Power Output Optimisation for Electric Vehicles Smart Charging Hubs

Andrea Bertolini
andrea.bertolini@tecnico.ulisboa.pt

Instituto Superior Técnico, Universidade de Lisboa, Portugal

January 2021

Abstract

Since most branches of the distribution grid may already be close to their maximum capacity, smart management when charging electric vehicles (EVs) is becoming more and more crucial. In fact, office buildings might not be able to handle several transactions at the same time, especially considering the next generation of fast chargers which are very power expensive. Thus, an optimal charging policy needs to be found. This master thesis tackles the problem of real-time EVs charging scheduling through deep reinforcement learning (DRL) techniques. DRL has been chosen because it can adaptively learn from interacting with the surrounding environment. The focus of the optimization is to ensure the completion of the charging transactions in a timely manner, while shifting the load from the times of peak demand. The novelty of the proposed approach lies in its innovative framework. Pools of electric vehicles with different characteristics are categorized using a clustering algorithm. A tree-based classifier has been developed to sort new instances of EVs and a multilayer perceptron has been trained to predict the expected duration of each charging session. These inputs are used as features based on which the DRL agent performs its actions, adjusting the maximum power associated to each charging station. The model has been compared to the system currently implemented and increasingly challenging scenarios have been considered. Results have showed that the developed algorithm fails less than the baseline, with a reduction of the load due to EVs charging of 80% during peak times.

Keywords: Reinforcement learning, electric vehicles, real-time charging scheduling, neural network, clustering algorithm

1. Introduction

1.1. Motivation

The transportation sector is one of the major responsible for energy consumption in the world. This environmental issue, accompanied by governments' incentives, have pushed for a strong adoption of electric vehicles (EVs) in many countries. In fact, the electrification of the transportation sector, together with the increasing electricity generation from renewable energy sources (*RES*) can lower the reliance on fossil fuels and lead to emission reduction [5]. Nevertheless, it has to be mentioned that EVs scale adoption will bring a massive high-power load into the electric grid, that might represent a threat to the health of the current energy system. In fact, if multiple EVs were charged simultaneously in an uncontrolled way, they could increase the peak demand on the grid, contributing to overloading and the need for upgrades at the distribution level [2]. This is the case of traditional charging, where charging stations possess no means of intercommunicating with other IT devices and power is immediately delivered at maximum speed rate. Instead, the technology behind *smart charging stations* is able to adjust the power output, optimizing the charging process while ensuring grid stability. *Smart charging* or intelligent charging formally refers to a system where an electric vehicle shares a data connection with a charging device, and the charging device shares a data connection with a charging operator [1]. In other words, it is a way of optimising the charging process according to distribution grid constraints and local renewable energy availability while respecting customers' needs for vehicle availability [2].

1.2. State of the art

Optimised charging of EVs can bring enormous benefits for customers while also stabilising the power grid. In the past years, several techniques have been applied to tackle this problem. Sortomme and El-Sharkawi [14] applied convex programming to schedule energy and ancillary services with the objective of maximizing the profit of an aggregator and providing low charging costs to the final customers. Elmedhi and Abdelilah [17] opti-

mally scheduled the charge of an EVs fleet with *genetic algorithms*. Zheng and Yang [18] introduced a global intelligent method to find optimal cooperation charging/discharging strategies for EVs to minimize the operation cost with *particle swarm optimization (PSO)*. Although the aforementioned methods achieved some success in day-ahead charging/discharging scheduling, they may be unsuitable for real-time scenarios where the variations in EV charging demand and the electricity prices are much more complex. In this regard, a different approach for solving the optimal EV charging problem can be pursued through Reinforcement Learning, which is used to solve complex tasks that cannot be solved by conventional techniques. While it is usually needed to precisely formulate the optimization problem (in its objective function and constraints, for example), here the intelligent agent can act *model-free*. Mocanu et al. [12] explored for the first time in the smart grid context the benefits of using Deep Reinforcement Learning (DRL) to perform on-line optimization of schedules for building energy management systems. Similarly, We et al. [16] proposed a system DRL-based algorithm for building HVAC control. Wan et al. [15] deal with the optimal EV charging problem with reinforcement learning. Dang et al. [4] applied a Q-Learning based charging scheduling scheme for EVs, considering bidirectional interaction between the vehicle and the grid, including the grid-to-vehicle charging and the *vehicle-to-grid (V2G)* electricity returning. Lee and Choi [11] presented a method for the scheduling of energy consumption of smart home appliances and distributed energy resources, including the charging of an electric vehicle. Lastly, Fang et al. [6] developed a *multi-agent* reinforcement learning approach for optimally scheduling energy in a residential microgrid environment, integrated with EVs and renewable generation.

1.3. Problem formulation

The objective of this master's thesis is to develop an innovative decision-based optimization algorithm able to send power limit adjustments to the charging points where EVs can connect. The aim is to alter in real-time the charging loads in order to satisfy power grid and vehicle constraints. The energy system taken in consideration is a generic office building, equipped with a photovoltaic (PV) plant for solar power production and with a garage where the charging stations are installed. The building considered is connected to the grid with 250 kW of power supply, it has up to 74 kWp of solar power installed and it is provided with 10 charging stations where EVs connect and disconnect during the whole day. The final goal of the optimisation is to ensure that every EV successfully completes their charging transactions in a timely manner without the associated load threatening the grid stability. This last condition has been encoded in the form of a threshold curve, which represents the total power at any instant of time and whose values the charging system should never exceed. This limit is obtained by subtracting, at each instant of time, the consumption of the building from the grid power supply, and adding up the solar production. Moreover, the algorithm developed in this dissertation must be able to handle the charging transactions efficiently, guaranteeing that every EV would receive as much energy as possible without breaking the constraint rule above listed. Combining all the elements, the problem can be formulated as follows:

$$
\begin{aligned}
\underset{P_t^i}{\text{maximize}} \quad & \textstyle\sum_{i=1}^{N}(E_T^i - E_0^i) \\
\text{subject to} \quad & \textstyle\sum_{i=1}^{N} P_{t=0}^i \leq P_{nom} - P_{cons,t=0} + P_{PV,t=0} \\
& \textstyle\sum_{i=1}^{N} P_{t=1}^i \leq P_{nom} - P_{cons,t=1} + P_{PV,t=1} \\
& \vdots \\
& \textstyle\sum_{i=1}^{N} P_{t=T}^i \leq P_{nom} - P_{cons,t=T} + P_{PV,t=T}
\end{aligned}
\tag{1}
$$

where:

- P_t^i is the charging power rate of the vehicle i at time t;
- $E_T^i - E_0^i$ is the total energy transferred to the vehicle i;
- P_{nom} is power grid supply from the grid;
- $P_{cons,t}$ is the building consumption at time step t;
- $P_{PV,t}$ is the photovoltaic solar power production at time step t.

1.4. Proposed framework

The problem of dynamically scheduling electric vehicles' charging is complex and the approach followed to tackle it involves the combination of multiple algorithms and techniques. Figure 1 shows the overall architecture of the artificial intelligence *(AI)* framework that has been built. The key element is the *intelligent agent*, namely a neural network *(NN)*, whose predictions can be thought as power rate variations during the EVs' charging transactions. These outputs are sent to a central system and, in turn, forwarded to the charge points in terms of power limit adjustments. The basic idea is that the agent, at each moment, interacts with the charge points in order to acquire useful information about the current state of the charging processes. Based on this knowledge newly gained, the agent shapes the charging scheduling of the EVs connected, impacting the information that it will receive in the next step.

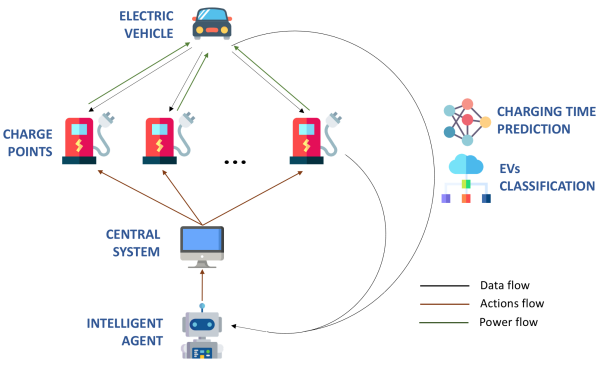

Figure 1: Architecture of the framework proposed: input data are gained from the EVs (black arrows), the agent sends actions to the charge points (red arrows) that are translated in power adjustments back to the EV (green arrows)e

However, just a part of those data is readily available. For this reason, a clustering algorithm has been used to identify pools of electric vehicles, a tree-based model has been developed to classify new instances of EVs and an additional neural network has been trained to predict the expected duration of the charging session for each new vehicle arriving. The terms *clusters* and *classes* will be used throughout the paper to indicate the same groups of samples but referring to different tasks. Clusters are the data structures recognised from the initial unlabeled data set, which in turn have been used as classes to sort new instances. Furthermore, a simulated environment has been created in order to iteratively respond to the agent's requests and to evaluate its performance. The interaction between the central system and the charge points has been emulated, as well as the EVs typical arrival and departure scheduling and other dynamics. Lastly, the work has been structured in a way that the intelligent agent had to fulfill tasks of increasing difficulty. In particular, two different scenarios have been evaluated. Initially, the power threshold was taking into account the only grid supply, making it a constant line. From this intermediary step, a more challenging problem has been constructed, considering also building consumption and solar production in the equation, with a power limit variable in time. The rest of the paper is organized as follows. *Section 2* contains a theoretical introduction of the elements of *Reinforcement Learning*. *Section 3* discusses the simulation dynamics, delving into the techniques used to cluster and classify pools of electric vehicles, as well as to predict the expected duration of their transactions. *Section 4* discusses the optimization problem, outlining how the intelligent agent and the environment have been designed and how they interact between each other. *Section 5* shows the results obtained along with the approach the has been followed, comparing the performances of the algorithm in the different scenarios considered and against the current implementation. *Chapter 6* draws the conclusions.

2. Reinforcement learning

2.1. Theoretical foundations

Reinforcement learning (RL) is a subfield of machine learning that addresses the problem of the automatic learning of optimal decisions over time [10]. RL algorithms do not learn from pre-processed data, but features and target are generated by the continuous interaction between two main modules, that iteratively exchange information to each other (figure 2): the *agent*, which is the learner and decision-maker in the system, and the *environment*, where the agent acts upon.

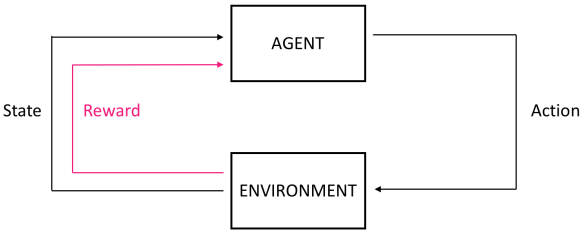

Figure 2: Agent-environment interaction

At each time step, the agent receives the current representation of the environment, defined as its internal *state*. Based on the information acquired, it performs an *action*, that might possibly alter the state, converting it to the *next state*. As a consequence of the action taken, the agent receives a *reward*, moves to the *next state* and the scheme is repeated. The mapping from the perceived states to the actions to be taken in those states, namely the way the agent is behaving, is called policy, usually denoted by π. The reward is a numerical signal whose calculation is designed to make it high if the agent takes good actions and low if the agent misbehaves. It is common for reinforcement learning algorithms to estimate *value functions*, a metric that assesses *how good* it is for the agent to be in a given state (or to perform a given action in a given state) [13]. Value functions are interesting and widely used in reinforcement learning tasks because they satisfy particular recursive relationships. The most relevant to the scope of this dissertation expresses the relation between the value of a state and the values of its successor states [13]:

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) \left[r + \gamma v_\pi(s') \right] \quad (2)$$

Equation 2 is known as the *Bellman equation*.

There is always a policy (or a set of policies) behaving better than the other ones: that is an *optimal policy* and denoted by π_*. The relationship can be now written in a special form:

$$v_*(s) = \max_{a \in A(s)} \sum_{s',r} p(s', r|s, a) \left[r + \gamma v_*(s') \right] \quad (3)$$

which is known as *Bellman optimality equation* for $v_*(s)$. The environment of the reinforcement learning problem described in this work is considered to be a *Markov decision process* (*MDP*), where the future system dynamics from any state have to depend on the current state only [10]. For finite MDPs, the Bellman optimality equation has a unique solution independent of the policy.

2.2. Q-Learning algorithm

Q-learning is an *off-policy* method, namely it evaluates a different policy from the one used to generate the data [13]. In Q-learning, the action-value function associated to each state-action pair is updated at every step, complying with the following rule:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right]$$

where S_t is the state at time t, A_t is the action performed at time t, $Q(S_t, A_t)$ is action-value function at time t, γ is the discount factor and α is the learning rate. The *learning rate* α determines to what extent newly acquired information overrides old information.

Algorithm 1 Q-Learning

1: Initialize $Q(s; a)$ $\forall$ s $\in S$; a $\in A(s)$ arbitrarily and Q(terminal state) $= 0$
2: **for** *episode* $= 1, \ldots, k$ **do**
3: Initialize S
4: **for** *step* $= 1, \ldots, n$ **do**
5: Choose A from S using policy derived from Q
6: Take action A, observe R, S'
7: $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right]$
8: **end for**
9: **end for**

In reinforcement learning problems, often happens that states encountered by the agent have never been experienced before. It is then necessary to generalize from previously experienced states to ones that have never been seen [13]. The kind of generalization required is called *function approximation*, a technique for estimating an unknown underlying function using historical or available observations from the domain [3]. For this purpose, *artificial neural networks* have been widely adopted as function approximators in reinforcement learning algorithms in order to make predictions and take actions. The combination of the popular Q-learning algorithm with neural networks (Deep Q-Learning, or *DQN*) is known to overestimate action values under certain conditions. A specific adaptation to the DQN algorithm, called *Double Deep Q-Learning*, or Double DQN, has been proposed and it has been shown that the resulting algorithm leads to much better performance [8].

3. Charging event generator

3.1. Simulation architecture

The simulation consists of three main objects: the *DQN agent*, in which the neural network is designed and the learning process takes place,the *central system*, responsible for emulating the behavior of the charging stations and that contains all the environment's characteristics and the *Electric Vehicle*, a separate object in charge of simulating the randomness associated with the EVs arrival/departure and their charging specifications. Those objects are Python classes.

3.2. Arrival and departure scheduling

Data of 120 charging transactions have been considered in order to reproduce the frequency of EVs arriving and leaving from the building's garage. As expected, the majority of the people reach the office in the morning, with a lower and lower number of EVs arriving as the day passes. Naturally, the departure function has the opposite trend. A typical day has been subsequently divided in 12 time bands, in order to calculate the *arrival and departure rates*, namely the probabilities with which a generic electric vehicle arrives or leaves during the day. The rates have been used to build daily occupancy profiles of the building's parking lot, as represented in figure 3.

3.3. Electric Vehicles Clustering

The shape of a charging pattern is an important feature that distinguishes electric vehicles between each other. Although having a similar structure, these trends present differences that might reveal essential information about the type of car under charge at that time. For this reason, a *clustering method* has been used in order to detect pools of cars that have similar charging characteristics. To this end, one of the most popular and well-known algorithms for unsupervised learning tasks has been chosen: *K-Means*. Because of the featureless nature of the available data, the key was to understand which features needed to be designed in order for the algorithm to successfully identify clusters among the data. In particular, due to data volatility, the average rate during the first 5 minutes of

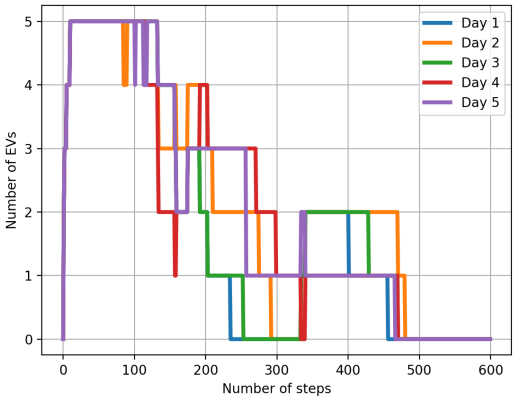

Figure 3: Parking lot occupancy profiles for five independent episodes, each made of 600 steps

charging has been considered:

$$P_{avg} = \frac{\sum_{i=1}^{5} P_i}{5} \tag{4}$$

where P_i [kW] is the speed charging rate at each time step i. Regarding the configuration of the curve, a feature called *slope* has been created, in the form:

$$Slope = 3 * (P_1 - P_{15}) \tag{5}$$

where P_1 and P_{15} are, respectively, the power rates at time step 1 and 15. This subtraction indicates if the curve stays constant or if it starts already to decrease during the first quarter of an hour of charging. A factor of 3, obtained by a try/test approach, has also been added because it helps the spreading of the data points in the space, thing that contributes to a better performing of K-Means. The results are shown in Figure 4, where the five clusters found are recognizable.

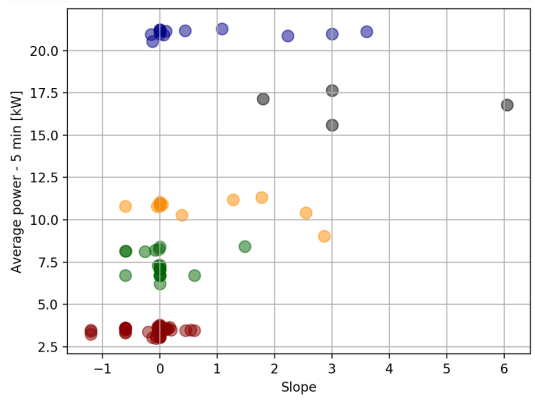

Figure 4: Data set of transactions after the clustering

As a new EV connects to a charge point, the car has to be re-conducted to one of the clusters previously introduced. This can be translated in a classification problem when the classes are the clusters themselves. The classes distribution is also *imbalanced*, since some clusters contain significantly more transactions than others. For the purpose of classifying clusters of electric vehicles, a decision tree has been trained having available the same data set containing 135 charging transactions that has been used for the clustering task. The features considered for this classification problem are the ones described in equations 4 and equation 5. The algorithm achieved a value of accuracy on the validation set equal to 94%. However, this result has been improved to 100% accuracy by training many decision trees and computing the aggregated answer of all of them, instead of using the prediction from the individual classifier. The method used in this thesis work is *random forests* [9], an ensemble of decision trees trained with the *bagging* method [7].

3.4. Charging session time prediction

The charging pattern of an electric vehicle can be well approximated knowing the type of EV (namely, its cluster), its maximum charging speed admissible and the *time* needed to complete the transaction. A tailored function has then been created in order to faithfully reproduce the charging pattern of the different EVs (figure 5).

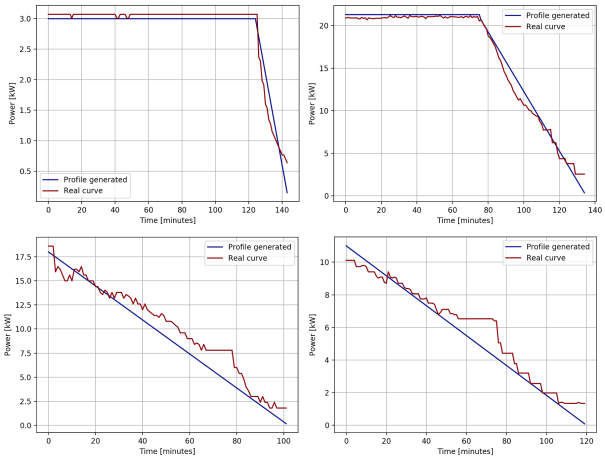

Figure 5: Profile generated in comparison with the real curves: cluster 1 (upper-left), cluster 3 (upper-right), cluster 4 (lower-left) and cluster 5 (lower-right)

.

A feed-forward neural network has been chosen for predicting the time at which the EVs are expected to be done with the charging. The data set includes data accounting for 135 charging transactions. The features considered are the *average power rate* during the first 5 minutes of the trans-

action (equation 4), the *slope* (equation 5) and the *starting time* of the transaction. The neural network in consideration has been able to reach a mean squared error in the test set equal to 0.02, which translate in 10 minutes of difference between the target and the value predicted.

4. Real-time electric vehicles charging

4.1. State

The state is the container of information the agent is allowed to visit and from which it extracts knowledge. More formally, the state is represented by a vector that contains all the characteristics of the current environment at a specific instant of time t:

$$s_t = [u_t^1, \ldots, u_t^n,\ c_t^1, \ldots, c_t^n,\ t_t^1, \ldots, t_t^n,\ p_t^1, \ldots, p_t^n,\ l_{t+1}, \ldots, l_{t+180}]$$

In this work, the state encapsulates five types of information:

- u_t^i is a boolean value indicating whether the charging socket is connected with the EV i or not, informing about the occupancy of the parking lot;
- c_t^i specifies the predicted cluster to which the EV i belongs to;
- t_t^i represents the predicted time needed by the EV i for completing the charging transaction;
- p_t^i is the real-time charging speed rate that the EV i is receiving at that instant of time;
- l_t $\ldots$, l_{t+180} are 3 hours ahead forecasted power threshold data, which define the limit of power consumption due to EVs charging.

4.2. Actions

The predictions of the neural network translate into the actions that the agent will perform in the next time step. The actions have been thought to be power limit adjustments that would intelligently modify the charging scheduling of electric vehicles. In this regard, an *on/off approach* has been explored, where the agent only has to choose between two actions: *charge* or *don't charge*. This format is a combination of effectiveness and simplicity, since the update of the environment doesn't require large computational time and the agent has to a variety of just two options to select from. The algorithm needs to send power adjustments to several EVs connected simultaneously, controlling many charge points at the same time. This means that the output of the neural network, namely the action taken by the agent, will be actually translated into as many individual actions as many charging sockets are considered in the problem. In particular, the total number of possibilities is given by n^k, where n is the number of individual actions and k is the number of charging sockets considered.

4.3. Reward system

The reward is a numerical value that expresses how good was the action taken by the agent. The final reward system has been formulated at the end of a step-by-step process. In fact, the problem has been initially considered in its easiest form, increasing gradually the complexity once good results were obtained. Figure 6 illustrates the methodology flow that has been followed. The goal was to start working with a basic reward implementation whose policy could have quickly been learned by the agent, and then moving to the most detailed and advanced model. The transition between a station and the next one would happen just when the previous model would give robust results.

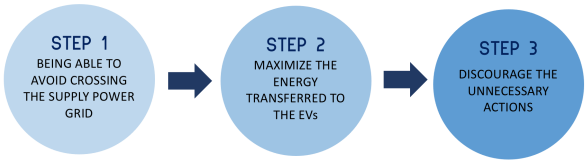

Figure 6: Reward system shaping methodology

The very first implementation of the model had the easiest task. The only concern here is to make sure that the total charging load, given by the sum of all the EVs simultaneous power consumption, would never exceed a power threshold. As it has been already explained in section 1.3, the threshold is considered constant and coincides with the power grid in the first scenario, while it depends on external factors such building consumption and photovoltaic solar production in the second scenario, making it variable throughout the day. The formulation of the reward for this first step at each instant of time is given by the following expression:

$$r_t = \begin{cases} -15 & if \quad \sum_{n=1}^{N} P_t^n \geq PT_t \\ \frac{PT_t - \sum_{n=1}^{N} P_t^n}{100} & if \quad \sum_{n=1}^{N} P_t^n < PT_t \end{cases} \tag{6}$$

where PT_t is the power threshold at each time step, P_t^n is the charging rate of EV n at time t and N is the total number of EVs under charge. The strategy that has been pursued was to provide a small positive reward for each time step that the agent successfully performs the task and a large penalty when the agent fails. The *failure* causes also the immediate end of the episode, conveying to the agent the message that those are terminal states

and need to be avoided. With the second step, the core of the optimization problem is introduced. In fact, the problem starts to be tricky now due to the fact that the two reward conditions are pushing in opposite directions: on one side, the total charging load has to lie below the power threshold but on the other side, the vehicles need to be as much charged as possible by the time they leave. The agent gets rewarded based on the percentage of energy it allowed to transfer during each transaction. The amount of energy transferred corresponds to the area underneath the charging curve, and can be computed with a integral operation. This calculation is performed each time an EV leaves, checking the state of charge at that point. However, before going to the mathematical expression, a premise is necessary. Often happens that an EV doesn't stay connected enough to the charge point to even allow the agent to charge it until maximum capacity. For this reason, the reward update is differentiated twice:

- *Case 1*: the EV leaves after the time needed to complete the transaction (7);
- *Case 2*: the EV leaves before the time needed to complete the transaction (8).

$$r^n = \begin{cases} -0.5 & if \quad \frac{E^n}{E^n_{max}} < 0.8, \\ \frac{E^n}{E^n_{max}} \times \frac{1}{100} & otherwise. \end{cases} \tag{7}$$

$$r^n = \begin{cases} -0.5 & if \quad \frac{E^n}{E^n_{act,max}} < 0.8, \\ \frac{E^n}{E^n_{act,max}} \times \frac{1}{100} & otherwise. \end{cases} \tag{8}$$

where E^n is the actual energy transferred to the EV at the time it disconnects from the charging socket and E^n_{max} is the maximum energy transferable to the EV if it would stay there enough time to be completely charged, and $E^n_{act,max}$ is the maximum energy transferable to the EV if it would not stay there enough time to be completely charged. It can be noticed that the update of the reward is at any time step t and for any EV connected n. Moreover, the weights of the two conditions can be compared and it can be noticed how the new penalty introduced is 30 times lighter than the punishment for exceeding the power threshold. This is due to the fact that, on average, 30 are the transactions simulated by the environment in one episode during training. The message for the agent is that surpassing the power limit is as serious as transferring to each EV less than 80% of the energy possible. Lastly, the ideal algorithm should interfere as little as possible in the charging dynamics. The continuous receiving of power adjustments might indeed cause problems to the battery management system of the electric vehicle. For this reason, an additional condition has been added to the system in order to encourage the agent to intervene as little as possible. Formally, this additional reward condition can be formulated in the following way:

$$r^n_t = \begin{cases} 0.001 & if \quad a^n_t = a^n_{t-1}, \\ -0.001 & otherwise. \end{cases} \tag{9}$$

where r^n_t is the reward value given at time t due to the event occurred to the EV n, a^n_t is the action performed by the agent at time step t to the EV n, and a^n_{t-1} is the action performed by the agent at time step $t-1$ to the EV n. The weight associated to this reward condition has been chosen consistently with the importance that has been attributed to it. Figure 7 shows the upgraded diagram.

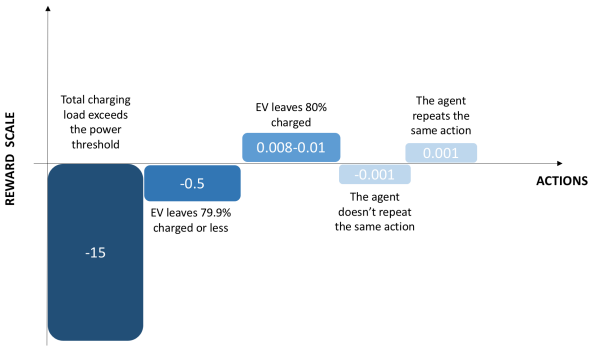

Figure 7: Overall reward system with the associated weights for each condition implemented

4.4. Optimization specifics

Due to the little time required and its good performance, it was opted for a deep neural network with 4 hidden layers and 32 neurons in each layer. The activation functions associated to the layers are *ReLu* function for the hidden layers and *linear* function for the output layer, due to the regression nature of the problem. The *exploration rate* ϵ is initially set to 1 and it reaches a minimum of 0.01, in order to always have a bit of exploration even when the agent should have learned a very good policy. Experimental results showed that a higher final exploration rate would divert the agent from learning a good policy and maximize the reward. The discount factor has been set as 0.99, and the target network is updated every 200 episodes.

5. Experimental results

5.1. Dual approach

Figure 8 shows the reward trends of the two scenarios during training. The simulation has been run for a total of 2000 episodes, after which it has not be recognized any valuable improvement and the computational time would rapidly increase. As it can be seen from the image, in the first scenario the agent find a solid way to improve his reward constantly after each episode. Instead, in the second scenario, the network seems to converge quite early and still fluctuating a lot.

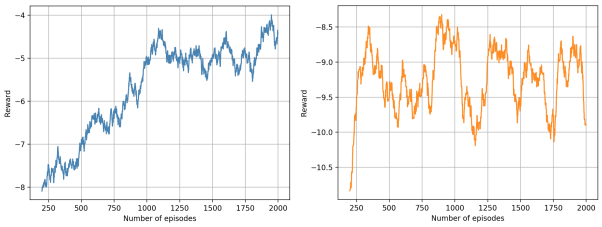

Figure 8: Reward trends for scenario 1 (left side) and scenario 2 (right side)

5.2. Scenario 1 - Constant grid power supply

As a very first goal of the optimization, building consumption and solar production are not taken into account. Figure 9 shows the results obtained after testing the agent in a typical day, comparing the scheduling generated by the intelligent agent as opposed to the traditional charging mechanism, the system currently implemented.

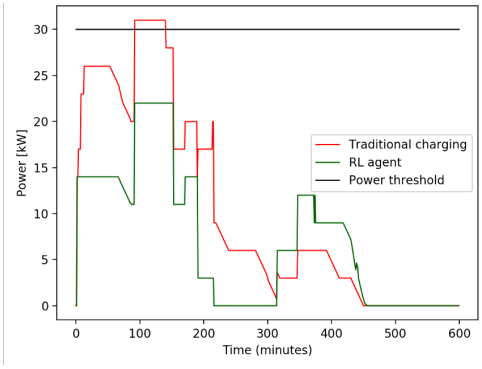

Figure 9: Charging scheduling found by the RL agent (green curve) compared against the baseline currently implemented (red curve) for scenario 1

First, it can be noticed that the total cumulative charging load due to EVs charging controlled by the intelligent agent, denoted by the green curve, never exceeds the power line. It can be seen how the traditional charging algorithm (red curve) fails in the morning, when the occupancy undergoes a sudden peak due to all the employees arriving at the office. The agent demonstrates to be aware of the danger and it doesn't charge a lot during the morning peak, avoiding in this way the failure. It is also important to notice that a part of the load has been shifted later in the day. As it has been explained in section 4.2, the agent can pick an action out of the 32 available, which index increases proportionally with the number of EVs charging. Figure 10 shows the policy adopted by the agent in the first scenario. The blue line represents the actions selected by the agent at every time step, while the green area in the background stands for the occupancy of that specific day considered.

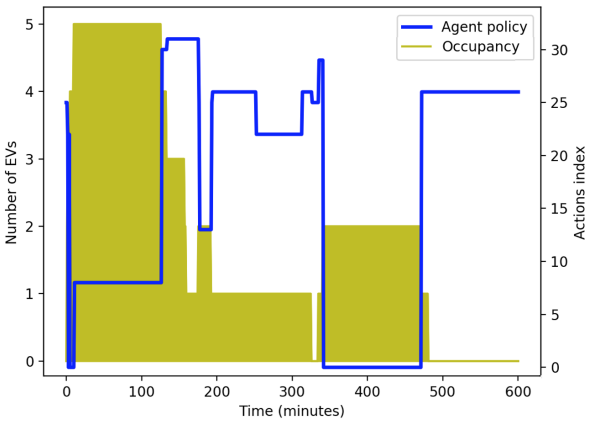

Figure 10: Policy adopted by the agent compared against the occupancy in scenario 1

It is interesting to notice that the agent's policy is strongly affected by the level of occupancy. In fact, it is more conservative in the morning and early afternoon, where the occupancy has its peaks and more audacious when the parking lot is nearly empty. The agent reacts very well to the arrival/departure scheduling of the EVs without getting any direct information about the time of the day from the environment, and just knowing about the presence or absence of the cars.

5.3. Scenario 2 - Smart office building

The big challenge given by this second scenario is that the power threshold is not constant anymore, but instead variable with time. Figure 11 shows the results of the simulation during a scheduling day. The test is now more ambitious for the agent, because the power threshold varies during time and it reaches its minimum points during the occupancy's peaks. As it can be seen from the picture, the smart agent (green curve) successfully manages to avoid the failure and charge as much as it can the vehicles without exceeding the limit. However, a different behavior might be expected in the second part of the day, since the algorithm solved the problem by not charging any socket during the afternoon. Here

it should be expected that the agent would take advantage of the uncharged cars left in the parking lot in order to complete the transactions. From how it can be seen in figure 12, the agents favors especially one action over the others, which corresponds to a specific combination of EVs charging and not charging.

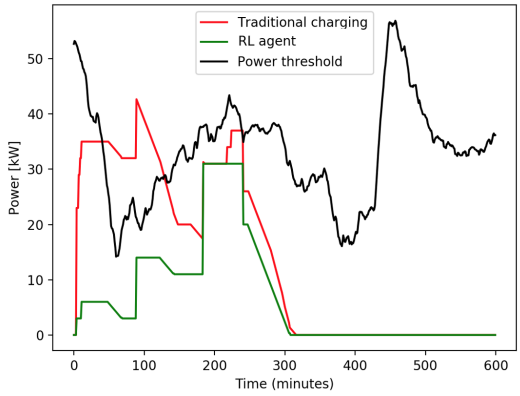

Figure 11: Charging scheduling found by the RL agent (green curve) compared against the baseline currently implemented (red curve) for scenario 2

As a consequence, the energy is not homogeneously spread among the different charging sockets, which is with all probabilities the biggest limitation encountered by this approach. This behavior is possibly due to the fact that the reward condition related to the unnecessary actions has too much weight and discourage the agent of exploring other possibilities.

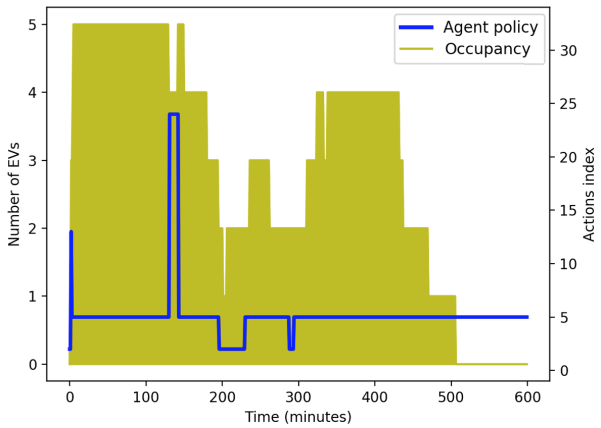

Figure 12: Policy adopted by the agent compared against the occupancy in scenario 2

5.4. Scenarios comparison

The two scenarios have been evaluated in the light of three main aspects: the *failures frequency*, the amount of *energy transferred* and the number of *power adjustments*, namely the number of actions performed by the agent. A set of ten independent days has been simulated, which data have been collected and analysed in the following sections. It has been thought that an interesting way to evaluate the gravity of failing was to measure the amount of energy that the agent would transfer where it would not be allowed to (namely, above the power threshold line), also defined as *failures volume*.

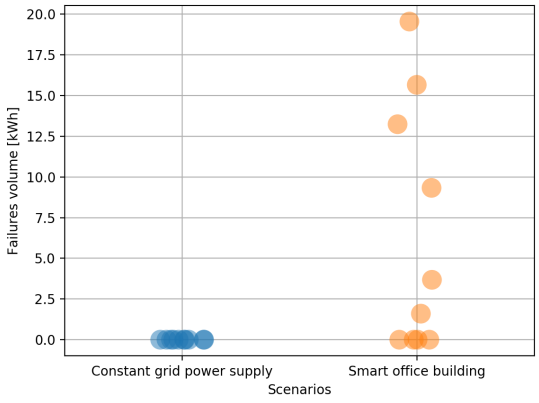

Figure 13: Comparison between scenario 1 and scenario 2 in the failures frequency

Figure 13 shows that, in the first scenario, the agent seems to learn a policy that efficiently accomplishes this task, while in the second scenario failure occurs a few times reaching up to 20kW. (figure 13). The failure volume has decreased up to around 80% on average in the 10 independent days considered.

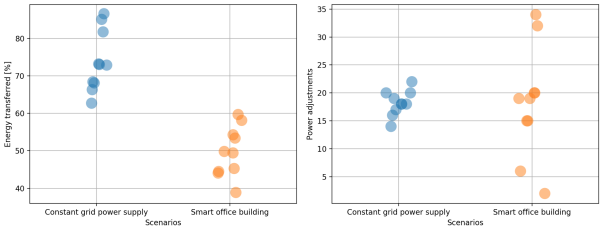

Figure 14: Comparison between scenario 1 and scenario 2 in the energy transferred

The other crucial metric of the problem is the percentage of energy transferred, on average, to every EV. Figure 14 (left side) shows a scatter plot of the values of energy transferred in the ten runs considered for the two different agents. Once again, it can be seen the difference in the two scenarios: in the first one, the average is permanently above the 70%, while in the second one it ranges between 40% and 60%. Figure 14 (right side) shows the number of power adjustments sent from the agent to the central system in order to change the scheduling of the EVs charging. The results in this case are highly

affected by the policy learned by the agent during training. The charging schema changes on average around 20 times, but if this number remains stable in scenario 1, it fluctuates in scenario 2.

6. Conclusions

In this paper, the problem of optimally scheduling the charging of a fleet of electric vehicles has been addressed. The main objective was to avoid that the cumulative load from the charging processes would threaten the health of the energy system, both locally and from the perspective of the power grid. To this purpose, *reinforcement learning* concepts have been explored. An *environment* has been created in order to reproduce at best complex dynamics, such as the basic functioning of a charging station and the stochastic EVs arrival and departure functions. Multiple techniques have been combined to create a unique architecture, due to the fact that important information was not readily available but needed to be obtained. The proposed algorithm has been confronted with a *traditional charging* simulated behavior, comparison that revealed strengths and limitations of the proposed implementation. Moreover, the performance of the intelligent agent has been evaluated in two different scenarios, looking at tailored metrics as *failures frequency*, *energy transferred* and *number of power adjustments*. Results have showed that the developed algorithm fails less than the baseline, with a reduction of the load due to EVs charging of 80% during the peak times of the day. The amount of energy transferred varies with the scenario considered: 75% in the case of constant grid power supply and around 50% considering the smart office building as a whole.

References

[1] Smart charging of electric vehicles.

[2] Innovation outlook: Smart charging for electric vehicles, 2019.

[3] J. Brownlee. Neural networks are function approximation algorithms.

[4] Q. Dang, D. Wu, and B. Boulet. A q-learning based charging scheduling scheme for electric vehicles. In *2019 IEEE Transportation Electrification Conference and Expo (ITEC)*, pages 1–5, 2019.

[5] M. N. et al. Review of positive and negative impacts of electric vehicles charging on electric power systems. *MDPI energies*, 13(4675), 2020.

[6] X. F. et al. Energy management of smart home with home appliances, energy storage system and electric vehicle: A hierarchical deep reinforcement learning approach. *MDPI energies*, 2019.

[7] A. Géron. *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow*. O'Reilly, 2019.

[8] A. G. Hado van Hasselt and D. Silver. Deep reinforcement learning with double q-learning. 12 2015.

[9] T. K. Ho. Random decision forests. 1995.

[10] M. Lapan. *Deep Reinforcement Learning Hands-On*. Packt, 2020.

[11] S. Lee and D.-H. Choi. Energy management of smart home with home appliances, energy storage system and electric vehicle: A hierarchical deep reinforcement learning approach. *MDPI sensors*, 2020.

[12] E. Mocanu, D. C. Mocanu, P. H. Nguyen, A. Liotta, M. E. Webber, M. Gibescu, and J. G. Slootweg. On-line building energy optimization using deep reinforcement learning. *IEEE Transactions on Smart Grid*, 10(4):3698–3708, 2019.

[13] A. G. B. Richard S. Sutton. *Reinforcement Learning: An Introduction*. 2014-2015.

[14] E. Sortomme and M. A. El-Sharkawi. Optimal scheduling of vehicle-to-grid energy and ancillary services. *IEEE Transactions on Smart Grid*, 3(1):351–359, 2012.

[15] 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*, 10(5):5246–5257, 2019.

[16] T. Wei, Yanzhi Wang, and Q. Zhu. Deep reinforcement learning for building hvac control. In *2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6, 2017.

[17] D. Wu, H. Zeng, C. Lu, and B. Boulet. Genetic algorithm for optimal charge scheduling of electric vehicle fleet. 2019.

[18] Z. Zheng and S. Yang. Particle swarm optimisation for scheduling electric vehicles with microgrids. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–7, 2020.