D^{3}PGSA: A Dual-value Deep Deterministic Policy Gradient based on Sample Augmentation for Dynamic Pricing of Electric Vehicle Charging Station

Ran Tian, Bo Wang, Minmin Jiang

***Abstract*—With the rapid growth of electric vehicles (EVs), the demand for charging infrastructure has surged. Effective pricing strategies for electric vehicle charging stations (EVCSs) are critical for ensuring sustainable service delivery, promoting EVs adoption, and facilitating a shift in energy usage. However, existing dynamic pricing approaches suffer from several limitations, such as the inefficiency of traditional algorithms at scale and the high data requirements and instability of reinforcement learning (RL) methods. In this paper, we propose a Sample Augmentation-Based Dual-Value Deep Deterministic Policy Gradient (D^{3}PGSA) algorithm to address the dynamic pricing problem for a single EVCS. The proposed method provides operators with a flexible pricing model tailored to dynamic market conditions, thereby enhancing both revenue and operational efficiency. Specifically, we first introduce an experience generation model to alleviate the issue of limited training samples. Next, we incorporate a feature generation module and a cluster-balanced (CB) experience replay mechanism to improve generalization to unseen states and enhance decision quality. Finally, a dual-value network is integrated into the traditional framework to reduce error accumulation and improve learning stability. Extensive experiments conducted on two real-world datasets and four simulated datasets demonstrate that D^{3}PGSA can dynamically adjust prices in response to market demand and significantly outperforms other methods in terms of revenue, solution time, and pricing accuracy.**

***Index Terms*—Electric vehicle charging stations, Dynamic pricing strategy, Clustering algorithm, Sample augmentation.**

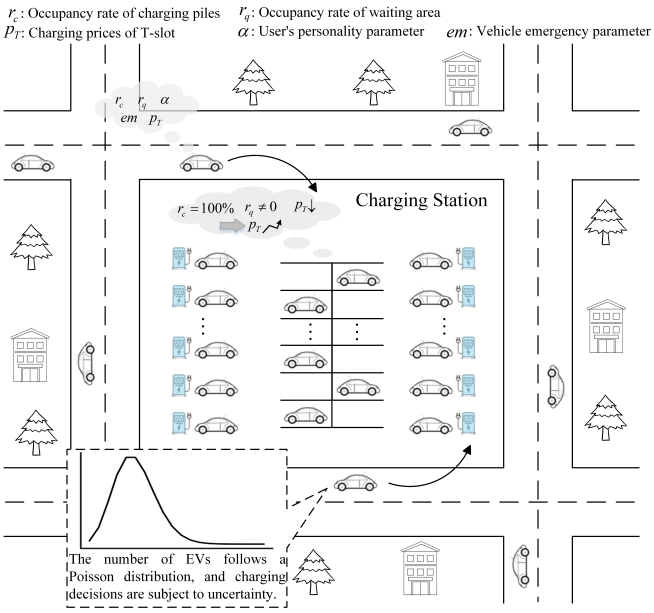

Fig. 1. Charging stations adjust prices based on real-time charging demand, while EVs make charging decisions considering both their internal states and the charging station environment

I. Introduction

ROAD traffic emissions are among the primary contributors to air quality degradation in densely populated urban areas. Promoting the adoption of electric vehicles (EVs) as a replacement for conventional fuel-powered vehicles offers an effective approach to reducing dependence on fossil fuels and mitigating greenhouse gas emissions [1]. However, as the penetration of EVs increases, issues such as insufficient and unevenly distributed electric vehicle charging stations (EVCSs) have become more pronounced, imposing additional stress on the power grid and potentially compromising grid stability. Accordingly, it is critical to establish a well-developed infrastructure that enhances clean energy supply and alleviates peak load conditions [2]. Nevertheless, expanding the number of EVCSs entails substantial capital investment and operational costs. Therefore, implementing effective pricing strategies to regulate charging demand while maintaining a balance between the interests of service providers and users is essential [3]. An appropriate pricing mechanism should dynamically adjust charging prices based on demand, ensuring the stable operation of EVCSs and preventing congestion or resource underutilization, as illustrated in Fig.1.

The pricing strategy of EVCSs is designed to incentivize users to shift their charging behaviors by dynamically adjusting electricity prices, thereby maximizing station revenue. EVCS pricing involves multiple interrelated and complex factors and impacts a wide range of stakeholders, including EV users, energy suppliers, infrastructure operators, and regulatory authorities. A well-designed pricing strategy is critical to achieving a sustainable electric mobility ecosystem. On one

Ran Tian is currently with the School of Computer Science and Engineering, Northwest Normal University, Lanzhou 730000, China (email: tianran@nwnu.edu.cn).

Bo Wang and Minmin Jiang are students at the School of Computer Science and Engineering, Northwest Normal University, Lanzhou 730000, China (email: 2023222196@nwnu.edu.cn, 2023222089@nwnu.edu.cn).

hand, pricing must be fair and transparent to enhance the attractiveness of EVs to end users. On the other hand, operators must balance revenue maximization, return on investment, and the long-term sustainability of the charging network.

In recent years, the academic community has proposed a variety of pricing strategies to address these challenges. Although existing methods have demonstrated a certain degree of effectiveness, several critical issues remain unresolved:

Challenge 1: In real-time or near real-time dynamic pricing scenarios, the central challenge lies in efficiently computing the optimal pricing scheme within a limited time. As the scale of charging demand grows, the computational complexity and solving time of pricing models increase accordingly. Traditional methods often fail to deliver feasible solutions within the required timeframe. This time sensitivity necessitates the development of efficient algorithms that ensure both speed and accuracy in pricing decisions, thereby enhancing the responsiveness and effectiveness of pricing strategies.

Challenge 2: In practical EV charging scenarios, vehicle arrivals and user charging decisions are inherently uncertain, making it difficult to accurately predict user behavior and demand. Moreover, charging station operators frequently encounter limited operational data when designing pricing strategies, particularly in real-world settings where data available for training optimization models is sparse. This lack of data leads to insufficient learning of demand patterns, ultimately hindering the generalization capability of pricing models and compromising the effectiveness and profitability of implemented strategies.

To address these challenges, we propose a deep reinforcement learning (DRL)-based dynamic pricing model that comprehensively incorporates operational constraints and demand fluctuations. By leveraging techniques such as data augmentation and cluster-based sampling, the proposed model enhances training stability and adaptability. It enables rapid profit-maximizing pricing decisions within the bounds of user affordability. In summary, the main contributions of this study are as follows:

1) We develop an improved reinforcement learning framework to enhance the computational efficiency of pricing strategies in dynamic environments. The proposed algorithm dynamically adjusts pricing decisions based on market demand and station-specific operational states, offering a practical solution for single-station revenue maximization.

2) To address the stochastic nature of demand and user behavior, as well as the problem of limited data availability, we utilize two real-world datasets and construct four simulation datasets based on Poisson distribution. A user decision-making model is designed to simulate behavioral uncertainty. To further alleviate data scarcity, we develop an experience generation model and a feature generation module to enhance sample diversity. Additionally, a cluster-based experience replay mechanism is introduced to improve sample utilization efficiency. The algorithm also incorporates a dual-value network architecture, which enhances pricing performance and reliability, mitigates error accumulation, and improves model adaptability and robustness in complex environments.

3) Extensive experiments conducted on datasets of varying scales demonstrate the effectiveness of the proposed algorithm in the context of single station dynamic pricing. Comparative evaluations against baseline methods, including heuristic and RL-based approaches from existing literature, show significant advantages in terms of revenue optimization and policy stability.

The remainder of this paper is organized as follows. Section II reviews the state-of-the-art in dynamic pricing for electric vehicle charging stations. Section III formulates the problem and presents the modeling framework. Section IV provides a detailed description of the proposed algorithm. Section V validates the effectiveness of the approach through extensive experiments. Finally, Section VI concludes the paper.

II. Related Work

In recent years, the rise of smart cities and intelligent transportation systems (ITS) has driven the widespread adoption of self-optimizing technologies in areas such as traffic scheduling, route planning, and energy management. Leveraging vehicular networks, artificial intelligence(AI), and hybrid learning mechanisms, systems like fusion-based intelligent traffic congestion control system for VNs(FITCCS-VN) use machine learning to provide real-time traffic guidance and routing optimization [4], while soft computing approaches enhance mobility prediction for autonomous vehicles in complex urban settings [5]. For data governance, blockchain has improved transparency and trust in vehicular networks [6], and data-aware resource allocation has reduced processing costs in cloud computing systems [7]. In smart buildings, federated learning (FL) and explainable AI (XAI) have been combined to strengthen security and user trust in distributed energy management [8]. Collectively, these advances highlight the critical role of data-driven optimization in urban mobility and provide methodological insights and practical support for dynamic pricing strategies in electric vehicle charging.

A. Dynamic Pricing for EVCSs

To mitigate the adverse impact of peak EV charging demand on grid stability and power safety, various operational strategies have been proposed in the literature. These include a trilevel dynamic pricing model for identifying optimal policies [9], a bilevel service balancing approach based on dynamic pricing [10], an integrated dynamic pricing and reservation-based system for autonomous parking and charging [11], and a two-stage framework for jointly optimizing station siting and pricing decisions [12]. Tan et al. [13] established a dynamic pricing mechanism that balances electricity prices with travel demand, while Lai et al. [14] considered the dual impact of charging prices on both the power and transportation networks. Additionally, Jadoun et al. [15] applied a meta-heuristic firehawk optimization technique to improve collective utility, reduce the peak-to-valley ratio of station loads, and lower electricity purchase costs for EV owners. Collectively, these studies demonstrate the effectiveness of dynamic pricing in alleviating grid stress and maintaining operational stability.

In addition, several studies have focused on enhancing user satisfaction and operator profitability by applying dynamic

TABLE I
RELATED WORK CLASSIFICATION

Reference	Algorithm Type	User Behavior Modeling	Number of Datasets		Multi-Scenario	Real-World Dataset	Statistical Significance	Mitigation of Operational Data Scarcity
			Single	Multiple				
Dai [16]	GT (Stackelberg)	✓	✓					
Moghaddam [39]	DRL	✓	✓		✓			
Jin [34]	DRL (DDPG)	✓	✓					
Cedillo [10]	TO	✓	✓		✓			
Lee [35]	DRL	✓	✓		✓			
Abdalrahman [37]	DRL (TD3)	✓	✓		✓			
Zhao [25]	DRL	✓		✓		✓		
Cui [32]	DRL (DDPG)	✓	✓		✓			
Liu [28]	GT (Evolutionary)	✓	✓			✓		
Kazemtarghi [3]	TO	✓	✓		✓	✓		
Tan [13]	MHA (DE)	✓	✓		✓			
Zhang [1]	LGDG / LGEG	✓	✓			✓		
Hou [36]	DRL (MARMD)	✓	✓		✓			
Bae [38]	DRL (PeDP)	✓		✓	✓			
Chakraborty [30]	HA (MOPSO)			✓	✓			
Lai [14]	GT (Stackelberg)	✓		✓	✓	✓		
Jadoun [15]	MHA (FHO)		✓		✓	✓	✓	
D^{3}PGSA	**DRL**	✓		✓	✓	✓	✓	✓

GT: Game Theory; DRL: Deep Reinforcement Learning; TO: Traditional Optimization; MHA: Meta-Heuristic Algorithm; HA: Heuristic Algorithm.

pricing strategies to reduce waiting time and charging costs [16], [17]. Gong et al. [18] introduced a dynamic spike pricing (DSP) scheme to minimize user costs while ensuring the stable operation of the power distribution system. Zhang et al. [19] incorporated user preference analysis into the design of time-of-use pricing to improve user satisfaction. Other approaches include a tri-level siting model to maximize station profitability [20], a revenue optimization strategy by Aljafari et al. [21] that also minimizes wait times and electricity costs for EV users, and a storage-integrated charging station proposed by Lin et al. [22] to reduce operational costs. To address information asymmetry, Lu et al. [23] proposed a dynamic pricing menu to incentivize users to reveal private information and thereby improve pricing efficiency.

Overall, most existing studies on EVCSs dynamic pricing focus on system-level strategies involving multiple charging stations. However, the specific challenges and practical demands of individual charging stations remain underexplored. Research on single-station dynamic pricing can better support localized operational adaptation and provide theoretical foundations for developing region-specific and user-tailored pricing mechanisms.

B. Solution Approaches

The aforementioned studies offer valuable insights into dynamic pricing for EV charging stations, yet traditional algorithmic approaches remain underexplored. Among the most commonly used methods are heuristic algorithms [24] and DRL techniques [25]. For example, Yang et al. [26] applied Monte Carlo simulations to model EV charging demand and proposed a bilevel pricing model; Chen et al. [27] introduced a Gaussian-Seidel algorithm with inertia weights; and Liu et al. [28] adopted evolutionary game theory for pricing scheme optimization. While effective, these approaches often struggle with scalability and efficiency in large, dynamic environments, highlighting the need for more adaptive algorithms.

Heuristic algorithms, inspired by natural processes, are valued for their flexibility and robustness in solving nonlinear problems. Zhang et al. [29] used Non-dominated Sorting Genetic Algorithm II(NSGAII) to optimize a hierarchical time-of-use pricing strategy, Chakraborty et al. [30] combined autoregressive integrated moving average(ARIMA) with multi-objective particle swarm optimization(MOPSO) for price forecasting, and the recently proposed White Shark Optimizer (WSO) [31] enriches bio-inspired optimization techniques. Despite their potential in dynamic pricing, heuristic algorithms are often time-consuming when applied to large solution spaces or multimodal objective functions and are prone to being trapped in local optima.

DRL offers strong adaptability in dynamic settings by learning optimal policies through interaction without requiring precise modeling. DRL further enhances this capability by employing neural networks to manage complex, high-dimensional data. In recent works, Cui et al. [32] integrated traffic flow prediction with Markov decision process(MDP) and proposed a deep deterministic policy gradient (DDPG)-based dynamic pricing strategy to maximize charging station revenue. Fraija et al. [33] applied RL to optimize revenue in price-driven demand response, and Jin et al. [34] applied DDPG to balance demand response and user satisfaction. Lee et al. [35] developed a privacy-preserving distributed DRL framework to optimize revenues across multiple charging stations. Hou et al. [36] proposed a multi-agent deep deterministic policy gradient(MADDPG)-based dynamic pricing approach to maximize long-term revenue via coordinated multi-station strategies. Abdalrahman et al. [37] utilized the twin delayed deep deterministic policy gradient(TD3) to optimize dynamic pricing. Bae et al. [38] proposed a Q-learning-based personalized dynamic pricing policy(PeDP) framework for rapid EVCS revenue maximization. Moghaddam et al. [39] integrated adaptive heuristic critics with recursive least squares to develop an online reinforcement learning model for grid load balancing and revenue enhancement. Yang et al. [40] proposed adaptive model-based safe deep reinforcement learning(AMSDRL) based on constrained Markov Decision Process(CMDP) for safe learning, while the recently introduced Distributional Soft Actor-Critic(DSAC) algorithm [41]

also shows strong performance in complex pricing tasks.

A selection of representative studies has been summarized in Table I, where their respective research objectives and methodologies are categorized and compared.

In summary, current research on EVCS dynamic pricing frequently employs the DDPG and its variants. However, the application of DDPG to charging station pricing still faces several limitations. First, DDPG requires a large volume of samples, but data collection is often time-consuming and costly, limiting its practical usability. Second, DDPG tends to suffer from overestimation of state-action values, resulting in overly optimistic policies, accumulated errors, and reduced stability in training and pricing decisions. Moreover, traditional DDPG employs random sampling from the experience pool without considering sample importance, lowering training efficiency and increasing the risk of overfitting, thereby affecting policy adaptability. These issues are particularly pronounced in the complex and uncertain environment of dynamic pricing for EVCS.

III. System Model

This section presents the problem definition and modeling framework, followed by the formulation of the dynamic pricing task as a MDP within the reinforcement learning.

A. Optimization Objective

In the context of dynamic pricing for charging stations, the core objective is to maximize the charging station owner's revenue while ensuring pricing remains within a range acceptable to users. Therefore, this paper sets the revenue function of the charging station as the optimization objective:

$$r = \max \sum_{t=1}^{24} \sum_{i=1}^{n} p_{t-1} \times c_i - \mathrm{C}, i \in V_{t-1} \tag{1}$$

where r represents the total revenue of the charging station, p_{t-1} denotes the station price at time $t-1$, c_i represents the charging amount of the i-th vehicle, and C represents the total operating cost of the charging station, V_{t-1} is the set of all EVs at each time interval.

In this study, we construct a detailed model to simulate the usage patterns of EVs at a charging station throughout the day, taking into account multiple factors, including battery state, charging demand, charging duration, user price sensitivity, and individual preference heterogeneity. Considering the temporal volatility of electricity markets, charging station operators typically engage in pre-purchase agreements with grid providers to secure stable electricity prices. This approach not only optimizes operational costs but also enhances market responsiveness. Accordingly, in our revenue modeling, we base the cost analysis on the pre-agreed electricity price, thereby excluding the influence of short-term price fluctuations.

The pricing strategy is executed over 24-time steps representing hourly intervals within a day. To improve the system's responsiveness to dynamic user behavior, we model vehicle arrivals at a finer granularity of 15-minute intervals. This enables more frequent updates to user states and timely adjustments to pricing strategies. Define bs_i, ct_i and em_i as the EV's battery status, charging duration, and urgency, respectively, where the battery status is the ratio of remaining charge to the total capacity. The calculation of charging amount c_i is given by (2):

$$c_i = ct_i \times cp, i = 1, 2, ..., n \tag{2}$$

where cp represents the charging power of the charging station.

The urgency of the vehicle em_i is calculated based on the current battery state:

$$em_i = e^{-2 \times bs_i}, i = 1, 2, ..., n \tag{3}$$

Assuming that users are relatively sensitive to charging prices, the current price will significantly impact their decisions. The user decision-making model includes three key parameters: w_q, w_p and α_i. Here, w_q represents the user's weighting of the waiting time in the charging station queue, w_p indicates the user's weighting of the current price, and α_i is used to simulate individual differences in users' sensitivity to price. The calculations for w_q and w_p are as (4):

$$w_q = 0.5 + \beta \times em_i, w_q + w_p = 1 \tag{4}$$

where β is a parameter that adjusts the weight of urgency.

The final decision is determined based on an aggregate score calculated from these parameters. When the score reaches the decision threshold ρ, the user opts to charge. The decision function is shown in (5):

$$d_i = \begin{cases} 1, & (w_q \times (1 - l_q) + w_p \times (p_{\max} - p_t)) \times \alpha_i \geq \rho \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where l_q represents the occupancy rate of the waiting area at the current EVCS, and $p_{\max}$ represents the highest charging price acceptable to the vehicle users.

The charging station serves vehicles in the waiting area and set V_t on a first-come, first-served basis. In this paper, the station lacks information on future EV arrivals and electricity prices, making it unable to predict demand in upcoming periods. Thus, these events are treated as non-causal for the station.

B. Markov Decision Process

The dynamic pricing problem of the EVCS corresponds to a MDP, which can be constructed as a four-tuple $\langle S, A, r, \eta \rangle$.

S: State space, including current electricity price, occupancy rates, and time step.

A: Action set. At state s_t, action a_t adjusts price (positive to increase, negative to decrease).

r:Reward function, representing revenue from charging at each step.

η: Random noise, decreasing with training iterations.

In our environment, the state at each time step is denoted as the set $s_t = \left\{\left(p_t, r_q^t, r_c^t, t\right)\right\}$, where r_q^t represents the occupancy rate of the waiting area at time step t, and r_c^t represents the usage rate of the charging stalls at time step t. The environment accepts the action a_t taken by the agent based on s_t at each time step to adjust the charging price p_t. These actions will affect the state evolution in future time

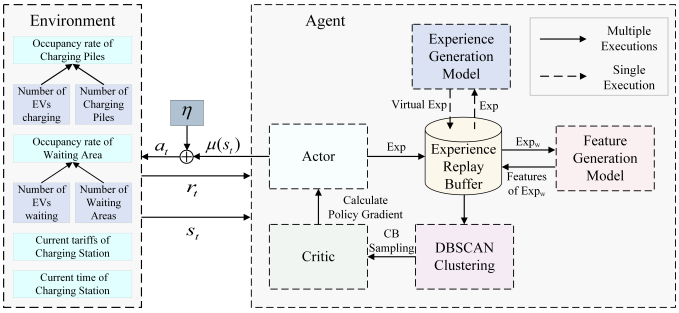

Fig. 2. Structure of the D^{3}PGSA

steps. Furthermore, we define the state transition method of the environment $s_{t+1} = F(s_t, a_t)$, where s_{t+1} is the system state at the next time step, and F is the state transition function, specifically defined as (6):

$$\begin{cases} n_c^{t+1} = \sum_{i \in V_t} d_i, r_c^t < 1 \\ n_q^{t+1} = \sum_{i \in V_t} d_i, r_c^t = 1 \text{ and } r_q^t < 1 \end{cases} \tag{6}$$

where n_c^{t+1} represents the number of charging stalls in use, and n_q^{t+1} represents the number of EVs in the waiting area.

The information in s_{t+1} is updated according to (7):

$$\begin{cases} r_c^{t+1} = \frac{n_c^{t+1}}{N_c} \\ r_q^{t+1} = \frac{n_q^{t+1}}{N_q} \end{cases} \tag{7}$$

where N_c is the total number of charging stalls and N_q is the total number of parking spots in the waiting area.

Following the formulation of the problem as a MDP, a DRL algorithm can be employed to derive the optimal pricing strategy for the charging station. This approach enables the station to more effectively respond to fluctuating EV charging demand, thereby improving both operational efficiency and revenue generation.

IV. METHODOLOGY

In this section, we propose a novel algorithm named D^{3}PGSA (dual-value Deep Deterministic Policy Gradient based on Sample Augmentation), which integrates a U-Net-based experience generation module, a SimSiam-based feature augmentation module, and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) clustering within a dual-value DDPG framework.

A. D^{3}PGSA Model

The D^{3}PGSA inherits the Actor-Critic framework,the structure is shown in Fig.2. The framework consists of a charging station environment and an agent. The environment defines the state space, action space, and state transition dynamics, and provides feedback to the agent. As the decision-maker, the agent interacts with the environment to learn a policy that maximizes cumulative rewards.

In D^{3}PGSA, a U-Net-based experience generation module is first employed to address the data efficiency limitations of DDPG by generating high-quality training samples. The objective of this module is defined as (8):

$$\min_{f_\varpi} \mathbb{E}_{Exp, \Gamma_\varpi, \Gamma_{\varpi^-}} [\mathcal{L}(f_\varpi(\Gamma_\varpi(Exp)), f_{\varpi^-}(\Gamma_{\varpi^-}(Exp)))] \tag{8}$$

where Exp represents the real experience samples collected from environment interaction. $\Gamma_\varpi(Exp), \Gamma_{\varpi^-}(Exp)$ represent two different transformed versions of the same experience sample, and f_ϖ, f_{ϖ^-} are two neural networks with different parameters.

Subsequently, a SimSiam-based feature generation module is employed to further enhance the representation of experience samples, thereby improving training efficiency. Its optimization objective is defined as (9):

$$\min_{f_\omega, g} \mathbb{E}_{Exp_w^i, \Gamma_\omega, \Gamma'_\omega} [\mathcal{L}(g(f_\omega(\Gamma_\omega(Exp_w^i))), f_\omega(\Gamma'_\omega(Exp_w^i)))] \tag{9}$$

where Exp_w represents the entire experience set, including both real and virtual samples, $Exp_w^i \in Exp_w$. $\Gamma_\omega(Exp_w^i), \Gamma'_\omega(Exp_w^i)$ refer to two perturbed versions of the Exp_w^i, f_ω is the encoder network, and g is the predictor network.

To improve the traditional experience replay mechanism, we adopt a DBSCAN-based feature representation and cluster-balanced (CB) sampling method to ensure sufficient diversity in each sampling batch. Before each interaction, the agent first initializes the environment and transition dictionary, then sets the environment feedback state s_t as input $Input_t = (p_t, r_q^t, r_c^t, t)$ to the model, the pricing action a_t is derived from the policy network and perturbed by Gaussian noise:

$$a_t = \mu_\theta(Input_t) + \eta \tag{10}$$

A delayed reward mechanism is adopted, where the reward r_t for step t is defined as the revenue obtained after $t-1$ steps. This guides the agent toward long-term returns, avoids myopic optimization, and improves global performance and real-world applicability. The reward r_t is calculated as (11):

$$r_t = \sum_{i=1}^{n} p_{t-1} \times c_i, i \in V_{t-1} \tag{11}$$

To prevent the agent from adopting overly aggressive pricing strategies in early stages, the following lower-bound constraint is applied:

$$p_t = \max(p_{t-1} + a_t, 0.1) \tag{12}$$

B. Experience Generation Model Based on U-Net

This module is designed to address the problem of limited training data. At its core lies a structured U-Net neural network, with the overall architecture illustrated in Fig.3. For training this model, we adopt a Consistency Training (CT) strategy. Firstly, Gaussian noise is added to the input experience sample Exp, which is denoted as x in the following formulation:

$$\begin{cases} x_{tp_n} = x + tp_n \times z \\ x_{tp_{n+1}} = x + tp_{n+1} \times z \end{cases}, z \sim \mathcal{N}(0, I) \tag{13}$$

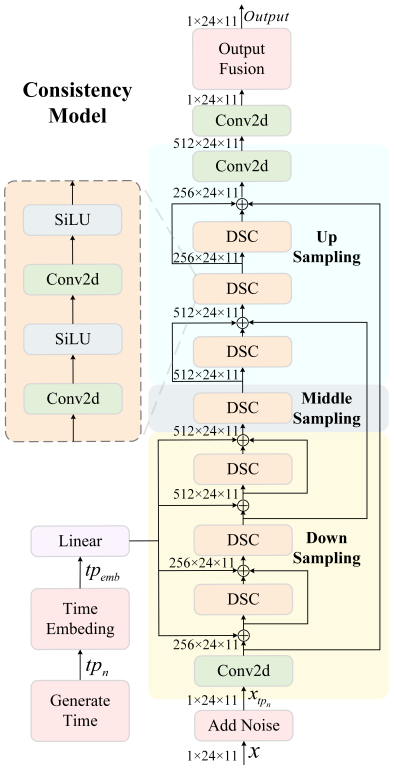

Fig. 3. Structure of experience generation model

Eq.(13) simulates perturbations under temporal uncertainty, allowing the model to learn robustness against variations at different time points. This enhances the model's temporal consistency and generalization capability. Where tp_n and tp_{n+1} represent earlier and later time points, respectively. The tp_i are generated according to (14):

$$tp_i = (\xi^{\frac{1}{\sigma}} + \frac{i}{\mathrm{L}-1}(\mathrm{T}^{\frac{1}{\sigma}} - \xi^{\frac{1}{\sigma}}))^{\sigma}, 0 \leq i < \mathrm{L} \tag{14}$$

where ξ is used for nonlinear transformation of the time scale, σ is a stretching factor to control sample centrality, L denotes the total number of dispersed time points, and T is the maximum temporal spread. Subsequently, temporal encoding is applied to tp_n as (15):

$$tp_{emb} = \mathrm{concat}(\sin(tp_n \cdot freq), \cos(tp_n \cdot freq)) \tag{15}$$

Compared to linear or sparse representations, this encoding scheme more effectively preserves temporal order and periodic structures. Where $freq$ denotes a set of distributed frequencies in log-space, which transforms tp_n into sine and cosine components, thereby embedding time information naturally into the input data.

Finally, both tp_{emb} and x_{tp_n} are fed into the model's feature extraction module, which consists of multiple layers of convolution and nonlinear activation based on depthwise separable convolution (DSC). The activation function used is SiLU (Sigmoid Linear Unit), which provides smoother gradient flows. Combined with the lightweight DSC module, this helps extract localized features with temporal continuity and positional stability. The extraction process is defined as:

$$x_{tp_n}{}' = SiLU(Conv(x_{tp_n})) \tag{16}$$

$$x_{tp_n}{}'' = SiLU(Conv(x_{tp_n}{}')) \tag{17}$$

The first layer of the feature extractor serves as a downsampling module, which leverages a 2D convolutional network followed by three DSC blocks to preserve critical data features while filtering out redundant information. Temporal embedding is performed at each layer, and residual connections are introduced every other layer.

$$x_{tp_n}{}'' = x_{tp_n}{}'' + \mathrm{FC}(tp_{emb}) \tag{18}$$

$$x_{tp_n} = x_{tp_n}{}'' + x_{tp_n} \tag{19}$$

In Eq.(18), the temporal embedding is projected via a fully connected layer to match the spatial feature dimension and is subsequently added to the spatial representation. This operation enables temporal context to be injected into each layer's spatial features, thereby enhancing the model's temporal awareness. As the network depth increases, these features may become difficult to reconstruct during the subsequent upsampling process. Therefore, in the downsampling layers, the model skip-connects certain key features. This residual connection structure is designed to prevent critical input features from being overly compressed or filtered out in deep networks, thereby aiding the decoder stage in restoring information and maintaining representational consistency.

The second layer is the intermediate layer, which applies DSC to extract the core semantic information within the model. The third layer functions as the upsampling module, which, similar to the downsampling layer, merges upsampled features with current-layer representations to ensure that hierarchical abstract information is captured throughout the network.

The final model output is refined by computing the weights $c_{s_{tp_n}}$ and $c_{o_{tp_n}}$, which are used to balance the features from the original input and the processed representations. The calculation is as (20)(21):

$$c_{s_{tp_n}} = \frac{0.25}{{tp_n}^2 + 0.25}, c_{o_{tp_n}} = \frac{0.25 \cdot tp_i}{(tp_n + \zeta)^2 + 0.25} \tag{20}$$

$$output = c_{s_{tp_n}} \cdot x + c_{o_{tp_n}} \cdot x_{tp_n} \tag{21}$$

where the coefficient 0.25 is an empirically validated constant to ensure training stability. The model parameters are optimized by minimizing the following loss function:

$$\mathcal{L}(\varpi, \varpi^-) = \lambda(tp_n)\mathrm{Mse}(output, f_{\varpi^-}(x_{tp_{n+1}}, tp_{n+1})) \tag{22}$$

where $\lambda(tp_n)$ is a time-based weighting factor, and ϖ^- denotes the exponentially moving average of model parameters used for stable training.

To reduce sampling noise in the generation of virtual experience samples, we employ a 5-step sampling strategy, which is defined as follows. This process is iterated five times to obtain a stable and reliable sample output.

$$\begin{cases} x_{\varsigma_n} \leftarrow x + \sqrt{\varsigma_n^2 - \ell^2} \times z \\ output \leftarrow f_{\varpi}(x_{\varsigma_n}, \varsigma_n) \end{cases}, z \sim \mathcal{N}(0, I) \tag{23}$$

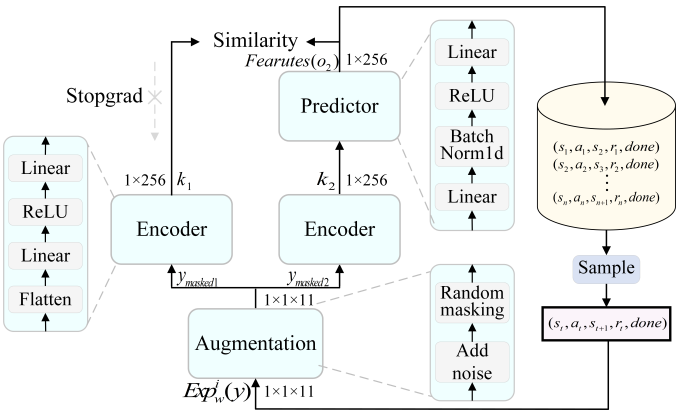

Fig. 4. Structure of feature generation model

C. Feature Generation Model Based on SimSiam

This module is designed to generate effective feature representations for the subsequent clustering operations. The overall architecture is illustrated in Fig.4. During training of this module, the input sample Exp_w^i is first augmented into two perturbed versions, denoted as y. The augmentation process is defined as:

$$y_n = y + \partial, \partial \sim \mathcal{N}(0, {l_{noise}}^2 \cdot I) \tag{24}$$

$$y_m = y_n \odot (1 - M), \mathbf{M} \sim Bernoulli(\mathrm{P}_{mask}) \tag{25}$$

where ∂ simulates continuous noise perturbations to the encoder input, while the masking matrix $\mathbf{M}$, sampled from a Bernoulli distribution, enables stochastic information suppression. This helps the network capture more robust and stable feature representations.

The two masked versions y_{m1} and y_{m2} are fed into the same encoder to extract feature vectors k_1 and k_2. A prediction head then maps k_1 to o_1 to predict k_2, and maps k_2 to o_2 to predict k_1. The symmetric loss (negative cosine similarity) is computed as:

$$\mathcal{L}(\omega) = \frac{1}{2}\mathcal{D}(o_1, \mathrm{stopg}(k_2)) + \frac{1}{2}\mathcal{D}(o_2, \mathrm{stopg}(k_1)) \tag{26}$$

where stopg prevents mutual learning between the prediction heads, thereby avoiding representational collapse and enabling stable self-supervised training. The function $\mathcal{D}$ is defined as:

$$D(a, b) = \frac{a \cdot b}{\|a\|\|b\|} \tag{27}$$

After saving the trained model parameters, the agent uses the model f_s to extract feature representations o for each sample y:

$$\{o_1, o_2, ..., o_w\} = \{f_s(y_1), f_s(y_2), ..., f_s(y_w)\} \tag{28}$$

All extracted features are denoted as the set $O = \{o_1, o_2, \quad ..., o_w\}$, where each feature o_i is indexed and stored back into the experience buffer for subsequent task retrieval.

D. Experience Replay Mechanism Based on DBSCAN

To enhance sample utilization and training performance, the agent performs clustering on the augmented experience samples using the DBSCAN algorithm based on the extracted feature set O. DBSCAN is chosen due to its ability to identify clusters without requiring the number of clusters to be predefined, making it well-suited for scenarios with stochastic fluctuations in charging demand. Furthermore, DBSCAN can effectively handle complex and irregular data distributions, which is particularly important in dynamic EV charging environments. The specific clustering process is described as follows. For each experience sample feature $o \in O$, its ε-neighborhood is defined as (29):

$$Y_\varepsilon(o) = \{j \in O \mid dist(o, j) \le \varepsilon\} \tag{29}$$

where $dist(o, j)$ represents the distance between o and j.

If the ε-neighborhood of point o contains at least $MinPts$ points, o is marked as a core point, and all points in its neighborhood are assigned to the same cluster. If its neighborhood overlaps with that of other core points, the clusters are merged.

If o is not a core point but lies within the neighborhood of a core point, it is labeled as a border point by (30).If o is neither a core point nor within the neighborhood of any core point, it is labeled as noise by (31):

$$\left|Y_\varepsilon(o)\right| < MinPts \text{ and } \exists j \in O | o \in Y_\varepsilon(j) \tag{30}$$

$$\left|Y_\varepsilon(o)\right| < MinPts \text{ and } \nexists j \in O | o \in Y_\varepsilon(j) \tag{31}$$

This process is repeated starting from unassigned points until all points are categorized as either belonging to a cluster or labeled as noise.

Finally, we introduce CB sampling to ensure that a suitable number of samples are selected from each cluster during each replay. The number of samples H to be drawn per cluster is computed as:

$$\mathrm{H} = \max(1, \left\lfloor \frac{\mathrm{B}}{\mathrm{K}} \right\rfloor) \tag{32}$$

where B is the total number of samples to be drawn, and K is the total number of clusters after DBSCAN. Then, H samples are randomly selected from each cluster until the required number B is reached.

E. DDPG Based on Dual-value Network

To avoid Q-value overestimation, we employ two independent value networks and use the minimum of their outputs as the target for updates. Its structure is shown in Fig.5.

After interacting with the environment, the agent samples experience tuples for training. The action value a_{t+1} is estimated using (33):

$$\begin{cases} Q_1(s_{t+1}, a_{t+1}) = Q_{\vartheta_1^-}(s_{t+1}, \mu_{\theta^-}(s_{t+1})) \\ Q_2(s_{t+1}, a_{t+1}) = Q_{\vartheta_2^-}(s_{t+1}, \mu_{\theta^-}(s_{t+1})) \end{cases} \tag{33}$$

where $Q_{\vartheta_1^-}, Q_{\vartheta_2^-}$ are target networks that are updated with a delay to stabilize Temporal Difference (TD) error estimation. Then, the TD target is computed using (34):

$$tg_t = r_t + \gamma \min(Q_1(s_{t+1}, a_{t+1}), Q_2(s_{t+1}, a_{t+1})) \tag{34}$$

As shown in (33) and (34), we adopt two Q-networks to estimate the action value for the next state and use the smaller value as the target Q value. This design follows the Double Q-learning principle, which helps mitigate the overestimation

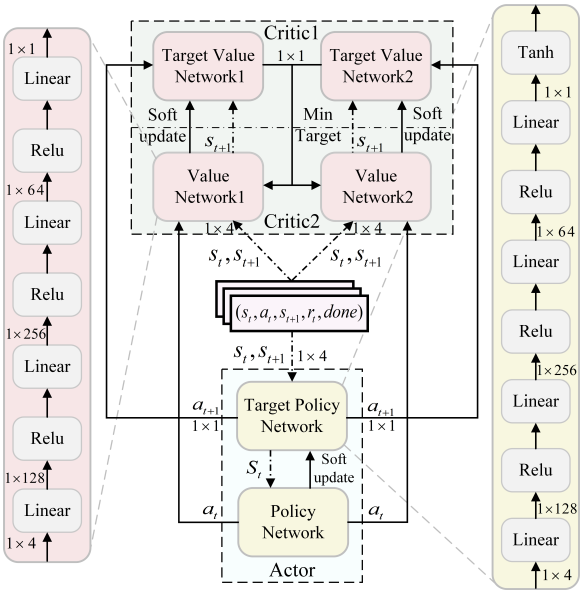

Fig. 5. Structure of dual-value network

bias by minimizing the Q estimate, thereby reducing cumulative Q estimation errors during long-term training. Finally, the loss functions for the two Q-networks are computed as:

$$\begin{cases} \mathcal{L}(\vartheta_1) = \dfrac{1}{\mathrm{B}} \displaystyle\sum_{t=1}^{\mathrm{B}} \left(Q_{\vartheta_1}(s_t, a_t) - tg_t\right)^2 \\ \mathcal{L}(\vartheta_2) = \dfrac{1}{\mathrm{B}} \displaystyle\sum_{t=1}^{\mathrm{B}} \left(Q_{\vartheta_2}(s_t, a_t) - tg_t\right)^2 \end{cases} \tag{35}$$

The gradients of the two Q-networks are independently calculated based on their respective TD errors to improve training stability. The gradient formulas are given in (36):

$$\begin{cases} \nabla_{\vartheta_1}\mathcal{L}(\vartheta_1) = (Q_{\vartheta_1}(s_t, a_t) - tg_t)\nabla_{\vartheta_1} Q_{\vartheta_1}(s_t, a_t) \\ \nabla_{\vartheta_2}\mathcal{L}(\vartheta_2) = (Q_{\vartheta_2}(s_t, a_t) - tg_t)\nabla_{\vartheta_2} Q_{\vartheta_2}(s_t, a_t) \end{cases} \tag{36}$$

To maximize the expected reward, the loss function of the policy network is defined as:

$$\mathcal{L}(\theta) = -(Q_{\vartheta_{\min}}(s_t, \mu_\theta(s_t))) \tag{37}$$

where $Q_{\vartheta_{\min}}$ represents the critic that outputs the smaller value when computing the TD target. The policy gradient is approximated by (38):

$$\nabla_\theta \mathcal{L}(\theta) \approx \frac{1}{\mathrm{B}} \sum_{t=1}^{\mathrm{B}} \nabla_\theta \mu_\theta(s_t) \nabla_{a_t} Q_{\vartheta_{\min}}(s_t, a_t)|_{a_t = \mu_\theta(s_t)} \tag{38}$$

In addition, the target networks are updated using a soft update strategy as described in (39) and (40):

$$\begin{cases} \vartheta_1^- \leftarrow \tau\vartheta_1 + (1-\tau)\vartheta_1^- \\ \vartheta_2^- \leftarrow \tau\vartheta_2 + (1-\tau)\vartheta_2^- \end{cases} \tag{39}$$

$$\theta^- \leftarrow \tau\theta + (1-\tau)\theta^- \tag{40}$$

F. Training Algorithm

In D^{3}PGSA, we employ an experience generation network, a feature generation network, and CB sampling to augment training samples. The policy and value networks are trained using a dual-value DDPG algorithm. The policy network is responsible for selecting pricing actions, while the value networks evaluate the expected returns of these actions. By leveraging both components, the model is capable of learning more effective strategies in complex pricing environments.

Algorithm 1: D^{3}PGSA

Input: Current electricity price of the station, charger occupancy rate, waiting area occupancy rate, current environment time

1 Initialize Actor network μ_θ and two Critic networks Q_{ϑ_1}, Q_{ϑ_2} with random parameters θ, ϑ_1 and ϑ_2.
2 Clone corresponding target networks with $\theta^- \leftarrow \theta$, $\vartheta_1^- \leftarrow \vartheta_1$, $\vartheta_2^- \leftarrow \vartheta_2$, Initialize the target network μ_{θ^-}, $\mathrm{Q}_{\vartheta_1^-}$, $\mathrm{Q}_{\vartheta_2^-}$.
3 Initialize the replay buffer, Tag, and exploration noise variable η.
4 **for** $e = 1 \to \mathrm{E}$ **do**
5 Obtain initial environment state s_0.
6 Anneal Gaussian noise standard deviation σ.
7 **for** $t = 1 \to \mathrm{N}$ **do**
8 Select action using policy and noise $a_t = \mu_\theta(s_t) + \eta$.
9 Execute action a_t, receive reward r_t and next state s_{t+1}
10 Store Exp into experience replay buffer.
11 **if** *the Tag is true and the number of samples in the experience replay buffer reaches the threshold* **then**
12 Invoke the experience generation model using original samples Exp as input. Apply Gaussian noise, temporal encoding, down/up-sampling to produce augmented experience as (8)(23).
13 Train the feature generation model using all experience samples as (9)(26), set Tag is False.
14 **end**
15 Use CB Sampling to draw B samples from the clustered replay buffer.
16 Compute TD target using the target network for each sampled experience as (34).
17 Minimize TD loss and update both Critic networks as (36).
18 Compute the policy loss and update the Actor network as (37).
19 Update the target networks using soft update rules as(39)(40).
20 **end**
21 **end**

Output: Optimal dynamic pricing strategy for the next time interval at the EV charging station

As shown in Algorithm 1, in each iteration, the agent interacts with the environment by outputting an action through the policy network. After accumulating a sufficient number of steps, the experience generation model is trained at Line 12 to produce augmented samples. At Line 13, the feature generation model is trained to encode experience samples into feature representations. Subsequently, Line 15 performs clustering based on these features. Following this, the policy network (Actor) and value networks (Critics) are updated at Lines 17 and 18, respectively. Line 19 updates the corresponding target networks. This iterative process continuously improves the policy, thereby enhancing the agent's interaction quality and overall learning efficiency in the environment.

G. Complexity Analysis

The computational complexity of the U-Net-based feature generation model is primarily determined by the encoder-decoder structure of the convolutional network, which consists of multiple layers of convolution operations. Given an input feature map of size $H \times W$ and a generated sample size of

N_g, the overall complexity is $O(N_g HW)$. The SimSiam-based feature generation model has a complexity of $O(N_f HW)$. The policy network is typically composed of a multi-layer perceptual structure. Assuming the input state dimension is d_s, the number of hidden neurons is d_h, and the number of training samples is N_p, the complexity is $O(N_p d_s d_h)$. In this study, we adopt a dual-value architecture for value networks. Given N_v training samples, the computational complexity becomes $O(2N_v d_s d_h)$. Since both the policy and value networks use relatively shallow neural architectures, their overall computation overhead during reinforcement learning training remains low.

In addition, this study adopts a DBSCAN-based clustering experience replay mechanism. Assuming the replay buffer size is M, and the data dimensionality is d, the computational complexity of DBSCAN, mainly affected by distance calculation and neighborhood querying, is $O(M \log M)$.In summary, the overall computational complexity of D³PGSA can be expressed as $O(N_g HW + N_f HW + N_p d_s d_h + 2N_v d_s d_h + M \log M)$, where the U-Net and SimSiam modules contribute the most due to data size and network depth. The policy and value networks impose relatively low computational loads, though the dual-value design introduces some additional overhead. The complexity of experience replay increases with the size of the replay buffer.

V. Experiments and Analysis

In this section, we conduct a series of experiments to evaluate the performance of the proposed D^{3}PGSA algorithm. The analysis is organized into six aspects: datasets and evaluation metrics, experimental setup, comparative experiments and analysis, training process visualization, hyperparameter studies, and ablation studies.

A. Datasets and Evaluation Metrics

To comprehensively evaluate the performance of the D^{3}PGSA model under various real-world operational scenarios, we conduct experiments on six datasets, including two real-world datasets from operational charging stations and four synthetically generated virtual datasets. The real-world datasets ACN-Data and EV Charging Reports include charging start and end times, energy consumption, and charging power. After preprocessing, the data are directly fed into the model to ensure realistic applicability. The four virtual datasets simulate different operational conditions: LiteDemand simulates to small-scale demand, HeavyHub to large-scale high-density demand, RushPeak to significant peak-hour demand, and OffValley to valley-period dominated demand patterns. These datasets are generated using Poisson processes to reflect the stochastic nature of EV arrivals, while maintaining alignment with real-world datasets in terms of data structure and physical constraints, ensuring consistency and comparability across training and validation. Fig.6 illustrates the hourly EV arrival distributions across different datasets.

As shown in Fig.6, the number of vehicle arrivals across the six datasets exhibits noticeable differences. Among them, LiteDemand and HeavyHub respectively reflect lower and higher levels of charging demand, with relatively small fluctuations, both remaining close to the normal fluctuation range. RushPeak presents two distinct spikes occurring during the morning and evening peak hours, significantly exceeding the normal fluctuation range. In contrast, OffValley shows two corresponding valley periods with values notably below the normal fluctuation range. Both ACN-Data and EV Charging Reports, derived from real-world charging operation data, display overall trends somewhat similar to RushPeak, but the fluctuations in vehicle arrivals are more complex and irregular.

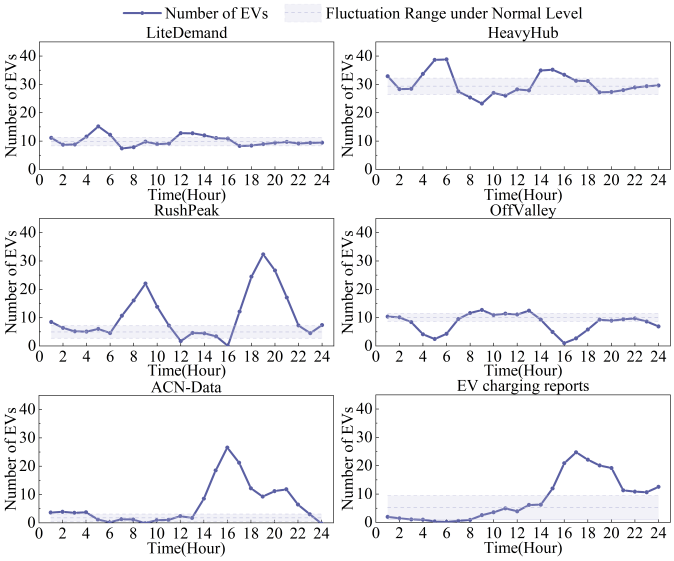

Fig. 6. Vehicle arrival curves of each dataset

This paper employs three metrics to evaluate algorithm performance, namely Average Revenue (AR), Worst-case Differential Gain (WDG), and Time-Weighted Reward Efficiency Index (TWREI). AR represents the average total revenue generated by the charging station across multiple runs, reflecting the algorithm's capability to maximize revenue. WDG quantifies the revenue improvement of each algorithm relative to the worst-performing method, thereby highlighting the relative advantage of each approach. TWREI integrates reward value and execution time to assess the trade-off between performance enhancement and computational efficiency.

$$\mathrm{AR} = \frac{1}{\mathrm{G}} \sum_{\mathrm{G}=1}^{\mathrm{G}} \sum_{t=1}^{24} \sum_{\mathrm{i}=1}^{n} r_t \tag{41}$$

$$\mathrm{WDG} = \frac{\mathrm{AR} - \mathrm{WR}}{\mathrm{WR}} \times 100\% \tag{42}$$

$$\mathrm{TWREI} = \frac{\mathrm{AR} - \mathrm{AR}_{\min}}{\mathrm{AR}_{\max} - \mathrm{AR}_{\min}} - \rho \log\left(\frac{\mathrm{st} - \mathrm{st}_{\min}}{\mathrm{st}_{\max} - \mathrm{st}_{\min}} + \delta\right) \tag{43}$$

where G is the number of training iterations, WR is the worst reward, ρ is the time weight, δ is a small value close to zero used to avoid numerical instability in logarithmic computations.

B. Experimental Setup

The experiments were conducted on a machine equipped with a 12 vCPU Intel(R) Xeon(R) Platinum 8255C CPU @

TABLE II
HYPERPARAMETER SETTINGS FOR COMPARISON ALGORITHMS

GWO	Alpha_pos=0, Alpha_score=∞, Wolves_num=30
WSO [31]	Pop_size=30, dim = 24, tau = 4.11
NSGAII [29]	Pop_size=30, generations=30, seq_len=24
DDPG [32]	Actor_lr=2e-3, Critic_lr=2e-3, Hidden_dim = 128, gamma=0.9, lmbda=0.9, eps=0.9
TD3 [37]	Actor_lr=2e-3, Critic_lr=2e-3, Alpha_lr = 3e-4, Hidden_dim = 128, gamma=0.99, tau=0.005, Target_entropy = -1
DSAC [41]	Actor_lr=2e-3, Critic_lr=2e-3, Hidden_dim = 128, gamma=0.98, tau=0.005, alpha = 0.2

2.50GHz, 40.0GB RAM, and an RTX 3080 GPU (10GB). The operating system was Ubuntu 20.04, with Python version 3.8, PyTorch version 2.0.0, and CUDA version 11.8.

Before conducting the dynamic pricing experiments for EV charging stations, we initialized the experimental parameters as follows. The discount factor γ and parameter β, Λ were set to 0.98, 1, and 0.3, respectively. The number of training iterations was set to 1000, and the soft update coefficient was set to 0.001. The experience buffer capacity was set to 40,000, and the number of sampled experiences per update was $\mathrm{B} = 128$. The initial standard deviation of Gaussian noise was set to 0.5. The number of chargers and waiting spots at each charging station were set to 30 and 50, respectively, and the initial pricing $p_0 = 0.1$.

C. Comparative Experiments and Analysis

To evaluate the effectiveness of the proposed D³PGSA algorithm, we compare it against six representative algorithms: GWO, WSO [31], NSGAII [29], DDPG [32], TD3 [37], and DSAC [41]. These algorithms cover classical HA and recent DRL methods. The hyperparameter settings for each algorithm are summarized in Table II.

We tested the performance of D³PGSA and the six aforementioned algorithms on all datasets described in Section 5.1. To address environmental randomness, all algorithms were evaluated on the test data 100 times. This process was independently repeated five times, and the final results are reported as the mean ± standard deviation in Table III. In the subsequent tables, bolded values indicate the best performance for each metric.

As shown in Table III, the proposed D³PGSA consistently demonstrates superior performance across most cases. For instance, in terms of the AR metric, D³PGSA achieves a mean of 1029.73 on the LiteDemand dataset, outperforming the best-performing heuristic algorithm WSO and the best-performing RL algorithm DSAC, which achieve mean values of 938.54 and 1014.94, respectively, corresponding to improvements of 9.64% and 1.46%. On the HeavyHub dataset, D³PGSA attains a mean of 3093.64, exceeding WSO and DSAC, which achieve 2825.37 and 3083.47, by 9.50% and 0.33%, respectively. Similar trends are observed on the other two virtual datasets, suggesting that D³PGSA maintains strong policy optimization and generalization capabilities under both low and high demand uncertainty scenarios. On the two real-world datasets, the performance of D³PGSA further corroborates these findings. Specifically, in terms of the TWREI metric, D³PGSA achieves a mean of 1.35 on the HeavyHub dataset, surpassing NSGAII and DSAC, which achieve 0.68 and 1.31, by 98.53% and 3.05%, respectively. Comparable results are observed across other datasets, demonstrating that D³PGSA effectively balances computation time and solution quality across various demand scenarios, thereby enhancing its suitability for dynamic pricing in EV charging stations.

To assess the computational efficiency and execution performance of the proposed method, we conducted a visual comparison of the solution time of D³PGSA and the two HA across different datasets. The results are illustrated in Fig.7.

As shown in Fig.7, D³PGSA exhibits significantly lower solution times compared to heuristic algorithms across all datasets. With the increase in demand scale, the solution times of heuristic methods rise markedly. For example, the time required by WSO and NSGAII increases from 68.58s and 79.22s on the LiteDemand dataset to 147.89s and 183.47s on the HeavyHub dataset, representing increases of 115.65% and 131.60%, respectively. In contrast, the solution time of D³PGSA increases only by 38.89%, from 0.18s to 0.25s, while consistently maintaining second-level efficiency and achieving comparable or superior objective values. Furthermore, on the two more complex real-world datasets, the solution time of D³PGSA increases by 24.00%, from 0.25s to 0.31s, while continuing to achieve the highest revenue and maintaining a solution time within seconds. In comparison, the solution times of WSO and NSGAII increase from 147.89s and 183.47s to 228.82s and 269.11s, corresponding to growth rates of 54.72% and 46.68%, approximately twice that of D³PGSA. These results highlight the long-term advantage of D³PGSA in addressing complex dynamic pricing problems and further demonstrate the strong policy search capability enabled by deep reinforcement learning.

To verify the statistical significance of the performance differences between the proposed D³PGSA algorithm and other methods under different scenarios, we conducted t-tests on the AR metric across algorithms. The t-value is the test statistic calculated in a t-test, the p-value is used to indicate the level of significance, with a threshold of p-value < 0.05 generally considered statistically significant, and stat.sig. is short for statistical significance. This analysis is used to examine whether the difference in mean values between two samples is meaningful. The results are presented in Table IV.

The results indicate that D³PGSA significantly outperforms the baseline algorithms in the vast majority of test scenarios, with p-values well below 0.05, demonstrating statistically significant differences. For example, in the LiteDemand scenario, the t-value between D³PGSA and DSAC is 18.18 with a p-value of 2.89E-13, indicating a clear advantage of D³PGSA. Compared to NSGAII, the t-value is 3.25 with a p-value of 0.01, also demonstrating a statistically significant difference. The t-value between D³PGSA and DDPG reaches 140.46 with a p-value of 7.38E-15, indicating an extremely significant difference. Similar statistically significant advantages are observed across other datasets, further demonstrating the robust superiority of D³PGSA in practical application scenarios. However, for NSGAII on the HeavyHub dataset, large performance fluctuations during multiple trials result in a lack of statistically significant difference in the t-test. Furthermore, to better characterize the performance stability and variation range of each algorithm, the 95% confidence intervals for the

TABLE III
COMPARATIVE EXPERIMENTAL RESULTS OF DIFFERENT ALGORITHMS ON SIX DATASETS

		GWO	WSO[31]	NSGAII[29]	DDPG[32]	TD3[37]	DSAC[41]	D^{3}PGSA
LiteDemand	AR	788.37±6.00	938.54±46.05	867.57±111.66	870.46±2.34	806.85±5.12	1014.94±1.53	**1029.73±0.98**
	WDG	1.20%±1.66%	20.47%±6.02%	11.17%±12.41%	11.75%±2.50%	3.58%±1.86%	30.31%±3.10%	**32.20%±2.93%**
	TWREI	0.03±0.05	0.67±0.18	0.41±0.40	0.76±0.04	0.42±0.05	1.32±0.01	**1.34±0.01**
HeavyHub	AR	2523.27±11.02	2825.37±72.15	2897.31±231.30	2750.26±4.16	2757.03±2.38	3083.47±1.57	**3093.64±1.63**
	WDG	0.00%±0.00%	11.98%±3.01%	14.80%±8.75%	9.00%±0.60%	9.27%±0.44%	22.20%±0.58%	**22.61%±0.52%**
	TWREI	0.01±0.01	0.54±0.14	0.68±0.38	0.80±0.02	0.71±0.02	1.31±0.01	**1.35±0.01**
RushPeak	AR	937.93±4.26	782.38±18.61	918.65±35.04	869.78±1.34	833.69±1.41	958.28±4.40	**992.09±0.75**
	WDG	19.93%±2.57%	0.00%±0.00%	17.51%±6.22%	11.22%±2.66%	6.60%±2.50%	22.54%±3.12%	**26.86%±2.92%**
	TWREI	0.74±0.02	0.03±0.01	0.68±0.20	0.81±0.06	0.62±0.07	1.20±0.03	**1.33±0.01**
OffValley	AR	700.75±1.27	718.77±60.22	659.70±63.70	795.65±1.42	806.63±1.94	809.80±4.26	**836.57±0.76**
	WDG	8.86%±6.39%	11.53%±10.09%	2.05%±4.10%	23.61%±7.26%	25.31%±7.20%	25.83%±7.89%	**29.96%±7.63%**
	TWREI	0.26±0.17	0.41±0.29	0.14±0.20	1.13±0.07	1.23±0.06	1.23±0.05	**1.32±0.02**
ACN-Data	AR	978.54±6.05	1024.21±43.03	999.18±39.98	1047.16±1.13	1035.25±1.60	1084.22±4.28	**1097.63±1.01**
	WDG	0.95%±1.37%	5.64%±3.92%	3.04%±2.94%	8.03%±1.32%	6.80%±1.26%	11.85%±1.37%	**13.24%±1.32%**
	TWREI	0.07±0.09	0.46±0.32	0.28±0.24	0.97±0.05	0.91±0.04	1.24±0.03	**1.34±0.01**
EV Charging Reports	AR	528.47±2.36	533.29±3.66	542.70±9.10	589.85±0.83	577.23±0.61	616.73±0.75	**618.79±0.56**
	WDG	0.00%±0.00%	0.91%±0.34%	2.70%±2.00%	11.61%±0.42%	9.23%±0.51%	16.70%±0.49%	**17.09%±0.48%**
	TWREI	0.01±0.01	0.07±0.02	0.18±0.13	1.04±0.02	0.94±0.02	1.32±0.02	**1.34±0.01**

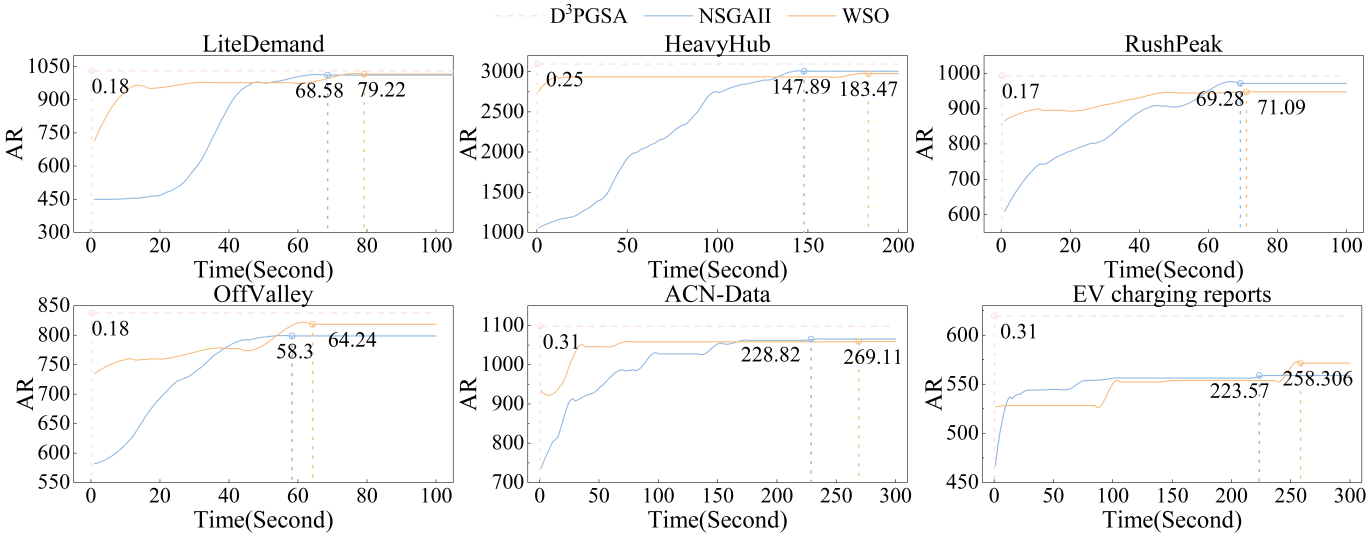

Fig. 7. Solution time curves of D^{3}PGSA and HA

TABLE IV
STATISTICAL SIGNIFICANCE TEST RESULTS OF D^{3}PGSA AGAINST OTHER ALGORITHMS ON THE AR METRIC

		GWO	WSO [31]	NSGAII [29]	DDPG [32]	TD3 [37]	DSAC [41]
Lite Demand	t-value	88.7959	4.4266	3.2474	140.4631	95.5044	18.1822
	p-value	2.89E-13	0.0022	0.0117	7.38E-15	1.61E-13	8.60E-08
	stat.sig.	✓	✓	✓	✓	✓	✓
Heavy Hub	t-value	114.4821	8.3116	1.9241	171.7188	261.0113	10.0433
	p-value	3.79E-14	3.31E-05	0.0945	1.48E-15	5.20E-17	8.22E-06
	stat.sig.	✓	✓	✗	✓	✓	✓
Rush Peak	t-value	27.9875	25.1695	4.6857	178.6859	222.1023	16.9246
	p-value	2.87E-09	6.65E-09	0.0016	1.08E-15	1.89E-16	1.51E-07
	stat.sig.	✓	✓	✓	✓	✓	✓
Off Valley	t-value	205.6329	4.3737	6.2083	56.8720	32.1750	13.8493
	p-value	3.50E-16	0.0024	0.0003	1.01E-11	9.48E-10	7.14E-07
	stat.sig.	✓	✓	✓	✓	✓	✓
ACN-Data	t-value	43.4158	3.8145	5.5052	74.4298	73.6775	6.8164
	p-value	8.74E-11	0.0051	0.0006	1.18E-12	1.28E-12	0.0001
	stat.sig.	✓	✓	✓	✓	✓	✓
EV charging reports	t-value	83.4531	51.6073	18.6517	64.9579	112.3183	4.9098
	p-value	4.74E-13	2.2E-11	7.05E-08	3.51E-12	4.41E-14	0.0012
	stat.sig.	✓	✓	✓	✓	✓	✓

key metrics are calculated and summarized in Table V. As shown in Table V, D³PGSA not only achieves higher average values across most datasets but also consistently yields confidence intervals that are positioned above those of the baseline algorithms, with narrower spans, indicating better robustness and convergence consistency. For instance, on the LiteDemand dataset, the 95% confidence interval for the AR metric of D³PGSA is 1029.73 ± 1.22, clearly positioned above the upper bounds of all other algorithms, further confirming the stability and significance of D³PGSA's performance advantage. On the HeavyHub dataset, the confidence interval for D³PGSA's TWREI metric is 1.35 ± 0.01, which is significantly higher and more stable than those of DSAC and WSO, demonstrating strong adaptability to complex environments.

In addition, to comprehensively assess the effectiveness of the SimSiam-based feature generation module, we conducted comparative experiments with an autoencoder model, to gain clearer insight into the strengths and weaknesses of different feature generation approaches. The results under the same experimental conditions are summarized in Table VI.

The SimSiam-based feature generation model outperforms the autoencoder across all four datasets, with a particularly notable advantage in terms of the AR metric. In addition, regarding computational efficiency, SimSiam demonstrates significantly lower runtime on all datasets. Its maximum computation time is only 0.31s, while the autoencoder consistently takes around 0.7s to 0.8s, indicating that SimSiam not only delivers superior feature generation performance but also achieves high computational efficiency.

TABLE V
CONFIDENCE INTERVALS OF AR AND TWREI IN THE COMPARATIVE EXPERIMENTS

		GWO	WSO[31]	NSGAII[29]	DDPG[32]	TD3[37]	DSAC[41]	D^{3}PGSA
LiteDemand	AR	788.37±7.45	938.54±57.18	867.57±138.64	870.46±2.90	806.85±6.36	1014.94±1.90	**1029.73±1.22**
	TWREI	0.03±0.06	0.67±0.22	0.41±0.50	0.76±0.05	0.42±0.06	1.32±0.02	**1.34±0.01**
HeavyHub	AR	2523.27±13.68	2825.37±89.59	2897.31±287.20	2750.26±5.17	2757.03±2.95	3083.47±1.95	**3093.64±2.02**
	TWREI	0.01±0.02	0.54±0.17	0.68±0.47	0.80±0.02	0.71±0.02	1.31±0.02	**1.35±0.01**
RushPeak	AR	937.93±5.29	782.38±23.11	918.65±43.50	869.78±1.66	833.69±1.75	958.28±5.47	**992.09±0.93**
	TWREI	0.74±0.03	0.03±0.01	0.68±0.24	0.81±0.07	0.62±0.08	1.20±0.04	**1.33±0.01**
OffValley	AR	700.75±1.57	718.77±74.78	659.70±79.09	795.65±1.76	806.63±2.40	809.80±5.28	**836.57±0.94**
	TWREI	0.26±0.20	0.41±0.35	0.14±0.25	1.14±0.07	1.24±0.04	1.22±0.09	**1.32±0.02**
ACN-Data	AR	978.54±7.51	1024.21±53.43	999.18±49.64	1047.16±1.41	1035.25±1.99	1084.22±5.32	**1097.63±1.25**
	TWREI	0.07±0.11	0.46±0.40	0.28±0.29	0.97±0.06	0.91±0.05	1.24±0.03	**1.34±0.01**
EV Charging Reports	AR	528.47±2.92	533.29±4.55	542.70±11.30	589.85±1.03	577.23±0.76	616.73±0.93	**618.79±0.69**
	TWREI	0.01±0.01	0.07±0.02	0.18±0.16	1.04±0.02	0.94±0.02	1.32±0.02	**1.34±0.02**

TABLE VI
COMPARISON RESULTS OF FEATURE GENERATION MODELS USING SIMSIAM AND AUTOENCODER

	LiteDemand		HeavyHub		ACN-Data		EV charging reports	
	AR	Time(s)	AR	Time(s)	AR	Time(s)	AR	Time(s)
SimSiam	**1030.7**	**0.18**	**3094.46**	**0.25**	**1098.97**	**0.31**	**619.56**	**0.31**
Autoencoder	1000.7	0.29	3045.55	0.7	1076.08	0.8	613.67	0.8

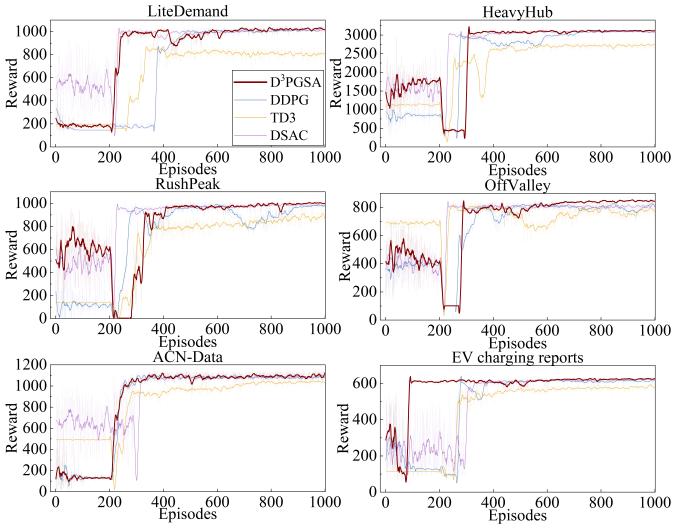

Fig. 8. Training curves of D³PGSA and baseline algorithms across different datasets

D. Training Process Visualization Analysis

To further investigate the optimization performance of D³PGSA across different datasets, we conducted a visual analysis of the training processes for all algorithms. In the Fig.8, the horizontal axis represents the number of training iterations, and the vertical axis denotes the reward obtained after each iteration. This visualization allows us to directly observe the dynamic behavior of each algorithm during optimization and its progression toward convergence to the optimal solution.

As illustrated in Fig.8, reinforcement learning algorithms often exhibit a plateau or decline in reward during the early training phase due to extensive exploration, reflecting the inherent exploration–exploitation trade-off. In the early stage, the algorithm focuses on exploring diverse strategies to escape local optima, while later shifting toward exploiting learned knowledge to stabilize performance. Although early exploration incurs a performance cost, rewards gradually increase as training progresses, leading to convergence toward an optimal solution. Unlike heuristic algorithms that rely on fixed rules, D³PGSA continuously interacts with the environment, enabling flexible policy refinement and superior adaptability to complex tasks. Its exploration mechanism allows optimization over a broad solution space without frequent state resets, highlighting the advantages of reinforcement learning over heuristic methods.

TABLE VII
RESULTS OF D^{3}PGSA ON FOUR DATASETS UNDER DIFFERENT LEARNING RATES

	Learning rate	0.0001	0.0005	0.001	0.002	0.005
LiteDemand	AR	925.07	992.00	1013.27	**1030.7**	999.41
	WDG	0.00%	7.23%	9.53%	**11.42%**	8.04%
	AVE	327.84	598.06	859.41	**904.34**	821.12
	TTC	-	72.76	60.69	43.52	**43.18**
HeavyHub	AR	2746.44	3022.62	3078.66	**3094.46**	3080.94
	WDG	0.00%	10.07%	12.10%	**12.67%**	12.18%
	AVE	1634.77	2379.28	2505.11	**2550.65**	2077.65
	TTC	-	77.74	73.06	**69.42**	105.82
ACN-Data	AR	1035.39	1044.80	1079.03	**1098.97**	1081.01
	WDG	0.00%	0.91%	4.22%	**6.14%**	4.41%
	AVE	702.19	795.98	870.34	**960.39**	870.65
	TTC	87.42	**66.96**	73.16	68.82	78.74
EV charging reports	AR	598.31	609.27	608.98	**619.56**	613.55
	WDG	0.00%	1.83%	1.78%	**3.55%**	2.55%
	AVE	396.97	435.45	**474.94**	442.63	376.14
	TTC	**70.06**	106.64	97.73	87.03	101.68

E. Hyperparameter Experiments

To identify the optimal hyperparameter settings for D³PGSA, we conducted experiments on two virtual datasets and two real-world datasets, focusing on the learning rate and the number of hidden neurons. In addition to standard performance metrics, we introduced two evaluation indicators: Average Reward (Ave) and Time to Convergence (TTC). A smaller TTC value indicates faster convergence under the corresponding learning rate, thereby reflecting the training efficiency of the model.

$$\mathrm{AVE} = \frac{1}{\mathrm{G}} \sum_{j=1}^{\mathrm{G}} r_j \tag{44}$$

$$\mathrm{TTC} = \mathrm{st} \times \min\{j \mid r_j > \mathrm{AVE}, j = 1, 2, ..., \mathrm{G}\} \tag{45}$$

where st represents the solution time for D³PGSA on each dataset as shown in Fig.7. The experimental results are presented in Table VII and Fig.9.

Analysis of the data in Table VII shows that lower learning rates result in increased TTC and slower convergence, as smaller learning rates lead to smaller parameter updates, making it difficult for the model to efficiently approach the optimal solution. In some cases, convergence failure is observed,

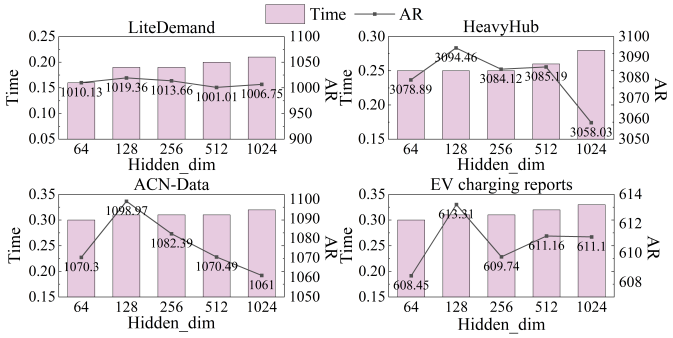

Fig. 9. Experimental results with varying numbers of hidden neurons

TABLE VIII
TEST RESULTS FOR THE DBSCAN PARAMETER EPSILON

	epsilon	0.1	0.5	1	2	5
Lite Demand	AR	1013.72	1022	**1024.13**	1015.69	1005.79
	WDG	0.79%	1.61%	**1.82%**	0.98%	0.00%
Heavy Hub	AR	2975.69	3066.92	**3080.99**	3074.36	3073.18
	WDG	0.00%	3.07%	**3.54%**	3.32%	3.28%
ACN-Data	AR	1069.18	1073.74	**1087.03**	1064.28	1072.58
	WDG	0.46%	0.89%	**2.14%**	0.00%	0.78%
EV charging reports	AR	607.4	612.72	617.35	**617.44**	610.19
	WDG	0.00%	0.88%	1.64%	**1.65%**	0.46%

denoted by "-" in the table, indicating that the model becomes trapped in local optima due to insufficient update magnitude. In contrast, higher learning rates accelerate convergence but often degrade the AVE, as excessive parameter updates near the optimum may prevent the model from stabilizing at the optimal point. Based on the overall results, a learning rate of 0.002 is selected as the optimal setting for subsequent experiments.

As shown in Fig.9, increasing the number of hidden neurons does not always lead to improved performance. For example, on the ACN-Data dataset, increasing the number of hidden units from 64 to 128 raises the AR from 1070 to 1098.97, however, further increases result in a decline in reward. Moreover, the model's solution time tends to increase with larger network sizes. A similar trend is observed across the other three datasets, where performance peaks when using 128 hidden neurons and then gradually declines. This phenomenon suggests that an excessive number of hidden neurons may lead to overfitting. Based on these observations, the number of hidden neurons is set to 128 to achieve optimal performance across the datasets.

In addition, to verify the effectiveness of the DBSCAN clustering algorithm, we investigated its key parameters: epsilon and MinPts. Table VIII presents the experimental results under different values of epsilon. As shown in Table VIII, an excessively large epsilon may cause the merging of distinct clusters, thereby reducing clustering quality, whereas an overly small epsilon increases the number of noise points. Therefore, an epsilon value of 1 is selected for subsequent experiments.

As shown in Table IX, when MinPts is set too low, the clustering process tends to form numerous small and unstable clusters, capturing noise rather than meaningful patterns. Conversely, excessively high MinPts values cause the algorithm to overlook genuine dense regions, merging nearby points into noise and reducing clustering effectiveness. This phenomenon reflects the sensitivity of density-based clustering to

TABLE IX
TEST RESULTS FOR THE DBSCAN PARAMETER MINPTS

	MinPts	1	2	5	10	50
Lite Demand	AR	1014.28	1024.25	**1030.7**	1028.19	1010.34
	WDG	0.39%	1.38%	**2.02%**	1.77%	0.00%
Heavy Hub	AR	3068.09	3075.09	**3094.46**	3088.71	3037.40
	WDG	1.01%	1.24%	**1.88%**	1.69%	0.00%
ACN-Data	AR	1073.18	1077.51	**1098.97**	1084.62	1066.36
	WDG	0.64%	1.05%	**3.06%**	1.71%	0.00%
EV charging reports	AR	613.21	611.16	**619.56**	614.50	592.27
	WDG	3.54%	3.19%	**4.61%**	3.75%	0.00%

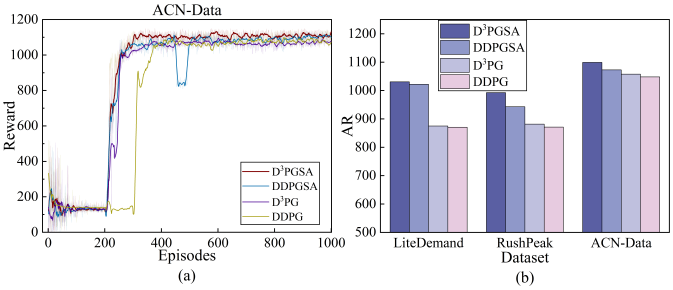

Fig. 10. Ablation experiment results

the balance between local point density and global structure recognition. Based on these observations, an MinPts value of 5 is selected for subsequent experiments.

F. Ablation Study

To further validate the effectiveness of individual components in D³PGSA, we conducted an ablation study. The results are shown in Fig.10. As shown in Fig.10(a), D³PGSA outperforms other variants in both reward improvement and convergence speed, validating the effectiveness of its design. The DDPGSA variant, which removes the dual-value network, exhibits greater training fluctuations and slightly lower performance. D³PG, which retains only the dual-value network without sample augmentation, achieves relatively stable training but lower overall rewards. The DDPG demonstrates the poorest performance and stability, further highlighting that the absence of both mechanisms severely impairs training efficiency in complex environments. As shown in Fig.10(b), D³PGSA achieves the highest objective values across all datasets, further verifying the effectiveness of its design. The performance drop from D³PGSA to DDPGSA highlights the critical role of the dual-value network in improving training stability and learning efficiency. Although DDPGSA removes the dual-value structure, it still outperforms D³PG and standard DDPG, confirming the significant contribution of the sample augmentation module. D³PG without sample augmentation performs better than standard DDPG but remains inferior to D³PGSA and DDPGSA. In summary, the dual-value network enhances stability while sample augmentation substantially improves learning performance.

To further validate the practical benefit of the dual-value network, we conducted a comparative experiment on the ACN-Data dataset. The results, shown in Fig.10(b), indicate that under the same number of updates, D³PG effectively suppresses Q-value overestimation. Its average Q-value error was reduced by 60.85%, from 43.65 in DDPG to 17.09, while the AR metric improved by approximately 0.93%.

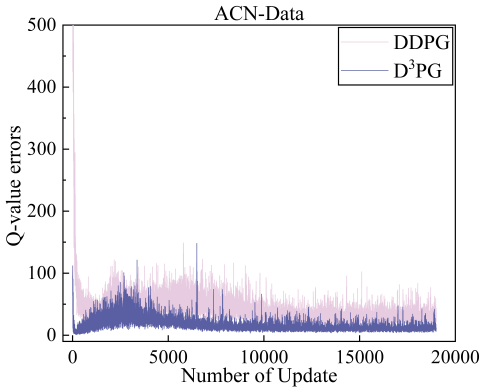

Fig. 11. Q-value error comparison

These results confirm that using the minimum Q-value in policy gradient updates offers significant stability advantages in practical applications.

VI. Conclusion

In this paper, we conducted an in-depth study on the dynamic pricing problem for EV charging stations (EVCS), which is particularly challenging due to market demand uncertainty, the need for real-time decision-making, and the requirement to maximize operator revenue within users' acceptable price ranges. To address these challenges, we proposed D³PGSA, an enhanced deep deterministic policy gradient algorithm that effectively overcomes the limitations of traditional reinforcement learning and heuristic methods under limited data, dynamic demand, and complex environmental conditions. Specifically, D³PGSA integrates an experience generation model, a feature generation model, a DBSCAN-based clustered experience replay buffer, and a dual-value architecture to enable more effective policy learning. These components collectively facilitate rational pricing decisions and establish a solid theoretical foundation for dynamic pricing strategies. The proposed improvements enhance the diversity of training samples, addressing the low sample utilization problem in standard DDPG, while the dual-value network mitigates estimation bias caused by single Q-networks, improving training stability and policy accuracy. Overall, these enhancements allow D³PGSA to surpass the fundamental limitations of existing RL and heuristic approaches in dynamic EVCS pricing. Extensive comparative experiments validated the effectiveness of D³PGSA. Results demonstrate that, compared to traditional baselines and recent methods, D³PGSA more flexibly adjusts pricing strategies, better responds to market demand changes, and improves station revenues.

In future work, we aim to further extend the applicability and real-world adaptability of the proposed model. First, considering that practical EV charging infrastructures typically involve multiple charging stations, we plan to expand the current framework into a multi-station collaborative pricing model. By leveraging distributed reinforcement learning and a centralized training with decentralized execution paradigm, we seek to enable coordination and policy sharing among stations to address large-scale optimization and scheduling challenges. Building upon this, we will also enhance the dimensionality of user behavior modeling to improve the policy's adaptability to diverse user decision-making logics. Specifically, we plan to incorporate features such as users' historical charging patterns, time preferences, and travel flexibility to construct more sophisticated user response models that closely mimic real-world behavior. Moreover, to strengthen the model's awareness of energy system constraints, we will explore integrating dynamic information related to renewable energy supply and grid load conditions into the pricing process. By introducing external variables such as photovoltaic (PV) generation variability and grid capacity limits, we aim to develop a pricing constraint mechanism informed by energy scheduling and grid feedback. This would ensure that pricing strategies not only maximize revenue, but also maintain system stability and support long-term sustainability.

References

[1] Y. Zhang, Y. Wang, F. Li, B. Wu, Y. Chiang, and X. Zhang, "Efficient deployment of electric vehicle charging infrastructure: Simultaneous optimization of charging station placement and charging pile assignment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 10, pp. 6654–6659, 2020.

[2] Y. Huang and K. Kockelman, "Electric vehicle charging station locations: Elastic demand, station congestion, and network equilibrium," *Transportation Research Part D: Transport and Environment*, vol. 78, p. 102179, 2020.

[3] A. Kazemtarghi, A. Mallik, and Y. Chen, "Dynamic pricing strategy for electric vehicle charging stations to distribute the congestion and maximize the revenue," *International Journal of Electrical Power & Energy Systems*, vol. 158, p. 109946, 2024.

[4] M. Saleem, S. Abbas, T. Ghazal, M. Khan, N. Sahawneh, and M. Ahmad, "Smart cities: Fusion-based intelligent traffic congestion control system for vehicular networks using machine learning techniques," *Egyptian Informatics Journal*, vol. 23, no. 3, pp. 417–426, 2022.

[5] M. Saleem, A. Khadim, M. Fatima, M. Khan, H. Nair, and M. Asif, "Assma-slm: Autonomous system for smart motor-vehicles integrating artificial and soft learning mechanisms," in *2022 International Conference on Cyber Resilience (ICCR)*. IEEE, 2022, pp. 1–6.

[6] H. Ahmadvand and F. Foroutan, "Dv-arpa: Data variety aware resource provisioning for big data processing in accumulative applications," *arXiv preprint arXiv:2008.04674*, 2020.

[7] T. Dargahi, H. Ahmadvand, M. Alraja, and C. Yu, "Integration of blockchain with connected and autonomous vehicles: vision and challenge," *ACM Journal of Data and Information Quality (JDIQ)*, vol. 14, no. 1, pp. 1–10, 2021.

[8] M. Khan, M. Farooq, M. Saleem, T. Shahzad, M. Ahmad, S. Abbas, and A. Abu-Mahfouz, "Smart buildings: Federated learning-driven secure, transparent and smart energy management system using xai," *Energy Reports*, vol. 13, pp. 2066–2081, 2025.

[9] L. Huang, Y. Jia, X. Han, and P. Wang, "Tri-level dynamic pricing for charging network operator in coupled power-transportation networks," *IEEE Transactions on Smart Grid*, 2024.

[10] M. Cedillo, H. Sun, J. Jiang, and Y. Cao, "Dynamic pricing and control for ev charging stations with solar generation," *Applied Energy*, vol. 326, p. 119920, 2022.

[11] G. Boateng, H. Si, H. Xia, X. Guo, C. Chen, I. Agyemang, and N. Ansari, "Automated valet parking and charging: A dynamic pricing and reservation-based framework leveraging multi-agent reinforcement learning," *IEEE Transactions on Intelligent Vehicles*, 2024.

[12] M. Adil, M. Mahmud, A. Kouzani, and S. Khoo, "Optimal location and pricing of electric vehicle charging stations using machine learning and stackelberg game," *IEEE Transactions on Industry Applications*, vol. 60, no. 3, pp. 4708–4722, 2024.

[13] J. Tan, F. Liu, N. Xie, W. Guo, and W. Wu, "Dynamic pricing strategy of charging station based on traffic assignment simulation," *Sustainability*, vol. 14, no. 21, p. 14476, 2022.

[14] S. Lai, J. Qiu, Y. Tao, and J. Zhao, "Pricing for electric vehicle charging stations based on the responsiveness of demand," *IEEE Transactions on Smart Grid*, vol. 14, no. 1, pp. 530–544, 2022.

[15] V. Jadoun, "Risk-based dynamic pricing by metaheuristic optimization approach for electric vehicle charging infrastructure powered by grid integrated microgrid system," *Electric Power Systems Research*, vol. 230, p. 110250, 2024.

[16] Y. Dai, Y. Qi, L. Li, B. Wang, and H. Gao, "A dynamic pricing scheme for electric vehicle in photovoltaic charging station based on stackelberg game considering user satisfaction," *Computers & Industrial Engineering*, vol. 154, p. 107117, 2021.

[17] S. Aznavi, P. Fajri, M. Shadmand, and A. Khoshkbar-Sadigh, "Peer-to-peer operation strategy of pv equipped office buildings and charging stations considering electric vehicle energy pricing," *IEEE Transactions on Industry Applications*, vol. 56, no. 5, pp. 5848–5857, 2020.

[18] L. Gong, W. Cao, K. Liu, and J. Zhao, "Optimal charging strategy for electric vehicles in residential charging station under dynamic spike pricing policy," *Sustainable Cities and Society*, vol. 63, p. 102474, 2020.

[19] Q. Zhang, Y. Hu, W. Tan, C. Li, and Z. Ding, "Dynamic time-of-use pricing strategy for electric vehicle charging considering user satisfaction degree," *Applied Sciences*, vol. 10, no. 9, p. 3247, 2020.

[20] Y. Li, J. Wang, W. Wang, C. Liu, and Y. Li, "Dynamic pricing based electric vehicle charging station location strategy using reinforcement learning," *Energy*, vol. 281, p. 128284, 2023.

[21] B. Aljafari, P. Jeyaraj, A. Kathiresan, and S. Thanikanti, "Electric vehicle optimum charging-discharging scheduling with dynamic pricing employing multi agent deep neural network," *Computers and Electrical Engineering*, vol. 105, p. 108555, 2023.

[22] R. Lin, H. Chu, J. Gao, and H. Chen, "Charging management and pricing strategy of electric vehicle charging station based on mean field game theory," *Asian Journal of Control*, 2023.

[23] C. Lu, J. Wu, J. Cui, Y. Xu, C. Wu, and M. Gonzalez, "Deadline differentiated dynamic ev charging price menu design," *IEEE Transactions on Smart Grid*, vol. 14, no. 1, pp. 502–516, 2022.

[24] S. Chakraborty, S. Mukhopadhyay, and S. Biswas, "Optimal placement of pv-dstatcom based ev charging stations with dynamic pricing," *IEEE Transactions on Industry Applications*, vol. 59, no. 6, pp. 7092–7102, 2023.

[25] Z. Zhao and C. Lee, "Dynamic pricing for ev charging stations: A deep reinforcement learning approach," *IEEE Transactions on Transportation Electrification*, vol. 8, no. 2, pp. 2456–2468, 2021.

[26] N. Yang, X. Shen, P. Liang, L. Ding, J. Yan, C. Xing, and L. Zhang, "Spatial–temporal optimal pricing for charging stations: A model-driven approach based on group price response behavior of evs," *IEEE Transactions on Transportation Electrification*, vol. 10, no. 4, pp. 8869–8880, 2024.

[27] Y. Chen, S. Hu, S. Xie, Y. Zheng, Q. Hu, and Q. Yang, "Optimal dynamic pricing of fast charging stations considering bounded rationality of users and market regulation," *IEEE Transactions on Smart Grid*, vol. 15, no. 4, pp. 3950–3965, 2024.

[28] Z. Liu, Y. Zhou, D. Feng, S. Xu, Y. Yi, H. Li, and H. Wang, "Dynamic pricing of electric vehicle charging station alliances under information asymmetry," *CSEE Journal of Power and Energy Systems*, 2025.

[29] Z. Zhaoyun and W. Xinghua, "Researh on dynamic time-sharing tariff orderly charging strategy based on nsga2 in pv-storage-charging stations," *Electric Power Systems Research*, vol. 225, p. 109784, 2023.

[30] P. Chakraborty and M. Pal, "Planning of fast charging infrastructure for electric vehicles in a distribution system and prediction of dynamic price," *International Journal of Electrical Power & Energy Systems*, vol. 155, p. 109502, 2024.

[31] M. Braik, A. Hammouri, J. Atwan, M. Al-Betar, and M. Awadallah, "White shark optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems," *Knowledge-Based Systems*, vol. 243, p. 108457, 2022.

[32] L. Cui, Q. Wang, H. Qu, M. Wang, Y. Wu, and L. Ge, "Dynamic pricing for fast charging stations with deep reinforcement learning," *Applied Energy*, vol. 346, p. 121334, 2023.

[33] A. Fraija, N. Henao, K. Agbossou, S. Kelouwani, M. Fournier, and S. Nagarsheth, "Deep reinforcement learning based dynamic pricing for demand response considering market and supply constraints," *Smart Energy*, vol. 14, p. 100139, 2024.

[34] R. Jin, Y. Zhou, C. Lu, and J. Song, "Deep reinforcement learning-based strategy for charging station participating in demand response," *Applied Energy*, vol. 328, p. 120140, 2022.

[35] S. Lee and D. Choi, "Dynamic pricing and energy management for profit maximization in multiple smart electric vehicle charging stations: A privacy-preserving deep reinforcement learning approach," *Applied Energy*, vol. 304, p. 117754, 2021.

[36] L. Hou, Y. Li, J. Yan, C. Wang, L. Wang, and B. Wang, "Multi-agent reinforcement mechanism design for dynamic pricing-based demand response in charging network," *International Journal of Electrical Power & Energy Systems*, vol. 147, p. 108843, 2023.

[37] A. Abdalrahman and W. Zhuang, "Dynamic pricing for differentiated pev charging services using deep reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 1415–1427, 2020.

[38] S. Bae, B. Kulcsar, and S. Gros, "Personalized dynamic pricing policy for electric vehicles: Reinforcement learning approach," *Transportation Research Part C: Emerging Technologies*, vol. 161, p. 104540, 2024.

[39] V. Moghaddam, A. Yazdani, H. Wang, D. Parlevliet, and F. Shahnia, "An online reinforcement learning approach for dynamic pricing of electric vehicle charging stations," *IEEE Access*, vol. 8, pp. 130 305–130 313, 2020.

[40] H. Yang, Y. Xu, and Q. Guo, "Dynamic incentive pricing on charging stations for real-time congestion management in distribution network: an adaptive model-based safe deep reinforcement learning method," *IEEE Transactions on Sustainable Energy*, vol. 15, no. 2, pp. 1100–1113, 2023.

[41] J. Duan, W. Wang, L. Xiao, J. Gao, S. Li, C. Liu, and K. Li, "Distributional soft actor-critic with three refinements," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.