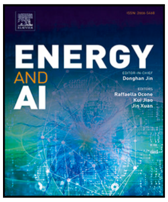
Contents lists available at ScienceDirect

Energy and AI

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



A graph neural network enhanced decision transformer for efficient optimization in dynamic smart charging environments

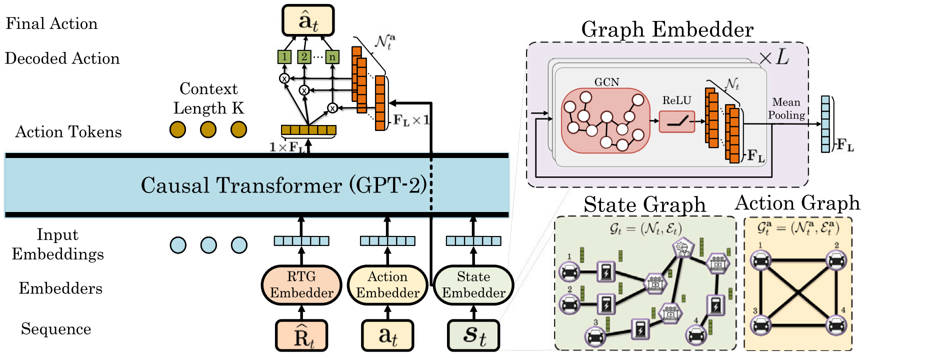
Stavros Orfanoudakis *, Nanda Kishor Panda, Peter Palensky, Pedro P. Vergara

Delft University of Technology, Intelligent Electrical Power Grids, Mekelweg 5, Delft, 2628 CD, The Netherlands

HIGHLIGHTS

- Topology-aware Decision Transformer for EV smart charging.
- Learns from offline trajectories and is real-time suitable.
- Outperforms strong RL/MPC baselines using fewer trajectories.
- Generalizes across EV fleet sizes and network topologies without retraining.

GRAPHICAL ABSTRACT



ARTICLE INFO

Keywords:
Decision transformer
Large language models
Electric vehicle
Smart charging
Graph neural networks

ABSTRACT

Electric-vehicle smart charging requires quick decision-making under uncertainty while enforcing strict electricity grid and user requirements. Mathematical optimization becomes too slow at scale, while online reinforcement learning struggles with sparse rewards and safety. This paper proposes GNN-DT, a topology-aware Decision Transformer that combines graph neural network embeddings with sequence modeling to learn charging policies from offline trajectories. The method operates over variable numbers of vehicles and chargers without retraining. Evaluated on realistic smart charging scenarios, GNN-DT achieves near-optimal performance, reaching rewards within 5 percent of an oracle solver while using up to 10× fewer training trajectories than baseline methods. It consistently outperforms online and offline reinforcement learning approaches and generalizes to unseen fleet sizes and network topologies. Inference runs in milliseconds, making the approach suitable for real-time deployment in large-scale charging systems.

1. Introduction

Electric vehicle (EV) smart charging requires coordinating large numbers of vehicles over long time horizons while respecting user requirements, electricity prices, and grid capacity constraints. In practice, charge point operators (CPOs) must make decisions in real-time under uncertainty regarding arrivals, departures, and energy demand [1]. Mathematical optimization and model predictive control can produce high-quality schedules, but they become computationally expensive as fleet size and network complexity increase, limiting their use in real-time operations. Online reinforcement learning (RL) offers adaptability

* Corresponding author.
E-mail address: s.orfanoudakis@tudelft.nl (S. Orfanoudakis).

https://doi.org/10.1016/j.egyai.2026.100679
Received 25 September 2025; Received in revised form 17 December 2025; Accepted 4 January 2026
Available online 16 January 2026

but often struggles with sparse rewards, safety constraints, and poor generalization across changing charging infrastructures. As a result, a gap remains between scalable, real-time control and methods that can effectively leverage historical charging data while generalizing across variable fleet sizes and network topologies.

1.1. Literature review

Mathematical and stochastic optimization provide strong baselines when models and forecasts are reliable [2]. Mixed-integer, robust, and chance-constrained formulations encode safety and quality-of-service constraints directly in the mathematical formulation. Furthermore, model predictive control (MPC) adds receding-horizon adaptation to track forecasts and market signals [3]. These methods are transparent, support sensitivity analysis, and offer explicit constraint guarantees [4]. However, as EV fleets scale, topologies vary, and forecast errors accumulate, repeated large-scale reoptimization becomes burdensome [5] and static formulations struggle to remain responsive [6]. Therefore, accurately modeling the uncertainty and scalability become key bottlenecks of mathematical optimization.

To relax modeling assumptions and improve adaptability, RL [7] has been applied to EV charging [8]. Actor–critic and value-based methods such as Deep Deterministic Policy Gradient (DDPG) [9], and Soft Actor–Critic (SAC) [10], have shown improvements over heuristics in both single-station and fleet settings. Graph Neural Networks (GNN) based formulations have the potential to exploit relational structure to scale across layouts and demand patterns [11]. Moreover, multi-agent (MA) RL methods assign local decision-making to stations or aggregators, hence simplifying the problem even more, but at the cost of optimality convergence [12]. Despite these advances, online RL often fails to learn under sparse or delayed rewards [13]. Most importantly, RL is sensitive to design choices, has difficulty enforcing strict operational constraints, and convergence can degrade in large, heterogeneous systems [14]. The need for more efficient training and stronger generalization motivates an offline learning paradigm.

Offline RL addresses safety and data-efficiency by training policies from logged trajectories collected under heuristic or optimized EV dispatch. Offline RL is particularly efficient in sparse-reward settings where the whole expert trajectories are provided [15]. Recent work builds detailed microgrid and charging models with PV, residential loads, storage, Vehicle to Grid (V2G), and nonlinear charging, and casts scheduling as a mathematical programming problem solved offline. Extrapolation error is limited by keeping the learned policy using behavioral cloning (BC) close to the data support, and tractability is improved by grouping EVs into sets and issuing set-level actions [16]. Pretraining with expert trajectories further accelerates learning and supports robustness to overloads and cost objectives on grid benchmarks. For example, BC on expert trajectories followed by online RL training using Proximal Policy Optimization (PPO) can help accelerate training, requiring fewer training epochs [17]. Beyond charging, offline energy-management studies show that dataset quality and distribution shift remain central. Strategies that constrain policies toward the dataset distribution and periodically update with new EV session data improve efficiency and adaptability [18]. Overall, offline RL can provide safe and effective charging policies when extrapolation is controlled and dataset coverage is strong, but sensitivity to distribution shift, long horizons, and data quality remains.

Decision Transformers (DT) offer a complementary path for offline policy extraction by modeling trajectories as sequences of states, actions, and return-to-go [19]. This formulation leverages successful historical behavior, uses the attention mechanism over long horizons, and enables return conditioning at inference to target different operating points. Hence, DTs can mitigate sparse rewards and reduce task-specific retraining. These properties directly address two offline RL limitations: long-range credit assignment and sensitivity to dataset composition. However, pure trajectory stitching can fail in non-stationary or stochastic settings where high returns occur by chance. Therefore, value-aware regularization (e.g., Q-regularized DT [20]) improves robustness by preferring actions with both sequence support and high estimated value [21]. Currently, transformer-based policies have not been designed to respect dynamic graph structure (variable numbers of EVs/chargers and changing connectivity) while preserving generalization and constraint awareness.

1.2. Our contributions

To address the need for efficient and adaptive decision-making in complex and dynamically evolving energy systems, GNN-DT[1] is introduced. GNN-DT is a topology-aware DT for EV smart charging that operates over variable network sizes without input padding or retraining. This design provides a general foundation for sequential control under dynamic connectivity and physical constraints, addressing the gaps identified in Table 1.

GNN-DT combines Graph Neural Network (GNN) embeddings with sequence modeling to support dynamic state and action spaces that arise from changing numbers of EVs, chargers, and grid connections. Unlike existing GNN-transformer approaches, GNN-DT incorporates a residual action decoding mechanism that maps fixed-length transformer outputs to node-level charging actions using state-dependent graph embeddings, thereby enabling consistent decision-making under changing topologies. The approach is evaluated on a realistic multi-objective EV charging optimization problem with sparse rewards, long horizons, and aggregated grid constraints, achieving near-optimal performance while maintaining real-time inference.

The main contributions are summarized as follows:

- Introducing a DT architecture that integrates GNN embeddings to handle variable state and action spaces, enabling learning without input padding or retraining and improving sample efficiency and generalization across different EV charging scenarios.
- Showing through systematic comparison that both online and offline RL baselines, trained on Optimal, Random, and Business-as-Usual datasets of varying sizes, achieve lower performance than GNN-DT on realistic EV charging optimization tasks.
- Demonstrating that the size and composition of the offline training dataset strongly affect DT performance, and that combining expert and non-expert trajectories leads to higher rewards than training on single-policy datasets alone.

2. Problem formulation

In this section, an introduction to offline RL and the mathematical formulation of the EV charging optimization problem is presented as an example of what type of problems can be solved by the proposed GNN-DT methodology (see Table 2).

2.1. Offline RL

Offline RL aims to learn a policy $\pi_\theta(a \mid s)$ that maximizes the expected discounted return $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$ without additional interactions with the environment [22]. A Markov Decision Process (MDP) is defined by the tuple (S, A, P, R, γ), where S is the state space, A the action space, P the transition function, R the reward function, $\gamma \in (0, 1]$ the discount factor [7]. In the offline setting, a static dataset $D = \{(s, a, r)\}$, collected by a (potentially suboptimal) policy, is provided. DTs leverage this dataset by treating RL trajectories as sequences, learning to predict actions that maximize returns based on previously

[1] The code can be found at https://github.com/StavrosOrf/DT4EVs and https://github.com/distributionnetworksTUDelft/DT4EVs.

Table 1
Literature review of EV smart-charging methods across three categories: mathematical/stochastic optimization, online, and offline RL. Columns summarize the algorithm class, key advantages/limitations, and typical settings.

Ref.	Category	Algorithm	Advantages	Limitations	Setting
[3]	Math. Opt.	MPC	Receding horizon and optimal	Needs short-term forecasts	Operation
[4]	Math. Opt.	MPC	Safety under uncertainty	Conservative; scenario scaling	Planning
[6]	Math. Opt.	DR sched.	Peak shaving; QoS-aware	Static; limited adaptability	Planning
[9]	Online RL	DDPG	Continuous actions; DR-aware	Sparse rewards; hard safety	Operation
[10]	Online RL	SAC	Structured policy improvement	Convergence/tuning sensitivity	Operation
[11]	Online RL	GNN-RL	Relational bias; layout transfer	Training complexity	Operation
[12]	MA RL	Q-Learning	Scales via locality	Non-stationarity; convergence	Operation
[13]	MA RL	Q-Learning	Fairness–efficiency trade-off	Stability at scale	Operation
[14]	Safe RL	Constrained SAC	Safety filter; constraint aware	Performance hit; tuning	Operation
[15]	Offline RL	Behavior Cloning	Good for sparse rewards	Dataset quality sensitivity	Operation
[16]	Offline RL	Behavior Cloning	Limits extrapolation	Dataset coverage limits	Operation
[17]	Off/Online RL	BC + PPO	Faster learning	Expert bias; transfer limits	Operation
[18]	Off/Online RL	BC + Q-Learning	Robust to shift	Comms infra; divergence tuning	Operation
Ours	Seq. Model	GNN-DT	Topology-aware, adaptive architecture, generalization	Higher training cost	Operation

Table 2
Notation for the EV charging optimization problem.

Symbol	Name	Description
Sets		
$\mathcal{T}$	Set of timesteps	Time horizon for optimization
$\mathcal{I}$	Set of charging stations	All EV charging stations
$\mathcal{W}$	Set of charger groups	Chargers grouped by local transformer connections
$\mathcal{J}_i$	Set of charging sessions	Charging sessions at charger i
Indexes		
t	Timestep index	Discrete time step
i	Charger index	Individual EV charging station
j	Session index	Charging session at a charger
w	Charger group index	Charger groups connected to transformers
Parameters		
$t^a_{j,i}$	Arrival time	Time EV j arrives at charger i
$t^d_{j,i}$	Departure time	Time EV j departs charger i
$e^*_{j,i}$	Desired battery capacity	Desired battery energy at departure for session j at charger i
$e^a_{j,i}$	Arrival battery energy	Battery energy at EV arrival
$\underline{e}_{j,i}, \overline{e}_{j,i}$	Battery limits	Min/max allowable battery energy
$\underline{p}^+_{j,i}, \overline{p}^+_{j,i}$	Charging power limits	Min/max charging power
$\underline{p}^-_{j,i}, \overline{p}^-_{j,i}$	Discharging power limits	Min/max discharging power
p^*_t	Total power limit	Desired aggregated power
Π^+_t, Π^-_t	Electricity prices	Prices for charging/discharging at timestep t
Δt	Time interval	Duration of each timestep
$\overline{p}_{w,t}$	Group power limit	Power limit for group w at timestep t
Variables		
$p^+_{i,t}, p^-_{i,t}$	Charging/discharging power	Power assigned at charger i, timestep t
$\omega^+_{i,t}, \omega^-_{i,t}$	Binary operation indicators	Indicates if charger i charges (+) or discharges (−) at timestep t
$e_{j,i,t}$	EV battery energy	Battery level for session j at charger i, timestep t
p^Σ_t	Total aggregated power	Net total power across all chargers at timestep t

collected experiences. A key component in DTs is the *return-to-go* (RTG), which for a time step t can be defined as: $G_t = \sum_{\tau=t}^{T} \gamma^{\tau-t} r_\tau$, representing the discounted cumulative reward from t until the terminal time T. Offline RL is particularly beneficial when real-time exploration is costly or impractical, while sufficient historical data are available.

2.2. The EV smart charging problem

Working closely with a CPO, it was evident that existing heuristic and mathematical programming charging strategies don't scale efficiently as EV fleets grow. To address this, the state–action space and objectives were designed around real-world operational constraints and assumptions provided by the CPO. A set of $\mathcal{I}$ charging stations indexed i is considered, all assumed to be controlled by a CPO over a time window $\mathcal{T}$, divided into T non-overlapping intervals. Since the chargers can be spread around the city, there are charger groups $w \in \mathcal{W}$, that can have a lower-level aggregated power limits representing connections to local power transformers. For a given time window, each charging station i operates a set of $\mathcal{J}$ non-overlapping charging sessions, denoted by $\mathcal{J}_i = \{j_{1,i}, \dots, j_{J_i,i}\}$, where $j_{j,i}$ represents the jth charging event at the ith charging station and $J_i = |\mathcal{J}_i|$ is the total number of charging sessions seen by charging station i in an episode. A charging session is then represented as $j_{j,i} : \{t^a_{j,i}, t^d_{j,i}, \bar{p}_{j,i}, e^*_{j,i}\}$, $\forall j, i$, where $t^a, t^d, \bar{p}$ and e^* represent the arrival time, departure time, maximum charging power, and the desired battery energy level at the departure time. The primary goal is to minimize the total energy cost given by:

$$f_1(p^+, p^-) = \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{I}} \Delta t\big(\Pi^+_t \, p^+_{i,t} - \Pi^-_t \, p^-_{i,t}\big) \tag{1}$$

$p^+_{i,t}$ and $p^-_{i,t}$ denote the charging or discharging power of the ith charging station during time interval t. Π^+_t and Π^-_t are the charging and discharging costs, respectively. Along with minimizing the total energy costs , the CPO also wants the aggregate power of all the charging stations ($p_t^{\Sigma} = \sum_{i \in \mathcal{I}} p_{i,t}^{+} - p_{i,t}^{-}$) to remain below the set power limit p_t^{*}. By doing so, the CPO avoids paying penalties due to overuse of network capacity. Hence, the total power capacity limit penalty is defined as:

$$f_2(p^+, p^-) = \sum_{t \in \mathcal{T}} \max\{0,\ p_t^{\Sigma} - p_t^{*}\}, \tag{2}$$

Maintaining the desired battery charge at departure is crucial for EV user satisfaction. This behavior is modeled as:

$$f_3(p^+, p^-) = \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}_i} \Big(\sum_{t=t_{j,i}^{a}}^{t_{j,i}^{d}} (p_{i,t}^{+} - p_{i,t}^{-}) - e_{j,i}^{*} \Big)^2 \tag{3}$$

Eq. (3) defines a sparse reward added at each EV departure based on its departure energy level. Building on the objective functions described by Eqs. (1)–(3), the overall EV charging problem is formulated as a mixed integer programming (MIP) problem, subject to lower-level operational constraints (e.g., EV battery, power levels) as detailed below:

$$\begin{aligned} \max_{p^+,\omega^+,p^-,\omega^-} \ \sum_{t \in \mathcal{T}} \Big[& - 100 \max\{0,\ p_t^{\Sigma} - p_t^{*}\} \\ & + \sum_{i \in \mathcal{I}} \Big(\Delta t \big(\Pi_t^{+} p_{i,t}^{+} \omega_{i,t}^{+} - \Pi_t^{-} p_{i,t}^{-} \omega_{i,t}^{-} \big) \\ & - 10 \sum_{j \in \mathcal{J}_i} \Big(\sum_{\tau=t_{j,i}^{a}}^{t_{j,i}^{d}} \big(p_{i,\tau}^{+} \omega_{i,\tau}^{+} - p_{i,\tau}^{-} \omega_{i,\tau}^{-} \big) - e_{j,i}^{*} \Big)^2 \Big) \Big] \end{aligned} \tag{4}$$

Subject to:

$$\overline{p}_{w,t} \geq \sum_{i \in \mathcal{W}_i} p_{i,t}^{+} \cdot \omega_{i,t}^{+} - p_{i,t}^{-} \cdot \omega_{i,t}^{-} \qquad \forall i,\ \forall w,\ \forall t \tag{5}$$

$$\underline{e}_{j,i} \leq e_{j,i,t} \leq \overline{e}_{j,i} \qquad \forall j,\ \forall i,\ \forall t \tag{6}$$

$$e_{j,i,t} = e_{j,i,t-1} + (p_{i,t}^{+} \cdot \omega_{i,t}^{+} + p_{i,t}^{-} \cdot \omega_{i,t}^{-}) \cdot \Delta t \qquad \forall j,\ \forall i,\ \forall t \tag{7}$$

$$e_{j,i,t} = e_{j,i}^{a} \qquad \forall j,\ \forall i,\ \forall t |\ t = t_{j,i}^{a} \tag{8}$$

$$\underline{p}_{j,i}^{+} \leq p_{i,t}^{+} \leq \overline{p}_{j,i}^{+} \qquad \forall j,\ \forall i,\ \forall t \tag{9}$$

$$\underline{p}_{j,i}^{-} \geq p_{i,t}^{-} \geq \overline{p}_{j,i}^{-} \qquad \forall j,\ \forall i,\ \forall t \tag{10}$$

$$\omega_{i,t}^{+} + \omega_{i,t}^{-} \leq 1 \qquad \forall i,\ \forall t \tag{11}$$

The multi-objective optimization function in Eq. (4) integrates Eqs. (1)–(3) using experimentally determined coefficients based on practical importance. The power of a single charger i is modeled using four decision variables, $p^+ \cdot \omega^+$ and $p^- \cdot \omega^-$, where ω^+ and ω^- are binary variables, to differentiate between charging and discharging behaviors and enable charging power to get values in ranges $0 \cup [\underline{p}^+, \overline{p}^+]$, and discharging power in $[\underline{p}^-, \overline{p}^-] \cup 0$. Eq. (5) defines the locally aggregated transformer power limits $\overline{p}$ for chargers belonging to groups $\mathcal{W}_i$. Eqs. (6)–(8) address EV battery constraints during operation with a minimum and maximum capacity of $\underline{e}$, $\overline{e}$, and energy e^a at time of arrival t^a. Eqs. (9) and (10) impose charging and discharging power limits for every charger–EV session combination. To prevent simultaneous charging and discharging, the binary variables ω^{ch} and ω^{dis} are constrained by (11).

2.3. EV charging MDP

The optimal EV charging problem can be framed as an MDP: $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, R)$, where $\mathcal{S}$ is the state space, A is the action space, P is the transition probability function, and R is the reward function. At any time step t, the state $\boldsymbol{s}_t \in \mathcal{S}$ is represented by a dynamic graph $\mathcal{G}_t = (\mathcal{N}_t, \mathcal{E}_t)$, where $\mathcal{N}_t$ is the set of nodes and $\mathcal{E}_t$ is the set of edges. The graph is dynamic since the number of nodes in the state and action graph can vary in each step, because of EVs' arrival and departures. Each node $n \in \mathcal{N}_t$ has a feature vector $\boldsymbol{x}_{n,t} \in \mathbb{R}^d$, capturing node-dependent information such as power limits and prices.

An EV node represents the current battery state and the remaining time until departure. A charging station node captures its physical charging and discharging power limits. A transformer node encodes the available feeder capacity that constrains the total power drawn by downstream chargers. A system-level node aggregates global context, including time-of-day information, electricity prices, and recent aggregate power consumption.

The action space $\mathbf{a}_t \in \mathcal{A}$ is represented by a dynamic graph $\mathcal{G}_t^{\mathbf{a}} = (\mathcal{N}_t^{\mathbf{a}}, \mathcal{E}_t^{\mathbf{a}})$, where nodes $\mathcal{N}_t^{\mathbf{a}}$ correspond to the decision variables of the optimization problem (e.g., EVs). Each node $n \in \mathcal{N}_t^{\mathbf{a}}$ represents a single action $a_{i,t} \in \mathbf{a}_t$, scaled by the corresponding charger's maximum power limit. For charging, $a_{i,t} \in [0, 1]$, and for discharging, $a_{i,t} \in [-1, 0)$. The transition function $\mathcal{P}(s_{t+1} \mid s_t, \mathbf{a}_t)$ accounts for uncertainties in EV arrivals, departures, energy demands, and grid fluctuations. Finally, the reward function follows the multi-objective formulation in Eq. (4) and is defined per timestep as $R(s_t, \mathbf{a}_t) = f_1 - 100 f_2 - 10 f_3$. It promotes low charging costs while penalizing violations of aggregated power limits and unmet energy requirements at EV departure. The first two terms capture immediate economic costs and network constraint violations, whereas the third term provides a sparse signal at departure time, introducing long-term temporal dependencies. The weighting coefficients were selected empirically to preserve the optimal solution of the underlying MIP while ensuring stable and effective learning.

3. GNN-based decision transformer

The training pipeline of GNN-DT, illustrated in Fig. 1, consists of four steps. First, EV charging scenarios are generated using stochastic processes for arrival, departure, and pricing. Second, each scenario is solved using multiple charging strategies, including heuristic, business-as-usual, and optimal solvers, to generate state–action trajectories. Third, the resulting trajectories are aggregated into an offline dataset. Finally, the GNN-DT model is trained in a supervised manner to predict charging actions from sequences of past states, actions, and returns-to-go. The model architecture is detailed in Fig. 2.

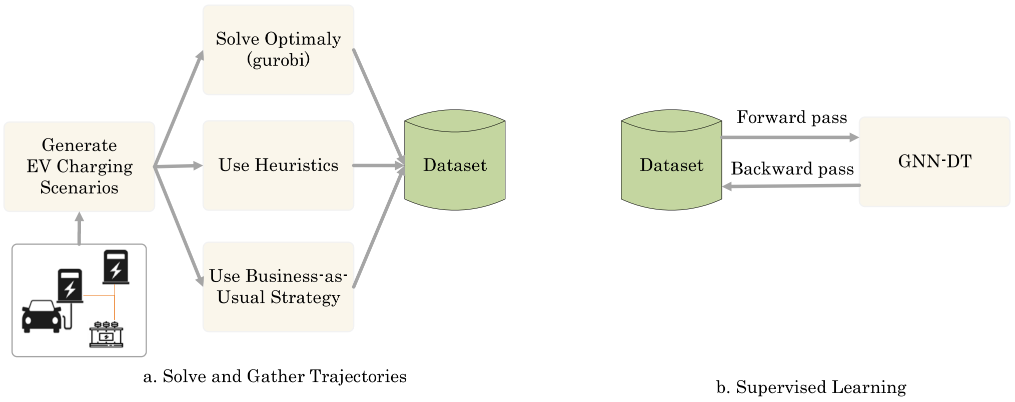
3.1. Sequence embeddings

In GNN-DT, each input "modality" is processed by a specialized embedding network. The state graph passes through the *State Embedder*, the action through the *Action Embedder*, and the return-to-go value through a simple Multi-Layer Perceptron (MLP). Compared to standard MLP embedders, GNNs provide embeddings for states and actions invariant to the number of nodes by capturing the graph structure. This design makes GNN-DT more sample-efficient during training and better at generalizing to unseen environments.

In detail, the *State Embedder* consists of L consecutive Graph Convolutional Network (GCN) [23] layers, which aggregate information from neighboring nodes as follows:

$$\boldsymbol{x}_t^{(l+1)} = \sigma\Big(D^{-1/2} A_t D^{-1/2} \boldsymbol{x}_t^{(l)} W^{(l)} \Big), \tag{12}$$

where $\boldsymbol{x}_t^{(l)} \in \mathbb{R}^{N_t \times F_l}$ denotes the node embeddings at layer l with N_t number of nodes, $W^{(l)} \in \mathbb{R}^{F_l \times F_{l+1}}$ are trainable weights, $\sigma(\cdot)$ is a nonlinear activation (ReLU), A_t is the adjacency matrix of the state graph $\mathcal{G}_t$, and D is the degree matrix for normalization. After the final layer, a mean-pooling operation produces a fixed-size state embedding: $\tilde{\boldsymbol{s}}_t = \frac{1}{|\mathcal{N}_t|} \sum_{n \in \mathcal{N}_t} \boldsymbol{x}_{n,t}^{(L)}$, where $\boldsymbol{x}_n^{(L)}$ is the embedding of node n at the Lth layer. This pooling step ensures that the state embedding is invariant to the number of nodes in the graph, enabling the architecture to scale with any number of EVs or chargers. Similarly, the *Action Embedder* processes the action graph $\mathcal{G}_t^{\mathbf{a}} = (\mathcal{N}_t^{\mathbf{a}}, \mathcal{E}_t^{\mathbf{a}})$ through C GCN layers followed by mean pooling, producing the action embedding $\tilde{\mathbf{a}}_t$. All embedding vectors (states, actions, or the return-to-go value) have the same dimensions. This design leverages the dynamic and invariant nature of GCN-based embeddings, allowing the DT to handle variable-sized graphs.



a. Solve and Gather Trajectories

b. Supervised Learning

Fig. 1. Overview of the proposed training pipeline. Initially, random EV charging scenarios are generated and are solved using business-as-usual and heuristic strategies employed by charge point operators. The problems are also solved optimally using solvers, such as Gurobi. After the dataset is generated, the GNN-DT model is trained in a supervised learning manner.

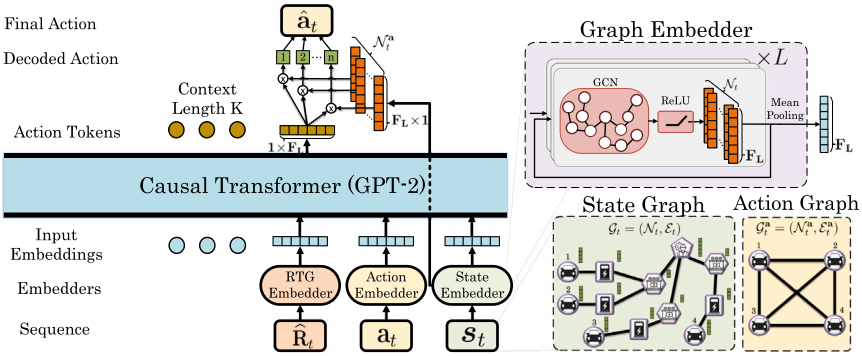


Fig. 2. Overview of the GNN-DT architecture. The input sequence, comprising return-to-go, action, and state, is processed through specialized embedding modules. The action graph $\mathcal{G}_t^{\mathbf{a}} = (\mathcal{N}_t^{\mathbf{a}}, \mathcal{E}_t^{\mathbf{a}})$, with nodes $\mathcal{N}_t^{\mathbf{a}} \subset \mathcal{N}_t$, and the state graph $\mathcal{G}_t = (\mathcal{N}_t, \mathcal{E}_t)$ are encoded using GNN-based embedders to produce embeddings of dimension F_L. These embeddings serve as inputs to a GPT-2–based causal transformer, which predicts the next action token. The predicted action token acts as a decoder, generating actions by multiplying with specific GNN state node embeddings.

3.2. Decoding actions

Once the embedding sequence of length K is constructed,[2] it is passed through the causal transformer GPT-2 to produce a fixed-size output vector $\boldsymbol{y}_t \in \mathbb{R}^{F_L}$ for each step. Because DT architectures inherently generate outputs of fixed dimensions, an additional mechanism is required to manage dynamic action spaces. To address this, GNN-DT implements a residual connection that merges the final GCN layer embeddings $\boldsymbol{x}_t^{(L)}$ with the transformer output $\boldsymbol{y}_t$ for every step of the sequence. Specifically, for each node $n \in \mathcal{N}_t^{\mathbf{a}}$, its corresponding state embedding $\boldsymbol{x}_{n,t}^{(L)} \in \mathbb{R}^{1\times F_L}$ is retrieved and is multiplied by the transformer output token $\boldsymbol{y}_t \in \mathbb{R}^{1\times F_L}$, yielding the final action for node n: $\hat{a}_{n,t} = \boldsymbol{y}_t^{\top} \cdot \boldsymbol{x}_{n,t}^{(L)}$. By repeating for every step t and every node $n \in \mathcal{N}_t^{\mathbf{a}}$ the final action vector $\hat{\mathbf{a}}_t$ is generated. This design allows the model to maintain a fixed-size output from the DT while dynamically adapting to any number of nodes (and hence actions). It effectively combines the high-level context learned by the transformer with the node-specific state information captured by the GNN, enabling robust, scalable decision-making even as the graph structure changes.

3.3. Action masking and loss function

The proposed GNN-DT model is trained via supervised learning using an offline trajectory dataset [19], similarly to offline RL. Specifically, the GPT-2 model is initialized with its default pre-trained weights, which are subsequently fine-tuned end-to-end for the EV charging optimization task. In GNN-DT, the learning of infeasible actions, such as charging an unavailable EV, is avoided through action masking. At each time step t, a mask vector $\mathbf{m}_t$, which has the same dimension as $\mathbf{a}_t$, is generated with zeros marking invalid actions and ones marking valid actions. For example, an action is invalid when the $a_{i,t} \neq 0$ and no EV is connected at charger i. The mean squared error between the predicted actions $\hat{\mathbf{a}}_t$ and ground-truth actions $\mathbf{a}_t$ from expert or offline trajectories is employed as the loss function. For a window of length K ending at time t, training loss is defined as:

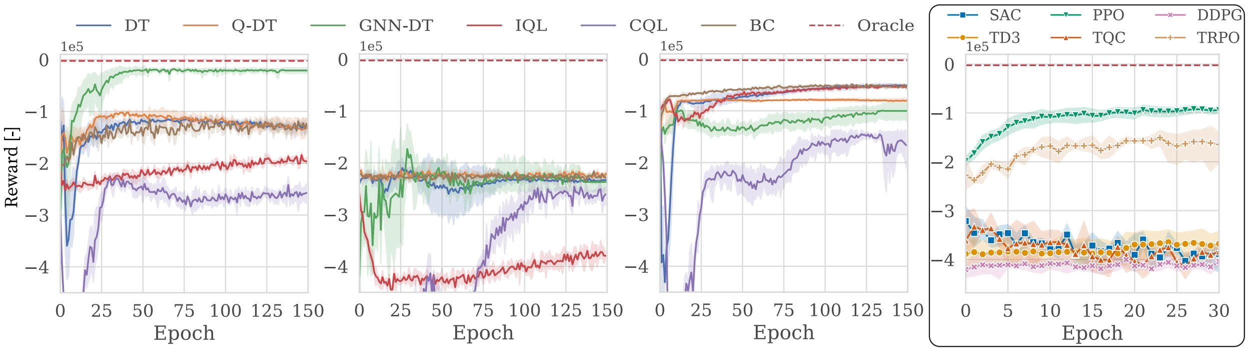
$$\mathcal{L} = \frac{1}{K}\sum_{\tau=t-K}^{t} \left\|(\hat{\mathbf{a}}_\tau - \mathbf{a}_\tau)\circ \mathbf{m}_\tau\right\|^2. \tag{13}$$

By incorporating the mask into the loss calculation (elementwise multiplication), a focus solely on valid actions is enforced, thereby preserving meaningful gradient updates.

4. Experimental setup

The dataset generation and the evaluation experiments are conducted using the EV2Gym simulator [24], which leverages real-world data distributions, including EV arrivals, EV specifications, electricity prices, etc. This setup ensures a realistic environment where the state and action spaces accurately reflect real charging stations' operational complexity. A scenario with 25 chargers is chosen, allowing up to 25 EVs to be connected simultaneously. In this configuration, the action vector has up to 25 variables (one per EV), while the state vector contains around 150 variables describing EV statuses, charger

[2] During inference the action ($\mathbf{a}_t$) and RTG ($\hat{\boldsymbol{R}}_t$) of the last step t are filled with zeros as they are not known.



a) Optimal Dataset (K = 10) b) Random Dataset (K = 10) c) BaU Dataset (K = 10) d) Online RL

Fig. 3. Training performance comparison for online and offline RL algorithms averaged over 5 random seeds. The line shows the mean, while the colored outline shows the standard deviation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 3
Algorithm hyperparameters for small- and large-scale settings.

Hyperparameter	Small scale	Large scale
Batch size	128	64
Learning rate	10^{-4}	10^{-4}
Weight decay	10^{-4}	10^{-4}
Steps per iteration	1000	3000
Decoder layers	3	3
Attention heads	4	4
Embedding dimension	128	256
GNN embedder feat. dim.	16	16
GNN hidden dimension	32	64
GCN layers	3	3
Epochs	250	400
CPU memory (GB)	8	40
Time limit (h)	10	46

conditions, power transformer constraints, and broader environmental factors. Consequently, the resulting optimization problem is in the moderate-to-large scale range, reflecting the key complexities of real-world EV charging. Each training process is repeated 10 times with random seeds to ensure statistically robust findings. All reported rewards represent the average performance over f50 evaluation scenarios, each featuring different configurations (electricity prices, EV behavior, power limits, etc.). Training was carried out on an NVIDIA A10 GPU paired with 11 CPU cores and 80 GB of RAM, using the AdamW optimizer and a LambdaLR scheduler. Baseline RL agents converged in 2–5 h, while the proposed GNN-DT required up to 10 h of training. Default hyperparameters were used for all baseline RL methods. Table 3 lists the full set of hyperparameters employed to train the DTs.

4.1. Dataset generation

Offline RL algorithms, including DTs, can learn policies from trajectories without the need for online interaction with the environment. Consequently, the quality of the gathered training trajectories has a substantial impact on the learning process. In this work, three distinct strategies were used to generate trajectories:

- **Random Actions**: Uniformly sampled actions in the range $[-1, 1]$ were applied to the simulator.
- **Business-as-Usual (BaU)**: A Round Robin charging policy commonly employed by CPOs, which sequentially allocates charging power among EVs to balance fairness and efficiency.
- **Optimal Policies**: Optimal solutions derived from solving offline the mathematical problem described in Section 2.2 for randomly generated scenarios.

Each trajectory consists of 300 state–action-reward-action mask tuples, with each timestep representing a 15-minute interval, resulting in a total of three simulated days. This combination of random, typical, and expert data provides a comprehensive basis for evaluating how GNN-DT learns from diverse offline trajectories.

5. Experiments

In this section, a comprehensive set of experiments is presented to evaluate the proposed method's performance, both during training and under varied test conditions. Different dataset types and sample sizes are examined to determine their impact on learning efficiency and convergence.

5.1. Training performance

Fig. 3 compares the proposed GNN-DT against multiple baselines, including the classic DT [19] and Q-DT [20], which both rely on flattened state representations due to their inability to directly process graph-structured data. In these baseline methods, empty chargers and unavailable actions are replaced by zeros, so the action vector is always the same size. Several well-known online RL algorithms from the Stable-Baselines-3 [25] framework are evaluated, such as SAC, DDPG, Twin Delayed DDPG (TD3), Trust Region Policy Optimization (TRPO), PPO, and Truncated Quantile Critics (TQC). Also, offline RL algorithms from D3RLPY [26], namely Implicit Q-Learning (IQL), Conservative Q-Learning (CQL), and BC, are also included. The offline RL algorithms (IQL, CQL, BC, DT, Q-DT, and GNN-DT) are trained on three datasets (*Optimal*, *Random*, and *BaU*), each comprising 10.000 trajectories. A red dotted line marks the optimal reward, which represents the experimental maximum achievable reward obtained by solving the deterministic MIP knowing the future (*Oracle*) defined in Eq. (4). This oracle reward serves as an upper bound and helps contextualize the relative performance of each method. It is important to note that the EV smart charging problem requires real-time (1–5 min intervals) optimization solutions for large-scale, highly stochastic scenarios, where metaheuristic algorithms, stochastic optimization, and MPC methods fail due to computational constraints. By contrast, once trained (over 2–24 h), RL agents can deliver real-time charging schedules on an ordinary computer in milliseconds.

In Figs. 3.a–c, the DT-based approaches use a context length $K = 10$. As expected, the *Optimal* dataset provides the highest-quality information, enabling GNN-DT to converge rapidly toward near-oracle performance, while classic DT, Q-DT, and the other offline RL algorithms lag far behind, showcasing GNN-DT's improved sample efficiency. With the *Random* dataset, the limited quality of data leads all methods to plateau at lower reward values, although GNN-DT still surpasses the other baselines. An intriguing behavior is observed with the *BaU* dataset, where classic DT, BC, and IQL converge at rewards exceeding those of GNN-DT. In contrast, the online RL algorithms displayed in Fig.

Table 4
Comparison of maximum episode rewards (×10^5) for baselines and GNN-DT across datasets and context lengths (K) over 10 random seeds. Bold indicates the highest value within each dataset and K category.

Dataset	Avg. training dataset reward	K = 2			K = 10		
		DT	Q-DT	GNN-DT	DT	Q-DT	GNN-DT
Random 100	−2.37 ± 0.39	−1.91	−1.97	**−0.82**	−2.12	−2.09	−1.16
Random 1000		−1.93	−2.04	−0.86	−2.11	−2.01	−1.18
Random 10000		−1.76	−2.04	−1.25	−1.81	−1.98	**−0.98**
BaU 100	−0.67 ± 0.07	−0.79	−0.74	**−0.59**	−0.79	−0.72	−0.56
BaU 1000		−0.71	−0.66	−0.65	−0.64	−0.71	−0.57
BaU 10000		−0.69	−0.66	−0.66	**−0.44**	−0.74	−0.53
Optimal 100	−0.01 ± 0.01	−0.67	−0.91	−0.15	−1.12	−0.90	−0.14
Optimal 1000		−0.63	−0.67	−0.10	−0.87	−0.86	−0.09
Optimal 10000		−0.63	−0.80	**−0.04**	−0.72	−0.90	**−0.07**

3.d struggle to achieve comparable improvements, suggesting that pure online exploration is insufficient for solving this complex EV charging optimization problem with sparse rewards. In the rest of this section, the online and offline RL baselines are omitted, as their performance is substantially inferior to that of DT, Q-DT, and GNN-DT.

5.2. Dataset impact

In Table 4, the maximum episode reward is compared for small, medium, and large datasets (100, 1.000, and 10.000 trajectories), under two different context lengths ($K = 2$ and $K = 10$), and 5 random seeds. The left side of Table 4 reports the dataset type, the number of trajectories, and the average reward in each dataset. All baselines achieve performance above the *Random* dataset's average reward. However, only GNN-DT consistently approaches the *Optimal* dataset's performance, reaching as close as -0.04×10^5 compared to the -0.01×10^5 optimal reward. This advantage becomes especially evident at the largest dataset size (10.000 trajectories), highlighting the benefits of the graph-based embedding layer. Overall, GNN-DT outperforms the baselines across all datasets and both context lengths, with the single exception of the *BaU* dataset at $K = 10$. Interestingly, a larger context window does not always translate into higher rewards, potentially due to the problem setting. Similarly, the dataset size appears to have minimal impact on Q-DT, whereas DT and GNN-DT generally improve with more trajectories. These findings underscore that both the quality and quantity of offline data, coupled with the GNN-DT architecture, are key to achieving superior performance.

5.3. Enhancing training datasets

The previous section highlighted that the quality of trajectories in the training dataset is the most influential factor for achieving high performance. In this section, the potential of creating new datasets is explored by mixing existing ones can further improve performance. The *Optimal* and *Random* datasets are combined in different proportions, as summarized in Table 5. A noteworthy result is that supplementing the *Optimal* dataset with “less useful” (*Random*) trajectories consistently boosts performance. In particular, GNN-DT with $K = 10$, trained on a mix of 250 *Optimal* and 750 *Random* trajectories, achieves near-oracle results, deviating by only -0.001×10^5 from the optimal reward. A similar trend emerges when blending *BaU* and *Random* datasets shown in Table 6. While the BaU dataset alone performs worse than the Optimal dataset, mixing it with Random data still yields improvements, with the 75% BaU and 25% Random combination showing the best results. Overall, these findings indicate that carefully integrating high- and lower-quality data can enhance policy learning beyond what purely *Optimal* or purely *Random* datasets can provide.

Table 5
Maximum reward of GNN-DT trained on merged *Optimal* and *Random* datasets for $K = 2$ and $K = 10$. Performance improves despite lower average training rewards, highlighting the importance of dataset diversity. Highest rewards per K are highlighted with **bold**.

Dataset	Total Traj.	Avg. dataset reward	GNN-DT reward (×10^5)	
			K = 2	K = 10
Random (Rnd.) 100%	1000	−2.37 ± 0.39	−0.863	−1.187
Opt. 25% + Rnd. 75%	1000	−1.78 ± 1.07	−0.045	**−0.020**
Opt. 50% + Rnd. 50%	1000	−1.18 ± 1.19	**−0.021**	−0.040
Opt. 75% + Rnd. 25%	1000	−0.60 ± 1.03	−0.073	−0.057
Optimal (Opt.) 100%	1000	−0.01 ± 0.01	−0.108	−0.099

Table 6
Maximum reward of GNN-DT trained on merged BaU-Random datasets for $K = 2$ and $K = 10$. The bold indicates the training dataset with the highest evaluation reward.

Dataset	Total Traj.	Avg. dataset reward	GNN-DT reward (×10^5)	
			K = 2	k = 10
Random (Rnd.) 100%	1000	−2.37 ± 0.39	−0.863	−1.187
BaU 25% + Rnd. 75%	1000	−1.93 ± 0.80	−0.578	−0.461
BaU 50% + Rnd. 50%	1000	−1.51 ± 0.87	−0.665	**−0.447**
BaU 75% + Rnd. 25%	1000	−1.09 ± 0.76	**−0.421**	−0.471
BaU 100%	1000	−0.01 ± 0.01	−0.654	−0.572

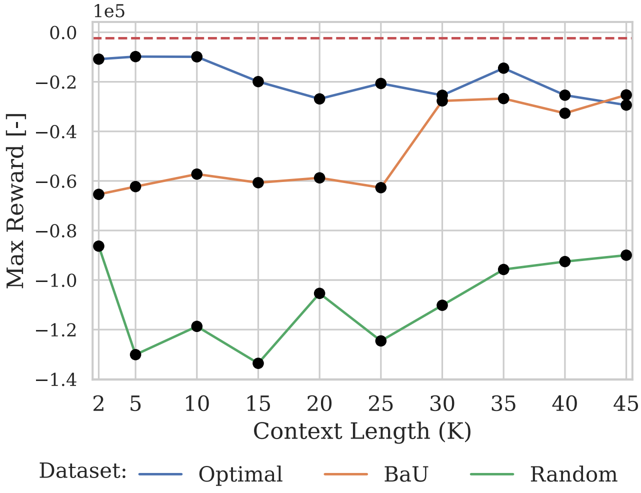


Fig. 4. GNN-DT performance for larger context lengths (K).

Table 7
Average reward trained over 5 runs with different seeds for *Optimal* and *Mixed* datasets.

DT	State GNN	Action GNN	Res. Con.	Action mask	Optimal ($\times 10^5$)	Mixed ($\times 10^5$)
✓	✗	✗	✗	✗	-0.69 ± 0.03	-0.95 ± 0.39
✓	GCN	✗	✗	✗	-0.71 ± 0.02	-0.77 ± 0.17
✓	GCN	✗	✓	✗	-0.18 ± 0.03	-0.16 ± 0.07
✓	GCN	GCN	✓	✗	-0.11 ± 0.03	-0.12 ± 0.04
✓	GAT	GAT	✓	✓	-0.14 ± 0.07	-0.15 ± 0.06
✓	GCN	GCN	✓	✓	$\mathbf{-0.09 \pm 0.02}$	$\mathbf{-0.10 \pm 0.04}$

5.4. *Impact of larger context lengths (K)*

Fig. 4 demonstrates that the context length K plays a key role in the performance of GNN-DT, with diminishing returns beyond a certain point. For high-quality datasets like *Optimal*, moderate context lengths ($K = 5$ to $K = 10$) yield the best results, while larger K values do not improve performance significantly. For suboptimal datasets like *BaU* and *Random*, the performance is lower overall, and longer context lengths seem to offer meaningful improvements, particularly when using the *BaU* dataset. Thus, selecting an appropriate context length is crucial for achieving better performance, while the quality of the dataset remains the most influential factor.

5.5. *Component ablation study*

To better understand the contribution of each architectural component, an ablation study is conducted by systematically removing or replacing elements of the model. The model is then trained with the *Optimal* and *Mixed* (Opt.25% +Rand.75%). The results in Table 7 reveal that neither a plain DT nor a DT augmented solely with a state-GNN submodule achieves competitive performance. Notably, adding the residual connection atop the state-GNN leads to a significant improvement, from -0.77×10^5 to -0.16×10^5 on the *Mixed* dataset, demonstrating its importance for effective credit assignment over dynamic inputs. Removing action masking or replacing the GCN module with a Graph Attention Network (GAT) [27] similarly degrades performance, indicating that each component provides distinct and complementary benefits. Ultimately, only the full GNN-DT architecture achieves strong performance across both the *Mixed* and *Optimal* datasets.

5.6. *Average results of EV charging*

Table 8 shows a comparison of key EV charging metrics for the 25-station problem after 100 evaluations, including heuristic algorithms, Charge As Fast as Possible (CAFAP) and BaU, and DT variants with the optimal solution, which assumes future knowledge.

The performance of the proposed algorithms was assessed using several evaluation metrics. For example, user satisfaction [%] captures the extent to which the state of charge at departure (e_{j,t^d}) of each electric vehicle $j \in \mathcal{J}$ meets its target e_j^*, thus defined as:

$$\text{User Satisfaction [\%]} = \frac{1}{|\mathcal{J}|} \sum_{j \in \mathcal{J}} \left(\frac{e_{j,t^d}{}^{*}}{e_j} \right) \cdot 100\%. \tag{14}$$

Energy charged [kWh] was measured as the total amount of energy delivered to the vehicles during the charging sessions, while energy discharged [kWh] was quantified as the energy returned from vehicles to the grid. Power violations [kW] were tracked to identify instances in which operational limits were exceeded, ensuring system feasibility. Finally, the overall charging cost [€] was evaluated by accounting for the time-varying electricity prices during charging and discharging periods, thus reflecting the economic performance of the strategy.

GNN-DT shows remarkable performance, achieving a close approximation to the optimal solution, particularly in user satisfaction (99.3% ± 0.03%) and power violation (21.7 ± 22.8 kW). It outperforms both BaU and DT variants in terms of energy discharged, power violation, and costs. Notably, GNN-DT performs well even compared to Q-DT, while maintaining competitive execution time, albeit slightly slower than the simpler models. The results underscore the effectiveness of GNN-DT in managing complex EV charging tasks, demonstrating its potential for real-world applications where future knowledge is not available.

5.7. *Illustrative example of EV charging*

After the model is trained, the behavior of the best baseline models trained (DT, Q-DT, GNN-DT) is compared against the heuristic BaU and mathematical optimization algorithm in an EV charging scenario. Fig. 5(a) presents the SoC progress for three EVs connected one after the other to a single charger throughout the simulation, while Fig. 5(b) shows the actions of all chargers taken by each algorithm. At the beginning of the simulation, EVs arrive at the charging station with unknown initial SoCs. Upon connection, they communicate their departure times and desired SoC levels to the CPO. Leveraging this information, along with real-time electricity price signals and power constraints, each algorithm determines optimal charging and discharging actions.

In Fig. 5(a), the heuristic BaU algorithm consistently overcharges the EVs, often exceeding the desired SoC levels. In contrast, both DT and Q-DT fail to satisfy the desired SoC. Conversely, GNN-DT successfully achieves the desired SoC for all EVs, closely mirroring the behavior of the optimal algorithm. This demonstrates GNN-DT's ability to precisely control charging based on dynamic state information. Fig. 5(b) provides further insights into the actions taken by each algorithm. The optimal solution primarily employs maximum charging or discharging power, since it knows the future. In comparison, GNN-DT exhibits a more refined approach, modulating charging power within a range of −6 to 11 kW. Baseline DT and Q-DT display a narrower range of actions, limiting their ability to optimize the charging schedules and adapt to varying conditions. These results underscore the superior capability of GNN-DT in managing the complexities of EV charging dynamics.

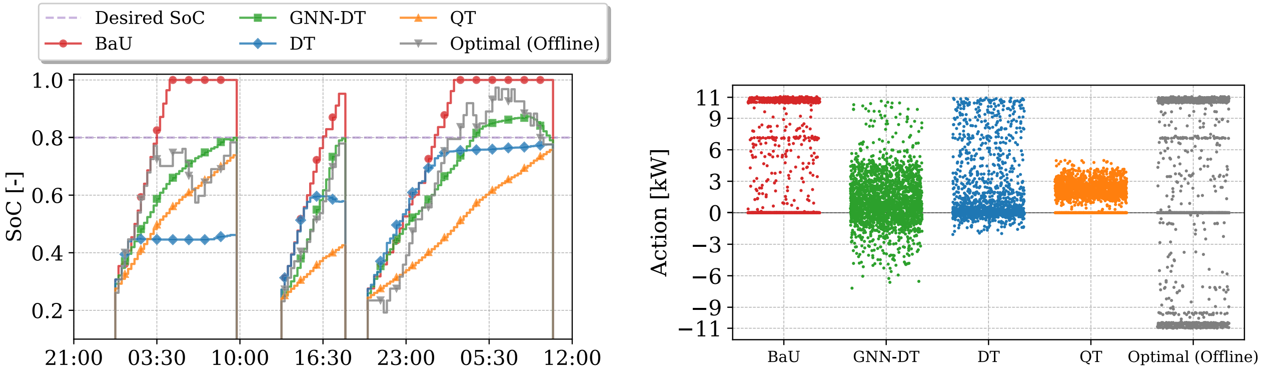
5.8. *Generalization and scalability analysis*

Evaluating the generalization of RL models across varying state transition probabilities is crucial for ensuring consistent performance under diverse conditions [28]. To evaluate the generalization capabilities of GNN-DT, three additional environments with different state transition probabilities are designed. The key environment variables that directly impact the state transition dynamics are visualized in Fig. 6. In detail, Fig. 6a–d presents the probability distributions of EV arrival time, departure time, duration of stay, and state of SoC at arrival across four scenarios: the original training environment and environments with small, medium, and extreme variations. These plots help quantify the extent of variation in each case. Additionally, Fig. 6e illustrates the temporal distribution of the power limit in each scenario, providing further insight into the differences in environment configuration.

In Fig. 7(a), the generalization capabilities of GNN-DT and other baselines are assessed in environments with small, medium, and extreme variations in state transition probabilities. While the baseline methods experience significant performance drops as the evaluation environment deviates from the training setting, GNN-DT maintains strong performance across all scenarios. This highlights the critical role

Table 8
Comparison of key EV charging metrics for the 25-station problem after 100 evaluation scenarios, for heuristic algorithms (CAFAP & BaU) and DT variants with the optimal solution, which assumes future knowledge.

Algorithm	Energy charged [MWh]	Energy discharged [MWh]	User satisfaction [%]	Power violation [kW]	Costs [€]	Reward [-10^5]	Exec. time [sec/step]
CAFAP	1.3 ± 0.2	0.00 ± 0.00	100.0 ± 0.0	1289.2 ± 261.8	−277 ± 165	−1.974 ± 0.283	0.001
BaU	1.3 ± 0.2	0.00 ± 0.00	99.9 ± 0.2	10.5 ± 9.4	−255 ± 156	−0.679 ± 0.067	0.001
DT	0.9 ± 0.1	0.03 ± 0.01	94.4 ± 1.6	58.7 ± 28.3	−173 ± 104	−0.462 ± 0.093	0.006
Q-DT	1.0 ± 0.1	0.00 ± 0.00	93.6 ± 2.1	20.1 ± 21.4	−187 ± 113	−0.665 ± 0.135	0.010
GNN-DT (Ours)	0.9 ± 0.1	0.19 ± 0.03	99.3 ± 0.2	21.7 ± 22.8	−142 ± 89	−0.027 ± 0.023	0.023
Optimal (Offline)	1.9 ± 0.2	1.08 ± 0.19	99.1 ± 0.2	2.0 ± 4.6	−119 ± 84	−0.020 ± 0.015	–



(a) EVs' battery level progression over time.

(b) Charger Actions.

Fig. 5. Comparison of smart charging algorithms for a single simulation day.

of GNN-based embeddings in improving model robustness and generalization. A key advantage of the GNN-DT architecture, not present in classic DTs, is its invariance to problem size, i.e., the same RL agent can be applied to both smaller and larger-scale environments. Fig. 7(b) illustrates the scalability and generalization performance of GNN-DT compared to the BaU algorithm and the Optimal policy. GNN-DT, trained on a 25-charger setup, was tested in environments with 5, 50, 75, and 100 chargers. GNN-DT's performance predictably declines at larger scales, since it wasn't trained on these problem instances. Nevertheless, GNN-DT still outperforms the BaU heuristic, demonstrating robustness to problem-size variation. Training GNN-DT on a mix of EV charger numbers could potentially further improve its adaptability.

The scalability and effectiveness of GNN-DT were tested when trained on a significantly larger optimization problem involving 250 charging stations. In this scenario, the model must handle up to 250 action variables per step and over 1000 state variables, which include critical information such as power limits and battery levels. The results presented in Table 9 demonstrate that GNN-DT shows promise for addressing more complex optimization tasks. However, the model requires a substantial increase in both the number of training trajectories and memory resources to maintain efficiency, highlighting a well-known limitation of DT-based approaches. Scaling the problem 10× roughly multiplies GPU memory usage, e.g. storing 3000 trajectories takes ≈2 GB for 25 chargers versus ≈20 GB for 250. While this can bottleneck large-scale training, parallelization and mini-batching mitigate it, and overall compute scales with the transformer's context length K (see Fig. 4), pointing to interesting directions for very large problem graphs as future work.

6. Discussion

The experimental results show that GNN-DT achieves strong performance across a range of EV charging scenarios, including unseen fleet sizes, network topologies, and stochastic variations in arrivals and prices. The ablation results confirm that permutation-equivariant graph embeddings and the residual decoding mechanism are both necessary to achieve these gains.

Table 9
Max. reward of GNN-DT in a large-scale EV charging optimization task with 250 chargers.

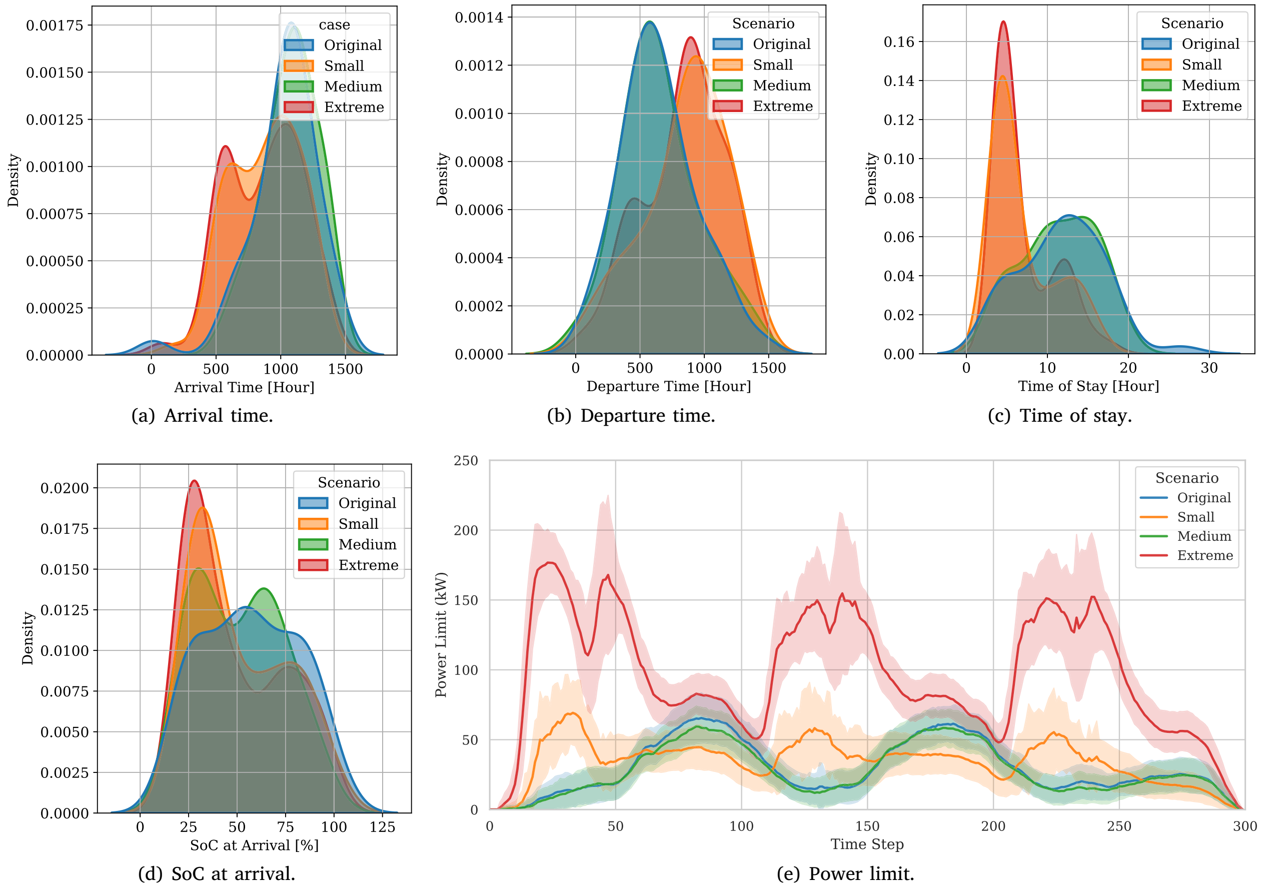
	Total trajectories	Avg. dataset reward	GNN-DT reward
Random	3000	−22.39 ± 1.49	−9.34
BaU	3000	−6.67 ± 0.32	−4.23
Optimal	3000	−0.08 ± 0.03	**−0.27**

At the same time, several limitations should be noted. First, GNN-DT incurs a higher training-time memory cost than conventional RL methods. The memory footprint scales with the number of graph nodes, sequence length, and batch size, which can become a bottleneck for very large charging networks. Second, the performance of GNN-DT depends on the quality and coverage of the offline dataset. Limited or biased datasets reduce robustness and increase sensitivity to distribution shift, highlighting the need for careful dataset construction. Third, although the model generalizes across moderate topology changes, performance degrades when applied to network structures that differ significantly from those seen during training, indicating sensitivity to large topology shifts.

Despite these limitations, inference remains fast and suitable for real-time deployment once training is completed. Addressing memory scaling, dataset efficiency, and robustness to extreme topology changes are important directions for future work.

7. Conclusions

This work demonstrates that offline sequence-based policies can achieve near-optimal performance for large-scale EV smart charging under realistic uncertainty. The proposed GNN-DT approach consistently outperforms online and offline RL baselines in terms of cost, constraint satisfaction, and user satisfaction, while remaining suitable for real-time deployment and generalizing across fleet sizes and network configurations. The results underscore the importance of integrating structured representations with high-quality offline data for informed



(a) Arrival time.

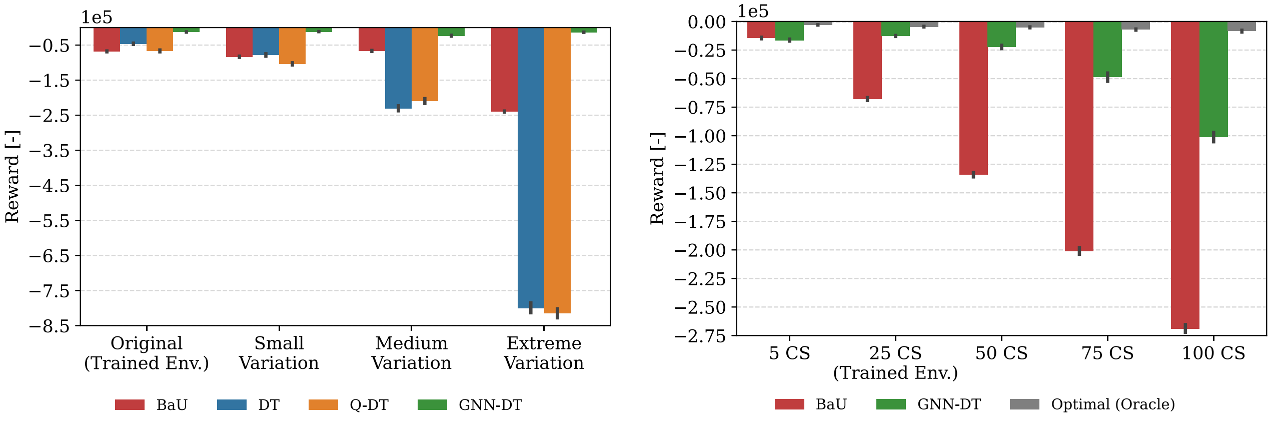
(b) Departure time.

(c) Time of stay.

(d) SoC at arrival.

(e) Power limit.

Fig. 6. Overview of the five key state transition variables across different scenarios: (a) arrival time, (b) departure time, (c) power limit, (d) state of charge upon arrival, and (e) time of stay.



(a) Unseen state transition probabilities.

(b) Evaluating on different size problems.

Fig. 7. Generalization performance of the proposed model, depicting the average rewards achieved across 100 randomly generated scenarios in previously unseen environments.

decision-making in complex energy systems. Future work will focus on improving memory efficiency and robustness to larger topology shifts.

CRediT authorship contribution statement

Stavros Orfanoudakis: Writing – review & editing, Writing – original draft, Visualization, Methodology, Conceptualization. **Nanda Kishor Panda:** Writing – review & editing, Writing – original draft, Conceptualization. **Peter Palensky:** Supervision, Funding acquisition. **Pedro P. Vergara:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization.

Declaration of competing interest

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

Acknowledgments

The study was partially funded by the DriVe2X research and innovation project from the European Commission with grant number 101056934. The authors acknowledge the use of computational resources of the DelftBlue supercomputer, provided by Delft High Performance Computing Centre (https://www.tudelft.nl/dhpc). This work used the Dutch national e-infrastructure with the support of the SURF Cooperative, using grant no. EINF-5716.

Data availability

Dataset and code is open sourced.

References

[1] Panda NK, Tindemans SH. Quantifying the aggregate flexibility of EV charging stations for dependable congestion management products: A dutch case study. 2024, arXiv:2403.13367.

[2] Minchala-Ávila C, Arévalo P, Ochoa-Correa D. A systematic review of model predictive control for robust and efficient energy management in electric vehicle integration and V2g applications. Modelling 2025;6(1). http://dx.doi.org/10.3390/modelling6010020.

[3] Diaz-Londono C, Orfanoudakis S, Vergara PP, Palensky P, Ruiz F, Gruosso G. Open source algorithms for maximizing V2G flexibility based on model predictive control. Electr Power Syst Res 2026;250:112082. http://dx.doi.org/10.1016/j.epsr.2025.112082.

[4] Tahmasebi M, Ghadiri A, Haghifam M, Miri-Larimi S. MPC-based approach for online coordination of EVs considering EV usage uncertainty. Int J Electr Power Energy Syst 2021;130:106931. http://dx.doi.org/10.1016/j.ijepes.2021.106931.

[5] Deshmukh S, Tariq H, Amir M, Iqbal A, Marzband M, Al-Wahedi AMAB. Impact assessment of electric vehicles integration and optimal charging schemes under uncertainty: A case study of Qatar. IEEE Access 2024;12:131350–71. http://dx.doi.org/10.1109/ACCESS.2024.3458410.

[6] Dengor I, Erdinc O, Yener B, Taşcıkaraoğlu A, Catalao JPS. Optimal energy management of EV parking lots under peak load reduction based DR programs considering uncertainty. IEEE Trans Sustain Energy 2019;10(3):1034–43. http://dx.doi.org/10.1109/TSTE.2018.2859186.

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

[8] Qiu D, Wang Y, Hua W, Strbac G. Reinforcement learning for electric vehicle applications in power systems:A critical review. Renew Sustain Energy Rev 2023;173:113052. http://dx.doi.org/10.1016/j.rser.2022.113052.

[9] Jin R, Zhou Y, Lu C, Song J. Deep reinforcement learning-based strategy for charging station participating in demand response. Appl Energy 2022;328:120140. http://dx.doi.org/10.1016/j.apenergy.2022.120140.

[10] Jin J, Xu Y. Optimal policy characterization enhanced actor-critic approach for electric vehicle charging scheduling in a power distribution network. IEEE Trans Smart Grid 2021;12(2):1416–28. http://dx.doi.org/10.1109/TSG.2020.3028470.

[11] Orfanoudakis S, Robu V, Salazar EM, Palensky P, Vergara PP. Scalable reinforcement learning for large-scale coordination of electric vehicles using graph neural networks. Commun Eng 2025;4(1):118. http://dx.doi.org/10.1038/s44172-025-00457-8.

[12] Salari A, Zeinali M, Marzband M. Model-free reinforcement learning-based energy management for plug-in electric vehicles in a cooperative multi-agent home microgrid with consideration of travel behavior. Energy 2024;288:129725. http://dx.doi.org/10.1016/j.energy.2023.129725.

[13] Kamrani AS, Dini A, Dagdougui H, Sheshyekani K. Multi-agent deep reinforcement learning with online and fair optimal dispatch of EV aggregators. Mach Learn Appl 2025;100620. http://dx.doi.org/10.1016/j.mlwa.2025.100620.

[14] Zhang S, Jia R, Pan H, Cao Y. A safe reinforcement learning-based charging strategy for electric vehicles in residential microgrid. Appl Energy 2023;348:121490. http://dx.doi.org/10.1016/j.apenergy.2023.121490.

[15] Wang Y, Wu J, He H, et al. Data-driven energy management for electric vehicles using offline reinforcement learning. Nat Commun 2025;16:2835. http://dx.doi.org/10.1038/s41467-025-58192-9.

[16] Jia R, Pan H, Zhang S, Hu Y. Charging scheduling strategy for electric vehicles in residential areas based on offline reinforcement learning. J Energy Storage 2024;103:114319. http://dx.doi.org/10.1016/j.est.2024.114319.

[17] Rossi F, Diaz-Londono C, Li Y, Zou C, Gruosso G. Smart electric vehicle charging algorithm to reduce the impact on power grids: A reinforcement learning based methodology. IEEE Open J Veh Technol 2025;6:1072–84. http://dx.doi.org/10.1109/OJVT.2025.3559237.

[18] Niu Z, He H. A data-driven solution for intelligent power allocation of connected hybrid electric vehicles inspired by offline deep reinforcement learning in V2X scenario. Appl Energy 2024;372:123861. http://dx.doi.org/10.1016/j.apenergy.2024.123861.

[19] Chen L, Lu K, Rajeswaran A, Lee K, Grover A, Laskin M, Abbeel P, Srinivas A, Mordatch I. Decision transformer: Reinforcement learning via sequence modeling. 2021, arXiv:2106.01345.

[20] Hu S, Fan Z, Huang C, Shen L, Zhang Y, Wang Y, Tao D. Q-value regularized transformer for offline reinforcement learning. In: Forty-first international conference on machine learning. 2024.

[21] Paster K, McIlraith SA, Ba J. You can't count on luck: why decision transformers and RvS fail in stochastic environments. In: Proceedings of the 36th international conference on neural information processing systems. NIPS '22, Red Hook, NY, USA: Curran Associates Inc.; 2024.

[22] Levine S, Kumar A, Tucker G, Fu J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. 2020, arXiv:2005.01643.

[23] Kipf TN, Welling M. Semi-supervised classification with graph convolutional networks. 2016, arXiv:1609.02907.

[24] Orfanoudakis S, Diaz-Londono C, Emre Yılmaz Y, Palensky P, Vergara PP. EV2gym: A flexible V2G simulator for EV smart charging research and benchmarking. IEEE Trans Intell Transp Syst 2025;26(2):2410–21.

[25] Raffin A, Hill A, Gleave A, Kanervisto A, Ernestus M, Dormann N. Stable-Baselines3: Reliable reinforcement learning implementations. J Mach Learn Res 2021;22(268):1–8.

[26] Seno T, Imai M. D3rlpy: An offline deep reinforcement learning library. J Mach Learn Res 2022;23(315):1–20.

[27] Veličković P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y. Graph attention networks. 2018, arXiv:1710.10903.

[28] Wang R, Foster DP, Kakade SM. What are the statistical limits of offline RL with linear function approximation?. 2020, arXiv:2010.11895.