 (USD/day). Higher values indicate better economic performance.

Method	Daily net benefit (USD/day)
Database 1 (USD/day)	
DQN	**69.72 ± 2.86**
DQN (Hau et al., 2020)	53.45
MGO (Hau et al., 2020)	69.42
PER	59.22 ± 1.36
C51	51.02 ± 0.83
PPO	49.27 ± 1.41

approximately 15.1%, 26.8%, and 29.3%, respectively, which suggests that—within this dataset, reward specification, and training budget—the tested extensions do not translate into higher terminal economic gain compared to the baseline DQN policy.

Table 11
Comparison of reinforcement learning algorithms: Cost, Carbon emissions, and Time.

Algorithm		Dollar cost	Carbon emission	Mean time
Value-based/Bald (2023)				
ϵ-greedy DQN	Stable updates	0.83	0.94	20.7 s
Ours DQN	Stable updates	0.82114	0.90166	15.6 s
DQN-Softmax	Stable updates	0.83	0.93	27.2 s
UA-DQN	Uncertainty-aware	0.82	0.91	95.3 s
Policy-based/Zangato et al. (2025)				
Ours-A3C		0.94	0.882	6.128 s
PPO	Constrained updates	1.0	0.937	–
SAC	Constrained updates	1.018	0.951	–
DDPG	Constrained updates	0.7581	0.937	37.1 s

6.2.1. *Case Study 3: Policy optimization benchmark*

Among the value-based methods in Table 11, our greedy DQN variant demonstrates strong performance with a relatively low dollar

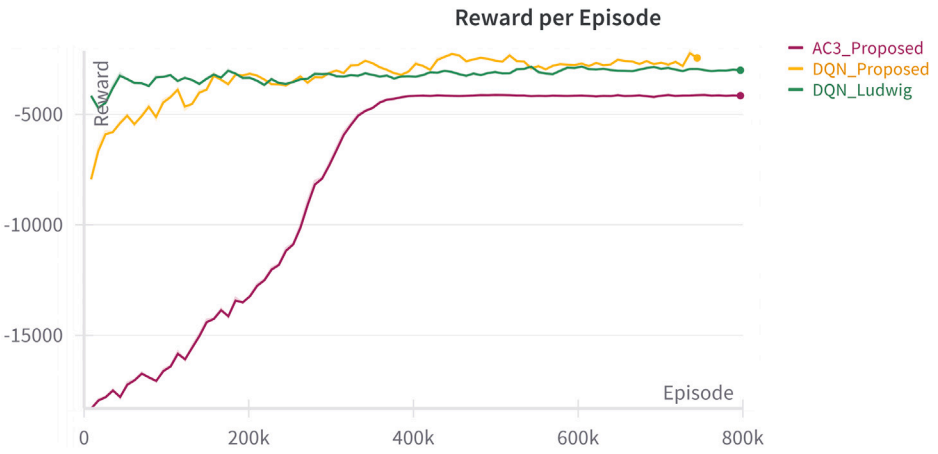

Fig. 10. Comparison of the convergence of tree algorithms applied to a specific set of buildings.

cost (0.8211), reduced carbon emissions (0.9017), and the lowest mean computation time (15.6 s) in its category—making. In comparison, UA-DQN exhibits higher computational time (95.3 s) despite having similar cost and emission metrics. The policy-based methods show broader variation. Our proposed A3C-Energy model achieves a well-balanced outcome, combining a competitive dollar cost (0.9400) with the lowest carbon emissions (0.8820) and the fastest execution time (6.128 s). Meanwhile, DDPG-Energy delivers the lowest dollar cost (0.7581) and reasonable emissions (0.9370) but requires significantly more computational time (37.1 s). These findings suggest that while value-based algorithms provide stable and consistent updates, policy-based models – particularly our A3C-Energy – offer faster, more adaptable performance depending on the specific optimization objective, whether minimizing cost, emissions, or computation time.

The Fig. 10 presents the episode reward trends for three reinforcement learning models: AC3-Proposed, DQN-Proposed, and DQN-Ludwig. Among them, DQN-Proposed shows the most promising performance, with a steady and consistent increase in rewards, eventually outperforming the benchmark DQN-Ludwig, which itself maintains stable and reliable learning behavior. In contrast, AC3-Proposed begins with very low rewards, shows initial improvement, but quickly plateaus and fails to reach the reward levels achieved by the DQN-based models. However, despite not achieving the highest reward values, the performance of AC3-Proposed might not be fully captured by the reward metric alone and could still contribute meaningfully in other aspects of task execution Table 11.

6.2.2. Comparison: State of the art, established techniques, and use case

In this section, a detailed comparison is conducted to rigorously validate the proposed models within a specific use case, benchmarking them against state-of-the-art methods and well-established reinforcement learning techniques.

Based on Table 12 and the methodological context, a critical discussion can be conducted to justify both the selection of the algorithms and the approach used to validate their performance. The table directly compares the proposed methods (DQN and A3C-Proposed) and several reference approaches reported in Chehade et al. (2024). This comparison serves as an indirect measure of the battery management optimization objective by evaluating the impact of each strategy on daily operational costs. Although some reference methods were not initially designed for the same specific purpose, their inclusion offers a fair and replicable framework for benchmarking across different datasets and experimental settings. In this regard, the proposed methods demonstrate competitive performance, which supports their practical applicability in real-world energy management scenarios.

Table 12
Comparative analysis of battery energy storage system (BESS) impact on daily operational costs.

Algorithm	Cost ($)
Proposed methods	
DQN	83,202
AC3-Proposed	84,941
Reference methods (Chehade et al., 2024)	
RL	85,100 ± 200
MPC	89,650 ± 665
MPC (exact model)	79,268
Baseline	90,989
Ground truth	79,268
No BMS	114,065

Table 13
Aggregate metrics on CartPole with 95% stratified bootstrap confidence intervals.

Metric	C51	PG	DQN
Median	328.19 ± 14.29	**411.06 ± 12.51**	311.15 ± 15.75
IQM	380.72 ± 22.35	**498.02 ± 6.12**	358.18 ± 29.28
Mean	328.19 ± 14.29	**411.06 ± 12.51**	311.15 ± 15.75
OG ↓	171.81 ± 14.29	**88.94 ± 12.51**	188.85 ± 15.75

Table 14
Metrics for HopperBulletEnv-v0.

Metric	A2C	DDPG	PPO	TD3
Median	667.88 ± 11.09	513.98 ± 17.46	1562.20 ± 35.14	**1952.79 ± 48.25**
IQM	731.76 ± 12.52	464.05 ± 25.79	1651.40 ± 51.47	**2176.17 ± 23.57**
Mean	667.88 ± 11.09	513.98 ± 17.46	1562.20 ± 35.14	**1952.79 ± 48.25**
OG ↓	1832.12 ± 11.09	1986.02 ± 17.46	937.80 ± 35.14	**547.21 ± 48.25**

6.2.3. Performance evaluation of established RL methods on standard tasks

To complement the domain-specific results, this subsection reports a set of standard benchmarks to assess how the evaluated reinforcement learning methods perform on widely recognized control tasks. The tables below summarize key aggregate metrics, including Median, Interquartile Mean (IQM), Mean, and Optimality Gap (OG), for CartPole and multiple Bullet environments. These results provide an additional layer of validation by highlighting consistent, scenario-robust behavior across the benchmarked control tasks, offering supportive evidence of performance transferability under the evaluated settings beyond the specific microgrid application.

The results in Tables 13 16, 15 and 14 indicate that, on standard benchmark tasks, the tested reinforcement learning algorithms deliver significant performance differences. For CartPole, the Policy Gradient

Table 15
Metrics for HalfCheetahBulletEnv-v0.

Metric	A2C	DDPG	PPO	TD3
Median	616.65 ± 48.21	448.42 ± 35.30	1285.46 ± 36.32	1220.58 ± 39.10
IQM	781.76 ± 55.37	665.87 ± 34.99	**1483.15 ± 28.30**	1368.05 ± 29.80
Mean	616.65 ± 48.21	448.42 ± 35.30	1285.46 ± 36.32	1220.58 ± 39.10
OG ↓	513.52 ± 40.62	551.58 ± 35.30	187.38 ± 23.58	**96.95 ± 24.59**

Table 16
Metrics for Walker2DBulletEnv-v0.

Metric	A2C	DDPG	PPO	TD3
Median	422.62 ± 12.18	337.23 ± 15.90	552.53 ± 14.60	**1243.43 ± 41.52**
IQM	438.54 ± 19.58	231.15 ± 17.80	604.64 ± 23.68	**1377.36 ± 42.62**
Mean	422.62 ± 12.18	337.23 ± 15.90	552.53 ± 14.60	**1243.43 ± 41.52**
OG ↓	577.38 ± 12.18	670.58 ± 14.87	447.47 ± 14.60	**115.99 ± 22.70**

(PG) method achieves a median score around 25% higher than C51 and over 30% higher than DQN, while its optimality gap is reduced by nearly 48% compared to C51 and by more than 53% compared to DQN. For Bullet environments, TD3 shows up to a 30% higher median return than PPO and more than 190% higher than A2C in the Hopper scenario. Similar trends appear in HalfCheetah and Walker2D, where TD3 reduces the optimality gap by more than 80% compared to A2C and by about 50% compared to PPO, confirming its superior sample efficiency and stable convergence under more complex dynamics.

6.3. Qualitative policy analysis across representative PV–BESS operating days

To complement the aggregate performance metrics, this subsection provides a qualitative, case-based interpretation of the learned control policies on three representative operating days. The selected cases are designed to highlight distinct PV–BESS operating regimes that frequently arise in practice.

Case 1: Day with surplus solar power and available BESS capacity

This case illustrates a surplus-PV regime with available BESS headroom, which is useful to visually interpret how different policies schedule charging/discharging under overlapping PV and price signals. In Fig. 11, DQN and PPO both exhibit comparatively stable SoC trajectories: DQN tends to execute more decisive charge–discharge blocks aligned with price valleys/peaks, while PPO typically produces smoother adjustments, reducing rapid oscillations when signals overlap. C51 can also appear smooth in this regime; however, its behavior can be more sensitive to reward scaling and support calibration in categorical distributional updates, which may affect how consistently it transfers across settings.

Crucially, these qualitative observations should not be interpreted as a universal ranking. The scenario-level evidence reported earlier suggests that differences between top methods are often modest and can fall within uncertainty bands. For example, Table 7 shows that although C51 attains the highest point estimates in Median/IQM, PPO remains close and the between-method gaps are comparable to the corresponding uncertainty bands effectively similar terminal performance under the current protocol Moreover, Table 8 indicates that PPO can achieve competitive typical performance but with wider confidence intervals, whereas DQN exhibits tighter bounds, reflecting more consistent outcomes across seeds. Taken together, the most consistent performers across the reported experiments are therefore DQN as a stable baseline with tight uncertainty and PPO as a generally competitive policy-gradient method, while gains from alternative variants may be more case-dependent.

Case 2 (H4): Surplus PV with ESS reaching full capacity

This case focuses on household H4 (StoreNet) on a representative day with surplus PV generation where the ESS reaches (or approaches) full capacity. Fig. 12 shows the stacked profiles of PV generation, load, battery actions, electricity price, and the resulting SoC for three controllers (AC, PPO, and DQN). The purpose of this qualitative example is to illustrate how different policies handle *capacity saturation*: when PV is abundant, effective strategies should charge early while headroom exists, avoid unnecessary cycling near the upper SoC bound, and preserve stored energy for later high-price/high-load periods.

Across the three methods, PPO generally exhibits smoother action adjustments and a more gradual SoC evolution, which helps avoid rapid charge–discharge oscillations around the saturation region. DQN, operating over discrete set-points, tends to apply more step-like charge/discharge blocks, which can capture price-driven opportunities but may lead to sharper SoC changes. The AC baseline shows intermediate behavior, following PV availability while adjusting actions to meet load and price variations. Overall, the figure suggests that PPO and DQN provide comparatively consistent control patterns in this saturation regime, albeit with different styles (smooth vs. step-like), underscoring that "stability" can reflect either conservative policy updates PPO or structured set-point decisions DQN.

7. Conclusions

This work investigated DRL for PV–BESS energy management under realistic price, load, and PV variability. The primary objective was to provide a transparent problem formulation, an implementable reward design that makes explicit the main cost–penalty trade-offs, and a consistent empirical comparison across heterogeneous datasets and operating regimes.

Across the evaluated scenarios, the results suggest that value-based methods (DQN and its variants) can achieve strong economic performance under discrete set-point control, while policy-gradient/actor–critic methods PPO/A3C can yield competitive and often smoother control behavior in several regimes. Importantly, *no single algorithm dominates across all datasets and metrics*: performance rankings depend on the dataset characteristics, the operating regime PV surplus vs. high-stress demand, and the reward scaling/weights. To strengthen statistical credibility, we therefore emphasize the uncertainty-aware summaries and interval estimates reported in Tables 7–8 rather than relying on point estimates alone.

A practical takeaway is that DQN and PPO tend to be among the most *consistent* choices across the reported experiments: DQN provides a strong baseline in cost-oriented objectives, whereas PPO often delivers stable policy behavior and competitive typical performance. Under within-dataset temporal hold-out, the results reflect robustness to temporal shifts under a fixed environment specification. Under cross-context transfer (StoreNet H4), the results reflect out-of-domain performance in a distinct household regime, where relative trends may differ from the temporal hold-out setting Nevertheless, the observed gains should be interpreted with care. First, the relative margins between top-performing methods are often small and, in some settings, overlap within the reported uncertainty estimates, indicating that algorithm rankings are not always statistically separable. Under temporal hold-out evaluation (within-dataset), performance is regime-dependent: controllers that appear advantageous in PV-surplus or mild-price-volatility periods may not retain the same advantage under high-stress demand peaks, tighter SoC constraints, or different reward weightings that prioritize reserves or battery wear. Under cross-context transfer, dataset characteristics such as time resolution and price structure can shift outcomes, so policies optimized for one context may degrade when applied to a different household or market environment. For these reasons, the conclusions should be read as *protocol- and scenario-conditional evidence* useful for guiding algorithm selection and design choices in similar settings rather than as universal statements of dominance.

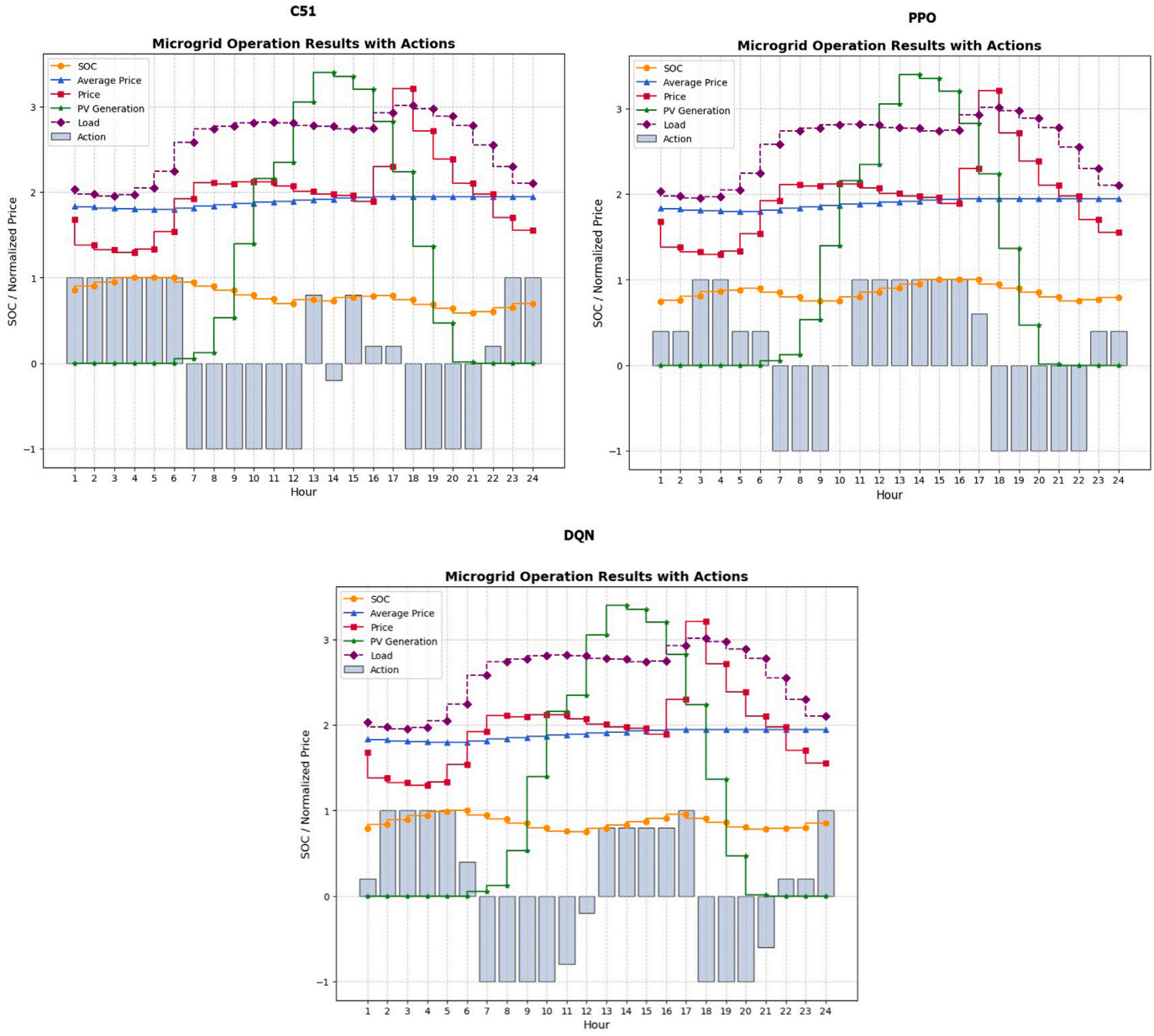

Fig. 11. ESS actions with DQN in Case 1.

Beyond these comparative findings, this study contributes a realistic and reproducible evaluation setup for PV–BESS EMS control. Specifically, the experiments are conducted with a physically grounded battery parameterization including non-ideal charge/discharge efficiencies and an operational DoD reference and a transparent multi-term objective that makes the economic–degradation trade-off explicit through interpretable cost and penalty components. This design enables a controlled comparison against a rule-based controller under the same battery dynamics. Across the reported experiments, the DRL policies generally attain higher terminal net benefit than the heuristic baseline, although the magnitude of the improvement varies by scenario and dataset. To support a statistically credible interpretation, results are reported using robust summaries (median/IQM) together with 95% bootstrap confidence intervals, which helps distinguish consistent gains from setting-dependent variations.

This study also has limitations. The experiments are simulation-based and rely on specific reward parameterizations, a discrete action space for several controllers, and simplified assumptions (e.g., aggregation resolution and data availability). Future work should extend the evaluation to continuous-action control with ramp-rate constraints, broader market/price models, and more explicit battery-health dynamics SoH- or temperature-coupled aging, ideally with field validation or hardware-in-the-loop testing.

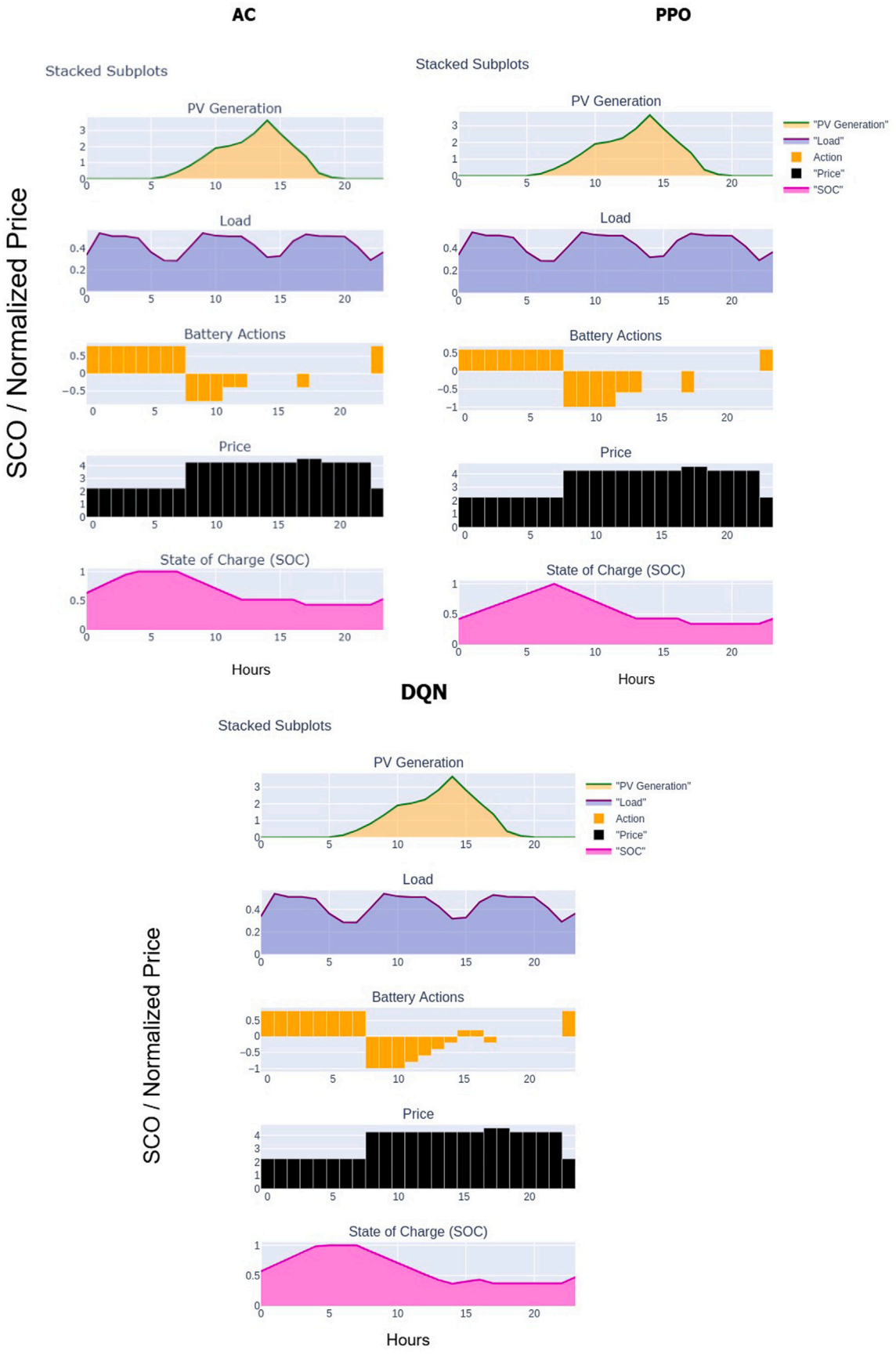

Fig. 12. BESS actions with PPO in Case 2.

CRediT authorship contribution statement

Sebastián López Flórez: Conceptualization, Data curation, Methodology, Writing – original draft. **Guillermo Hernández:** Formal analysis, Methodology, Writing – review & editing. **Javier Prieto:** Supervision, Project administration, Funding acquisition. **Fernando de la Prieta:** Supervision, Funding acquisition, Writing – review & editing.

Declaration of AI Tool Usage

During the preparation of this manuscript, the author(s) used ChatGPT for the purpose of generating initial verifying grammar. After using this tool, the author(s) carefully reviewed and edited the content as necessary and take full responsibility for the published article's accuracy and completeness.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Sebastian lopez florez reports financial support was provided by IoTalentum - H2020 MSCA ITN European Training Network on IoT. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The IoTalentum Project within the Framework of Marie Skłodowska-Curie Actions Innovative Training Networks (ITN)-European Training Networks (ETN), which is funded by the European Union Horizon 2020 Research and Innovation

Appendix A. Rule-based baseline controller

This appendix describes the rule-based (heuristic) baseline controller used for comparison against the DQN-based. The controller operates on the same battery model as the learning-based agents, ensuring that all methods are evaluated under identical electrochemical and degradation dynamics.

A.1. Action space

The controller selects a continuous control signal $a_t \in [-1, 1]$, where $a_t > 0$ denotes charging and $a_t < 0$ denotes discharging. This continuous signal is then mapped to the nearest element of the discrete action set

$$\mathcal{A} = \{a^{(0)}, a^{(1)}, \ldots, a^{(N_{\text{act}}-1)}\},$$

with $a^{(k)}$ uniformly spaced in $[-1, 1]$, exactly as in the DQN agent. The selected discrete action index is passed to the battery model, which computes the next SOC and the instantaneous economic reward.

A.2. Control logic

The rule-based controller combines two ideas that are widely used in the PV–battery literature (Azuatalam et al., 2019; Moghimi et al., 2016; Salpakari & Lund, 2016): *self-consumption maximization* (SCM), prioritizing the use of local PV generation, and *time-of-use price arbitrage*, charging when prices are low and discharging when prices are high.

To prevent excessive degradation, the controller enforces an operational SOC window $[s_{\min}, s_{\max}]$, typically chosen as $s_{\min} \approx 0.2$ and $s_{\max} \approx 0.8$, in line with recommended practice for lithium-ion batteries. Two price thresholds are defined relative to the running average price:

$$\lambda_{\text{cheap}} = \bar{\lambda}_t - \Delta_{\text{cheap}}, \qquad \lambda_{\text{exp}} = \bar{\lambda}_t + \Delta_{\text{exp}},$$

with $\Delta_{\text{cheap}}, \Delta_{\text{exp}} > 0$ tuned heuristically.

Let $P_t^{\text{net}} = P_t^{\ell} - P_t^{\text{pv}}$ denote the net load (positive if additional power must be imported, negative if there is surplus PV). The continuous control a_t is chosen according to the following rules:

1. **PV surplus priority (SCM):**
   - If $P_t^{\text{net}} < 0$ (PV surplus) and $s_t < s_{\max}$, the controller charges the battery:
   $$a_t = \begin{cases} a_{\max} & \text{if } s_t \le s_{\min}, \\ 0.7\, a_{\max} & \text{if } s_{\min} < s_t < s_{\max}, \end{cases}$$
   where $a_{\max} \in (0, 1]$ is the maximum charging command.
2. **Operation without PV surplus:** When $P_t^{\text{net}} \ge 0$, the controller falls back to a price-based arbitrage policy, constrained by the SOC window:
   - If $s_t \le s_{\min}$ (low SOC), discharging is avoided. The controller will only charge from the grid if the price is sufficiently low:
   $$a_t = \begin{cases} 0.7\, a_{\max} & \text{if } \lambda_t \le \lambda_{\text{cheap}}, \\ 0 & \text{otherwise.} \end{cases}$$
   - If $s_t \ge s_{\max}$ (high SOC), further charging is avoided. If there is net load and the price is high, the controller discharges:
   $$a_t = \begin{cases} a_{\min} & \text{if } P_t^{\text{net}} > 0 \text{ and } \lambda_t \ge \lambda_{\text{exp}}, \\ 0 & \text{otherwise,} \end{cases}$$
   where $a_{\min} \in [-1, 0)$ is the maximum discharging command.
   - If $s_{\min} < s_t < s_{\max}$ (SOC in the mid-range), the controller performs a simple price arbitrage:
   $$a_t = \begin{cases} 0.7\, a_{\min} & \text{if } P_t^{\text{net}} > 0,\ \lambda_t \ge \lambda_{\text{exp}}, \\ 0.5\, a_{\max} & \text{if } \lambda_t \le \lambda_{\text{cheap}}, \\ 0 & \text{otherwise.} \end{cases}$$

Finally, the continuous action a_t is clipped to $[-1, 1]$ and mapped to the nearest discrete level in $\mathcal{A}$. The resulting control sequence is therefore directly comparable with the discrete actions produced by the DQN agent.

Appendix B. Additional training details

B.1. Exploration schedule (ϵ-decay)

We employ an exponential ϵ-greedy decay schedule,

$$\epsilon_t = \epsilon_{\min} + (\epsilon_{\max} - \epsilon_{\min}) \exp(-\lambda t), \tag{B.1}$$

with the parameters reported in Table B.17.

Table B.17
Exploration decay schedule parameters used in the experiments.

$\epsilon_{\max}$	$\epsilon_{\min}$	λ
1.0	0.01	1×10^{-4}

Appendix C. Model parameters for GEFCom2014 configuration

This section outlines the specific model parameters used to configure the experiments based on the GEFCom2014 dataset. By explicitly defining the GEFCom2014 configuration, the study ensures consistency with the benchmark dataset, enables fair comparisons with other methods, and supports the reproducibility of results for energy forecasting and storage optimization tasks (see Table C.18).

Table C.18
Comparison of reinforcement learning algorithms for energy management in microgrids.

Algorithm	Action space	Operating conditions/Observables	Training modality	Key features/Technical notes	Limitations/Remarks
DQN	Discrete	Stable prices; deterministic SoC, PV and demand profiles	Offline; batch with history	Experience replay and target network; supports batch updates	Requires action discretization; sensitive to reward shaping and learning parameters
PER-DQN	Discrete	Mild stochasticity; time-correlated disturbances	Offline; prioritized replay	Samples by TD-error (IS correction); focuses on critical transitions	Risk of overfitting rare transitions; careful tuning of prioritization and weights needed
Double DQN	Discrete	Low-noise environments; deterministic load/price patterns	Offline; dual Q-networks	Reduces overestimation by decoupling selection and evaluation	Synchronization critical; limited to discrete action spaces
NoisyNet-DQN	Discrete	High volatility in signals; uncertain demand	Offline; stochastic exploration	Trainable noise in network weights enables adaptive exploration	Sensitive to noise design; additional training overhead
C51	Discrete	Heavy-tailed returns; probabilistic load/price dynamics	Offline; fixed support setup	Learns return distribution; projection on $[v_{min}, v_{max}]$	Robust to noise; higher complexity and memory demand
PG	Discrete	Differentiable rewards; low–moderate dynamics	Offline or episodic simulation	Direct policy gradient optimization	High variance; requires strong reward signal and variance reduction
PPO	Continuous	Medium–high stochasticity; RES integration	Offline or simulated environments	Clipped surrogate objective; GAE improves stability	Hyperparameter sensitive; high training cost
A2C/A3C	Discrete/ Continuous	Time-varying pricing; high-frequency PV dynamics	Online or asynchronous simulation	Advantage Actor–Critic; A3C accelerates convergence with parallel agents	Sensitive to reward scaling; sync issues; higher computational cost
DDPG	Continuous	Smooth variations; mildly stochastic conditions	Offline or simulated continuous control	Deterministic policy gradient; target networks + replay buffer	Exploration unstable; sensitive to normalization and learning rates
SAC	Continuous	Highly stochastic or partially observable systems	Offline or simulated continuous control	Entropy-regularized Actor–Critic; stable exploration with soft Q-learning	Computationally expensive; entropy coefficient tuning required

Table C.19
Model parameters for GEFCom2014 configuration.

(a) PPO parameters (GEFCom2014).

Parameter	Value
State space	
State size (PV, load, RTP, 24 h avg RTP, SOC)	5
Network architecture	
Actor hidden layers	40–160 ELU
Critic hidden layers	40–160 ELU
Output activation (Actor)	Softmax
Output activation (Critic)	Linear
Training parameters	
Learning rate (Actor)	0.0005
Learning rate (Critic)	0.001
Discount factor	0.99
GAE parameter	0.98
Clip ratio	0.2
Training setup	
Episodes	100
Timesteps per episode	24
Epochs	5
Optimizer	Adam

(b) DoubleDQN parameters (GEFCom2014).

Parameter	Value
Network architecture	
State size	5
Action size	11
Hidden layer 1	40 ELU
Hidden layer 2	160 ELU
Output layer	Linear
Training hyperparameters	
Learning rate	0.001
Batch size	64
Target update frequency	100 steps
Discount factor (γ)	0.95
Memory size	10,000
Exploration strategy	
Initial ϵ	1.0
Final ϵ	0.01
Decay rate	0.0001
Training setup	
Episodes	10
Timesteps per episode	24
Pretrain samples	10,000
Optimizer	Adam

Table C.20
DQN parameterizations for GEFCom2014 (standard vs. NoisyNet exploration).

(a) DQN2013 (GEFCom2014).

Parameter	Value
Network architecture	
State size	5
Action size	11 discrete actions
Hidden layer 1	40 ELU
Hidden layer 2	160 ELU
Output layer	Linear
Weight init	Glorot uniform
Training hyperparameters	
Learning rate	0.001
Batch size	64
Discount factor (γ)	0.99
Replay memory size	2000
Optimizer	Adam
Exploration strategy	
Initial exploration (ϵ_{start})	1.0
Final exploration (ϵ_{stop})	0.01
Decay rate	0.0001
Training setup	
Episodes	10
Timesteps per episode	24
Warm-start samples	1000
Target network update	Hard replacement

(b) NoisyNet-DQN (GEFCom2014).

Parameter	Value
Network architecture	
State size	5
Action size	11 discrete actions
Noisy layer 1	40 ELU units
Noisy layer 2	160 ELU units
Noisy output layer	Linear
Noise initialization (σ_0)	0.5
Noise parameters	
Noise type	Factorized Gaussian
Noise transform	$f(x) = sign(x)\sqrt{\lvert x\rvert}$
Weight initialization	Uniform $[-\frac{1}{\sqrt{p}}, \frac{1}{\sqrt{p}}]$
Training hyperparameters	
Learning rate	0.001
Batch size	64
Target update frequency	100 steps
Discount factor (γ)	0.95
Replay memory size	10,000
Training setup	
Episodes	10
Timesteps per episode	24
Optimizer	Adam
Exploration	Pure noisy network policy

Table C.21
Model parameters for GEFCom2014 configuration (PER-DQN vs. Actor–Critic).

(a) PER-DQN parameters (GEFCom2014).

Parameter	Value
Network architecture	
State size	5
Action size	11 discrete actions
Hidden layer 1	40 ELU units
Hidden layer 2	160 ELU units
Output layer	Linear
Weight initialization	Glorot uniform
Prioritized replay parameters	
Memory size	10,000
Priority exponent (α)	0.6
Importance sampling (β)	0.4 to 1.0
β increment	0.001 per sample
Base priority (e)	0.001
Training hyperparameters	
Learning rate	0.001
Batch size	64
Target update frequency	20 steps
Discount factor (γ)	0.95
Training setup	
Episodes	10
Timesteps per episode	24
Initial exploration (ϵ_{start})	1.0
Final exploration (ϵ_{stop})	0.01
Exploration decay rate	0.0001
Optimizer	Adam

(b) Actor–Critic parameters (GEFCom2014).

Parameter	Value
Network architecture	
State size	5
Action size	11 discrete actions
Actor network	
Hidden layer 1	40 ELU units
Hidden layer 2	160 ELU units
Output activation	Softmax
Learning rate	0.0005
Entropy coefficient (β)	0.01
Critic network	
Hidden layer 1	40 ELU units
Hidden layer 2	160 ELU units
Output activation	Linear
Learning rate	0.001
Training hyperparameters	
Discount factor (γ)	0.99
Update method	n-step TD
Batch processing	Episode-wise
Training setup	
Episodes	100
Timesteps per episode	24
Optimizer	Adam
Exploration	Policy sampling

Table C.22
Model parameters for GEFCom2014 configuration (C51 vs. Policy gradient).

(a) C51 parameters (GEFCom2014).

Parameter	Value
Network architecture	
State size	5
Action size	11 discrete
Hidden layer 1	40 ELU
Hidden layer 2	160 ELU
Output atoms	51
Distributional parameters	
Value range (V_{min} to V_{max})	−30 to 30
Atom size (Δ_z)	1.176
Support points	51
Training hyperparameters	
Learning rate	0.001
Batch size	64
Target update frequency	20 steps
Discount factor (γ)	0.95
Training setup	
Episodes	10
Timesteps per episode	24
Replay buffer size	10,000
Optimizer	Adam
Loss function	Categorical crossentropy
Exploration strategy	
Initial ϵ	1.0
Final ϵ	0.01
Decay rate	0.0001

(b) Policy Gradient parameters (GEFCom2014).

Parameter	Value
Network architecture	
State size	5
Action size	11 discrete
Hidden layer 1	40 ReLU units
Hidden layer 2	160 ReLU units
Output activation	Softmax
Policy gradient parameters	
Learning rate	0.0005
Entropy coefficient (β)	0.01
Loss function	Policy Gradient + Entropy
Reward discounting	Episode-wide
Training hyperparameters	
Discount factor (γ)	0.99
Update method	Monte Carlo
Batch processing	Full episodes
Training setup	
Episodes	150
Timesteps per episode	24
Optimizer	Adam
Exploration	Policy sampling

Table C.23
Model parameters for the StoreNet energy community configuration (PPO vs. DQN2013).

(a) PPO parameters (Energy community).

Parameter	Value
State space	
State representation	Community profile
Network architecture	
Hidden layers	64-128-64 ReLU
Output activation	Linear
Training parameters	
Learning rate	0.0003
Batch size	256
Discount factor	0.99
GAE parameter	0.95
Clip ratio	0.2
Training setup	
Training cycles	1000 epochs
Optimizer	Adam

(b) DQN2013 parameters (StoreNet).

Parameter	Value
Network architecture	
State representation	Community energy profile
Action size	11 discrete actions
Hidden layer 1	40 ELU units
Hidden layer 2	160 ELU units
Output activation	Linear
Weight initialization	Glorot uniform
Training hyperparameters	
Learning rate	0.001
Batch size	64
Discount factor (γ)	0.95
Replay buffer size	10,000
Target update frequency	20 steps
Exploration strategy	
Initial ϵ	1.0
Final ϵ	0.01
Decay rate	0.0001
Training setup	
Episodes	10
Timesteps per episode	24
Optimizer	Adam
Loss function	Mean squared error
Energy community specifics	
State components	5 features
Action space	Discrete control
Reward structure	Economic,community benefit

Table C.24
Model parameters for the StoreNet energy community configuration (C51 vs. Policy gradient).

(a) C51 parameters (StoreNet energy community).		(b) Policy Gradient parameters (StoreNet energy community).	
Parameter	Value	Parameter	Value
Network architecture		**Network architecture**	
State representation	5	State representation	5
Action size	11 discrete actions	Action size	11 discrete actions
Hidden layer 1	40 ELU units	Hidden layer 1	30 ReLU units
Hidden layer 2	160 ELU units	Hidden layer 2	60 ReLU units
Output atoms per action	51	Output activation	Softmax
Distributional parameters		**Policy gradient parameters**	
Value range (V_{min} to V_{max})	−30 to 30	Learning rate	0.0005
Atom size (Δ_z)	1.176	Entropy coefficient (β)	0.6
Support points	51	Loss function	Policy Gradient + Entropy
Training hyperparameters		Update method	Monte Carlo
Learning rate	0.001	**Training hyperparameters**	
Batch size	64	Discount factor (γ)	0.95
Target update frequency	20 steps	Batch processing	Full episodes
Discount factor (γ)	0.95	Episodes	10
Replay buffer size	10,000	Timesteps per episode	24
Energy community specifics		**Energy community specifics**	
Battery capacity	5 kWh (normalized)	Battery capacity	5 kWh (normalized)
Charge/discharge rate	1 kW (normalized)	Charge/discharge rate	1 kW (normalized)
SOC target	0.5 (50%)	SOC target	0.5 (50%)
Multiplier system	Piecewise nonlinear	**Exploration strategy**	
Exploration strategy		Method	Policy sampling
Initial ϵ	1.0	No ϵ-greedy	Pure stochastic policy
Final ϵ	0.01		
Decay rate	0.0001		

Table D.25
Actor-critic configuration for urban energy storage.

Parameter	Value
1. Network architecture	
Shared feature extractor	256 → 256 ReLU
Policy head	256 → 256 ReLU → Softmax
Value head	256 → 256 ReLU → Linear
Activation functions	ReLU (hidden), Softmax (policy)
2. Training configuration	
Learning rate	3e−4
Discount factor (γ)	0.99
Entropy coefficient	0.01
Batch size	64
Gradient clipping	0.5
Optimizer	Adam
3. Urban energy specifics	
State components	5 features per building
Action space	11 discrete storage actions
Reward function	Combined economic/environmental
Evaluation metrics	Price cost, Emission cost
4. Performance tracking	
Data logged	SOC, actions, rewards
Evaluation frequency	Full episodes
Decision time	<1 ms per timestep
Parallelization	Multi-building support

Table D.26
DDPG configuration for urban energy storage optimization.

Parameter	Value
1. Network architecture	
Actor network	CommNet (LSTM-based)
Critic network	MLP (3 hidden layers)
Actor hidden layers	256 → 256 LeakyReLU
Critic hidden layers	32 → 32 LeakyReLU
Activation functions	LeakyReLU (hidden), Tanh (actor output)
2. Training configuration	
Learning rate	3e−4
Discount factor (γ)	0.95
Batch size	128
Memory buffer size	10,000
Target network update (τ)	0.001
Optimizer	Adam
Exploration decay	Linear 0.5 to 0.05
3. Urban energy specifics	
State components	Normalized observation features
Action space	Continuous [−1, 1] (per building)
Reward function	Custom economic/environmental
Evaluation metrics	Price cost, Emission cost
4. Performance tracking	
Training duration	3 years (26,280 steps)
Update frequency	Every timestep
Decision time	<1 ms per timestep
Parallelization	Multi-building support

C.1.

See Tables C.19–C.24.

Appendix D. Model parameters for urban energy storage optimization

This section describes the main model parameters that define the urban energy storage optimization framework. By detailing these settings, the study ensures that the experiments are transparent, replicable, and aligned with realistic urban microgrid conditions (see Tables D.25 and D.26).

Appendix E. Performance analysis on Walker2DBulletEnv-v0

To evaluate the learning dynamics and score distribution of the tested algorithms under the Walker2DBulletEnv-v0 scenario, Fig. E.13

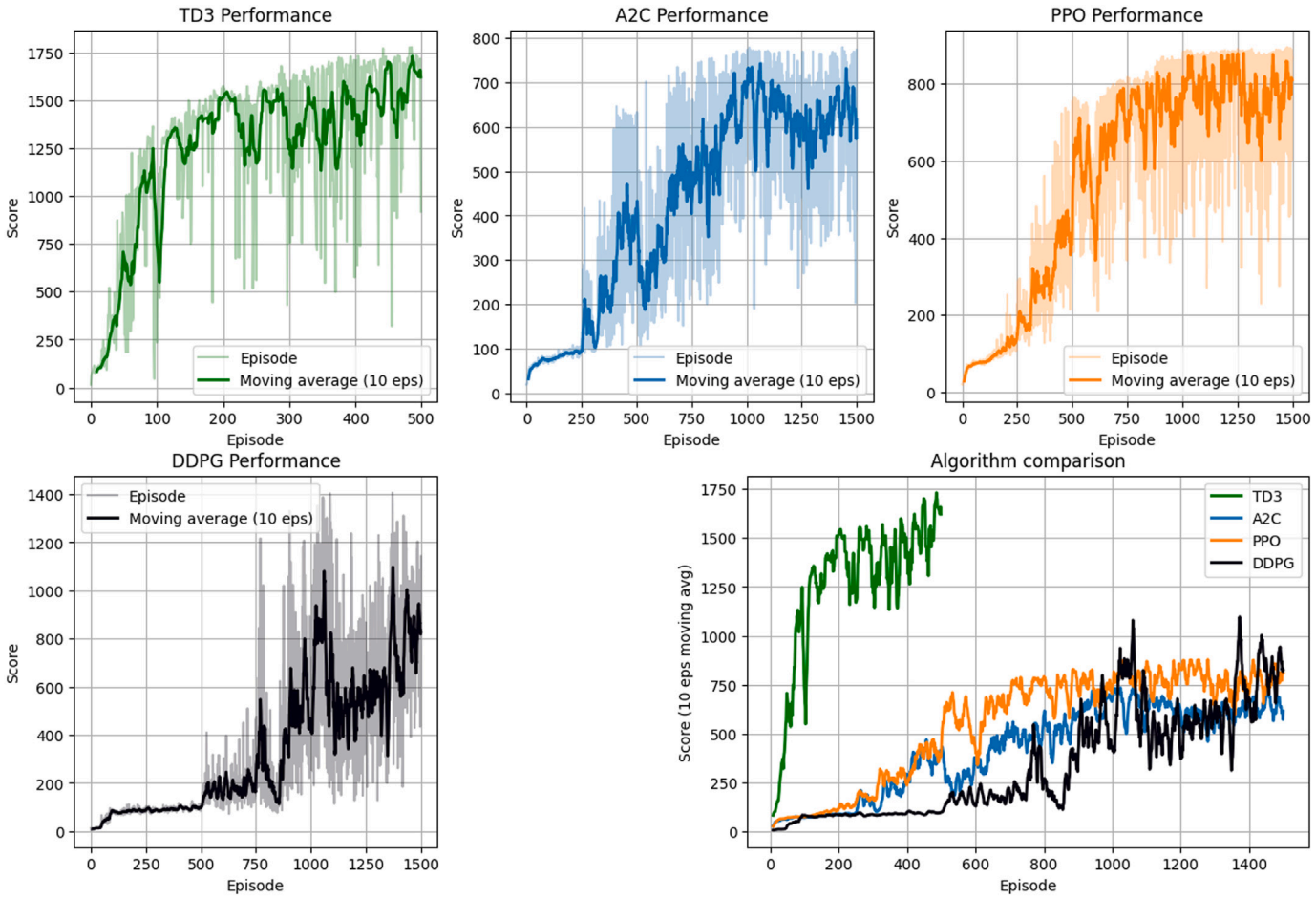

Fig. E.13. Aggregate performance metrics for DDPG, PPO, A2C, and TD3 on Walker2DBulletEnv-v0. The bars show the Median, IQM, Mean, and OG values for each algorithm.

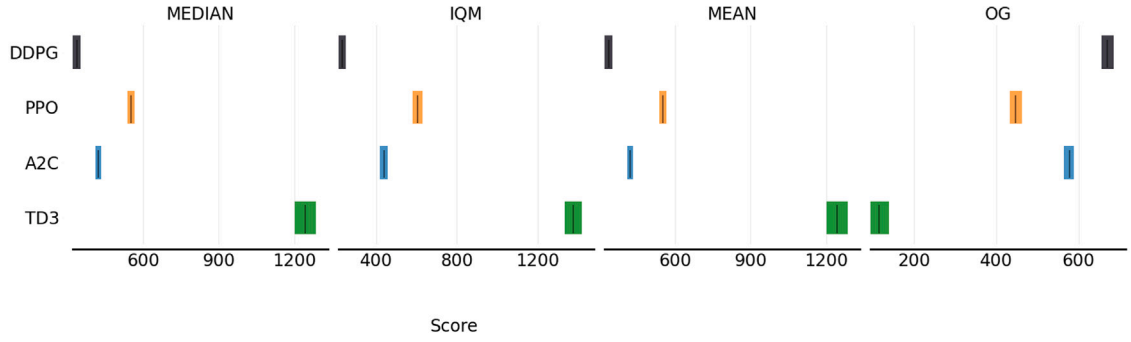

Fig. E.14. Learning curves of DDPG, PPO, A2C, and TD3 on Walker2DBulletEnv-v0. The plots show individual episode scores, moving averages, and an overall comparison to highlight training stability and convergence speed.

and E.14 present a detailed comparison of aggregated metrics and episode-based performance trends. These results illustrate how each agent performs in terms of median score, IQM, mean, and optimality gap, and how learning progresses across episodes.

To analyze the performance of the algorithms under the HalfCheetahBulletEnv-v0 scenario, Fig. E.15 and E.16 present the aggregated performance metrics and the episode-based learning dynamics. The summary presents the performance of each agent in terms of median score, IQM, mean, and optimality gap, while the learning curves illustrate the convergence speed and training stability across episodes.

Fig. E.17 and E.18 show the performance evaluation for HopperBulletEnv-v0, highlighting both aggregate metrics and detailed learning behavior of the tested algorithms. The bar plot summarizes the median, IQM, mean, and optimality gap for each agent, while the learning curves illustrate training stability and performance trends throughout the episodes.

Data availability

Data will be made available on request.

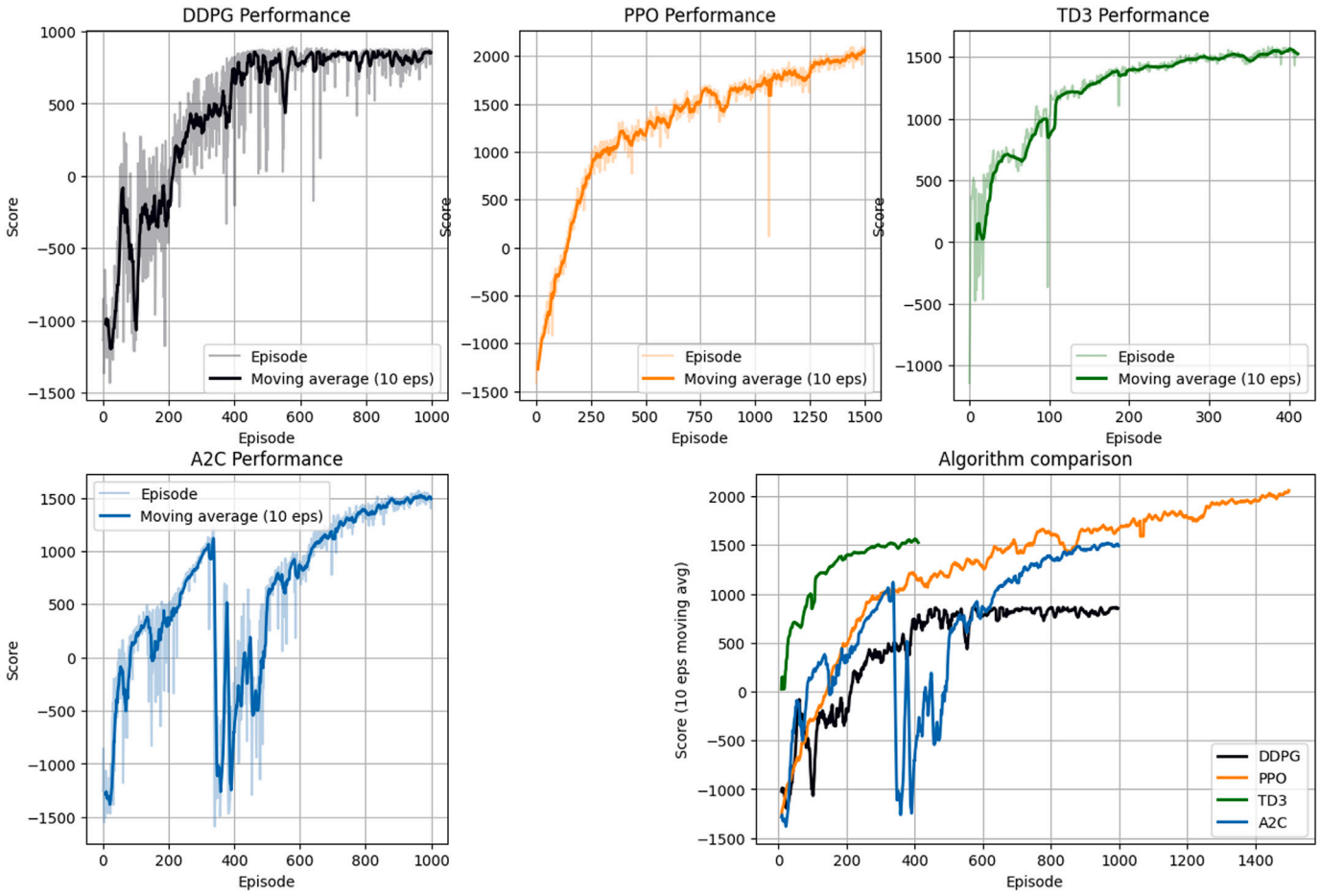

Fig. E.15. Aggregate performance metrics for DDPG, PPO, A2C, and TD3 on HalfCheetahBulletEnv-v0. The bars show the Median, IQM, Mean, and OG for each algorithm.

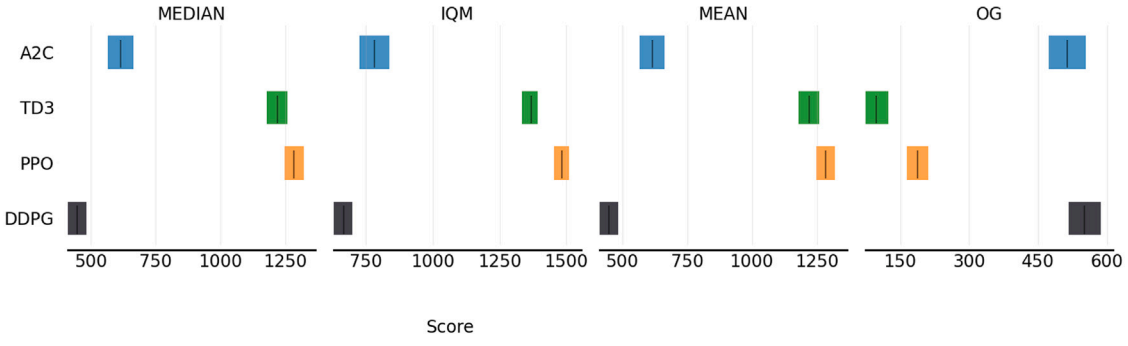

Fig. E.16. Learning curves of DDPG, PPO, A2C, and TD3 on HalfCheetahBulletEnv-v0. The plots display episode scores, moving averages, and an overall comparison of training performance.

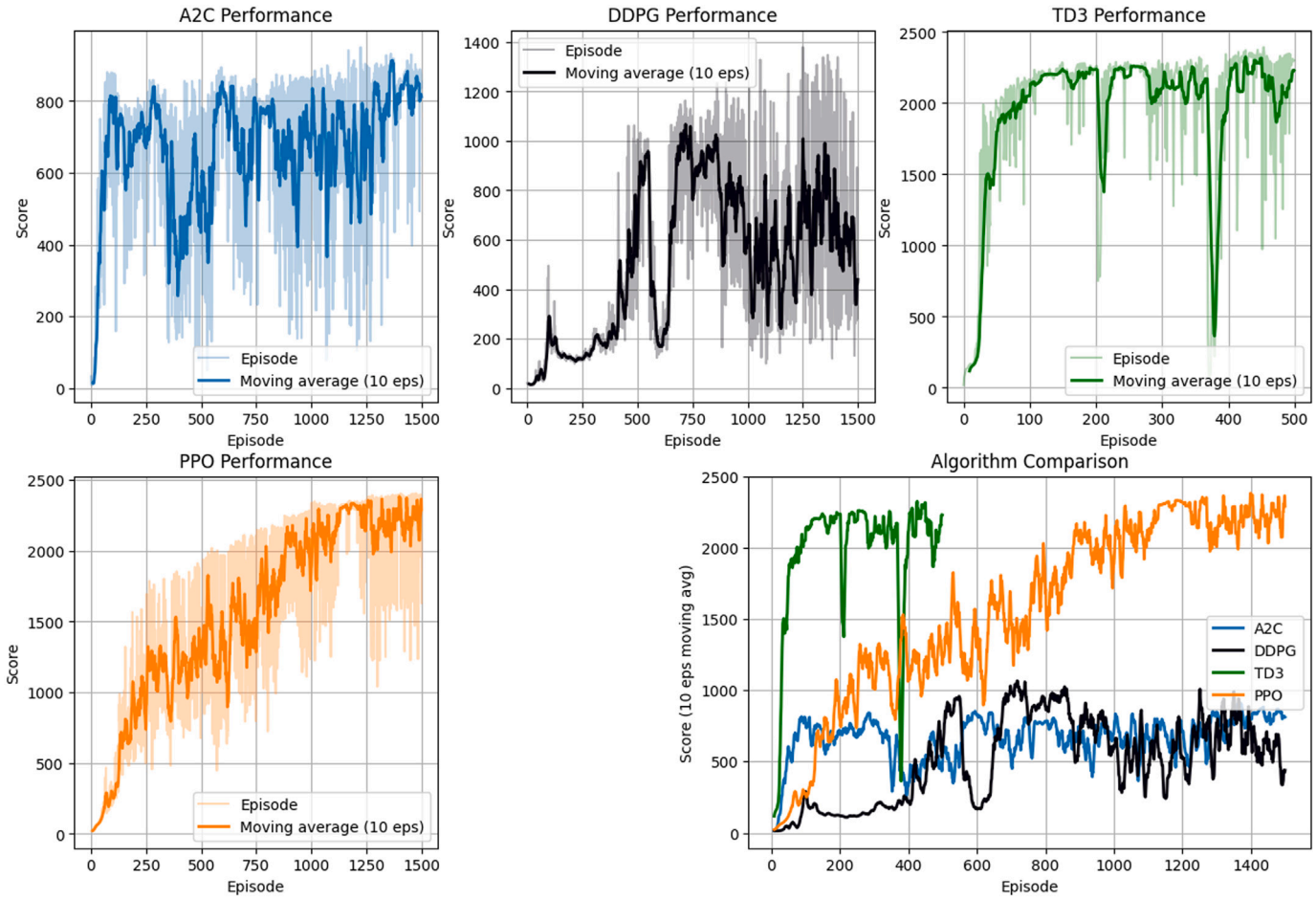

Fig. E.17. Aggregate performance metrics for DDPG, PPO, A2C, and TD3 on HopperBulletEnv-v0. The figure reports Median, IQM, Mean, and OG for each method.

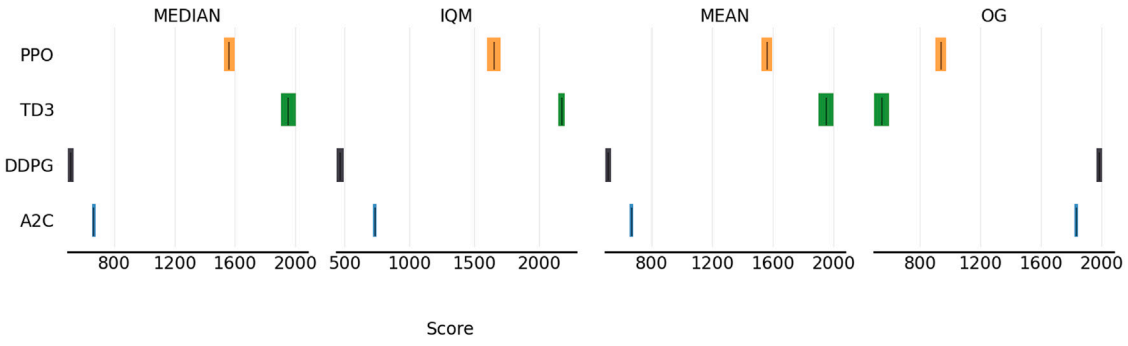

Fig. E.18. Learning curves for DDPG, PPO, A2C, and TD3 on HopperBulletEnv-v0. The graphs show raw episode scores, smoothed moving averages, and cross-method comparisons.

References

Affi, S., Cherif, H., & Belhadj, J. (2021). Smart system management and techno-environmental optimal sizing of a desalination plant powered by renewables with energy storage. *International Journal of Energy Research*, *45*(5), 7501–7520.

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A. C., & Bellemare, M. (2021). Deep reinforcement learning at the edge of the statistical precipice. *Advances in Neural Information Processing Systems*, *34*, 29304–29320.

Aih, H. C. (2020). *Energy management & economic evaluation of grid-connected microgrid operation*. National University of Singapore, Department of Electrical & Computer Engineering, Matric Number: A0111953L.

Ali, N., Wahid, A., Shaw, R., & Mason, K. (2024). A reinforcement learning approach to dairy farm battery management using Q learning. *Journal of Energy Storage*, *93*, Article 112031.

Anderson, R. I., & Fok, R. (2000). Hotel industry efficiency: An advanced linear programming examination. *American Business Review*, *18*(1).

Azuatalam, D., Angenendt, G., Kümpel, T., & Coll-Mayor, D. (2019). Energy management of small-scale PV-battery systems: A systematic review and a case study. *Renewable and Sustainable Energy Reviews*, *113*, Article 109247. http://dx.doi.org/10.1016/j.rser.2019.109247, Analiza estrategias heurísticas como self-consumption maximisation (SCM), time-of-use arbitrage (ToUA) y su combinación SCM+ToUA para sistemas PV–batería..

Bald, L. (2023). *Uncertainty-aware reinforcement learning for demand response in energy systems* [Master's thesis], Tübingen, Germany: Eberhard Karls Universität Tübingen, Supervised by Nicole Ludwig. URL: https://github.com/ludwigbald/msc-thesis.

Cai, C., Gan, F., Cui, Y., Li, B., & Hou, S. (2025). Entropy-driven multi agent deep reinforcement learning for resilient distribution networks: Coordinating MESS and microgrids. *International Journal of Electrical Power & Energy Systems*, *170*, Article 110968.

Cao, J., Harrold, D., Fan, Z., Morstyn, T., Healey, D., & Li, K. (2020). Deep reinforcement learning-based energy storage arbitrage with accurate lithium-ion battery degradation model. *IEEE Transactions on Smart Grid*, *11*(5), 4513–4521.

Cardoso, G., Stadler, M., Siddiqui, A., Marnay, C., DeForest, N., Barbosa-Póvoa, A., & Ferrão, P. (2013). Microgrid reliability modeling and battery scheduling using stochastic linear programming. *Electric Power Systems Research*, *103*, 61–69.

Chehade, M. F. E. H., Cho, Y.-h., Chinchali, S., & Zhu, H. (2024). Should we use model-free or model-based control? A case study of battery management systems. arXiv preprint arXiv:2407.15313.

Chen, C., Duan, S., Cai, T., Liu, B., & Hu, G. (2011). Smart energy management system for optimal microgrid economic operation. *IET Renewable Power Generation*, *5*(3), 258–267.

Chen, T., & Gao, C. (2023). Intelligent electric vehicle charging scheduling in transportation-energy nexus with distributional reinforcement learning. *IEEE/CAA Journal of Automatica Sinica*, *10*(11), 2171–2173.

Çiftçioğlu, A. Ö., Kazemi, F., & Shafighfard, T. (2025). Grey wolf optimizer integrated within boosting algorithm: Application in mechanical properties prediction of ultra high-performance concrete including carbon nanotubes. *Applied Materials Today*, *42*, Article 102601. http://dx.doi.org/10.1016/j.apmt.2025.102601.

Durán Gómez, P., Echevarría Camarero, F., Ogando-Martínez, A., & Carrasco Ortega, P. (2023). Profitability of alternative battery operation strategies in photovoltaic self-consumption systems under current regulatory framework and electricity prices in spain. *Energies*, *16*(21), 7375.

Flórez, S. L., Herniández, G., Gonziález-Briones, A., & de la Prieta, F. (2023). AI-optimized energy management for more efficient and sustainable microgrids. In *International symposium on distributed computing and artificial intelligence* (pp. 438–447). Springer.

Gassi, K. B., & Baysal, M. (2023). Improving real-time energy decision-making model with an actor-critic agent in modern microgrids with energy storage devices. *Energy*, *263*, Article 126105.

Hau, C., Radhakrishnan, K. K., Siu, J., & Panda, S. K. (2020). Reinforcement learning based energy management algorithm for energy trading and contingency reserve application in a microgrid. In *2020 IEEE PES innovative smart grid technologies europe (ISGT-europe)* (pp. 1005–1009). IEEE.

Hong, T., Pinson, P., Fan, S., Zareipour, H., Troccoli, A., & Hyndman, R. J. (2016). Probabilistic energy forecasting: Global energy forecasting competition 2014 and beyond.

IEC61970, D. (2003). *Energy management system application program interface (ems-api) part 301: Common information model (cim) base*. Geneva, Switzerland: IEC.

Jeong, J., Kim, S. W., & Kim, H. (2023). Deep reinforcement learning based real-time renewable energy bidding with battery control. *IEEE Transactions on Energy Markets, Policy and Regulation*, *1*(2), 85–96.

Kim, S. H., & Shin, Y.-J. (2023). Optimize the operating range for improving the cycle life of battery energy storage systems under uncertainty by managing the depth of discharge. *Journal of Energy Storage*, *73*, Article 109144.

Li, Z., Xiang, Y., & Liu, J. (2025). Forecasting error-aware optimal dispatch of wind-storage integrated power systems: A soft-actor-critic deep reinforcement learning approach. *Energy*, *318*, Article 134798.

Lin, M., You, Y., Meng, J., Wang, W., Wu, J., & Stroe, D.-I. (2023). Lithium-ion battery degradation trajectory early prediction with synthetic dataset and deep learning. *Journal of Energy Chemistry*, *85*, 534–546.

Liu, F., Hu, B., Li, R., & Li, Y. (2018). A novel control strategy of energy storage system considering prediction errors of photovoltaic power. In *2018 15th international conference on control, automation, robotics and vision* (pp. 1247–1251). IEEE.

Liu, F., Liu, Q., Tao, Q., Huang, Y., Li, D., & Sidorov, D. (2023). Deep reinforcement learning based energy storage management strategy considering prediction intervals of wind power. *International Journal of Electrical Power & Energy Systems*, *145*, Article 108608.

Liu, D., Zang, C., Zeng, P., Li, W., Wang, X., Liu, Y., & Xu, S. (2023). Deep reinforcement learning for real-time economic energy management of microgrid system considering uncertainties. *Frontiers in Energy Research*, *11*, Article 1163053.

Liu, Y., Zhang, D., & Gooi, H. B. (2020). Optimization strategy based on deep reinforcement learning for home energy management. *CSEE Journal of Power and Energy Systems*, *6*(3), 572–582.

Moghimi, M., Ghosh, A., Ledwich, G., et al. (2016). Rule-based energy management system in an experimental microgrid with the presence of time-of-use tariffs. *Renewable Energy*, Propone un EMS basado en reglas para una microrred con PV y batería, donde las decisiones de carga/descarga se toman en función de tarifas time-of-use y el estado de carga.

Nambisan, P., & Khanra, M. (2024). Optimal power-split of hybrid energy storage system using Pontryagin's minimum principle and deep reinforcement learning approach for electric vehicle application. *Engineering Applications of Artificial Intelligence*, *135*, Article 108769.

Neves, D. E., Ishitani, L., & do Patrocinio Junior, Z. K. G. (2024). Advances and challenges in learning from experience replay. *Artificial Intelligence Review*, *58*(2), 54.

Rezaeimozafar, M., Duffy, M., Monaghan, R. F., & Barrett, E. (2024). A hybrid heuristic-reinforcement learning-based real-time control model for residential behind-the-meter PV-battery systems. *Applied Energy*, *355*, Article 122244.

Sage, M., Campbell, J., & Zhao, Y. F. (2024). Enhancing battery storage energy arbitrage with deep reinforcement learning and time-series forecasting. *87899*, In *Energy sustainability*. American Society of Mechanical Engineers, Article V001T01A007.

Salpakari, J., & Lund, P. (2016). Optimal and rule-based control strategies for energy flexibility in buildings with PV. *Applied Energy*, *161*, 425–436. http://dx.doi.org/10.1016/j.apenergy.2015.10.036, Compara control óptimo y estrategias rule-based para maximizar autoconsumo de PV y reducir costes bajo señales de precio horarias..

Shafighfard, T., Asgarkhani, N., Kazemi, F., & Yoo, D.-Y. (2025). Transfer learning on stacked machine-learning model for predicting pull-out behavior of steel fibers from concrete. *Engineering Applications of Artificial Intelligence*, *158*, Article 111533. http://dx.doi.org/10.1016/j.engappai.2025.111533.

Shafighfard, T., Kazemi, F., Bagherzadeh, F., Mieloszyk, M., & Yoo, D. (2024). Chained machine learning model for predicting load capacity and ductility of steel fiber-reinforced concrete beams. *Computer-Aided Civil and Infrastructure Engineering*, *39*(23), 3573–3594. http://dx.doi.org/10.1111/mice.13164.

Shang, Y., Wu, W., Guo, J., Ma, Z., Sheng, W., Lv, Z., & Fu, C. (2020). Stochastic dispatch of energy storage in microgrids: An augmented reinforcement learning approach. *Applied Energy*, *261*, Article 114423.

Slama, S. B., & Mahmoud, M. (2023). A deep learning model for intelligent home energy management system using renewable energy. *Engineering Applications of Artificial Intelligence*, *123*, Article 106388.

Sobu, A., & Wu, G. (2012). Dynamic optimal schedule management method for microgrid system considering forecast errors of renewable power generations. In *2012 IEEE international conference on power system technology* (pp. 1–6). IEEE.

Su, T., Wu, T., Zhao, J., Scaglione, A., & Xie, L. (2025). A review of safe reinforcement learning methods for modern power systems. *Proceedings of the IEEE*.

Terkes, M., Demirci, A., & Gokalp, E. (2024). A comprehensive review of the aging mechanism and degradation costs of fresh and second-life batteries based on analytical and deterministic methods. *Energy Storage*, *6*(4), Article e661.

Wei, G., Chi, M., Liu, Z.-W., Ge, M., Li, C., & Liu, X. (2023). Deep reinforcement learning for real-time energy management in smart home. *IEEE Systems Journal*, *17*(2), 2489–2499.

Wu, N., & Wang, H. (2018). Deep learning adaptive dynamic programming for real time energy management and control strategy of micro-grid. *Journal of Cleaner Production*, *204*, 1169–1177.

Zangato, T., Osmani, A., & Alizadeh, P. (2025). Data-driven policy mapping for safe RL-based energy management systems. *Energy Reports*, *13*, 1888–1909.

Zhang, Z., Zhang, D., & Qiu, R. C. (2019). Deep reinforcement learning for power system applications: An overview. *CSEE Journal of Power and Energy Systems*, *6*(1), 213–225.

Zhao, J., Feng, X., Pang, Q., Fowler, M., Lian, Y., Ouyang, M., & Burke, A. F. (2024). Battery safety: Machine learning-based prognostics. *Progress in Energy and Combustion Science*, *102*, Article 101142.