Scalable Coordinated Charging of Large EV Fleet via Imitation Learning

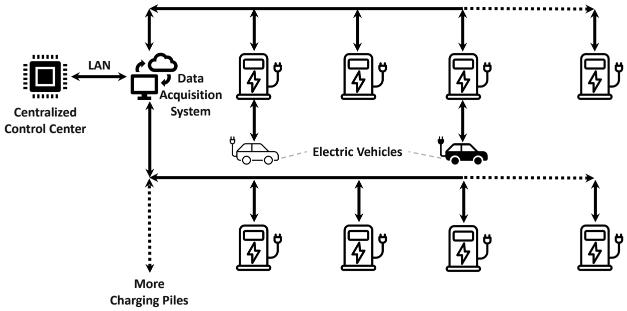

Fig. 1: Illustration of the proposed system

***Abstract*—**

***Index Terms*—**

I. Introduction

COORDINATED charging of Large EV Fleet via Imitation Learning.

II. System Modeling and Problem Formulation

As depicted in Fig. 1, the proposed coordinated charging system is structured around a large parking lot, which is equipped with many electric vehicle (EV) charging piles. Once the connection between the EV and the charging pile is established, a wired data communication link is formed [1], [2]. The EV can transmit information such as rated charging power, current state of charge (SoC), and battery capacity to the charging pile, while the charging pile will transmit the current charging power to the corresponding EV. In addition to the charging piles, the parking lot incorporates a data acquisition system (DAS), responsible for the regular collection of pertinent charging data. This data is then transmitted to the centralized control centre (CCC) via a local area network (LAN) [3]–[5]. The CCC employs the proposed coordinated charging strategy based on imitation learning and reinforced with safety post-processing measures to swiftly determine optimal charging actions for each charging pile. These decisions are communicated back to the DAS, prompting adjustments in power output at each charging pile. This continuous real-time loop ensures the efficiency of the charging operations.

A. System Modeling

Consider a parking lot with N charging piles, each indexed by $k \in \{1, 2, ..., N\}$. Since EVs continuously arrive at and depart from the parking lot, a discrete-time model is usually utilized to describe the dynamic behavior of the system to facilitate receding horizon optimization (RHC) [6]–[9]. Let each time slot occupy time of length $\Delta \mathrm{t}$ and the total number of time slots across the scheduling horizon be T. For convenience, the first time step of a scheduling horizon is always recorded as $t = 1$, which is particularly useful when integrating our algorithm into a receding horizon optimization framework [9], [10]. That is, we adopt a relative time system specific to each scheduling period in this study, and all other timestamps such as the arrival and departure time of an EV are shifted implicitly according to the relative time origin. After one time step passes, we start a new scheduling horizon, and the optimization for coordinated charging is redone in a RHC framework.

Consider an arbitrary scheduling horizon Let us consider hereafter an arbitrary index $k \in [1, N]$ and an arbitrary time step $t \in [1, T]$ without loss of generality. After an EV establishes a connection to charging pile k, it is denoted as EV$_k$. Note that EV$_k$ and charging pile k share identical parameters upon the connection. Each EV$_k$ is characterized by 7 parameters $\langle t_k^{\mathrm{a}}, t_k^{\mathrm{d}}, E_k, p_k^{\mathrm{rated}}, e_k, \mathrm{SoC}_k^{\mathrm{a}}, \mathrm{SoC}_k^{\mathrm{d}} \rangle$, corresponding to EV$_k$'s arrival time, departure time, battery capacity, rated charging power, charging efficiency, SoC at arrival, and expected SoC upon departure. In practice, t_k^{d} and $\mathrm{SoC}_k^{\mathrm{d}}$ are provided by the EV driver, e.g., through a mobile phone application [7], [11], and the other parameters are typically fetched automatically through communication with the EV's battery management system once the EV is plugged in [4], [9], [12]. In our convention, EV$_k$ is qualified for charging at t_k^{a} but cannot get charged at the departure time step t_k^{d}. Besides, it is assumed that EV$_k$ does not leave earlier before its provided departure time t_k^{d}.

Let $p_{t,k}$ denote the charging power of EV$_k$ at time t. The rated power of EV$_k$ is p_k^{rated}, commonly treated as the maximum charging power. We scale $p_{t,k}$ into the unit range $[0, 1]$ to facilitate subsequent algorithm design [6], which is termed the charging action $a_{t,k}$ and computed by

$$a_{t,k} = p_{t,k} / p_k^{\mathrm{rated}} \tag{1}$$

Evidently, EV$_k$ can only be charged when it is connected to charging pile k. The range of $a_{t,k}$ is thus

$$\begin{cases} a_{t,k} \in [0, 1], & t_k^{\mathrm{a}} \leq t < t_k^{\mathrm{d}} \\ a_{t,k} = 0, & \text{otherwise.} \end{cases} \tag{2}$$

To assess the charging cost, $m_{t,k}$ is defined as the electricity cost incurred by EV$_k$ at time t:

$$m_{t,k} = \rho_t p_{t,k} \Delta_{\mathrm{t}} = \rho_t a_{t,k} p_k^{\mathrm{rated}} \Delta_{\mathrm{t}} \tag{3}$$

where ρ_t represents the real-time electricity price at time t. One goal of coordinated charging is to reduce the total electricity bill C computed by

$$C = \sum_{k=1}^{N} \sum_{t=1}^{T} m_{t,k}. \tag{4}$$

Apart from reducing the charging cost, the equality of charging among customers is also worth considering. Roughly speaking, we want the charging of each EV to progress at a similar pace instead of charging one EV first for some intervals and then another. The SoC of EV$_k$ at time t is termed $\text{SoC}_{t,k} \in [0,1]$. If the EV keeps charging at the rated power, then the minium number of time steps still required to fulfill the expected SoC_k^{d} at departure is given by

$$r_{t,k} = \frac{E_k(\text{SoC}_k^{\text{d}} - \text{SoC}_{t,k})}{e_k p_k^{\text{rated}} \Delta\text{t}}. \tag{5}$$

Note here that we allow $r_{t,k}$ to be fractional, because its main purpose is to compute the charging urgency of EV$_k$, which is denoted by $g_{t,k}$ at time t and calculated as

$$g_{t,k} = \begin{cases} \frac{r_{t,k}}{t_k^{\text{d}} - t} & t_k^{\text{a}} \leq t < t_k^{\text{d}} \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

In Eq. (6) above, $(t_k^{\text{d}} - t)$ counts the remaining number of time steps within the expected during of stay of EV$_k$. Intuitively, the charging task becomes increasingly urgent as $g_{t,k}$ increases. A reasonable objective is to maintain the balancing of urgency among the EVs being served. That is, the charging pace of each EV should be similar such that the charging fairness is assured [5], [11].

A straightforward rule to maintain charging fairness is that, if an EV's charging urgency is higher, then we allocate it more power to charge provided that all constraints are satisfied. Formally, the proposed method will try to proportionally allocate charging action $a_{t,k} \in [0,1]$ according to the corresponding charging urgency $g_{t,k}$ for each EV$_k$ at time t [13]. Let $G_{t,k}$ represent the proportion of EV$_k$'s charging urgency $g_{t,k}$ over the total urgency of all EVs:

$$G_{t,k} = \frac{g_{t,k}}{\sum_{k=1}^{N} g_{t,k}} \tag{7}$$

Clearly, the range of the normalized charging urgency above is $G_{t,k} \in [0,1]$. Given a candidate set of charging actions $a_{t,k}, k \in [1,N]$ at time t, we compute the *fair* charging action of each EV according to proportion $G_{t,k}$ as follows:

$$a_{t,k}^{\text{fair}} = G_{t,k} \sum_{k=1}^{N} a_{t,k}. \tag{8}$$

Ideally, a fair charging scheme should attain $a_{t,k} = a_{t,k}^{\text{fair}}$ for every t and k, but the strict equality requirement is practically impossible due to a variety of constraints for coordinated charging. In addition, another objective of our charging strategy is to minimize the total electricity cost, which may conflict with the charging fairness. Thus, we try to minimize the difference between $a_{t,k}$ and $a_{t,k}^{\text{fair}}$ instead, which is quantified for the EV fleet by

$$D = \sum_{k=1}^{N} \sum_{t=1}^{T} (a_{t,k} - a_{t,k}^{\text{fair}})^2. \tag{9}$$

B. Optimization Problem Formulation

Coordinated charging of EVs is typically formulated as a constrained optimization problem. In this study, the objective is to minimize the charging cost An optimal charging scheduling is proposed to minimize the electricity cost C while maximizing the charging fairness for all connected EVs (or equivalently, minimizing the fairness discrimination D). Since the two objectives are conflicting in general, we combine them into a single one via weighted sum as follows

$$F = (1-\beta)C + \beta D, \tag{10}$$

where the weight coefficient $\beta \in [0,1]$ specifies the importance of each objective, whose value depends on the practical preference of electricity cost or charging fairness [11], [13]. Obviously, as β increases, the model prioritizes fairness over electricity cost.

There are primarily two constraints associated with the optimization problem. The first is the satisfaction of the EV's SoC demand, that is, to guarantee that the SoC of EV$_k$ at departure time matches the expected SoC SoC_k^{d} as specified previously by the driver when plugging in the EV. To formulate this constraint, we first describe the SoC dynamics during the stay period of an EV. Since the SoC keeps increasing even within a single time slot if it gets charged, it should be clarified further that $\text{SoC}_{t,k}$ refers to the SoC of EV$_k$ at the very beginning of the time step t. We then have

$$\text{SoC}_{t,k} = \begin{cases} \text{SoC}_k^{\text{a}}, & t = t_k^{\text{a}} \\ \text{SoC}_{t-1,k} + \frac{p_{t-1,k} e_k \Delta_{\text{t}}}{C_k}, & t_k^{\text{a}} < t \leq t_k^{\text{d}}. \end{cases} \tag{11}$$

The SoC demand constraint is thus enforced by

$$\text{SoC}_{t_k^{\text{d}},k} = \text{SoC}_k^{\text{d}}. \tag{12}$$

Additionally, the sum of all EV$_k$'s charging power $p_{t,k}$ cannot exceed the parking station's power capacity P_{max}:

$$\sum_{k=1}^{N} p_{t,k} \leq P_{\text{max}} \tag{13}$$

Now we can present the complete optimization problem called **P1** for coordinated charging towards both cost reduction and fairness improvement as follows:

$$(\textbf{P1}) \ \min_{\boldsymbol{a}} F \tag{14a}$$

$$\text{subject to (1) − (13).} \tag{14b}$$

The decision variables of **P1** are collected into a vector as

$$\boldsymbol{a} = [a_{t,k}],\ \forall t \in [1,T], k \in [1,N]. \tag{15}$$

Since the fairness related term in the objective function (14a) involves fractions and squares, **P1** forms a general nonlinear program (and more precisely, not a quadratic program) that is notoriously difficult to solve because of its excessive computation burden [10], [11], [13], [14]. Though some techniques such as piecewise linear approximation [14], [15], greedy search [9], and metaheuristic algorithms like particle swarm optimization (PSO) [11] have been proposed to handle such nonlinear programming (NLP) problems, they are not applicable here because of either the dependence on special problem properties [9], [10] or the poor scalability to a large number of EVs [11]. For instance, we have conducted a small-scale case study with 100 EVs, and two well-established NLP solvers, namely Bonmin and Couenne[1], both failed to solve **P1** to optimality even after 24 hours. However, the each time step for practical charging scheduling is usually of length $\Delta_{\text{t}} = 10$ min or 15 min [5], [9], [10], and the solving time is expected to be only a fraction of Δ_{t} to yield charging actions

[1] Please check the COIN-OR Foundation: https://github.com/coin-or.

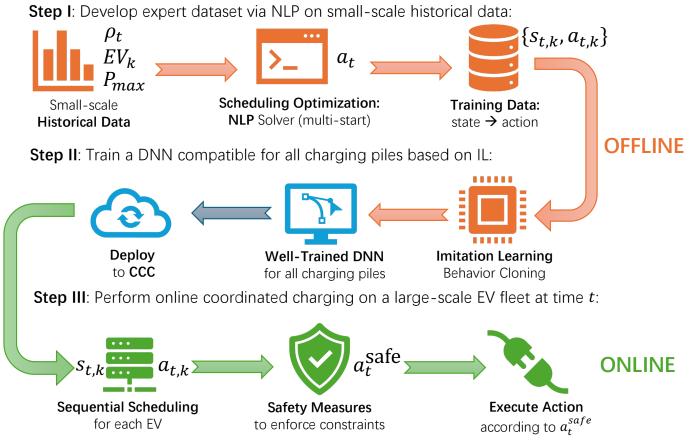
Step I: Develop expert dataset via NLP on small-scale historical data:
ρ_t
EV_k
P_{max}
Small-scale Historical Data
a_t
Scheduling Optimization: NLP Solver (multi-start)
$\{s_{t,k}, a_{t,k}\}$
Training Data: state → action
OFFLINE
Step II: Train a DNN compatible for all charging piles based on IL:
Deploy to CCC
Well-Trained DNN for all charging piles
Imitation Learning Behavior Cloning
Step III: Perform online coordinated charging on a large-scale EV fleet at time t:
$s_{t,k}$
$a_{t,k}$
a_t^{safe}
Sequential Scheduling for each EV
Safety Measures to enforce constraints
Execute Action according to a_t^{safe}
ONLINE

Fig. 2: Overview of the proposed framework

fast enough for receding horizon optimization. Obviously, this high-demanding requirement is far beyond the capacity of traditional numerical optimization algorithms like branch and bound [14] or metaheuristic search algorithms like PSO [11].

To sum up, the fundamental challenge of coordinated charging considering both cost and fairness is how to solve **P1** accurately and efficiently such that the optimization strategy is practical and performant even in the case of a large EV fleet, which is the main motivation of the current study. In the next section, we will present a novel solution based on imitation learning to overcome this challenge effectively.

III. Proposed Method

Solving large-scale **CO** requires significant computational resources, making it impractical for real-time decision-making scenarios. Therefore, we propose a novel approach based on imitation learning that effectively reduces the computational costs, enabling the real-time provision of near-optimal solutions even for ultra-large-scale EV fleet ($N \geq 500$).

A. Framework Overview

The framework of the proposed method is illustrated in Fig. 2 for a more concise exposition. It can be delineated into three principal phases. The first stage is expert data generation. An NLP solver is utilized to obtain an optimal solution of the historical data by solving **CO**. Upon getting the optimal solution, a re-simulation of the charging process is conducted. At each time step t, the corresponding state and action of EV_k are recorded according to the sequence of charging piles, thus forming an expert training dataset. For the second stage, the dataset is used for training an imitation learning (IL) agent, which is appproximated by a Deep Neural Network (DNN). This single IL agent will be responsible for making decisions for all charging piles. Phase one and two of the proposed method run completely offline. Finally, the well-trained DNN is deployed online within a large-scale parking lot equipped with charging facilities. Like the simulation in step one, the IL agent sequentially estimates the optimal charging action according to the real-time state of each EV_k. Note that for each time step t, after the IL agent completes the sequential decision-making for all EVs, the CCC will employ safety measures to adjust the decisions, ensuring all hard constraints about charging are met. Through these steps, the proposed method efficiently handles large-scale coordinated charging problems, generating near-optimal solutions while ensuring computational feasibility and practical applicability.

B. Expert Dataset Generation

One significant characteristic of the proposed approach is the excellent scalability, allowing for the flexible expansion of both the scale of EV fleets and sampling interval Δt. This feature endows the proposed method with the full advantage of optimization-based methods. To construct an expert training dataset, several small-scale EV charging scenarios will be generated. By learning from these small-scale scenarios, the proposed method can accommodate EV charging scheduling problems of any scale.

Given a charging scenario χ, the optimal scheduling $\boldsymbol{A}$ is obtained by solving **CO** on scenario χ. We need to extract the corresponding state $s_{t,k}$ for each action $a_{t,k}$ to form the expert dataset, which will be used for behaviour cloning later in imitation learning. The proposed method's principal attribute lies in its scalability. When formulating the state $s_{t,k}$ for EV_k, it is imperative to ensure its adaptability to EV fleets of varying magnitudes. In the context of an EV fleet comprising N vehicles, partitioning it into three distinct groups as follows:

1) Group_A encompasses EVs connected to pile_1 to pile_{k-1}, representing vehicles whose charging actions have already been determined.
2) Group_B constitutes a specialized subset containing solely EV connected to pile_k, for which the charging action $a_{t,k}$ is presently under determination.
3) Group_C encompasses EVs connected to pile_1 to pile_{k-1}, representing vehicles whose charging actions are pending decisions.

Consequently, irrespective of the EV fleet's scale, obtaining the state $s_{t,k}$ for EV_k entails abstracting it into three distinct groups. As illustrated in Figure 3, $s_{t,k}$ is composed of $s^\text{O}_{t,k}$, $s^\text{A}_{t,k}$, $s^\text{B}_{t,k}$ and $s^\text{C}_{t,k}$.

$s^\text{A}_{t,k}$, $s^\text{B}_{t,k}$ and $s^\text{C}_{t,k}$ represent states for the Group_A, Group_B and Group_C mentioned above, respectively. These groups can be regarded as three 'large' EVs, and they share some same attributes and have some unique attributes. Consider the same attributes first, given a specific Group_X where consists of N_X piles, the state $s^\text{X}_{t,k}$ always contains average battery capacity C_X, average rated charging power p^rated_X, weighted average expected state of charge $\text{SoC}^\text{d}_\text{X}$ when leaving, weighted average state of charge $\text{SoC}_{t,\text{X}}$, and average remain parking time $r_{t,\text{X}}$.

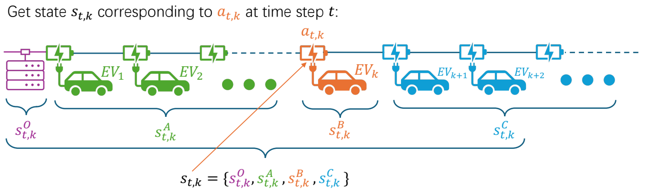
Get state $s_{t,k}$ corresponding to $a_{t,k}$ at time step t:
$a_{t,k}$
EV_1
EV_2
EV_k
EV_{k+1}
EV_{k+2}
$s^O_{t,k}$
$s^A_{t,k}$
$s^B_{t,k}$
$s^C_{t,k}$
$s_{t,k} = \{s^O_{t,k}, s^A_{t,k}, s^B_{t,k}, s^C_{t,k}\}$

Fig. 3: Illustration of the state $s_{t,k}$

There parameters are calculated by the following equations:

$$\begin{aligned} s^{\mathrm{X}}_{t,k} &= \{C_{\mathrm{X}}, p^{\text{rated}}_{\mathrm{X}}, \text{SoC}^{\text{d}}_{\mathrm{X}}, \text{SoC}_{t,\mathrm{X}}, r_{t,\mathrm{X}}\}, \quad \mathrm{X} \in \{\mathrm{A}, \mathrm{B}, \mathrm{C}\} \\ C_{\mathrm{X}} &= \frac{\sum_{k\in\mathrm{X}} C_k}{N_{\mathrm{X}}} \\ p^{\text{rated}}_{\mathrm{X}} &= \frac{\sum_{k\in\mathrm{X}} p^{\text{rated}}_k}{N_{\mathrm{X}}} \\ \text{SoC}^{\text{d}}_{\mathrm{X}} &= \frac{\sum_{k\in\mathrm{X}} \text{SoC}^{\text{d}}_k \cdot C_k}{\sum_{k\in\mathrm{X}} C_k} \\ \text{SoC}_{t,\mathrm{X}} &= \frac{\sum_{k\in\mathrm{X}} \text{SoC}_{t,k} \cdot C_k}{\sum_{k\in\mathrm{X}} C_k} \\ r_{t,\mathrm{X}} &= \sum_{k\in\mathrm{X}} (t^{\text{d}}_k - t) \end{aligned} \tag{16}$$

Please note that, aside from the aforementioned shared attributes, $\text{Group}_{\mathrm{A}}$, $\text{Group}_{\mathrm{B}}$ and $\text{Group}_{\mathrm{C}}$ also have unique attributes. For $\text{Group}_{\mathrm{A}}$, where the charging actions have already been determined, it is pivotal to get the proportion of total power consumption p^{total}_{A} in relation to the maximum power capacity of the charging station:

$$p^{\text{total}}_{\mathrm{A}} = \sum_{k\in\mathrm{A}} p_{t,k} \tag{17}$$

Similarly, for $\text{Group}_{\mathrm{A}}$ and $\text{Group}_{\mathrm{C}}$, which have multiple piles, it is imperative to obtain the occupancy rate of connected EVs Occ_{A} and Occ_{C} relative to the total number of charging piles:

$$\begin{aligned} \text{Occ}_{\mathrm{A}} &= N_{\mathrm{A}}/N \\ \text{Occ}_{\mathrm{C}} &= N_{\mathrm{C}}/N \end{aligned} \tag{18}$$

As for EV group B, which only contains one EV_k, it is essential to know the relative location of the connected charging pile within the overall charging station. Given the sequential decision-making model, location of $\text{Group}_{\mathrm{B}}$ l_{B} can be easily calculated through the following equation:

$$l_{\mathrm{B}} = \frac{k}{N} \tag{19}$$

In addition of the three groups' states, $s^{\mathrm{O}}_{t,k}$ denotes the overall state, including the time t and current electricy price ρ_t:

$$s^{\mathrm{O}}_{t,k} = \{t, \rho_t\} \tag{20}$$

Due to the complexity of the state $s_{t,k}$ involved, all components are meticulously listed in Table I below for easy reference.

Type	Name	Description
$s^{\mathrm{O}}_{t,k}$	t	time
	ρ^t	current electricity price
$s^{\mathrm{A}}_{t,k}$	C_{A}	average battery capacity
	$p^{\text{rated}}_{\mathrm{A}}$	average rated charging power
	$\text{SoC}_{t,\mathrm{A}}$	weighted average current SoC
	$\text{SoC}^{\text{d}}_{\mathrm{A}}$	weighted average target SoC
	$r_{t,\mathrm{A}}$	average remain parking time
	Occ_{A}	occupancy rate of piles
	$p^{\text{total}}_{\mathrm{A}}$	total power consumption
$s^{\mathrm{B}}_{t,k}$	C_{B}	battery capacity
	$p^{\text{rated}}_{\mathrm{B}}$	rated charging power
	$\text{SoC}_{t,\mathrm{B}}$	current SoC
	$\text{SoC}^{\text{d}}_{\mathrm{B}}$	target SoC
	$r_{t,\mathrm{B}}$	remain parking time
	l_{B}	relative location of connected pile
$s^{\mathrm{C}}_{t,k}$	$C^{\text{norm}}_{\mathrm{C}}$	average battery capacity
	$p^{\text{rated}}_{\mathrm{C}}$	average rated charging power
	$\text{SoC}_{t,\mathrm{C}}$	weighted average current SoC
	$\text{SoC}^{\text{d}}_{\mathrm{C}}$	weighted average target SoC
	$r_{t,\mathrm{C}}$	average remain parking time
	Occ_{C}	occupancy rate of piles

TABLE I: COMPONENTS OF STATE $s_{t,k}$

Due to the special design of the state $s_{t,k}$, the proposed method follows a sequential decision-making process. At each time step t, when deciding the charging action $a_{t,k}$ for pile_k, it's essential to ensure that the charging actions for pile_1 to pile_{k-1} have already been determined, while pile_{k+1} to pile_N are still awaiting decisions. The state $s_{t,k}$ of pile_k depends on the actions already decided before it. Conseqently, acquiring the corresponding states sequentially, which is outlined in Algorithm 1, is necessary.

Algorithm 1 Extract $s_{t,k}$ of each pile_k sequentially

```
Input: EV charging scenario $\boldsymbol{\chi}$
Output: state-action matrix $\boldsymbol{S}$
for $i \leftarrow 1$ to 64 do
    Randomly initialize charging scheduling $\boldsymbol{a}_i$ on $\boldsymbol{\chi}$
    Find the local optimal $\boldsymbol{a}^*_i$ on $\boldsymbol{\chi}$ by solving CO
    Calculate the objective value $\boldsymbol{f}_i$ of $\boldsymbol{a}^*_i$
end for
Select $\boldsymbol{a}^*_i$ with the lowest $\boldsymbol{f}_i$ as $\boldsymbol{a}^*$
for $t = \tau : \tau + T - 1$ do
    for $k = 1 : N$ do
        if $\text{EV}_k$ is connected to $\text{pile}_k$ then
            Extract the state $s_{t,k}$ of $\text{pile}_k$ from $\boldsymbol{\chi}$ and $\boldsymbol{a}^*$
            Get charging action $a^*_{t,k}$ of $\text{pile}_k$ from $\boldsymbol{a}^*$
            Add $\langle s_{t,k}, a^*_{t,k} \rangle$ to $\boldsymbol{S}$
        end if
    end for
end for
```

With the definition of $s_{t,k}$ and state extraction algorithm, we can easily extract the pair of state vector and scalar optimal action $\langle \boldsymbol{s}, a^* \rangle$. Combining all pairs to the state-action matrix $\boldsymbol{S}$, we then get the expert dataset that can be used for training

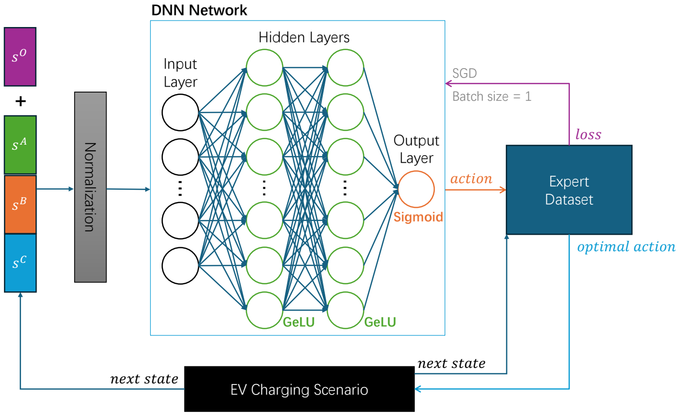

Fig. 4: Architecture of the DNN Network.

in IL later.

C. Behaviour Cloning

Due to the significant computational requirements of optimization methods like **P1**, it is impractical to make real-time decisions for charging actions of each pile in real-world applications. Therefore, our goal is to identify an alternative method that significantly reduces the computational load while maintaining optimization effectiveness as much as possible. Deep neural networks (DNN), with their powerful high-dimensional representation capabilities and rapid inference speed, are the preferred solution for addressing this issue. The EV charging scheduling scenario, which is defined in **P1**, can be regarded as a finite Markov Decision Process (MDP) over discrete time steps. At each time step τ, we can observe the real-time state $s_{\tau,0}$ of the first charging pile$_0$. Based on this information, the most suitable charging action $a_{\tau,0}$ is determined. Since the proposed model is sequential, once the charging action for pile$_0$ is decided, the new state $s_{\tau,1}$ of pile$_1$ can be obtained. After all piles have made their decisions at time step t, the state $s_{\tau+1,0}$ of pile$_0$ at time step $\tau+1$ can be calculated. In summary, after executing an action, a new system state can always be obtained. For a typical MDP, there are two main approaches to training the DNN: reinforcement learning (RL) and imitation learning (IL). Given the availability of a well-defined optimal solution (the expert dataset in section III-B), IL is evidently the more stable and cost-effective choice.

Utilizing behaviour cloning (BC), IL also tries to learn a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ compatible with all charging piles. The policy π maps the state $s_{\tau,k}$ of each pile$_k$ to the optimal action $a_{\tau,k}$ at time step τ. In this particular application, we use the state $s_{t,k}$ as the input features and the optimal action $a^*_{t,k}$ obtained from the expert dataset as the expected labels to train the DNN as the approximator to the optimal policy π^*. Theoretically, DNN is very powerful and has the ability to approximate any bounded continous function. However, there's no theoretical ganrantee that the optimal policy π^* can be written as a bounded continous function.

The architecture of the proposed DNN is shown in Figure 4, which is a fully-connected feed-forward network with two hidden layers. The input is a 21-dimention normalized vector $\boldsymbol{s}$ defined in Table I, and the output is a scalar action $a^*_{t,k} \in [0, 1]$. The activation function used in the hidden layers is the Gaussian Error Linear Unit (GeLU), which demonstrates significantly superior performance compared to the Rectified Linear Unit (ReLU) in this specific problem. For the output layers, as the range of output is $[0, 1]$, Sigmoid activation function, which can transform any value to a scalar between 0 and 1, is chosen.

As a regression task, the mean squared error (MSE) function is used to evaluate the loss. To minimize the loss function, the Adam optimizer with mini-batch mode is commonly used. However, this training method is not suitable for the problem at hand. The need to consider fairness involves collaboration among different charging piles. This collaboration is implicit since the network is provided with states that do not contain complete information. Maintaining fairness in charging through implicit collaboration represents one solution space of the model. Simultaneously, the model also need to minimize the charging costs. Unfortunately, the solution spaces for cost reduction and fairness maintenance are in conflict. In practice, cost reduction typically receives more attention, thus it is given greater weight. The significant contribution of the highly weighted cost reduction objective to the total loss function causes its solution space's attraction basins to become deeper or broader. Consequently, the Adam optimizer with an adaptive learning rate easily falls into the local optima.

Algorithm 2 Training the DNN network via IL

```
Input: Training dataset $\boldsymbol{S}$ and validation dataset $\boldsymbol{V}$
Output: DNN's parameters $\boldsymbol{\theta}^*$ of best performance
Randomly initialize $\boldsymbol{\theta}$
for epoch $\leftarrow 1$ to $100$ do
    for $<\boldsymbol{s}, a^* > \in \boldsymbol{S}$ do
        Calculate the estimated action $\pi(\boldsymbol{s}; \boldsymbol{\theta})$
        Calculate the loss $L(\boldsymbol{\theta}, \boldsymbol{S}) = \frac{1}{n}\sum_{\boldsymbol{s}}^{\boldsymbol{S}}(\pi(\boldsymbol{s}; \boldsymbol{\theta}) - a^*_{\boldsymbol{s}})^2$
        Update $\boldsymbol{\theta}$ by SGD optimization algorithm
    end for
    Calculate the estimated action vector $\pi(\boldsymbol{V}; \boldsymbol{\theta})$
    Calculate the loss $L(\boldsymbol{\theta}, \boldsymbol{V}) = \frac{1}{n}\sum_{\boldsymbol{v}}^{\boldsymbol{V}}(\pi(\boldsymbol{v}; \boldsymbol{\theta}) - a^*_{\boldsymbol{v}})^2$
    if $L(\boldsymbol{\theta}, \boldsymbol{V})$ doesn't decrease for 5 epochs then
        Break
    end if
end for
```

As an alternative, we employ the Stochastic Gradient Descent (SGD) optimization algorithm. By utilizing different samples for each update, the gradient update direction can vary with each iteration. This stochastic nature enhances SGD's ability to explore the global parameter space, thereby reducing the risk of becoming trapped in local optima. Furthermore, SGD does not incorporate an adaptive learning rate mechanism. Consequently, all objectives contribute equally to the overall gradient during each update, mitigating the bias that may arise from disproportionate weights assigned to specific objectives.

The early termination technology is inculded to avoid possible overfitting. To verify the scalability of the DNN agent, the validation dataset is bulid from a larger scale EV fleet

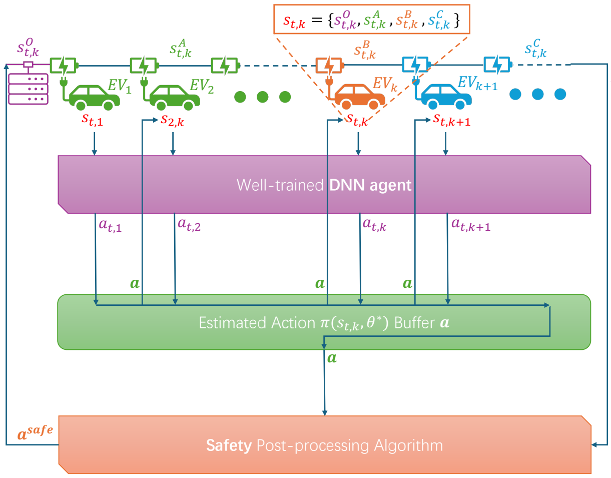

Fig. 5: Illustration of the proposed sequential scheduling.

than the training dataset's. If MSE loss doesn't decrease for a certain number of epochs, the early termination is activated and return the $\boldsymbol{\theta}^*$ which has the best performance on the validation dataset.

D. Real-Time Online EV Charging Scheduling

Please note that the construction of the expert dataset in Section III-B, as well as the training of the DNN in Section III-C, are entirely conducted offline. Once the well-trained DNN agent is obtained, it is deployed to the CCC to commence online real-time scheduling for charging of EVs. As shown in Figure 5, the order of the decision-making and the extraction of states from the optimal solution $\boldsymbol{A}$ remain consistent. At each time step τ, each pile$_k$'s state $s_{\tau,k}$ is computed sequentially as shown in Table I. The state $s_{\rho,k}$ is used as the input of the DNN agent, which returns the estimated charging action $\pi(s_{\tau,k};\boldsymbol{\theta*})$.

Algorithm 3 Real-Time Online EV Charging Scheduling

```
Input: DNN's parameters $\boldsymbol{\theta}^*$, real-time EV scenario $\boldsymbol{\chi_t}$
Output: Real-time EV charging scheduling
Load the parameters $\boldsymbol{\theta}^*$ of the well-trained DNN.
Initiallize the action buffer $\boldsymbol{a}$
for $t = \tau : \tau + T - 1$ do
    Clear $\boldsymbol{a}$
    for $k = 1 : N$ do
        if EV$_k$ is connected to charging pile$_k$ at time step $t$ then
            Calculate pile$_k$'s state $s_{t,k}$ from $\boldsymbol{\chi_t}$ and $\boldsymbol{a}$
            Get estimated action $\pi(\boldsymbol{s}_{t,k};\boldsymbol{\theta*})$ from DNN
            Store $\pi(s_{t,k};\boldsymbol{\theta*})$ in $\boldsymbol{a}$
        end if
    end for
    Apply the safety post-processing by Algorithm 4 to $\boldsymbol{a}$ and get the safe charging action vector $\boldsymbol{a}_{\textbf{safe}}$
    Excute the final action $\boldsymbol{a}_{\textbf{safe}}$ on each charging pile
end for
```

Please note that the estimated charging action $\pi(s_{t,k};\boldsymbol{\theta*})$ provided by the DNN agent is not the final decision. All estimated charging actions $\pi(s_{t,k};\boldsymbol{\theta*})$ have to be re-uploaded to the CCC for safety post-processing, which will be introduced in Section III-E. After post-processing, the CCC will issue the ultimate safe charging decision $a^{\text{safe}}_{t,k}$ to all charging piles for execution. This process above will keep iterating until the last EV departs.

E. Safety Measures

Algorithm 4 Safety post-processing algorithm

```
Input: Estimated action vector $\boldsymbol{a}^*$ by DNN, real-time EV scenario $\boldsymbol{\chi_t}$
Output: Ultimate safe action vector $\boldsymbol{a}^{\text{safe}}$
for $t = 0 : t_{\text{end}}$ do
    for $k = 1 : N$ do
        if EV$_k$ is connected to charging pile $k$ at time step $t$ then
            Update SoC$_{t+1,k}$ of EV$_k$ by $\boldsymbol{a}^*_k$ and $\boldsymbol{\chi_t}$
            if SoC$_{t+1,k}$ $>$ SoC$^{\text{d}}_k$ then
                Adjust $\boldsymbol{a}^*_k$ to make SoC$_{t+1,k}$ = SoC$^{\text{d}}_k$
            end if
        end if
    end for
    Save the adjusted $\boldsymbol{a}^*$ to the intermediate vector $\boldsymbol{a}'$
    for $k = 1 : N$ do
        if EV$_k$ is connected to charging pile $k$ at time step $t$ then
            Calculate the estimated charging power $p_{t,k}$ by $\boldsymbol{a}'[k]$ and $\boldsymbol{\chi_t}$
        else
            Set $p_{t,k}$ to 0
        end if
    end for
    if $\sum_k^N p_{t,k} > P_{\text{rated}}$ then
        Update $\boldsymbol{a}'$ by scaling it down equally to make $\sum_k^N p_{t,k} = P_{\text{rated}}$
    end if
    for $k = 1 : N$ do
        if EV$_k$ is connected to charging pile $k$ at time step $t$ then
            Update SoC$_{t+1,k}$ of EV$_k$ by $\boldsymbol{a}'_k$ and $\boldsymbol{\chi_t}$
            if SoC$_{t+1,k}$ + $\frac{p^{\text{rated}}_{t,k}\cdot e_k\cdot\Delta t\cdot(t^{\text{d}}_k-(t+1))}{C_k}$ < SoC$^{\text{d}}_{t+1,k}$ then
                Allocate the remain power $P_{\text{rated}} - \sum_k^N p_{t,k}$ to EV$_k$, until the charging requirement SoC$^{\text{d}}_{t+1,k}$ is fulfilled.
                Update $\boldsymbol{a}'_k$ according to the allocated power and make sure $\boldsymbol{a}'_k \le 1$
            end if
        end if
    end for
    Save the adjusted $\boldsymbol{a}'$ to $\boldsymbol{a}^{\text{safe}}$
end for
```

The estimated charging actions $\pi(s;\boldsymbol{\theta*})$ obtained from the DNN agent can't be excuted directly due to the hard constraints. In the particular charging scheduling problem, violating the constraints could lead to serious consequences like grid instability, battery overcharge and failing to meet user demands. There will inevitably be some error in any neural network's predictions for real world applications. Therefore, a safety post-porcessing algorithm is proposed to ensure the constraints are fully met for any policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$. After the safety post-porcessing, even a stochastic policy can be adjusted to satisfy the constraints. The safety measures can be mainly divided into three steps:

1) Overcharge protection: to check if any EV_k's $\mathrm{SoC}_{t+1,k}$ will be larger than the user's expected $\mathrm{SoC}^{\mathrm{d}}_k$ after executing the estimated charging action $\pi(s;\boldsymbol{\theta}^*)$. If such vehicles do exist, make appropriate adjustments and save the changes to intermediate action vector $\boldsymbol{a}'$.
2) Station capacity protection: to check if the total charging power of all EVs exceed the charging station's power capacity after executing $\boldsymbol{a}'$. If such a situation occurs, update $\boldsymbol{a}'$ by scaling down it equally.
3) Undercharging protection: to check if any EV_k can't fulfill the SoC target when leaving after executing $\boldsymbol{a}'_k$, even charging with maximum power p^{rated}_k for the remaining time steps (except the current one). If such an occurrence arise, the forced charging module will be activated, ensuring the vehicle is charged as much as possible without exceeding the charging station's power capacity. After saving the changes to $\boldsymbol{a}'$, the ultimate safe action vector $\boldsymbol{a}^{\mathrm{safe}} = \boldsymbol{a}'$ is upload to the charging station and all charging piles will excute it in the following time step.

Please refer to Algorithm 4 for the details.

References

[1] S. L. Xuezhong Hu and Y. J. Haibin Wang, "Ac charging pile of electric vehicle and intelligent charging control strategy research," *REVIEWS OF ADHESION AND ADHESIVES*, vol. 11, no. 3, 2023.

[2] A. Zhou, J. Yu, Z. Li, and J. Pu, "A security authentication method between the charging pile and battery management system," in *2021 IEEE 4th International Conference on Electronics Technology (ICET)*. IEEE, 2021, pp. 400–405.

[3] E. ElGhanam, M. Hassan, A. Osman, and I. Ahmed, "Review of communication technologies for electric vehicle charging management and coordination," *World Electric Vehicle Journal*, vol. 12, no. 3, p. 92, 2021.

[4] H. Yu, C. Xu, W. Wang, G. Geng, and Q. Jiang, "Communication-Free Distributed Charging Control for Electric Vehicle Group," *IEEE Transactions on Smart Grid*, vol. 15, no. 3, pp. 3028–3039, May 2024.

[5] L. Yao, W. H. Lim, and T. S. Tsai, "A Real-Time Charging Scheme for Demand Response in Electric Vehicle Parking Station," *IEEE Transactions on Smart Grid*, vol. 8, no. 1, pp. 52–62, Jan. 2017.

[6] H. M. Abdullah, A. Gastli, and L. Ben-Brahim, "Reinforcement learning based ev charging management systems–a review," *IEEE Access*, vol. 9, pp. 41 506–41 531, 2021.

[7] M. Dorokhova, Y. Martinson, C. Ballif, and N. Wyrsch, "Deep reinforcement learning control of electric vehicle charging in the presence of photovoltaic generation," *Applied Energy*, vol. 301, p. 117504, Nov. 2021.

[8] Z. Wan, H. Li, H. He, and D. Prokhorov, "Model-Free Real-Time EV Charging Scheduling Based on Deep Reinforcement Learning," *IEEE Transactions on Smart Grid*, vol. 10, no. 5, pp. 5246–5257, Sep. 2019.

[9] W. Wu, Y. Lin, R. Liu, Y. Li, Y. Zhang, and C. Ma, "Online EV Charge Scheduling Based on Time-of-Use Pricing and Peak Load Minimization: Properties and Efficient Algorithms," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 1, pp. 572–586, Jan. 2022.

[10] Z. Yi, D. Scoffield, J. Smart, A. Meintz, M. Jun, M. Mohanpurkar, and A. Medam, "A highly efficient control framework for centralized residential charging coordination of large electric vehicle populations," *International Journal of Electrical Power & Energy Systems*, vol. 117, p. 105661, May 2020.

[11] S. Hajforoosh, M. A. S. Masoum, and S. M. Islam, "Online optimal variable charge-rate coordination of plug-in electric vehicles to maximize customer satisfaction and improve grid performance," *Electric Power Systems Research*, vol. 141, pp. 407–420, Dec. 2016.

[12] S. Thangavel, D. Mohanraj, T. Girijaprasanna, S. Raju, C. Dhanamjayulu, and S. M. Muyeen, "A Comprehensive Review on Electric Vehicle: Battery Management System, Charging Station, Traction Motors," *IEEE Access*, vol. 11, pp. 20 994–21 019, 2023.

[13] H. Li, G. Li, T. T. Lie, X. Li, K. Wang, B. Han, and J. Xu, "Constrained large-scale real-time EV scheduling based on recurrent deep reinforcement learning," *International Journal of Electrical Power & Energy Systems*, vol. 144, p. 108603, Jan. 2023.

[14] C. A. Floudas, I. G. Akrotirianakis, S. Caratzoulas, C. A. Meyer, and J. Kallrath, "Global optimization in the 21st century: Advances and challenges," *Computers & Chemical Engineering*, vol. 29, no. 6, pp. 1185–1202, May 2005.

[15] S. Gao, R. b. Z. Lee, Z. Huang, C. Xiang, M. Yu, K. T. Tan, and T. H. Lee, "A Hybrid Approach for Home Energy Management With Imitation Learning and Online Optimization," *IEEE Transactions on Industrial Informatics*, pp. 1–13, 2023.