KU LEUVEN

FACULTEIT INGENIEURSWETENSCHAPPEN

Data-efficient reinforcement learning for low-voltage grid optimization using transfer learning

Davy Didden
Nadia Wiesé

Thesis voorgedragen tot het behalen van de graad van Master of Science in de ingenieurswetenschappen: energie

Promotor:
Prof. dr. ir. Johan Driesen

Assessoren:
Prof. dr. ir. Geert Deconinck
Dr. ir. Christina Protopapadaki

Begeleider:
Dr. ir. Hussain Kazmi

Academiejaar 2019 – 2020

Preface

Ten years ago our educational journey started when meeting in high school. Our common passion for energy and engineering led to our first master's degree at the University of Antwerp eight years later. Up for one more challenge, we decided to start a follow-up degree at the KU Leuven and bundled our forces, resulting in the work presented here. It is fair to say our educational path has come to an end, but a life-long friendship has been given in return.

As engineers-to-be with an expertise in energy, we believe that investments in distribution systems are essential to facilitate the electric evolution towards a smart grid. In order to support a more efficient system, new grid control applications should be developed and implemented. To provide adequate inputs for these applications and to solve the current blind spots of distribution system operators in their electricity systems, more detailed monitoring is needed. Due to our passion for control strategies and demand side management systems, the subject of this thesis drew our attention. With this work, we hope and aim to contribute to this ongoing smart and green revolution.

We would first like to thank our promoter Prof. Dr. Ir. Johan Driesen and our thesis advisor Dr. Ir. Hussain Kazmi. As a thesis advisor, Hussain was always willing to help whenever we had a question about our research or writing. We could not have wished for better support! Also, we want to express our gratitude to our parents, friends and beloved ones for the continuous encouragement throughout our years of study and during our research and writing of this thesis. Finally, we want to thank our professors for passing on their knowledge and for sharing their passion about their own research field.

Davy Didden
Nadia Wiesé
10 January 2020

Contents

Abstract

Increasing proliferation of distributed energy resources and electrification of demand in the residential building sector can help realize global decarbonization targets in a cost-effective manner. However, this green revolution has entailed a rapid transformation of the classical power system creating new challenges for the distribution system operator, such as reduced power quality and network congestion issues. Automated demand response control strategies provide a solution to these problems by exploiting available energy flexibility in the low-voltage grid at interesting times. Battery energy storage and photovoltaics (PV) curtailment are particularly suited for this approach, offering a competitive alternative to grid reinforcement investments.

Reinforcement learning, a data-driven control method, is especially suitable for control in which self-adaptability is required through continuous agent-environment interaction. However, a significant shortcoming of such data-driven methods is their large data-inefficiency, which often translates into infeasible amounts of training data necessary to achieve adequate controller performance. The concept of transfer learning helps to circumvent these disadvantages: a controller is trained with data in one domain and the gained knowledge is subsequently transferred to a different but related control task. It is shown that both initial and asymptotic performance can be greatly enhanced by utilizing these principles.

In this thesis, we propose a reinforcement learning based controller trained with real-world data from Belgium and The Netherlands and research the potential of transfer learning for optimal control strategies in limited data domains. Since many demand response applications require scaling to a multi-agent setting, we research these control methods in both a single-agent and multi-agent setting. The former entails the operation of a centralized grid battery, whereas the latter focuses on three distributed, independently acting agents controlling a local battery and PV installation.

It is found that the grid impact of renewable energy sources and electrified demand equipment can be mitigated adequately with the suggested controller and that transfer learning improved the performance of the RL controller significantly. These results demonstrate that transfer learning methods provide a solution for optimal control strategies in limited data domains and can be used to accelerate the learning process of RL based systems in real world problems.

Samenvatting

De groei van hernieuwbare energie en de elektrificatie in de residentiele sector leveren een bijdrage in de strijd tegen de uitstoot van broeikasgassen en helpen bij het behalen van de klimaatdoelstellingen. Desalniettemin brengt deze snelle transformatie van het klassieke elektriciteitsnetwerk nieuwe uitdagingen met zich mee voor de distributienetwerkbeheerders, zoals netwerkoverbelastingen en power quality problemen. Geautomatiseerde demand response controle strategieën bieden een oplossing jegens deze problemen, daarbij gebruikmakend van de aanwezige flexibiliteit in het laagspanningsnet, zoals opslag in batterijen of het afvlakken van PV generatie, wat kan helpen om uitbreidingen en versterkingen van het netwerk uit te stellen.

Reinforcement learning, een data gedreven controlemethode, is uitermate geschikt voor controlestrategieën waarbij een automatisch aanpassingsvermogen vereist is door middel van een continue interactie tussen de controller (agent) en omgeving. Deze data gedreven methoden zijn echter zeer data-inefficiënt en daardoor vereisen ze een significante hoeveelheid trainingsdata alvorens ze een adequaat resultaat bieden in de praktijk. Transfer learning helpt om deze nadelen te omzeilen: eerst wordt een controller offline getraind met beschikbare data, waarna de verkregen informatie overgedragen wordt (transfer) aan de controller die online functioneert in de praktijk.

In deze thesis presenteren we een ontwerp van een controller, gebruikmakend van reinforcement learning, die getraind wordt met reële data van België en Nederland. We onderzoeken de mogelijkheden van transfer learning in optimale controle strategieën in gelimiteerde datadomeinen. Bijkomend vereisen vele reële praktische problemen een multi-agent setting. Hiervoor onderzoeken we de vermelde controlemethoden zowel in een single-agent setting als in een multi-agent setting met drie onafhankelijk werkende controle eenheden.

Het onderzoek wees uit dat de impact van hernieuwbare bronnen en warmtepompen op het laagspanningsnetwerk beperkt kan worden met de voorgestelde controller en dat transfer learning de performantie van de reinforcment learning controller beduidend verbeterd. Deze resultaten tonen aan dat transfer learning-methoden een oplossing bieden voor optimale controle strategiëen in gelimiteerde datadomeinen en dat ze het leerproces van reinforcement learning systemen kunnen versnellen in praktische problemen.

List of Figures

List of Tables

List of Abbreviations and Symbols

Abbreviations

ANN	Artificial Neural Network
CHP	Combined Heat and Power
DG	Distributed Generation
DSO	Distribution System Operator
DR	Demand Response
DRL	Deep Reinforcement Learning
DQN	Deep Q-Network
DQL	Deep Q-learning
EV	Electric Vehicle
HV	High-Voltage
HVAC	Heating Ventilation and Air Conditioning
IL	Independent learner
LV	Low-Voltage
MARL	Multi-Agent Reinforcement Learning
ML	Machine Learning
MPC	Model Predictive Control
MV	Medium-Voltage
PV	Photovoltaics
RES	Renewable Energy Sources
RL	Reinforcement Learning
SARL	Single-Agent Reinforcement Learning
SoC	State of Charge
TL	Transfer Learning
TSO	Transmission System Operator

Symbols

Power systems:

E	Energy
P	Active power
Q	Reactive power
R	Resistance
U	Voltage (European notation)
V	Voltage (American notation)
X	Reactance
Z	Impedance

Reinforcement learning (following the notation of [1]):

$X \sim p$	Random variable X selected from distribution $p(x) \doteq Pr(X = x)$
$Pr(X{=}x)$	Probability that a random variable X takes on the value x
$\mathbb{E}[X]$	Expected value of a random variable X, i.e., $\mathbb{E}[X] = \sum_x p(x)x$
s, s'	States
a	An action
r	A reward
$\mathcal{S}$	Set of all states
$\mathcal{A}(s)$	Set of all actions available in state s
$\mathcal{R}$	Set of all rewards
A_t	Action at time step t
S_t	State at time step t
π	Policy
π_*	Optimal policy
$\pi(a\|s)$	Probability of taking action a in state s under stochastic policy π
G_t	Return following time step t
$p(s', r\|s, a)$	Dynamics function
$p(s'\|s, a)$	Transition function
$v_\pi(s)$	Value of state s under policy π
$v_*(s)$	Value of state s under the optimal policy π_*
$q_\pi(s, a)$	Value of taking action a in state s under policy π
$q_*(s, a)$	Value of taking action a in state s under the optimal policy π_*
C	Target network update frequency
V, V_t	Array estimates of state-value function v_π or v_*
Q, Q_t	Array estimates of action-value function q_π or q_*
α	Learning rate
γ	Discount factor
ϵ	Probability of taking a random action in an ϵ-greedy policy

Transfer learning:

$\mathcal{D}$	Domain
$\mathcal{T}$	Task
$\mathcal{D}_\mathcal{S}$	Source domain
$\mathcal{D}_\mathcal{T}$	Target Domain
$\mathcal{T}_\mathcal{S}$	Source task
$\mathcal{T}_\mathcal{T}$	Target task
$\mathcal{X}$	Feature space
$\mathcal{Y}$	Label space
$P(X)$	Marginal probability distribution
x_i	Feature vector corresponding to some input
X	Learning sample

Chapter 1

Introduction

1.1 Situation

The European Union has set itself stringent targets concerning the reduction of greenhouse gas emissions: a minimum of 40% CO_2 reduction by 2030 and 80% by 2050 has to be achieved [2]. The decarbonization of the residential building sector, which is responsible for 22% of the global energy consumption, is one of the possible methods to combat anthropogenic climate change [3]. A shift is needed towards a more efficient and sustainable electric power system in order to contribute to the ongoing green energy revolution at the residential level [4].

One of the key developments is the increasing amount of renewable energy sources (RES) integrated into the low-voltage grid, such as photovoltaics (PV), which can reduce CO_2 emissions and improve the autonomy of residential consumers [4]. Another emerging phenomenon is the electrification of residential heating appliances (heat pumps) and transportation (electric vehicles). Combination of the aforementioned - increasing electrification powered by distributed RES - is a potential solution to the required decarbonization of the residential sector [5].

However, the rapid transformation of the classical power system entailed by this green energy revolution creates new challenges for the distribution system operator (DSO). Implementation of RES in low-voltage grids and the increasing electrification can lead to reduced power quality and network congestion. Moreover, grid violations are a limiting factor for the increasing implementation of RES in low-voltage networks. Since distribution grids are not designed to accommodate these changes, potential benefits of decarbonization policies might be diminished [4, 5].

In order to provide a solution towards a smart grid, automated demand response applications (DR) can help to cope with the aforementioned issues. DR is a valuable asset for distribution network voltage regulation and congestion management [2, 4]. A broad range of control strategies exists for the implementation of DR applications making use of the available energy flexibility in the grid. Amongst others, the options include electrified demand appliances, battery storage, and PV curtailment [6].

In this study we will use reinforcement learning (RL) to design a controller with the aim of determining an optimal policy to keep the grid within safe operating limits using PV curtailment and battery flexibility. One of the advantages of RL is its self-adaptability through continuous agent-environment interaction. However, a significant shortcoming of such data-driven methods is their high sample complexity, which means they require a large amount of interactions with the environment in order to learn an (approximate) optimal control policy [7]. Due to this large data-inefficiency, large amounts of, often unavailable, training data is necessary in order to achieve adequate results.

In light of these limitations governing data-inefficiency, we propose a sample-efficient RL based controller that makes use of transfer learning (TL). Here, agents are trained in a simulated environment (in this case a low-voltage grid) with available data and subsequently implemented as a starting point for the controller in the "real" environment. This allows transferring gained knowledge from one domain to another. There is still a great deal of work to be done in the research area on transfer learning. Energy related papers on this topic study predominantly forecast related issues [8, 9, 10] or use transfer learning in a multi-agent cooperative setting [7].

Studies on RL techniques commonly focus on a single agent interacting with its environment. However, many real-world applications require scaling to a multi-agent setting [4, 6]. According to Vázquez-Canteli and Nagy [4], multi-agent reinforcement learning (MARL) is still in its infancy and comprises purely theoretical research, whereby convergence and stability are often observed for no more than two agents. We study in this thesis the potential of independent multi-agent systems in the field of distributed generation (DG) and storage, including 3 independently acting agents.

1.2 Problem statement and goals

This thesis was carried out in collaboration with i.LECO, a spin-off of the company Enervalis. Founded in 2019, they have the goal to enable and speed up the needed green energy transition with a focus on the future expected network structure of local energy communities. They focus on smart solutions related to residential housing, EV charging, and storage technologies [11].

In 2018, Enervalis concluded the REnnovates [12] project in the Netherlands. Here, 249 houses were renovated through thorough isolation and equipped with an air-sourced heat pump and a PV installation. The key idea was to transition the houses towards "zero on the meter houses" making them carbon neutral. The electricity consumption was monitored for one year long with three separated measurements for each house: the power generation from the PV installation, energy consumption of the heat pump, and other loads. The DSO, which was involved in this project, had no detailed visibility on the effects of the heat electrification and increased penetration of RES into the grid.

Within this setting, our study addresses the following three research questions:

- What are the effects of installing RES and heat pumps in zero on the meter houses on the distribution network power quality and grid congestion issues in context of the REnnovates project?
- Is transfer learning a solution for the data-inefficiency of RL based methods with high sample complexity in automated DR applications?
- Can grid violations be mitigated with the use of such sample-efficient RL based controller in case of a single and multi-agent scenario?

To answer these questions we define four key goals:

- Quantify the effect of the implementation of RES and heat pumps in terms of four grid violations: over- and undervoltages, line overloading and transformer overloading. Therefore, we perform an extensive data analysis on the REnnovates data using the topology of the Linear project in Flanders, since the grid topology of the REnnovates project was not available.
- We aim to create a fair and equitable playground for the RL based controller. Three rule based reference controllers are developed: house level battery control, house level PV curtailment and district level battery control. A comprehensive study on the sizing and placement of the batteries is needed to make a simulation with these controllers in the low-voltage grid environment.
- The third target is the design of the RL controller (which does not employ transfer learning). The goal of the controller is to maximize the efficiency and power quality of the low-voltage grid by using the flexibility of batteries and/or curtailment of the PV installations. More specifically, we aim to: i) minimize grid violations: over- and undervoltages, line overloading and transformer overloading; and ii) minimize losses due to battery charging or discharging and curtailment of the PV installations. Although it is out of the scope of this work to perform a detailed economical analysis, the DSO has a strong financial incentive to limit these grid issues and ensure security of supply towards its customers. We designed a deep Q-learning (DQL) controller for both a single- and multi-agent scenario, the latter incorporating three independently acting agents.
- The main objective is the design of a sample-efficient RL based controller that makes use of transfer learning. This is done for two cases: one scenario where the grid topology is known and another where the topology is unavailable, so training has to be performed on a different grid topology with different battery sizing. The former case is studied for both the single- and multi-agent controller, whereas the different grid topology case is researched for the single-agent case only. We compare the performance of the proposed controller with the rule based controllers and the baseline RL based controller which does not employ transfer learning.

1.3 Overview

This thesis is organized as follows: chapter 2 provides the reader with a literature study on demand response control strategies with focus on reinforcement learning methods and transfer learning. In chapter 3, an extensive data analysis is performed in order to outline the problem statement in this thesis. Chapter 4 subsequently presents the description and comparison of the rule-based controllers to create a baseline for the DQL-controller. The utilized model of the environment is described as well. Additionally, the sizing and placement of the batteries are explained. Chapter 5 discusses the design and performances of the DQL controller for the single- and multi-agent case without employing transfer learning. In chapter 6, all of these elements are brought together by focusing on the potential of RL controllers using transfer learning in demand response applications. We compare the performance of the proposed controller with the rule-based controllers from chapter 4 and the normal baseline RL based controller from chapter 5. Finally, chapter 7 concludes the thesis and highlights the possibilities for future research.

Chapter 2

Reinforcement learning based controllers for demand response applications

Large scale roll-out of distributed RES due to the global concern on climate change, has led to a rapid transformation of the classical power system. As a result of their inherent intermittency and instantaneous mismatch between generation and consumption, implementation of RES can lead to reduced voltage quality and network congestion. To cope with these issues, a broad range of applied control strategies making use of available energy flexibility in distribution networks exists. In this chapter we take a closer look at the position of reinforcement learning in this context. The theory underlying this concept is presented and a specific algorithm, deep Q-learning (DQL), utilised further in this work is described. Finally, the concepts of transfer learning is reviewed.

2.1 Energy flexibility for grid optimization

2.1.1 Grid issues

In wake of the green energy revolution described in chapter 1, the share of distributed generation in classical power systems is increasing rapidly [4]. Amongst others, residential solar generation, combined heat and power (CHP) installations and wind plants have been installed in large numbers, with a further increase expected in the upcoming decades [13]. Electrification of non-electric energy vectors, residential heat pumps and electric vehicles being the prime examples, adds further to the equation [4]. Inevitably, the role of the DSO has to be adjusted accordingly. The operation of the network has changed from passive to active network management, with voltages and power flows no longer set by a simple top-down centralized generation to decentralized consumption architecture. Consumers are now prosumers and bi-directional power flow is no longer an exception.

TABLE 3.2 STANDARD EN 50160 – SUMMARY FOR CONTINUOUS PHENOMENA

Voltage disturbance	Voltage level	Voltage quality index (limit)
Supply voltage variations	LV	• 95% of the 10 minute mean r.m.s values for 1 week (± 10% of nominal voltage)
		• 100% of the 10 minute mean r.m.s values for 1 week (+ 10% / - 15% of nominal voltage)
	MV	• 99% of the 10 minute mean r.m.s values for 1 week below +10% of reference voltage and 99% of the 10 minute mean r.m.s values for 1 week above -10% of reference voltage
		• 100% of the 10 minute mean r.m.s values for 1 week (± 15% of reference voltage)
Flicker	LV, MV, HV	• 95% of the P_{lt} values for 1 week, should be less than or equal to 1
Unbalance	LV, MV, HV	• 95% of the 10 minute mean r.m.s values of the negative phase sequence component divided by the values of the positive sequence component for 1 week, should be within the range 0% to 2%
Harmonic voltage	LV, MV	• 95% of the 10 minute mean r.m.s values for 1 week lower than limits provided by means of a table
		• 100 % of the THD values for 1 week (£ 8%)
	HV	• 95% of the 10 minute mean r.m.s values for 1 week lower than limits provided by means of a table
Mains signalling voltages	LV, MV	• 99% of a day, the 3 second mean value of signal voltages less than limits presented in graphical format

Figure 2.1: Voltage quality disturbances with indicative limits following the EN 50160 standard. Remark that in low-voltage networks the voltage magnitude has to be within ±10% of the nominal level. [14]

The role of the DSO is to accommodate the distribution of electricity from the high-voltage (HV) grid to the individual installations at medium- and low-voltage (MV and LV) level. Summarized, the DSO is responsible for the safe operation, development and maintenance of the distribution network [15]. The former includes ensuring the network is operating within acceptable limits. A key concept to quantify this behaviour is voltage quality, the different elements of which include [16]:

- Voltage frequency has to be confined at 50 (or 60) Hz;
- Voltage magnitude has to be within acceptable limits of the nominal level;
- Sinusoidal shape has to be maintained as close as possible;
- Operational reliability has to be ensured at all times (security of supply).

Any deviation from one of these characteristics is said to result in a reduced voltage quality. In Europe, the standard EN 50160 [14] gives an overview of all voltage quality disturbances with indicative limits. Figure 2.1 summarizes.

Returning to the issue of increasing distributed generation and electrification, a multitude of studies [17, 18, 19] show that excessive local power generation - e.g. residential PV production exceeding instantaneous load consumption - predominantly effects voltage quality through pushing the voltage magnitude outside of the statutory limits. Additionally, network congestion can arise. As will be shown in chapter 3, the same issues are quantified with the data and topologies used throughout this work. To alleviate these problems, a scala of voltage control techniques is available. The next section highlights the different control possibilities and indicates the sources of flexibility used towards this end in the rest of the thesis.

2.1.2 Voltage control in distribution networks

To quantify the voltage magnitude issue described in the previous section, the simplified representation of a radial distribution network in figure 2.2 is analyzed. A strong grid (assume V_1 constant) is connected to a PV installation through a distribution line with impedance $Z = R + jX$. Both load and PV exchange active power (P) and reactive power (Q) with the network.

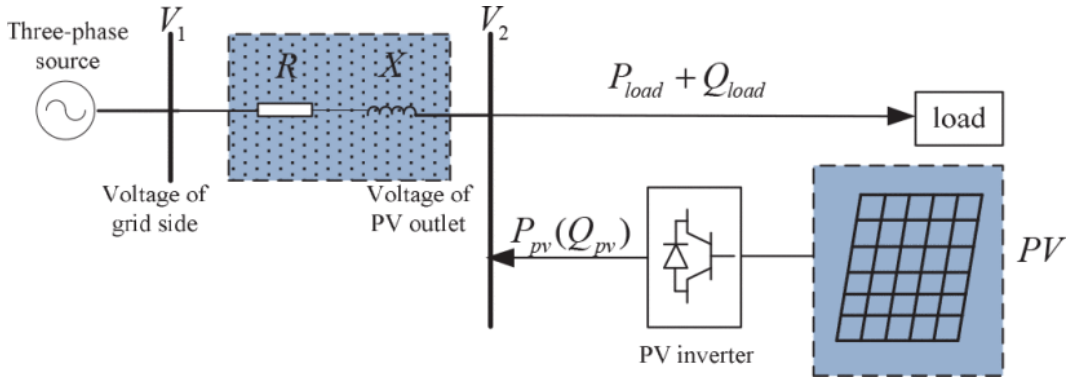

Figure 2.2: Simplified equivalent circuit of a radial feeder connecting a voltage source and grid-connected PV system over a line impedance. [17]

The voltage at the consumer side V_2 can be approximated by [17]:

$$V_2 \approx \frac{V_1}{2} + \sqrt{\left(\frac{V_1}{2}\right)^2 - (P_{load} - P_{PV})\,R + (Q_{load} - Q_{PV})\,X} \tag{2.1}$$

In general, the R/X ratio in a distribution network is large compared to that of transmission systems [18]. Closer examination of equation 2.1 consequently tells us that any significant active power injection will lead to a non-negligible increase of the local voltage level V_2. Vice versa, substantial load consumption would lead to a decrease in voltage magnitude. To keep these voltage variations within acceptable limits, a range of solutions have been proposed [17, 18]:

- **On-load tap changers (OLTCs)**: a common method for voltage regulation is to regulate the secondary voltage of the distribution system transformer. It is a very effective voltage regulation method, but due to slow tap change duration (3-10 minutes) not suitable for highly dynamic system regulation.
- **Reactive power control devices**: the approach above shows that any alteration in active power can be conversely countered by a change in reactive power to limit voltage fluctuations. This is the main idea behind generator power factor control (PFC) or static synchronous compensators (STATCOMs). In this work we solely focus on active power voltage regulation techniques.
- **Network asset upgrades**: equation 2.1 shows that a smaller impedance can mitigate voltage fluctuations. This network reinforcement method, through upgrading of existing feeder configurations, is a straightforward but often economically infeasible solution [18].

- **Demand side management (DSM)**: DSM refers to any initiative or technology that encourage consumers to adjust their energy usage in a favorable way [6]. Distinction is made between energy efficiency improvements and demand response (DR). The latter focuses on providing incentives to consumers to adjust their consumption at interesting times. Through DR existing energy flexibility in the network can be employed to solve voltage issues. The next section further elaborates this option.

2.1.3 Energy flexibility and demand response

Many definitions of energy flexibility exist. The view used throughout this work envisions energy flexibility as a service, which allows DR based on the requirements of the grid [6]. Important in this context is the availability of a supportive and enabling regulatory framework. The EU legislation entails existing provisions - particularly the Third Energy Package Electricity Directive [20] and the Energy Efficiency Directive [21] - making demand response possible. This framework creates the necessary obligations on member states, regulators, TSO and DSO to enable and promote demand response, allowing the market to develop.

Energy systems with high potential for DR applications can be divided in four major groups: heating ventilation and air conditioning (HVAC), smart appliances, electric vehicles (EVs), and distributed generation with energy storage [4, 6]. In this thesis the focus is on the latter. In the researched topology described in chapter 3, energy flexibility is available under the form of battery energy storage and residential PV curtailment. The exact implementation of both options in the reinforcement learning setting is discussed in detail in chapter 5.

2.2 Demand response control strategies

2.2.1 Review of control possibilities for DR in smart grids

In the previous section, the flexibility of battery storage and PV curtailment have been indicated as valid resources for distribution network voltage regulation and congestion management. The main objective is to develop a control model capable of determining an (approximate) optimal policy for the given control objective, in this case keeping the grid within safe operating limits. Control theory is the branch of engineering focusing on implementation and development of such models [6].

Two main types of control strategies can be distinguished: control of a single system component (local control) or control of the entire energy system as a whole (supervisory control) [6]. Local controllers ensure process stability and accurate tracking of setpoints, whereas supervisory controllers regulate the local controllers with an eye on smooth system operation. A further distinction can be made between classical control, hard control, soft control, hybrid control and other control techniques [22]. Figure 2.3 gives a non-exhaustive overview of the most common control methodologies for DR based applications within these categories.

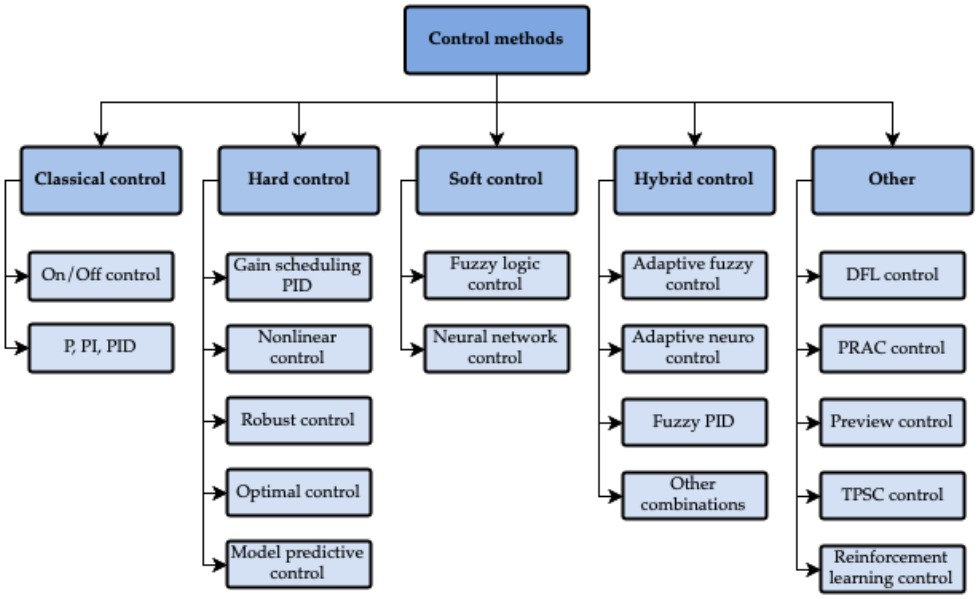

Figure 2.3: Overview of control methods for demand response. [6]

Classical control entails some of the most common techniques, including on/off and P/PI/PID approaches. The former show strength in their simplicity, but are unable to control dynamic systems with time delays [22]. The latter tune a control variable based on an error signal comprising a custom setpoint and sensory process information, but only perform well when operating conditions do not differ from tuning conditions too strongly [6].

Hard control architectures include gain-scheduling PID (improved stability in comparison with regular PID, but requires manual tuning); nonlinear control (effective but complex mathematical design and identification of stable states); robust control (good performance under change of parameters and time-varying dynamics, but robustness can be difficult to obtain due to the variable nature of energy systems); optimal control (optimization of an objective function, same issues as robust control) and model predictive control (MPC, relying on a dynamic model of the process to forecast the system's future state allowing optimization of the current timeslot, while keeping future timeslots in consideration) [6].

Soft control comprises, amongst others, neural network control and fuzzy logic control. In the former, an artificial neural network (ANN) is trained to learn non-linear inverse system dynamics, but large data training sets are required [23]. The latter varies from digital control by working with analogue input signals varying between 0 an 1, allowing elements of human thinking to be integrated in controller design [24].

Combinations of the previous architectures leads to intertwined models denoted as hybrid control strategies. The main goal is to combine advantages of both soft- and hard control. For more information the reader is referred to [6, 22].

2.2.2 Reinforcement learning based controllers

Reinforcement learning is a machine learning (ML) approach where an agent learns from direct interaction with its environment to achieve a predetermined goal, without the need of exemplary supervision or complete models of that environment [1]. A summary of the theoretical framework is given in section 2.3.

Reinforcement learning has been applied in the DR setting as a control strategy to a wide range of energy systems, including HVAC, EVs, smart appliances and DG with energy storage [4]. Vázquez-Canteli and Nagy give an extensive literature review of algorithms and modelling techniques which implement reinforcement learning for demand response in [4]. They found a steep increase in number of publications involving RL after 2012. The reason for this increasing interest is multifold:

- To ensure future success, the economic savings generated by DR must outweigh the dissatisfaction caused to consumers [4]. Through its capability of learning through interaction without an extensive model of the environment, RL is especially suited for ***integration of human feedback*** in control algorithms.
- Reinforcement learning methods often require large datasets to train, but have the advantage of being able to ***learn offline*** from historically collected experiences. The concept of transfer learning, training a controller on one dataset to improve performance and accelerate learning in new environments, shows interesting possibilities [25, 26]. Section 2.5 elaborates on this topic.
- A major advantage of RL over other algorithms is its ***self-adaptability***. Settings with non-stationary environments require active adaptation of the learning agent, which is achieved through continuous agent-environment interaction.
- Finally, reinforcement learning is highly advantageous in complex environments because of its ***model-free nature*** [1, 4]. We highlight the differences with a model-based approached in the next section.

2.2.3 Model free vs. model based

Reinforcement learning methods can be both model-based or model-free. In the former, the agent first captures the dynamics of the system explicitly by estimating the transition probabilities of state-action pairs [1]. The RL problem is then reduced to a planning problem: deciding on an optimal action sequence based on possible future situations before having actually experienced them. In a model-free approach, the controller neither learns nor possesses such model [6]. The agent learns solely on a trial-and-error basis through direct interaction with its environment [1].

Model-free RL algorithms can be very computationally efficient, adjust to non-stationary settings and offer control capabilities to environments for which regular models are too complex [1, 4]. Disadvantages compared to model-based approaches include the curse of dimensionality (see section 2.3) and the delayed reward problem. These issues often translate to a large data-inefficiency, demanding unreasonable amounts of training data [6, 26].

2.2.4 Multi-agent reinforcement learning (MARL)

Traditional RL techniques focus on a single agent interacting with its environment, e.g. one controller adjusting the setpoint of a grid-connected storage unit. Many real-world applications however require scaling to a multi-agent setting [4, 6]. Conventional control techniques for these kinds of issues involve centralized planners making supervisory decisions for all local control units [6]. With increasing number of agents these methods can quickly become infeasible [4, 6]. MARL provides an interesting alternative for these traditional algorithms.

The decentralized approach offered by MARL can remove the effect of communication delays and eliminate a single point of failure. A number of additional challenges arise however. In a DR context, for example, actions of the different agents injecting energy in the distribution grid influence voltages seen by other agents, thus creating a non-stationary environment. Without information on each others actions, calculations can become both computationally and data inefficient [27].

2.3 Deep reinforcement learning

2.3.1 Theoretical framework for the RL problem

Markov decision process (MDP)

The reinforcement learning problem can be formulated mathematically in the form of a so-called Markov decision process. Figure 2.4 indicates the key elements of the framework. This entire section is based on the reference RL handbook by Sutton and Barto [1]. We follow their notational conventions: capital letters are for random variables, lower case letters for specific values of random variables.

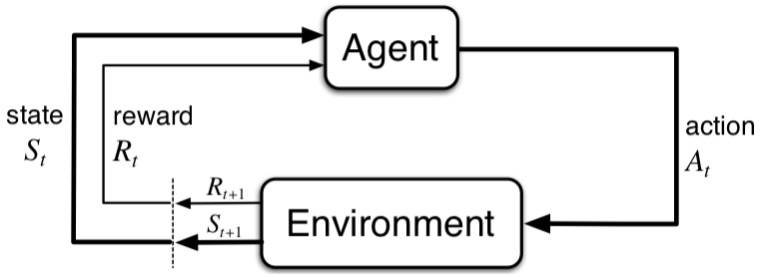

Figure 2.4: The agent-environment interaction in an MDP [1].

An agent and environment interact with each other at a sequence of discrete time steps t_k. After observing the state of the environment $S_t \in \mathcal{S}$ at time t, the agent takes an action $A_t \in \mathcal{A}(s)$. As a response to this action, the environment returns a reward $R_{t+1} \in \mathcal{R}$ and transitions towards a new state S_{t+1}. When the sets of all possible states $\mathcal{S}$, actions $\mathcal{A}$ and rewards $\mathcal{R}$ are finite, we speak of a finite MDP. In this case, the dynamics of the MDP are fully captured by the dynamics function p:

$$p(s', r|s, a) \doteq Pr(S_{t+1} = s', R_{t+1} = r|S_t = s, A_t = a) \in [0, 1]. \tag{2.2}$$

In words, this translates to the probability of the environment transitioning to state s' whilst returning a reward r, given action a was taken in the original state s. From this dynamics function all information about the environment can be extracted. In the literature, researchers often mention the state-transition probabilities, or transition function, which follows directly from the dynamics function:

$$p(s'|s,a) \doteq Pr(S_{t+1} = s'|S_t = s, A_t = a) = \sum_{r \in \mathcal{R}} p(s', r|s, a) \in [0, 1], \tag{2.3}$$

which is simply the probability of transitioning from state s to s' following action a. When p depends solely on the previous state S_t and action A_t, but to no extent on any earlier states and actions, the state is said to have the Markov property.

Goals, rewards and returns

The MDP framework gives a mathematical abstraction of learning through interaction with an environment to reach a specific goal. This goal is formalized through the use of a reward signal: $R_t \in \mathbb{R}$. In general, the agent's purpose is to maximize the total cumulative reward received in the long-term. Section 5.1.5 gives an in-depth description of the reward design for the RL problem presented in this work.

To quantify the concept of "cumulative reward" a new quantity is introduced: the return G_t. In general, it is defined as any combination of the rewards received following an action taken at time step t. To keep the return finite for continuing tasks (i.e. non-episodic tasks) a discount factor γ is introduced:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} = R_{t+1} + \gamma G_{t+1}. \tag{2.4}$$

$\gamma = 0$ leads to a myopic agent only focusing on maximizing immediate reward, whereas $\gamma = 1$ represents a farsighted agent attaching equal importance to future and immediate rewards.

Policies and value functions

A common thread uniting almost all reinforcement learning algorithms is their computation of (action-)value functions. As the name suggests, these are estimates for "how good" a certain state (or taking a specific action in a given state) is. Once again, this concept is formalized in terms of the expected return:

$$v_\pi(s) \doteq \mathbb{E}_\pi \left[G_t | S_t = s \right] = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) \left[r + \gamma v_\pi(s') \right] \tag{2.5}$$

$$q_\pi(s,a) \doteq \mathbb{E}_\pi \left[G_t | S_t = s, A_t = a \right] = \sum_{s',r} p(s', r|s, a) \left[r + \gamma q_\pi(s', a') \right]. \tag{2.6}$$

Here, $\pi(s|a)$ represents the policy the agent is following; a mapping from states to the probabilities of taken a certain action from those states. It is the goal of the

RL algorithm to find the optimal policy π_*. The state-value function $v_\pi(s)$ of a state s with respect to a policy π is defined as the expected return starting from s whilst following π. Similarly, the action-value function $q_\pi(s,a)$ is the expected return starting from s, taking action a and thereafter following π. Equations 2.5 and 2.6 are called the Bellman equations for v_π and q_π respectively. Applied to the optimal policy π_* this leads to the Bellman optimality equations, which is particularly interesting for the optimal action-value function:

$$q_*(s,a) = \mathbb{E}\left[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') | S_t = s, A_t = a\right]. \tag{2.7}$$

Thus, once one knows $q_*(s,a)$ it is fundamental to find the optimal action given a state s as $argmax_a(q_*(s,a))$. It is for this reason that approximation of $q_*(s,a)$ or $v_*(s)$ lies at the heart of almost all RL algorithms.

2.3.2 Solving the reinforcement learning problem

To find (or approximate) the optimal policy π_* for the RL problem, formulated as a finite MDP, many approaches exist. When the dynamics of the system are completely determined, i.e. the dynamics function $p(s', r|s, a)$ is known for all $s \in \mathcal{S}, a \in \mathcal{A}$, the problem is reduced to a planning problem: deciding on an optimal action sequence based on possible future situations before having actually experienced them. Dynamic programming (DP) algorithms, such as policy iteration or value iteration, can compute π_* in this way when a perfect model of the environment is provided.

Apart from their great computational expenses, the requirement to have perfect knowledge of their environment makes these algorithms of limited utility. Luckily, RL is also applicable when no perfect model is available. Two approaches exist: model-based and model-free RL. We refer to section 2.2.3 for a brief comparison of both methods. One of the most popular model-free RL algorithms is Q-learning [4, 1]. Here, the expected return $Q(s,a)$ after each transition $s \to s'$ is updated through:

$$\underbrace{Q_{i+1}(s,a)}_{\substack{\text{updated}\\ \text{Q-value}}} \leftarrow \underbrace{Q_i(s,a)}_{\substack{\text{current}\\ \text{Q-value}}} + \alpha \Big[\underbrace{r(s,a) + \gamma \max_{a'} Q_i(s',a')}_{\substack{\text{update}\\ \text{target}}} - \underbrace{Q_i(s,a)}_{\substack{\text{current}\\ \text{Q-value}}} \Big], \tag{2.8}$$

where α is the learning rate ($\alpha = 0$: no new information is learned; $\alpha = 1$: all previous information is overwritten). The update target is an approximation of equation 2.7 in two ways: i) it samples the expected values (so $\mathbb{E}$ is simply dropped and replaced by a single sample) and ii) the current estimate Q is used instead of q_*.

Major advantages of this algorithm are its simplicity, model-free and off-policy nature. The latter means it can make the Q-values converge to the optimal values Q_* (so the target policy is an optimal policy π_*) independent of the policy being followed during training (the behaviour policy, e.g. ϵ-greedy, see section 5.2.2). This allows learning from historical data without specific action selection [4].

2.3.3 From Q-learning to deep-Q-learning

Artificial neural networks (ANN) as function estimators

Q-learning is a discrete algorithm, meaning states and actions are represented in a tabular way. Many real-life problems however, such as the MDP faced in this work, are continuous in nature. Additionally, these tabular methods suffer from the curse of dimensionality: large state-action spaces greatly increase computational and memory requirements. To overcome these issues, the simple Q-table can be replaced by function estimators such as ANNs or other regression techniques [1, 4].

The interested reader is referred to [28] for a more in-depth review of ANNs. Summarized, an ANN is a non-linear function approximator consisting of a network of interconnected units called neurons [1]. The units in the input and output layers can be connected through one or more invisible layers, in this case we speak of a deep neural network. The ANN is trained on labeled input-output data, mostly by using the backpropagation algorithm in combination with a variant of gradient descent. For the purpose of this work, only a high-level understanding of these methods is required. The ANN can replace a Q-table with its ability to generalize from previous experiences, even for states which have never been encountered.

DQL with experience replay and periodic target updates

For the implementation of our DQL algorithm, we follow the approach proposed by Mnih et al. [29] published in Nature in 2015. The same iterative update rule as in regular Q-learning (equation 2.8) is used, but since the back-propagating optimizer already has a learning rate, the step-size parameter α is set to 1. The first key difference is the approximation of the optimal-action value function $q_*(s, a)$ by a DQN with parameters (weights and biases) θ:

$$q(s, a, \theta) \approx q_*(s, a). \tag{2.9}$$

Unfortunately, the usage of non-linear estimators can lead to unstable or divergent RL behaviour. Two additional concepts are introduced to address these issues:

- **Experience replay**: the agent's experience $e_t = (s_t, a_t, r_t, s_{t+1})$ is stored in the replay memory $D_t = (e_1, \cdots, e_t)$ at each time step. Subsequently, a uniformly random minibatch is sampled from D_t to perform a Q-learning update on the DQN (the ANN is "fitted" to the samples in the minibatch). This causes experiences to be used (potentially) in many network updates, enhancing data efficiency. Additionally, the correlation between consecutive samples is broken and changes in the data are smoothed out.
- **Periodic target updates**: to further improve stability, a separate ANN $\hat{q}(s, a, \hat{\theta})$ is used to generate the targets for the Q-learning updates. This second DQN is only synchronised with the main model every C steps, avoiding the need to track a constantly changing target by adding a delay between the time Q is updated and the time this update affects the Q-learning targets, effectively reducing oscillations and enhancing convergence.

Algorithm 1: Deep Q-learning with experience replay [29].

input : Replay memory size N, minibatch size S, target network update frequency C, exploration probability $\epsilon \in [0, 1]$

Initialize replay memory D with size N;
Initialize action-value function q with random weights θ;
Initialize target action-value function $\hat{q}$ with weights $\hat{\theta} = \theta$;
for *episode=1,M* **do**
  Initialize state s_1 by resetting the environment;
  for *t=1,T* **do**
    With probability ϵ select random action a_t;
    otherwise select greedy action $a_t = argmax_a q(s_t, a, \theta)$);
    Pass action a_t to environment and observe state s_{t+1} and reward r_{t+1};
    Store experience $e_t = (s_t, a_t, r_{t+1}, s_{t+1})$ in replay memory D;
    Sample random minibatch $(s_j, a_j, r_{j+1}, s_{j+1})$ with size S from D;
    Set update targets (for each sample in minibatch):

$$y_j = \begin{cases} r_j, & \text{if episode terminates at step j+1.} \\ r_j + \gamma \max_{a'} \hat{q}(s_{j+1}, a', \hat{\theta}), & \text{otherwise.} \end{cases}$$

    Perform a gradient descent step on $(y_j - q(s_j, a_j, \theta))^2$ w.r.t. θ;
    Reset $\hat{q} = q$ every C steps;
  end
end

The general implementation of these methods is illustrated in algorithm 1. Usage of the ϵ-greedy policy and its importance in the exploration-exploitation dilemma is explained in a more practical context in section 5.2.2.

2.3.4 Multi-agent deep reinforcement learning

The general context of using MARL within decentralized DR applications was briefly discussed in section 2.2.4. For the practical implementation of algorithms utilizing this approach, multiple options exist. Tousi et al. [27] give an overview and compare the performance of four MARL control strategies suitable for voltage control in power systems. Other methods exist (see [4]), but are out of the scope of this work.

First, a collaborative MDP learner is considered, where the different agents are represented as one single, large learning agent. Each of the individual actions are combined and translated into a single action set. This eliminates the need for communication between the agents, but each of them needs to be able to acquire information on the states, actions and reward of all other agents. For problems with many agents this approach is therefore often infeasible.

A second possibility is the usage of completely independent learning agents (IL). The latter have no information on the actions, states or rewards of the other agents and individually solve their MDP. Each agent thus has its own Q-table (or DQN) which it updates according to a given update-rule. When calculating the Q-values, the state of the environment and reward function can be considered both globally or locally. This allows great computational and memory savings as the costs scale linearly with number of agents. Additionally, the adaptation of a single-agent RL algorithm towards an IL-MARL method is straightforward since the single-agent model can simply be duplicated for each agent. Because of these advantages we opt to work with this approach for the MARL setting presented in chapter 5.

The final two possibilities presented in [27] are coordinated RL and RL with distributed value functions. In the former, an agent coordinates its actions with a number of other agents, but acts independently from the remainder. The global Q-function is decomposed into a linear combination of localized Q-functions. This completely distributed method allows large storage and computational savings. The second method allows cooperation between neighboring, but also non-neighboring agents. Additionally, information about each agents local Q-function can be shared.

2.4 RL for low-voltage grid optimization: a literature review

Vázquez and Nagy give an overview of algorithms and modeling techniques for RL in DR applications in their work [4]. It is clear that the majority of the reviewed papers covering control strategies for distributed generation and storage units have an economical motivation. The most common objective in these papers is the minimization of the energy cost for the consumer [30, 31, 32, 33, 34, 35]. Raju et al. [36] present a cooperative multi-agent scenario with aim of minimizing the cost of power generation for the whole community. Moreover, Sekizaki et al. [37] look into user comfort combined with energy cost. Mwubir et al. [38] propose a strategy to maximize self-consumption of local photovoltaics production in a microgrid. To summarize, previous research typically investigated a consumer viewpoint objective function. Studies on the maximization of efficiency and power quality in low voltage grids, thus from the viewpoint of the DSO, are lacking in the main literature.

Furthermore, action spaces covering battery actions are mostly limited in size. Mwubir et al. [38] propose a method with three possible actions: charging the battery at a power equal to the instantaneous PV generation, discharging the battery, and remaining idle. Sekizaki et al. [37] only uses two battery actions but combine them with another source of flexibility, a water heater, which is regulated through an on/off control mechanism. Li et al. [33] research the case where an action is taken by the consumer: sell or buy electricity from the grid. Most of the papers reviewed in the overview study [4] do not implement large action spaces for battery actions combined with curtailment actions, as is the case in this study.

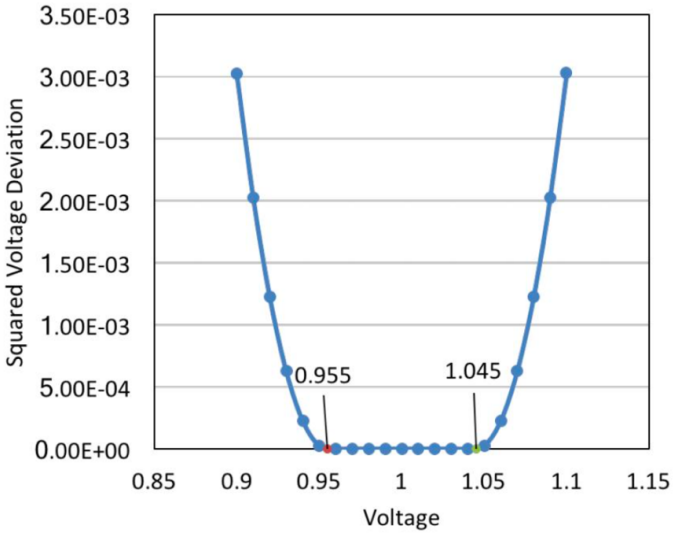

Figure 2.5: As a part of their reward function, Navidi et al. use a quadratic relation to express the voltage quality. [39]

Due to the predominantly economic targets in the papers reviewed in [4], various reward functions are also based on costs. Jiang et al. and Qiu et al. [31, 40] combine a cost component based on the battery losses with a reliability component: the agent is punished if energy from the battery is needed but not available or the SoC is too low. Furthermore, Navidi et al. [39] research a reward function with a cost component and a power quality component to punish the agent when voltages are outside the selected boundaries of 0.995 p.u. and 1.045 p.u., as can be seen in figure 2.5. We used the same idea of a quadratic reward function for violations but with different boundaries to improve the performance.

2.5 Transfer learning

2.5.1 General considerations

Greater part of machine learning algorithms concentrate on isolated tasks with a specific feature space $\mathcal{X}$ and feature probability distribution $P(X)$ (see further) [25]. A feature can be defined as a measurable property or characteristic of the studied phenomenon. In case of different distributions, models need to be redesigned using new training data. This is very expensive for real life applications and sometimes it is even impossible to collect new training data [41]. According to Torrey et al. [25] transfer learning is trying to change this by developing methods to transfer knowledge learned in one or more tasks and use the obtained information to improve learning in related tasks. By using previously collected data or models in this way, new models can be bootstrapped to enhance their initial and asymptotic performance.

2.5.2 A taxonomy of transfer learning

Two concepts form the basis of transfer learning: the domain $\mathcal{D}$ and task $\mathcal{T}$ [41, 26]:

- A domain $\mathcal{D}$ consists of two components: a feature space $\mathcal{X}$ and a marginal probability distribution $P(X)$ with $X = \{x_1, x_2, ..., x_n\} \in \mathcal{X}$. Here, x_i is a particular feature vector corresponding to some input and X a particular learning sample. In a demand side management context, the probability of observing a specific feature vector is quantified by the marginal distribution $P(X)$ and depends for example on the occupant behaviour.
- Given a domain $\mathcal{D} = \{\mathcal{X}, P(X)\}$, a task $\mathcal{T}$ consists of two components: a label space $\mathcal{Y}$ and a conditional probability distribution $P(Y|X)$, which is learned from the training data in the form of pairs $\{x_i, y_i\}$, where $x_i \in X$ and $y_i \in \mathcal{Y}$. The task $\mathcal{T}$ is then given by $\{\mathcal{Y}, P(Y|X)\}$.

Combining these elements, Pan et al. define transfer learning as follows [41]:

> "Given a source domain $\mathcal{D}_\mathcal{S}$ and learning task $\mathcal{T}_\mathcal{S}$, a target domain $\mathcal{D}_\mathcal{T}$ and learning task $\mathcal{T}_\mathcal{T}$, transfer learning aims to help improve the learning of the target predictive function $f_\mathcal{T}(\cdot)$ in $\mathcal{D}_\mathcal{T}$ using the knowledge in $\mathcal{D}_\mathcal{S}$ and $\mathcal{T}_\mathcal{S}$, where $\mathcal{D}_\mathcal{S} \neq \mathcal{D}_\mathcal{T}$, or $\mathcal{T}_\mathcal{S} \neq \mathcal{T}_\mathcal{T}$."

Depending on the relation between source and target domain tasks, transfer learning can be divided into three subsets: inductive, transductive, and unsupervised transfer learning. Table 2.1 summarizes.

Table 2.1: Relationship between traditional machine learning and various transfer learning settings [41].

Learning setting	Source and target domain	Source and target task
Traditional ML	The same	The same
Inductive TL	Different but related/the same	Different but related
Transductive TL	Different but related	The same
Unsupervised TL	Different but related/the same	Different but related

In unsupervised learning no labeled data is available in both source and target domain. This approach is beyond the scope of this thesis. In case of inductive transfer learning the conditional probability distribution varies between the source and target task (i.e. $P(Y_s|X_s) \neq P(Y_t|X_t)$), which implies that transfer occurs between systems with different dynamics. When transfer takes place between identical systems which operate in different regions of the state space (e.g. due to different household behavioural patterns), we speak of transductive transfer learning. In this case, the marginal probability distribution differs between the source and target domain: $P(X_s) \neq P(X_t)$ [26].

Kazmi et al. [26] presents two methods to achieve transfer with neural networks:

- **Feature sharing**: feature sharing involves direct usage of the source training data while learning the target model to improve the learning performance.
- **Parameter sharing**: this method is the form of transfer learning where model parameters, such as the weights of a neural network, are used to initialize the target model. Usually, the first model is trained with a large amount of source data. After initialization the weights are fine-tuned with observed data from the target domain, while using a much smaller learning rate to retain the representations from the first model.

In this work we will study the effect of parameter sharing on the performance of the DQL controller developed in chapter 5. For our specific case, this translates into training the DQL agents in one domain and subsequently transferring their ANNs to the control task at hand by initializing the new agents with these networks. This method, typically used for reinforcement learning, is called the starting-point method [25]. In TL terms, the initial solution of the target task is set based on information from a source task. Compared to the typical randomized initialization in RL algorithms, the target task solution under starting-point transfer begins much closer to a good solution [25].

To assess the used transfer learning methods, three measures can be studied according to Torrey et al. [25]. Firstly, the initial performance after transfer can be compared to the initial performance exhibited by a randomly initialized agent. A second parameter is the learning rate of the agent with transferred knowledge versus the oblivious agent. Finally, the asymptotic performances after reaching convergence can be equated. When the transfer learning process is successful, often all three of these indicators show improvement over the randomly initialized case, as conceptualized in figure 2.6.

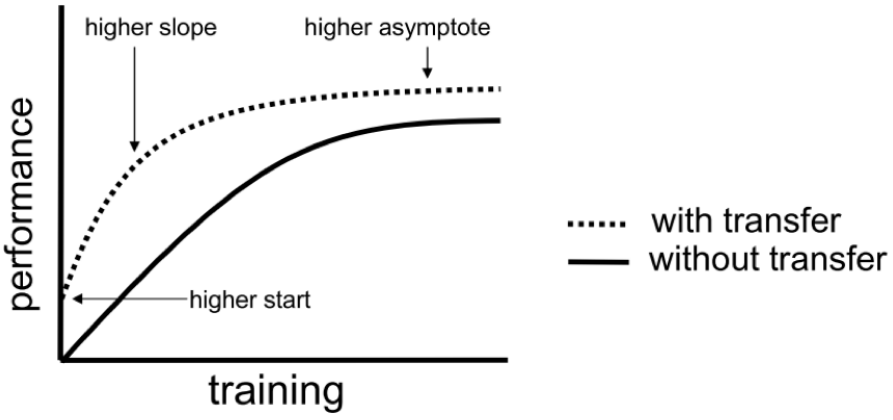

Figure 2.6: Performance improvements through transfer learning [25].

2.6 Conclusion

In this chapter a comprehensive literature study on demand response control strategies with focus on reinforcement learning methods and transfer learning was given. Furthermore, we substantiated the theoretical framework utilized throughout the rest of this work. First, the emerging grid issues due to increasing RES generation and growing electrification of demand were highlighted. It was shown that activating the flexibility of energy sources in a low-voltage system could mitigate these problems through smart control techniques.

One of those techniques, RL, is particularly suitable for this purpose. Some of its key advantages include the ease with which human feedback can be integrated in the algorithms (a crucial aspect for the future growth and scalability of demand response), their self-adaptability, model-free nature, and the capability to train offline on historical data. These characteristics explain the observed increase in popularity of RL based control methods for demand response in energy systems.

Subsequently, the concept of RL was formalized through a mathematical framework based on the concept of a Markov decision process. A specific algorithm, deep Q-learning, for solving such problem was elaborated. The choice for DQL is multifold: the method has shown great recent breakthroughs in various areas of RL, has been extensively studied and is very well documented. Because of these favourable advantages this Q-learning approach is used throughout the remainder of the thesis.

A major disadvantage of RL, however, is the known data-inefficiency linked to the training process of the controller. The majority of machine learning algorithm are designed for solving a specific problem. Transfer learning is used to transfer information from one problem to solve a different, but related problem to accelerate the learning process of algorithms with similar tasks. In addition, the need to collect new training data can be circumvented when this is difficult or impossible to do. In this thesis we will study the effect of parameter sharing on the performance of the designed DQL controller in a different environment by initializing the neural network with the weights of an earlier trained network.

Chapter 3

Data analysis: problem formulation

Extensive integration of RES and increasing electrification of traditionally non-electrified equipment can jeopardize power grid quality. To quantify these effects, data of the REnnovates project combined with a grid topology from the Linear project is analyzed and discussed. First, a brief overview of the data and applied research tools is given. Next, different types of grid violations - overvoltages, undervoltages, and equipment congestion - introduced by solar panels and heat pumps in the distribution network are quantified. To conclude, some interesting dynamics observed when analyzing the specific REnnovates-Linear setup are discussed.

3.1 Data and grid topology

3.1.1 REnnovates data and Linear grid-topology

During the REnnovates project, the electricity consumption of 249 households was monitored on a quarter-hourly basis over the period of one year. The power consumption of the installed heat pumps and other loads was measured separately. Additionally, the PV generation for each household was monitored. The obtained data for each "house ID" thus consists of three components: PV production, electricity consumption from the heat pump, and other loads. A total of 141 of such house IDs were available, of which 82 proved to be suitable for further analysis due to the presence of corrupted measurements or missing data. It was found that this amount of data was sufficient for analyzing trends on low voltage grids (see section 3.1.2).

The actual consumer data originates from the REnnovates project, but no grid topology data was available here. Hence, the grid topologies used are taken from the Linear project [42]. In this thesis we opted for the physical setup where the largest amount of grid problems were observed in practice. This grid includes 2 large feeders with 29 households, each connected by a separate house feeder to the grid.

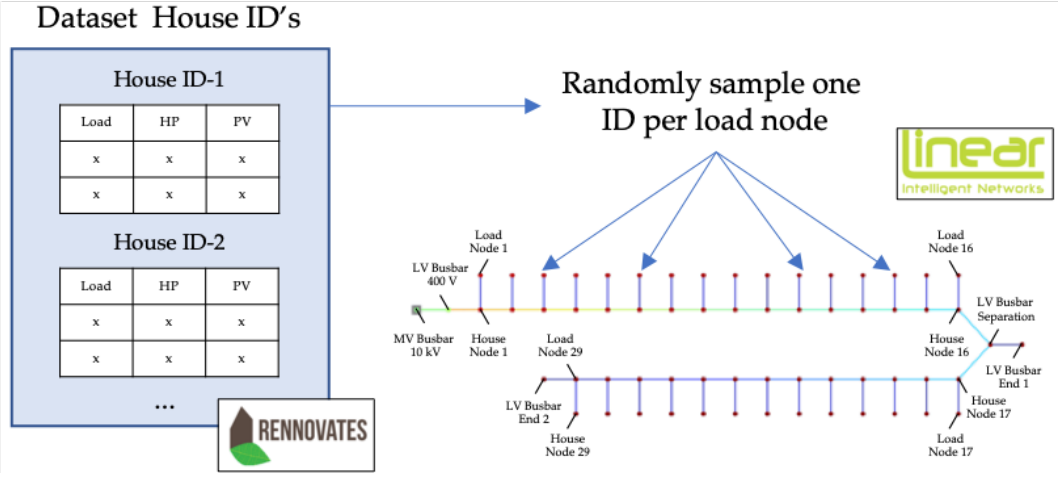

Figure 3.1: Practical coupling of the REnnovates data profiles and Linear grid topology. At the beginning of each simulated year, this update process is repeated to create a randomized simulation environment.

At the beginning of each power flow simulation, 29 house IDs are randomly sampled from the 82 available datasets and allocated to the load and PV nodes representing the different households. Figure 3.1 clarifies while highlighting the REnnovates data and Linear grid topology contributions. In this way, a total of $82!/(82-29)! = 1.11e53$ unique house ID combinations are possible, each of which generating a different aggregate network behaviour. The next section further elaborates on this topic.

3.1.2 House ID analysis

To gain a general idea about the magnitude of the different data components in each house ID, a data-analysis was performed. Table 3.1 on the next page summarizes the results, showing average energy and power consumption or generation for load, PV and heat pump.

An additional parameter to take into consideration is the correlation between the data collected for the different households. That is, if the mutual relation between load consumption, heat pump consumption and PV generation is too large, a random sampling of different IDs (see figure 3.1) would not generate enough variance in the separate simulations for adequately training the RL controller. Figure 3.2 shows that a low correlation is present between the load and heat pump generation of the different data IDs. Logically, PV generation shows a much higher correlation. For the net power balance ($P_{PV} - P_{load} - P_{hp}$) the mean Pearson correlation coefficient is calculated to be 0.608: a moderate linear relationship. The observations presented here in combination with multiple experimental runs, see section 3.3, show that enough variance is present between the different data IDs to generate distinctive network behaviour when randomizing the IDs.

Table 3.1: Analysis of the 82 used REnnovates house IDs. Total energy consumption or generation calculated for a time frame of one year.

Parameter	**Mean ± std**
mean(E_{load})	3506.21 ± 1179.70 kWh
mean(E_{hp})	2451.42 ± 606.42 kWh
mean(E_{PV})	7747.71 ± 906.79 kWh
max(P_{load})	5.12 ± 1.01 kW
max(P_{hp})	2.96 ± 0.15 kW
max(P_{PV})	6.10 ± 0.46 kW

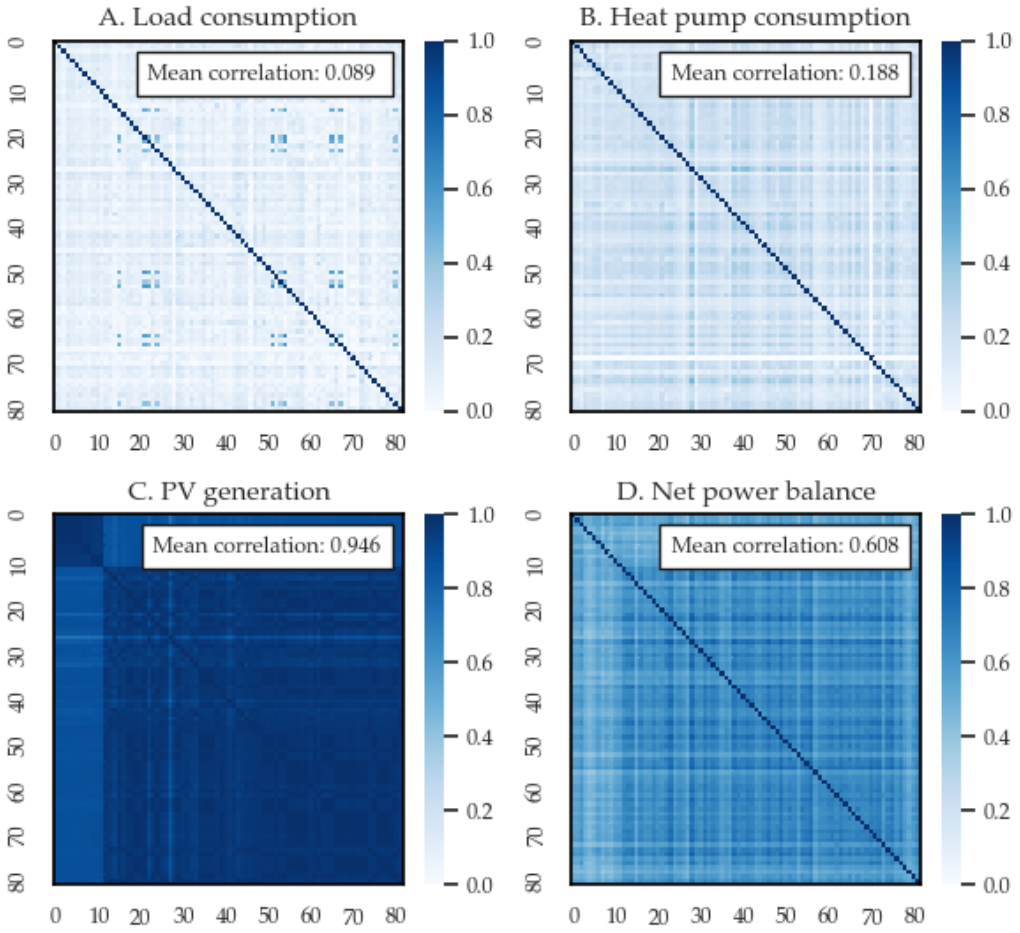

Figure 3.2: Correlation between the different REnnovates data IDs using the Pearson correlation coefficient (1: perfect positive linear correlation, 0: no linear correlation). The numbering on the axes represents an indexing corresponding to each of the 82 unique data IDs.

3.2 Power flow simulations with pandapower

In order to simulate grid behaviour under the control actions taken by the different kinds of controllers, a power flow software to run simulations was needed. For the design of the ANN at the heart of the DQL controller, we opted to use the well documented python package Keras [43] with Google's Tensorflow [44] backend. A logical extension is to work with a power flow solver which too works within the python programming environment. To this end, the low-voltage distribution network was modeled in pandapower [45] - a python package which builds on the data analysis library pandas [46] and the power system analysis toolbox PYPOWER [47] - aimed at automation of analysis and optimization in power systems. The reader is referred to appendix A.1 for the justification of the program selection and a brief overview of the underlying power flow solver.

3.3 Grid violations

To quantify the voltage and network congestion problems at the basis of this thesis, 4 different types of grid violations are studied: overvoltages ($U > 1.1$ pu, $U_{base} = 400$ V), undervoltages ($U < 0.9$ pu), line overloading ($I_{line} > I_{line,max}$), and transformer overloading ($I_{trafo} > I_{trafo,max}$). To obtain statistically significant results, 100 simulations of a year with randomized house ID distributions were executed. Throughout the entire text we will refer to the findings presented here as the "no-controllable resources scenario", which forms the baseline for all considered controllers.

Figure 3.3 shows the results for the total number of violations and violated weeks (i.e. a week where at least one of the aforementioned violations occurs). On average, the simulations contained approximately 340 violated quarters. Translated into violated weeks this amounts to a mean of 17.1 weeks, with the minimum and maximum values ranging between 13 and 19 violated weeks per year, not considering any outliers. An analysis of the most extreme violations observed in the simulations is given in figure 3.4. The following observations can be made:

- It can be seen from boxplot A that on average the yearly maximum voltage per simulation amounts to 1.118 p.u. or approximately 447 V, which exceeds the allowable upper voltage limit by ±7 V. All simulations have a maximum overvoltage deviation above the acceptable operating limit, indicating the presence of strong overvoltage issues in the studied REnnovates-Linear setup.
- Similarly, boxplot B shows the minimum voltage per year observed in each of the 100 simulations. In contrast to the overvoltage issues identified in boxplot A, not a single simulation exhibits an undervoltage deviation above the allowable limit. One of the reasons for this can be found in the low heat pump consumption of the REnnovates households.
- Finally, box plot C and D show the maximum line and transformer loading. It is readily verified that these parameters remain below the rated value in each of the randomized runs.

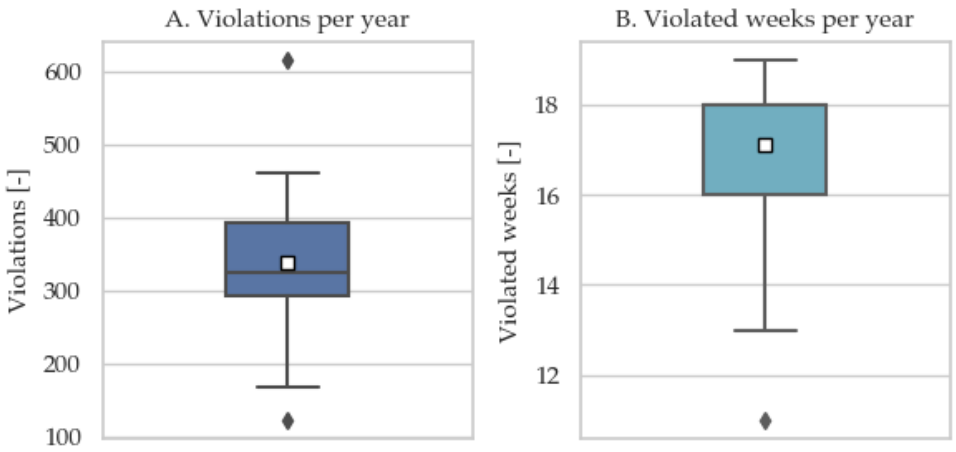

Figure 3.3: Number of yearly violations (A) and violated weeks (B) in the no-controllable resource scenario for 100 randomized house ID distributions.

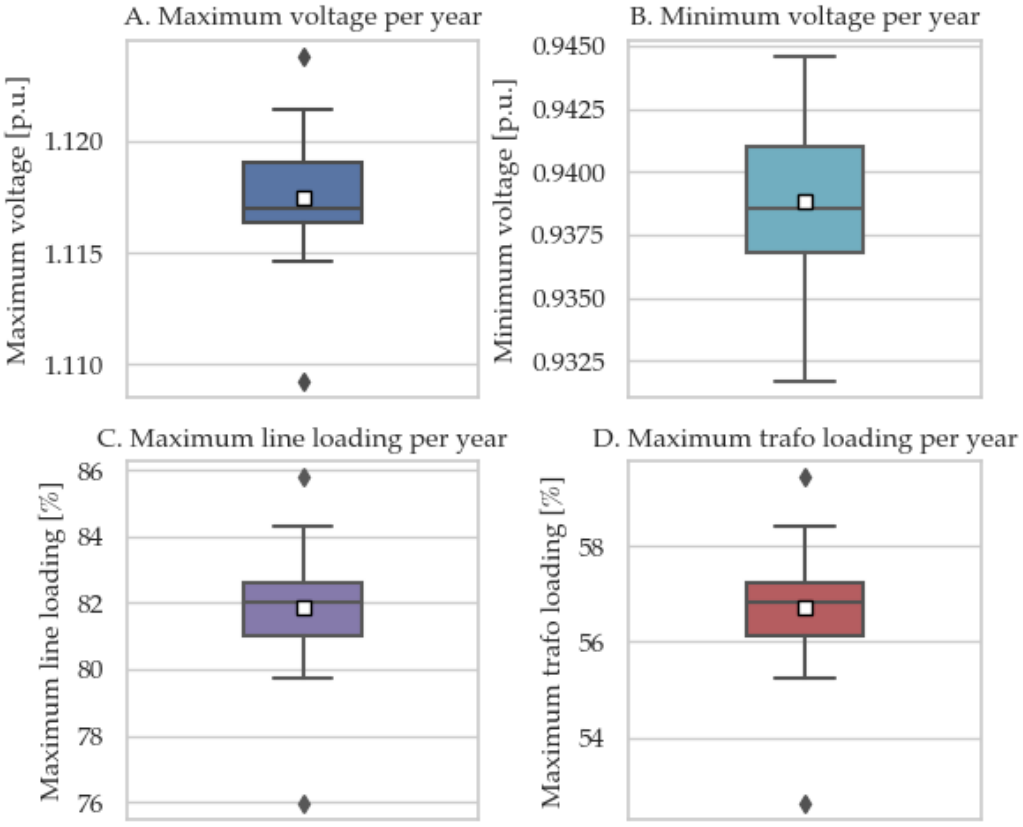

Figure 3.4: Voltage and equipment loading limits in the no-controllable resource scenario for 100 randomized house ID distributions.

From the aforementioned it is clear that the major issues identified in the REnnovates-Linear data-topology combination are overvoltage problems. To clearly visualize the seasonal trend linked to these network violations, a plot of the temporal grid voltage is given in figure 3.5. It is clear - and according to common sense - that overvoltages take place around noon in the summer months at moments of high solar generation.

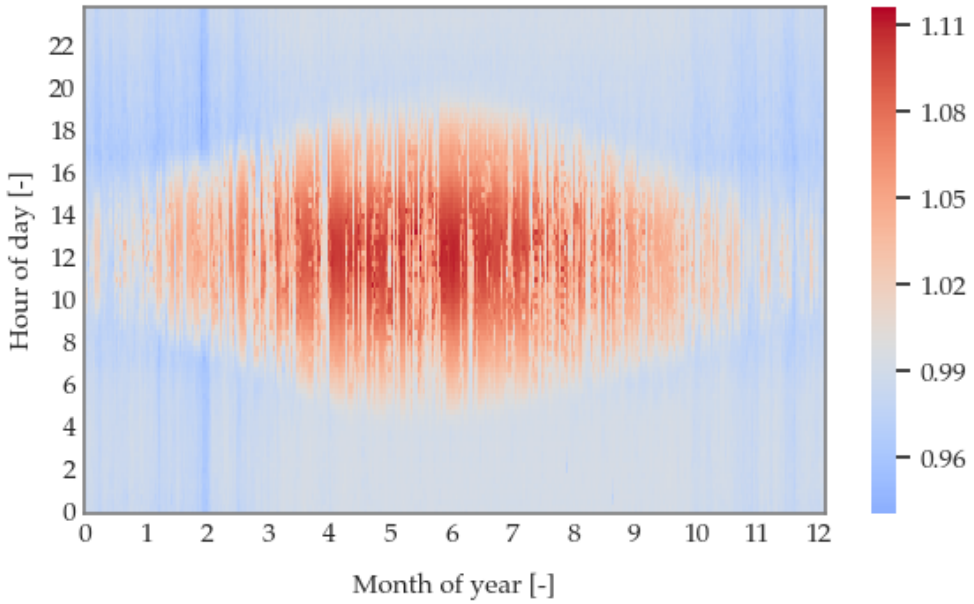

Figure 3.5: Heatmap showing the voltage at the end of the grid feeder - where the voltage variations are the most pronounced - averaged over the 100 no-controllable resources scenario simulations. The overvoltage issues ($U > 1.1$ p.u.) during the summer at noon are readily verified.

The lower limit of the colorbar in figure 3.5 confirms the findings presented on the previous page: no undervoltages can be observed during any of the simulated years. One of the reasons for this can be found in the low heat pump consumption in the REnnovates data IDs. In table 3.1 a mean yearly heat pump consumption of ±2450 kWh was found. As a comparison: the average Dutch household yearly consumes 23260 kWh of natural gas [15]. A heat pump with a COP of 3 could fill in this heating demand with an electricity consumption of 23260/3=±7750 kWh; a factor three more than the heat pump consumption in the data.

The explanation for this is found in the thorough insulation of the REnnovates houses, effectively reducing the primary heating demand which translates to a lower power consumption. Therefore, we emphasize that the used data is not representative for all "zero-on-the-meter houses", since this concept only takes the net energy consumption into account and not the instantaneous power balance. Moderately insulated houses could induce undervoltages as a result of their larger heat pump consumption patterns.

In combination with the observations made in section 3.1.2 about the limited correlation in the data collected for the different households, the analysis performed in this section confirms the presence of sufficient variance between the simulations, as indicated by the statistical spread in the violations in figures 3.3 and 3.4 with randomized data ID assignments.

3.4 Generalizing the problem

The REnnovates-Linear setup discussed in this chapter will form the basis for the analysis made throughout the rest of this work. However, it does not represent the most general problem formulation from point of view of the RL controller: no undervoltages or equipment overloading were observed. To analyse the applicability in this extended scenario, a second, modified grid was created. The topology is similar to the Linear grid (see section 3.1), but the number of houses on the feeder was increased to 40, a smaller transformer was placed and the heat pump consumption was increased. The exact same analysis procedure as for the REnnovates-Linear setup is followed. In the analysis a noticeable amount of undervoltages was observed.

Because of the extensive modifications made to this grid in comparison with the original setup, the real-life applicability of this scenario is reduced - especially from point of view of the projects from which this work originates (aiming for "zero-on-the-meter" houses). When aiming for residential decarbonisation through thorough insulation of houses, significant installation of solar PV, and possibly the addition of energy flexibility through battery energy systems, our analysis has shown that overvoltage issues are the predominant factor putting strain on the DSO's operations. It is for this reason that we do not further consider the results of this additional analysis in this work.

3.5 Conclusion

The data-analysis presented in this chapter exposed the low voltage distribution network issues due to the residential solar panels: a considerable amount of overvoltages in the summer. The need for a control strategy to alleviate these problems is clear. Moreover, undervoltages were not observed in the data analysis due to the thorough insulation of the houses. Therefore, we conclude and stress that enhancing residential housing insulation is a priority and is a valid solution for solving grid issues, in addition to more active control strategies.

Despite the fact that it represents a fictitious scenario - i.e. data from two different projects are combined - the obtained results are of great interest since the matching of these projects could well represent a real-life scenario. Moreover, the highlighted voltage issues were observed in practice and are one of the main motivations underlying this thesis. Finally, it can be noted that many RL researches lack the availability of real-life data. Often, custom datasets with completely fictitious information are employed. Therefore, the usage of real-life data is a key benefit in this work.

Chapter 4

Rule-based controllers: creating a baseline

To create a fair and equitable playground, the RL based controller's performance is benchmarked with three reference control strategies: house level battery control, house level PV curtailment, and district level battery control. In this chapter we first describe the used distribution grid model, including the battery and curtailment model necessary to simulate the impact of the controllers on the grid. Then we describe the functioning of the baseline controllers, argue the choice for battery sizing and placement, and compare the performance to the no-controllable resource scenario.

4.1 Modeling the distribution grid

4.1.1 Network model

We consider the distribution grid described in chapter 3. This element based network is modeled in pandapower as a static, balanced power system. The reader is referred to the pandapower documentation [45] for more information about the underlying numerical solver (based on the Newton-Raphson method) and the available electric components in the pandapower library.

Both the rule-based and DQL controllers have the possibility to control two types of electric components: batteries and PV units (curtailment). A PV unit is modeled as a generator with negative active power following the passive sign convention. The storage units are modeled as either loads or generators depending on the charging or discharging state. Following the same convention, the active power is positive for charging and negative for discharging:

$$\begin{cases} P_{PV} & \leq 0 \\ P_{load} & \geq 0 \\ P_{hp} & \geq 0 \end{cases} \quad \text{and} \quad \begin{cases} P_b < 0 & \text{charging} \\ P_b > 0 & \text{if discharging.} \\ P_b = 0 & \text{idle} \end{cases} \tag{4.1}$$

4.1.2 Battery model

At time of writing it is not yet possible to perform time dependent power flow simulations in pandapower. That is, only instantaneous power balances and resulting network parameters are calculated. As a consequence, the storage unit's state of charge is not updated during any power flow calculation. Therefore, we implemented a simplified battery model as a supplementary component in the simulation code. The charging and discharging of the battery is assumed to be a linear process. Self-discharge or aging symptoms are neglected. The only occurring losses are due to the charging and discharging process in the batteries and converter.

The following constraints are applied to every battery in the network:

$$SoC_{k,min} \leq SoC_k \leq SoC_{k,max} \quad \text{with} \quad SoC_k = \frac{E_{b,k}}{E_{b,k,max}} \tag{4.2}$$

$$P_{b,k,min} \leq P_{b,k} \leq P_{b,k,max} \quad \text{with} \quad P_{b,k,min} = -P_{b,k,max} \tag{4.3}$$

with $E_{b,k}$ the momentary battery energy content, $E_{b,max}$ the maximum battery energy content, P_b the charging/discharging power and $P_{b,k,min}$ and $P_{b,k,max}$ the minimum and maximum power to charge or respectively discharge the k-th battery. Additionally, all batteries placed in the same network are assumed to be identical:

$$SoC_{k,min} = SOC_{min} \tag{4.4}$$

$$E_{b,k,max} = E_{b,max} \tag{4.5}$$

$$P_{b,k,max} = P_{b,max} \tag{4.6}$$

$$P_{b,k,min} = P_{b,min} \tag{4.7}$$

The energy content of each battery at the next quarter hour $E_{b,k}^{q+1}$ is calculated taking into account the current energy content $E_{b,k}^{q}$, the battery charging efficiency $\eta_{b,charge}$, battery discharging efficiency $\eta_{b,discharge}$ and converter efficiency η_c. It is assumed that $\eta_{b,charge} = \eta_{b,discharge} = \eta_{b,c/d}$. Combining these parameters leads to the overall efficiency $\eta_{charge} = \eta_c \cdot \eta_{b,c/d} = \eta_{discharge}$. Additionally, the battery delivers a constant power P_b during each time step $\Delta t = 1/4$ h:

$$E_{b,k}^{q+1} = E_{b,k}^{q} + \Delta E_{b,k} \tag{4.8}$$

$$\begin{cases} \Delta E_{b,k} = P_{b,k} \cdot \Delta t \cdot \eta_{charge} & \text{charging} \\ \Delta E_{b,k} = P_{b,k} \cdot \Delta t \cdot (1/\eta_{discharge}) & \text{if discharging.} \\ \Delta E_{b,k} = 0 & \text{idle} \end{cases} \tag{4.9}$$

Finally, the energy losses $E_{b,loss}$ are computed because of their importance in the reward function for the MDP (see section 5.1.5):

$$\begin{cases} \Delta E_{b,loss,k} = P_{b,k} \cdot \Delta t \cdot (1 - \eta_{charge}) & \text{charging} \\ \Delta E_{b,loss,k} = P_{b,k} \cdot \Delta t \cdot (1 - 1/\eta_{discharge}) & \text{if discharging} \\ \Delta E_{b,loss,k} = 0 & \text{idle} \end{cases} \tag{4.10}$$

4.1.3 Curtailment model

As a result of the simple PV unit modeling described in section 4.1.1 (generators directly connected to the grid), the PV inverter is not included in the grid model and thus assumed to be ideal. The latter is justified because the used PV data is measured as the output power of the inverter. Curtailment through adjustment of the inverter power is implemented by overwriting the original output power $P_{PV,DC,j}$ before curtailment of the j-th pv unit with the output power $P_{PV,AC,j}$ after curtailment:

$$P_{PV,AC,j} = (1 - \beta_j) \cdot P_{PV,DC,j}, \tag{4.11}$$

with $\beta_j \in [0, 1]$ the fraction of the power output of PV inverter j after curtailment as compared to before curtailment ($\beta = 0$: no curtailment, $\beta = 1$: full inverter clipping). Finally, the losses used in the reward function (see section 5.1.5) are calculated as follows:

$$\Delta E_{PV,loss} = \beta_j \cdot P_{PV,DC,j} \cdot \Delta t \tag{4.12}$$

4.2 Design of the rule-based controllers

4.2.1 General considerations

One of the critiques for reinforcement learning based controllers, extending to AI in general, is the complexity of the utilised models. Control engineers should not opt for these kinds of methods simply for the sake of implementing a "popular" control strategy. To create a fair baseline for our RL controller within this spirit, we develop three types of reference controllers: district level battery control, house level battery control, and house level PV curtailment. The naming of each controller refers to the source of flexibility used in their respective control strategies and at what "level" in the grid they operate:

- **District level battery control**: regulates the charge or discharge power of a single, large grid battery connected to the beginning or end of a feeder.
- **House level battery control**: with this decentralized approach each house in the network - 29 in total for the REnnovates-Linear setup - manages its own (smaller) battery operations. PV curtailment is not possible.
- **House level PV curtailment**: combination of a centralised-decentralised control system with the capability to clip the solar PV generation of each of the 29 houses. We do not consider the usage of batteries in this control strategy.

In each of the models considered throughout this work, data communications are possible between three entities: a grid monitoring system (checking the network for violations; most likely operated by the DSO in practice), a centralized controller and one or more decentralized controllers. Figure 4.1 gives a qualitative overview. Which of these entities are effectively used and the type of calculations they perform varies depending on the considered control strategy. This will be further elaborated when discussing the working principle of the individual controllers.

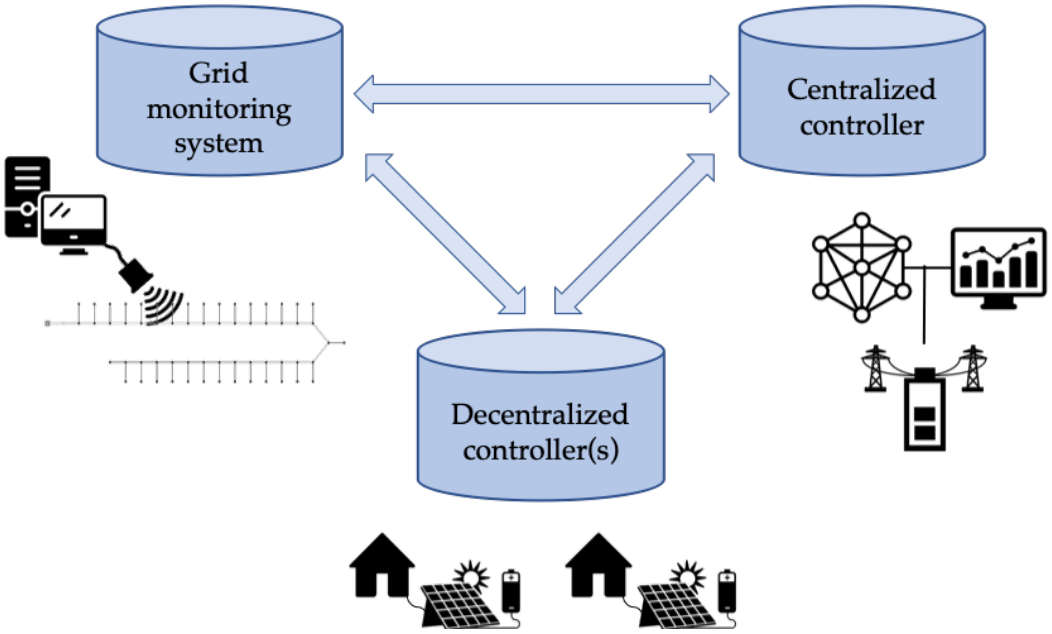

Figure 4.1: General overview of the types of grid entities possibly involved in the different control strategies considered throughout this work.

4.2.2 Operating principle

The target of the rule-based controllers is to avoid voltage and network congestion issues. In this section we further elaborate the algorithms utilised in these reference control strategies. This method later allows benchmarking the RL controller's performance with respect to both the no-controllable resource scenario and the baseline performance presented in this chapter.

District level battery control

In this work we consider both single- and multi-agent control strategies. The former is implemented by placing a battery at the district level on a strategical position in the network. Both the choices for battery sizing and optimal battery placement are discussed briefly in section 4.3. This central storage unit offers flexibility to the network operator in the form of storing or releasing energy to mediate voltage issues and managing grid congestion.

Figure 4.2 gives a schematic overview of the algorithm representing the rule-based battery controller. First, a centralized controller fetches the PV and load forecasts for the upcoming quarter hour. It runs a power flow calculation and checks the network for voltage violations (i.e. when no storage would be used). Next, the aggregate net power balance is determined by summing over all local PV generation (-) and load consumption (+). Based on the combination of expected network violations and this aggregate power balance the controller decides on a control action.

Summarized, the applied rule-set aims at charging the battery when an overvoltage is expected, discharges when an undervoltage is expected, and keeping the state-of-charge (SOC) at 50% in between these scenarios. A backup-controller ensures that the applied battery power is kept within the capabilities of the installed battery ($P_{b,min} \leq P_b \leq P_{b,max}$ and $SOC_{min} \leq SOC \leq SOC_{max}$).

Referring to the principles highlighted in figure 4.1, this district level battery is operated by a centralized controller, which fetches the aggregate PV-load forecast and locally runs a power flow solver on a model of the network. In a more extended version of this controller, the instantaneous power injection and consumption of the connected households could be communicated from the grid monitoring system to the centralized battery controller, allowing for a finer regulation.

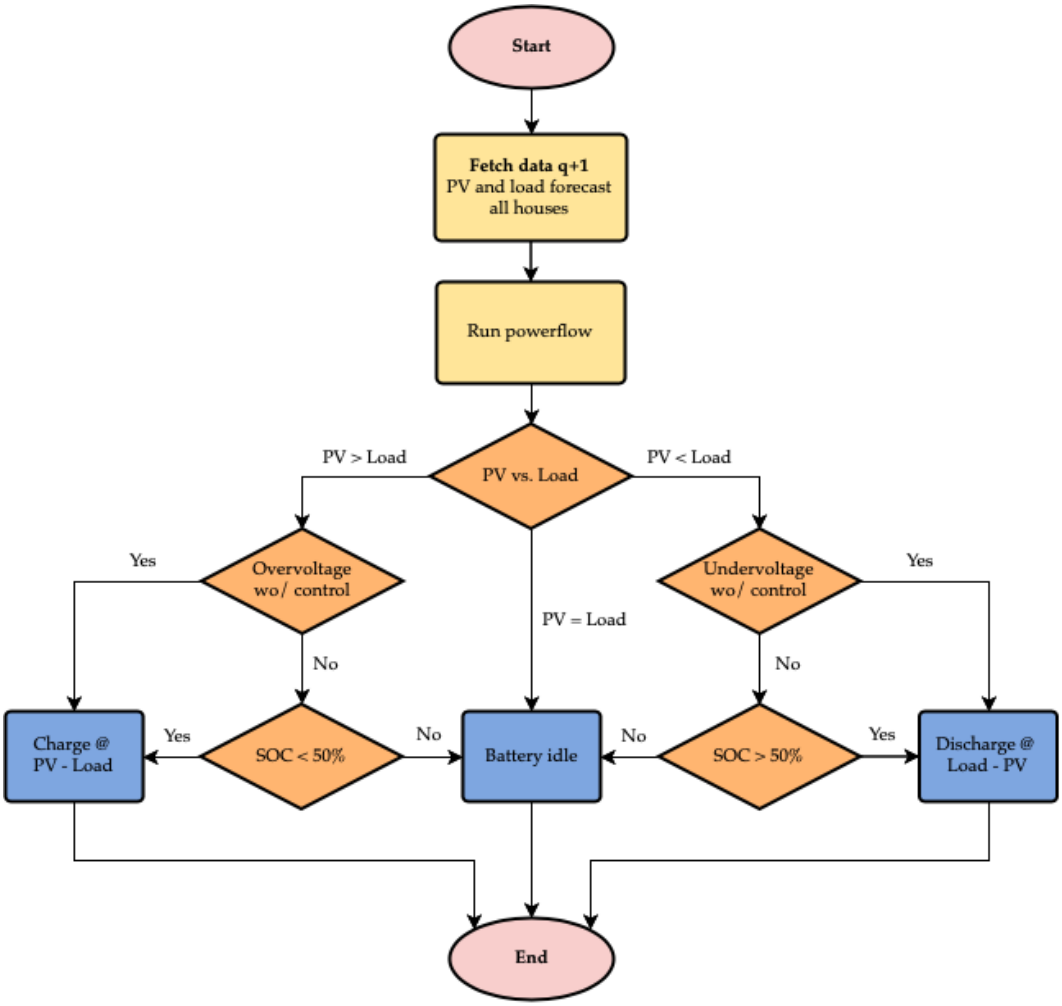

Figure 4.2: Rule-based logic for the baseline battery controller. Battery charges at an overvoltage, discharges at an undervoltage, and tries to keep the SoC at 50% in between.

House level battery control

As explained in chapter 2, sources of flexibility in distribution networks are often present in a decentralized, multi-agent setting. Through DR-agreements this distributed flexibility can be enabled to alleviate grid issues. In this work we assume all households connected to the grid have PV installations, leading to 100% solar proliferation. Additionally, depending on the type of control strategy considered, a designated amount of households is assigned a controllable battery storage unit. Furthermore, it is assumed that the DSO or aggregator (who has a contract with the DSO) has full control over these installations via bilateral DR-agreements.

For the rule based controller, the same principle is applied as for the district level battery (figure 4.2). The main difference is the switch from an aggregate to localized power balance: a central controller still checks the network for violations in the no-controllable resource scenario, but now sends this information to each distributed battery control unit. There, each controller checks the PV-load forecast (decentralized control) for its own household and determines the charging power based on this value: $P_b = (P_{PV} - P_{load} - P_{hp})_{local}$.

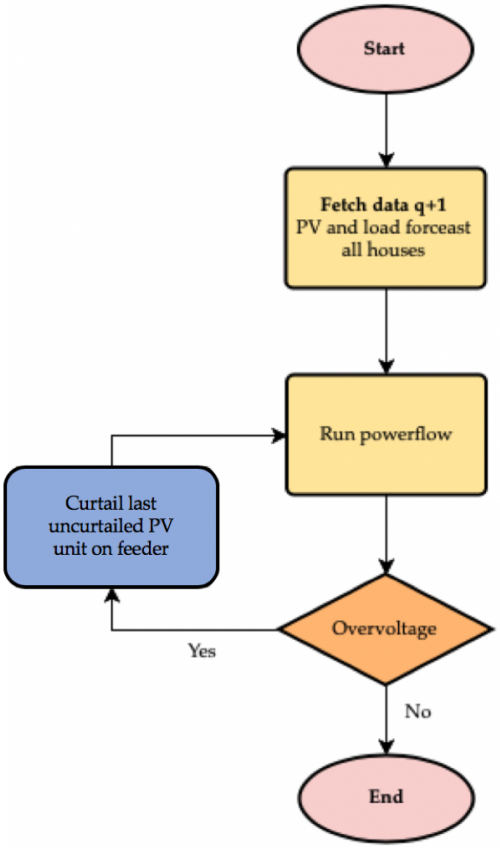

Figure 4.3: Rule-based logic for the baseline PV curtailment controller. The different PV installations are iteratively curtailed starting at the end of the feeder.

House level PV curtailment

Where the house-level battery controller utilizes the flexibility of the local storage installations, another option lies in curtailment of the solar generation. This approach has some obvious downsides: useful energy is lost and only overvoltages can be prevented. It is for this reason that PV curtailment should be kept as a last resort, only for situations in which grid security would be compromised. A similar approach to load-shedding (to resolve undervoltages) is not researched, as this would jeopardize security of supply towards consumers.

The principle of the rule-based PV curtailment controller is illustrated in figure 4.3. A central controller is now needed for both the violation check on the network and supervisory control of the local PV units. This is in contrast to the house level battery control where a power flow calculation is performed based on forecasts and send to the decentralized controllers which actually control the batteries. An iterative approach in the centralized curtailment control on house level is employed: when an overvoltage is predicted, the last uncurtailed solar installation on the feeder (which has the greatest impact on the voltage magnitude, see section 4.3) is shut down. If an overvoltage remains, this method is repeated until all violations are eliminated.

4.3 Battery sizing and placement

An important parameter when considering district or house level battery control is the battery sizing. Two components are considered: maximum battery power $P_{b,max}$ and maximum energy content $E_{b,max}$. First, an analysis was performed based on overvoltages - as these are the predominant issues in the network - to find upper limit values for both parameters:

- **Battery power**: the worst-case scenario is considered to find an upper bound for $P_{b,max}$. To this end, all PV units are set to deliver maximum power (the houses are equipped with a 6 kWp or 8 kWp installation, but are found to never deliver more than 7 kW, see table 3.1). Subsequently, the battery power is iteratively increased and the maximum voltage in the network is observed. Figure 4.4 summarizes the results for both the district and house battery case. It is found that a battery power of ±60 kW on district level, and batteries at house level with a maximum power of ±2.5 kW suffice to avoid overvoltages in this worst-case scenario.
- **Battery capacity**: to place an upper-bound on the battery's rated energy content, we track the maximum number of consecutive quarter hours with an overvoltage in the no-controllable resource scenario. From the data-analysis performed in chapter 3 a value of 16 is found. Assuming the battery has to charge at maximum power (see explanation above) during these quarters to prevent overvoltages, following maximum battery capacities are retrieved:

$$E_{b,max,district} = 16 \text{ q} \cdot 0.25 \text{ h/q} \cdot 60 \text{ kw} = 240 \text{ kWh} \tag{4.13}$$

$$E_{b,max,house} = 16 \text{ q} \cdot 0.25 \text{ h/q} \cdot 2.5 \text{ kw} = 10 \text{ kWh} \tag{4.14}$$

Section 4.4 gives a more in-depth review of the influence of different battery sizes on the controller's performance. For the district level storage unit, an additional variable must be taken into consideration: battery placement. In figure 4.5 the battery power analysis described above is repeated, but with the storage unit placed at different positions in the network. From this analysis it is clear that placement towards the end of the feeder is optimal, which is in correspondence with equation 2.1. That is, energy generated at the end of the feeder sees the highest resistance ($R = \rho l/A$ with l big) and thus causes the largest voltage drop.

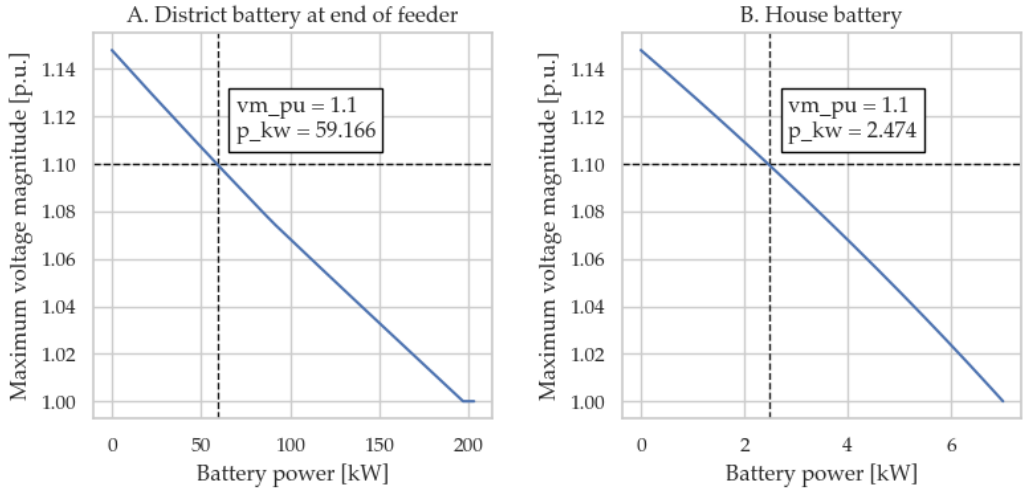

Figure 4.4: Worst-case battery power analysis. The highlighted intersection shows the minimum battery power needed to resolve any overvoltage in the network when all PV units deliver maximum power.

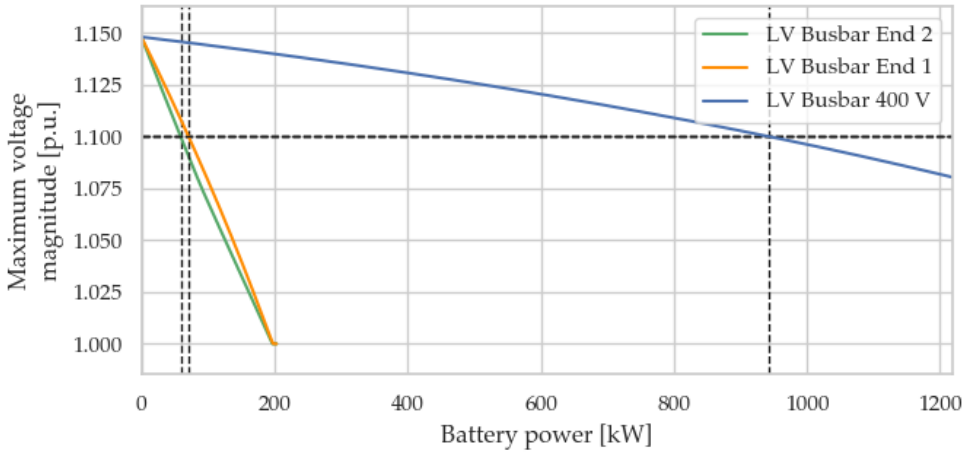

Figure 4.5: Worst case district battery analysis for different positions of the storage unit in the network. From left to right: end, middle, and beginning of the feeder.

4.4 Comparison of the rule-based controllers

4.4.1 Comparing performance

To gain insight in the performance of the different controllers, each of them is tested in 100 randomized house ID simulations. To create a fair comparison, the same random seeds are applied for the different controllers (i.e. they are all tested on the same 100 different combinations of REnnovates data IDs). The results are presented in figure 4.6 and figure 4.7. In the former, the yearly observed violations in each simulation are normalized by the number of violations for that year in the no-controllable resources scenario. The latter shows the incurred losses and controller efficiency, which we defined as the total number of prevented violated quarter hours over the energy losses incurred in doing so. It should be noted that it is not the objective of the controllers to maximize this number, since not taking any action ($E_{loss} = 0$) would lead to an infinite efficiency. It is, however, an interesting parameter to consider.

It is readily verified that PV curtailment outperforms both rule-based battery strategies based on violations prevented. Since in the REnnovates-Linear setup overvoltages are the only issues, curtailing the PV units at moments of high solar generation suffices to resolve all violations. The corresponding lost energy (battery losses vs. curtailed PV energy) is slightly lower for the moderately sized district battery controller. Throughout this work we assume a combined converter-battery charge or discharge efficiency of 90%, combining to a round-trip efficiency of 81%.

An interesting comparison can be made between the house level curtailment and house level battery control strategies by looking at the the worst-agent losses. Figure 4.8 and figure 4.9 indicate that in this context PV curtailment performs the poorest, which is logical since the last house on the feeder is always curtailed first. When looking at this from an individual consumer point of view, such control strategy is difficult to justify, except when the consumer is remunerated for his actions through a DR scheme. When considering an energy-community perspective - where the PV curtailment losses from the worst-off agents could be allocated equally amongst all community participants - this control strategy is well justifiable and from the analysis performed here perhaps the most sensible option (especially since installation of batteries entails additional, substantive fixed costs).

For the battery controllers, the worst-case scenario battery sizes derived in section 4.3 are studied, as well as looking at a more moderate, realistic sizing. It can be seen in figure 4.6 that both the rule-based grid and house battery control strategies are capable of solving approximately 85% of violations as compared to the scenario where no control would be implemented. When taking into consideration battery losses, see figure 4.7, the moderately sized district battery performs slightly better than the PV curtailment controller. This results in an overall similar controller efficiency, but the district battery avoids curtailment of local PV installations. It is up to the DSO to determine which choice they prefer, the possible cost of avoiding a grid violation and ensuring security-of-supply being the key decision factors.

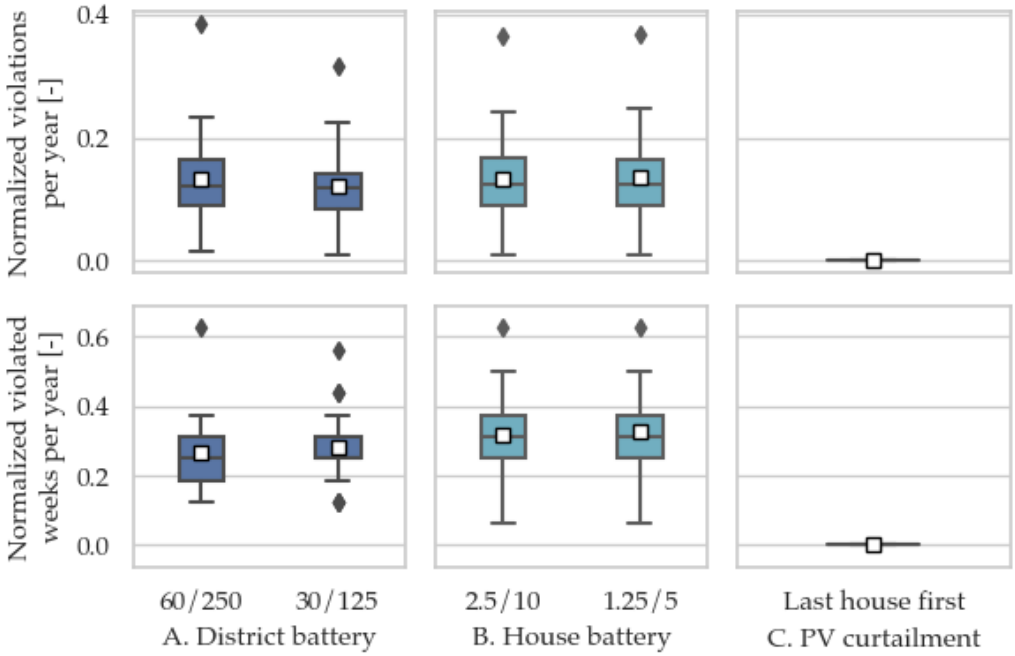

Figure 4.6: Comparison of the baseline controllers violations normalized with the no-controllable resources scenario. Battery sizes $P_{b,max}/E_{b,max}$ (kW/kWh).

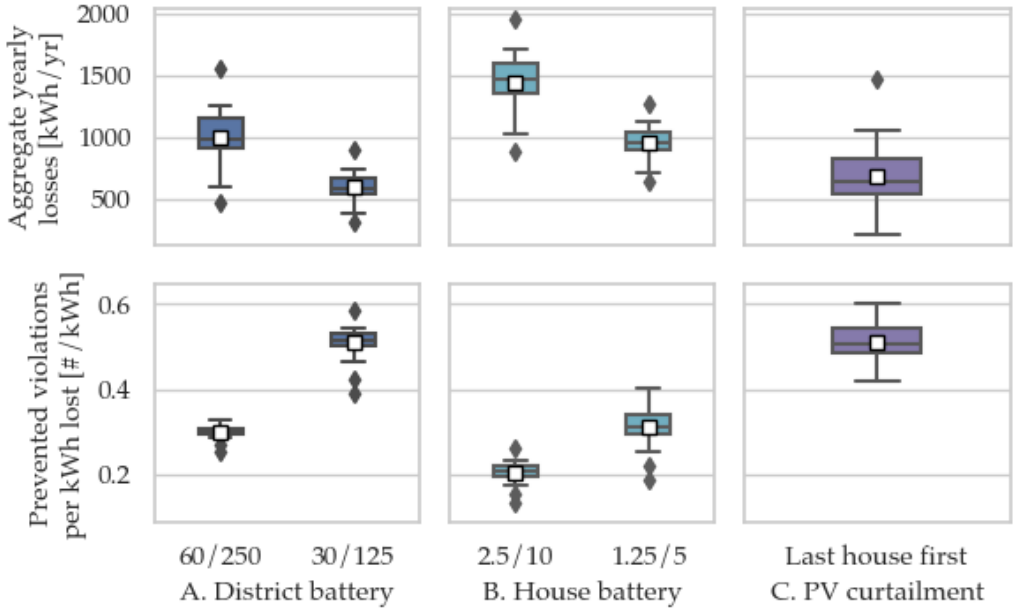

Figure 4.7: Comparison of the baseline controllers losses and efficiency based on 100 randomized simulations. Battery sizes $P_{b,max}/E_{b,max}$ (kW/kWh).

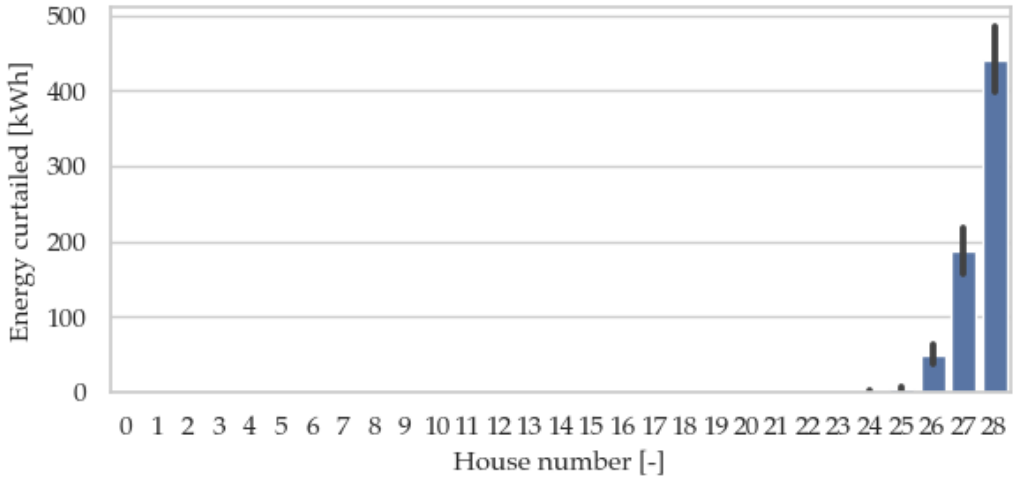

Figure 4.8: Yearly total curtailed energy per house with the rule-based curtailment controller. The error bars indicate the 95% confidence intervals around the estimated mean value over 100 randomized simulations. House-numbering starts with 0 at the beginning of the feeder and increments towards the end of the grid.

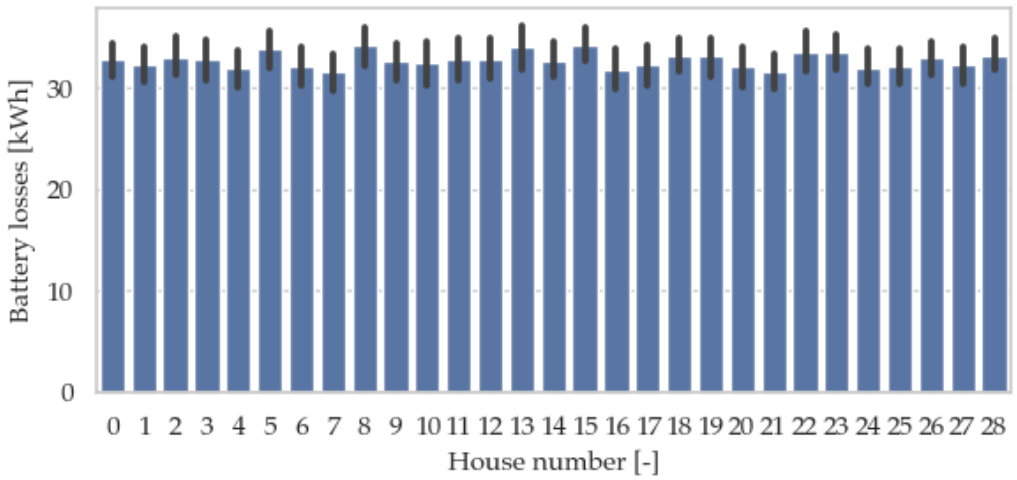

Figure 4.9: Yearly total battery losses per house with the rule-based house battery controller. The error bars indicate the 95% confidence intervals around the estimated mean value over 100 randomized simulations. House-numbering starts with 0 at the beginning of the feeder and increments towards the end of the grid.

4.4.2 Rule-based battery sizing issue

An interesting observation can be made when comparing the worst-case and moderate battery sizing for both the centralized and decentralized battery controllers in figure 4.6 and 4.7: the larger battery performs (slightly) worse than the smaller storage unit - a rather non-intuitive perception. The major issue lies in the battery power selection of the rough control strategy: it always charges or discharges at $P_b = P_{PV} - P_{load} - P_{hp}$. It is clear that the controller discharges/charges his battery more than necessary and therefore not leveraging the battery to its fullest. This phenomenon is more closely examined in appendix A.2.

4.4.3 Limitations of the rule-based controllers

From the aforementioned discussions some clear limitations of the baseline controllers are observed. For the battery controllers, it was found that the limited controllability of P_b leads to sub-optimal performance, especially at larger battery sizing. Additionally, they are not capable of catching seasonal trends, i.e. the batteries aim at keeping their SoC at 50% throughout the whole year. In the summer, reducing the SoC to lower levels could be beneficial to keep more charging capability for upcoming overvoltages; in the winter a higher SoC is similarly interesting to avoid undervoltages. Of course, a good balance is needed in order to avoid unnecessary battery losses.

Another disadvantage of the rule-based battery controllers is their lack of using available forecasts. This results in a completely myopic behaviour, taking away the possibility to intelligently regulate the battery's SoC. Furthermore, the position of the house batteries in the network is not taken into consideration. This is a strong limitation, as it was shown in section 4.3 that agents can most efficiently prevent overvoltages if positioned at the end of the network.

The PV curtailment controller comes out best in the analysis, but has the clear limitation of only being capable of solving overvoltages. Moreover, this control strategy is unattractive from an individual consumer view as stated above.

4.5 Conclusion

In this chapter the foundations for the control strategy aimed at resolving grid issues in the REnnovates-Linear setup have been laid. Both the general network model and rule-based controllers operating within this framework have been described and analysed. The most import limitations - sub-optimal battery power control based and the inability to resolve undervoltages through curtailment - were elaborated. It is clear that a more intelligent approach through implementation of an RL controller deserves further research. To conclude, it should be noted that despite their limitations, the rule-based controllers have purposely not been designed in a too naive manner and their performance can be considered good in comparison to the no-controllable resources scenario.

Chapter 5

Deep reinforcement learning based controller

In the previous chapter it was found that the coarse control strategy utilized by the rule-based controllers leads to sub-optimal performance. It should not be the case that a specific battery design choice leads to poor control performance. Therefore, we present a more intelligent approach through a DQL based controller in this chapter. First, the general setup of the reinforcement learning problem is translated mathematically in the form of a Markov decision process (MDP). At each time step the agent will interact with its environment in order to maximize the (discounted) cumulative reward over time. Next, we elaborate the DQL controller utilized to solve the MDP. An important aspect in this context is the optimal tuning of the model hyperparameters. Finally, the performance of the randomly initialized RL based controller is examined given one year of data for the single-agent and multi-agent scenario.

5.1 Design of the DQL controller

5.1.1 Markov decision process

The RL based controller solves a finite MDP, formulated as (s, a, p, r), where s is the state of the environment, a is the control action taken by the agent based on the given state, r is the reward function, and $p(s', r|s, a)$ is the dynamics function indicating the probability of the environment transitioning to state s' whilst returning a reward r, given action a was taken in the original state s. For a comprehensive overview of the mathematical descriptions underlying these principles the reader is referred to section 2.3.1 of this work. Summarized, an MDP gives a mathematical representation to the RL problem, with the key idea that an agent learns from direct interaction with its environment to achieve a predetermined goal. The latter translates quantitatively to maximizing the discounted cumulative reward (= return) in the long-term. In what follows we discuss and define the key elements making up the MDP to be solved by the DQL agent(s) in the second part of this chapter.

5.1.2 Environment

Following the view of Sutton and Barto presented in [1], the environment can be considered as anything that is outside of an agent's absolute control. That is, any parameter which it cannot change arbitrarily is presumed to be out of the agent-environment boundary. The behavior of this environment is modeled as the low voltage distribution network described in section 4.1. It consists of the physical grid topology taken from the Linear project and 29 households consuming or injecting energy based on the REnnovates data IDs.

In case of the single-agent approach, a district battery is placed at a strategical position in the distribution network. Section 4.3 discussed the optimal battery placement (at the end of the feeder) and argued an upper-boundary for the required battery sizing ($P_{b,max} = 60$ kW and $E_{b,max} = 250$ kWh). However, the additional analysis of the baseline district battery controller showed that a more moderate battery sizing, i.e. 30 kW / 125 kWh, sufficed to resolve grid issues; we will therefore consider this battery sizing throughout the entire SARL section.

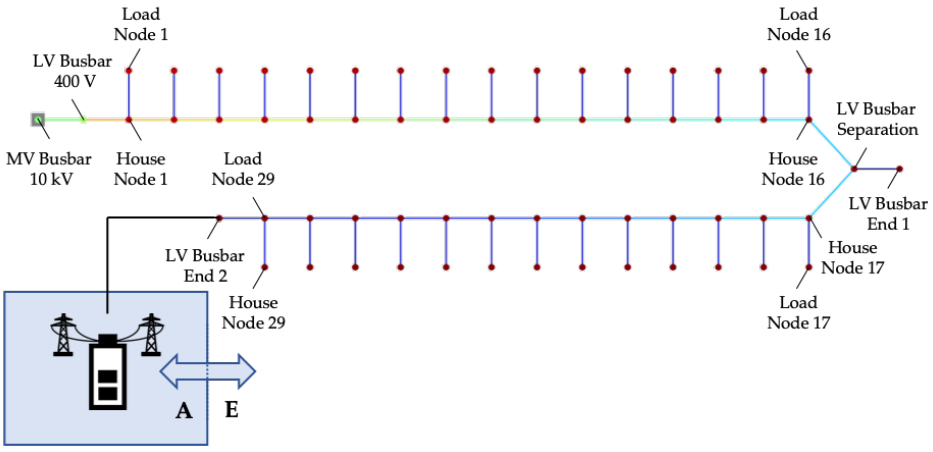

Figure 5.1: In the single-agent scenario, a district battery is connected to the end of the grid feeder. The box and arrows indicate the agent-environment boundary.

In context of the multi-agent approach, multiple smaller residential batteries are placed at two or more households in the network. In the literature, many MARL studies are performed with only two agents [4]. In this work we want to extend this towards working with three agents. For the same reasons the district level battery is placed at the end of the grid feeder, the three houses equipped with batteries are chosen at house nodes 27-29 (see figure 5.2). In addition, they can curtail their own PV output. The number of agents is chosen on basis of the analysis given in figure 4.8, indicating that all overvoltages can be resolved through curtailment of the last three houses. For the battery sizing, we consider the values of $P_{b,max} = 2.5$ kW and $E_{b,max} = 10$ kWh. A detailed explanation of this battery dimension is given in appendix A.4.

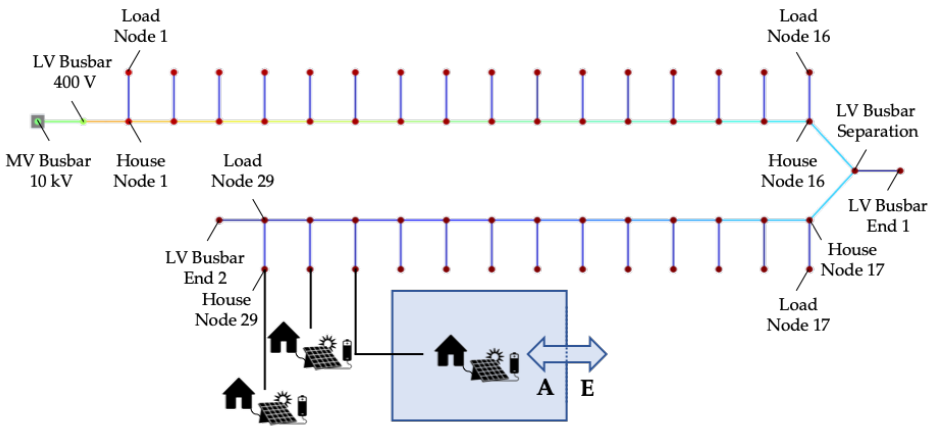

Figure 5.2: Three smaller batteries are connected to the households near the end of the grid feeder in the multi-agent scenario. The box and arrows indicate the agent-environment boundary.

This configuration represents the case where consumers own their own house battery but control actions are taken according to the overall interest of the energy community to maintain a tolerable voltage level. The agents will act as independent learners (IL, see section 2.3.4) and thus have to discover the influence of other agents on the environment. During training, the environment is initialized at the start of a new simulation. As described in chapter 3, new random load profiles are randomly allocated to the different load nodes in the grid.

Following each reset of the environment, the SoC of the batteries is set to 50%. This choice is based upon the reasoning that 50% is not the worst-case scenario in winter or summer and thus the simulations can start in every season of the year. It should be noted that, when not working with excessively large battery capacities, the initial SoC has no noticeable influence on the results since the entire SoC range can be traversed in the time frame of a single day.

5.1.3 States

In an MDP the agent aims at learning an optimal policy $\pi_*(a|s)$ - a stochastic mapping between states and actions - to determine an optimal action given an observed state of the environment. In general, the state can be any form of information that can aid the agent in this decision process. In an idealized setting, all available information could be considered, but due to the curse of dimensionality this is often practically infeasible. In this work, 12 different components are used to represent the relevant environmental aspects. Summarized, the state of the agent is a vector that contains information about the current SoC of the battery, the future energy demand and production, as well as calendar features (time of day and year). Table 5.1 gives an overview. In what follows, each of the components is discussed briefly.

Table 5.1: The state of the environment entails 12 different components, including the SoC of the battery, the forecasted power balance, and specific calendar features.

Index	State
1	Quarter hour of day
2	Day of the week
3	Season of the year
4	SoC of the battery
5-12	Aggregate net power forecast for next 8 quarter hours: $\sum_{agent_i}(P_{PV,i}^{q+j} - P_{load,i}^{q+j} - P_{hp,i}^{q+j}),\ j \in [1, \cdots, 8]$

In order to prevent an impending over- or undervoltage on the grid, the forecasted difference between aggregated PV generation and load consumption for the upcoming 8 quarter hours is considered. An advantage of this approach is the scaling aspect: when going from device to household, to community level the quality of the forecasts normally improves because of cancellation of human stochasticity [48]. This means that in practice only a single aggregate prediction needs to be generated for the entire network, but no predictions are needed at the local level. Throughout this entire work the availability of a perfect forecast is assumed; future research can extend the findings presented here towards a scenario with imperfect prediction capabilities.

In the MARL case, the agents still use this aggregated power balance. A separated $P_{PV} - P_{load} - P_{hp}$ state where the agents only receive information on their own expected energy demand and generation, leads to a decision making process that is not in line with the overall target (and consequently results in a bad performance). That is, when a single agent has a high load or heat pump consumption, but all other agents do not, this scenario will most likely not lead to an undervoltage. Since we are working with ILs, this information cannot be deduced from his own power balance. The reader is referred to appendix A.3.1 for the experimentally found correlation between aggregate power balance and grid violations.

The second used state is the SoC of the battery. In the MARL case, each agent knows only its own SoC. An important part in the learning process lies in finding a policy which keeps the energy content of the battery at optimal levels, anticipating upcoming quarters with grid violations. To this end, the agent needs a notion of time and the ability to look into the future. Three time driven states are used to describe the seasonal and daily trends in the data profiles: quarter hour of the day, day of the week and season of the year (for more detailed information: see A.3.3. The results of the data analysis presented in section 3.1.2 revealed the daily and seasonal tendencies in the voltage profiles. Given only these calendar features, the agent still lacks a more specific indication of the system dynamics. It is for this reason that the aggregate power forecasts for the next 8 quarter hours are given. Justification for this number is given in appendix A.3.2.

5.1.4 Actions

An agent can take different actions based upon the given state of the environment. The single-agent controls the charging and discharging process of the district battery ($P_{b,max} = 30$ kW and $E_{b,max} = 125$ kWh) in steps of 25% of the power limits. Since this agent is not connected to an individual household it has no control over the PV installations. In the MARL scenario, the agents can control their own battery in the same way as in the single-agent case, but since multiple batteries are present a more coarse power refinement can be used. In addition, they can curtail their own PV installation. Combination of PV curtailment and battery control are possible, leading to a MARL action space with size $5 \cdot 3 = 15$.

Table 5.2: The actions the agents can take for a single and multi-agent scenario. Only with multi-agent, PV curtailment is a possibility.

Action type	**Single-agent**	**Multi-agent**
Battery power P_b	$P_{b,max} \cdot 100\%$	
	$P_{b,max} \cdot 75\%$	
	$P_{b,max} \cdot 50\%$	$P_{b,max} \cdot 100\%$
	$P_{b,max} \cdot 25\%$	$P_{b,max} \cdot 50\%$
	0	0
	$P_{b,min} \cdot 25\%$	$P_{b,min} \cdot 50\%$
	$P_{b,min} \cdot 50\%$	$P_{b,min} \cdot 100\%$
	$P_{b,min} \cdot 75\%$	
	$P_{b,min} \cdot 100\%$	
Curtailment fraction β	n/a	$\beta = 1$
		$\beta = 0.5$
		$\beta = 0$

5.1.5 Reward function

The central goal of an RL agent is to maximize the discounted, cumulative reward over time. After taking an action, the environment transitions to a new state and returns a reward towards the agent indicating how favorable this decision was. The reward function thus represents the goal of the MDP as it sets each agent's objectives. Four different components are considered for the low-voltage grid optimization problem studied in this work: voltage violations, line overloading, transformer overloading and energy losses. The main goal of the controller is to eliminate grid violations. In addition, a secondary goal is imposed: minimizing battery and curtailment losses in order to maximize efficiency (which can be defined as the amount of prevented violated quarter hours per kWh lost) and prevent cycling of the battery. The general form of the reward function then becomes:

$$R = R_{voltage} + R_{line} + R_{trafo} + R_{losses}. \tag{5.1}$$

The voltage component $R_{voltage}$ consists of the rewards, or rather penalties, for over- and undervoltages on the grid. Not only the boolean value of having a voltage violation or not, but also the magnitude of the violation is of importance. For this reason, the penalty is given a symmetric quadratic relation, effectively penalizing larger violations more severely (see figure 5.3 A):

$$R_{voltage} = R_{overvoltage} + R_{undervoltage} \tag{5.2}$$

$$\begin{aligned} R_{overvoltage} &= \begin{cases} (-1) \cdot \alpha_1 \cdot (\max(U_i) - 1.09)^2 & \text{if } \max(U_i) \geq 1.9 \\ 0 & \text{otherwise} \end{cases} \\ R_{undervoltage} &= \begin{cases} (-1) \cdot \alpha_2 \cdot (0.91 - \min(U_i))^2 & \text{if } \min(U_i) \leq 1.9 \\ 0 & \text{otherwise,} \end{cases} \end{aligned} \tag{5.3}$$

with U_i the voltage in node i of the grid. The attentive reader may notice that the thresholds from which a voltage is penalized, differ from the violation boundaries of 0.9 p.u. and 1.1 p.u derived from the European standard in section 2.1.1. When training the agent with boundaries of 0.9 p.u. and 1.1 p.u. the voltage penalties are minimized by making the over- or undervoltages very small, but often the violations are not removed completely. For this reason the limits are shifted slightly. In practice, this gives the grid operator an additional operational margin of 4V.

Building upon this reasoning, line and transformer overloading are penalized in a similar fashion: the reward component is a quadratic function and the reward threshold is placed at 95% of the rated loading (see figures 5.3 C and 5.3 D):

$$\begin{aligned} R_{line} &= \begin{cases} (-1) \cdot \alpha_3 \cdot \left(\max\left(\frac{100 \cdot I_{line,j}}{I_{rated,j}} \right) - 95 \right)^2 & \text{if } \max\left(\frac{100 \cdot I_{line,j}}{I_{rated,j}} \right) > 95 \\ 0 & \text{otherwise} \end{cases} \\ R_{trafo} &= \begin{cases} (-1) \cdot \alpha_4 \cdot \left(\max\left(\frac{100 \cdot I_{trafo,k}}{I_{rated,k}} \right) - 95 \right)^2 & \text{if } \max\left(\frac{100 \cdot I_{trafo,k}}{I_{rated,k}} \right) > 95 \\ 0 & \text{otherwise,} \end{cases} \end{aligned} \tag{5.4}$$

where the j-th line and k-th transformer in the network are considered.

Finally, the share of the losses in the reward function consists of two components: battery losses and curtailment losses. The calculation of these losses is formulated in section 4.1. As can be observed from figure 5.3 B, both reward components are a linear function of charging and discharging losses or curtailment losses:

$$R_{losses} = R_{b,loss} + R_{PV,loss} \tag{5.5}$$

$$R_{b,loss} = (-1) \cdot \alpha_5 \cdot \Delta E_{b,loss} \tag{5.6}$$

$$R_{PV,loss} = (-1) \cdot \alpha_6 \cdot \Delta E_{PV,loss}. \tag{5.7}$$

Different scaling factors α_i are used in the reward function to scale the different components relative to each other. These factors need to be optimized in such a way that solving violations is the primary objective and minimization of the losses remains a secondary target. An in-depth description of this design process can be found in appendix A.6.

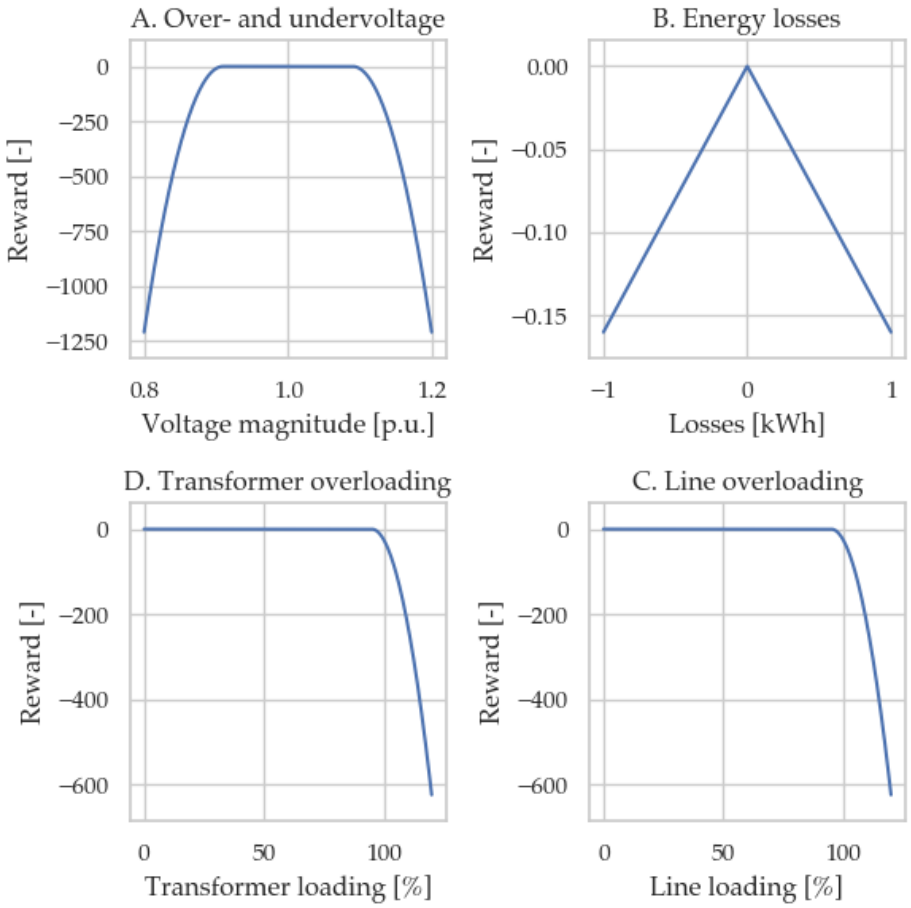

Figure 5.3: The reward function for each agent consists of four different components: voltage violations, line overloading, transformer overloading and energy losses.

5.1.6 Solving the RL problem

To solve the MDP each agent faces the deep Q-learning algorithm with experience replay and target update model described in section 2.3.2. The ANN, which serves as a function approximator for $q(s, a)$, acts as the "brain" of the agent and entails the learned state-action mapping. In the single-agent case, the sole agent is assigned a main and target ANN. In the MARL context, one main and one target network are assigned to each agent, but no information about each others states or actions is shared between the agents. This leads to the IL approach discussed in section 2.3.4. For the design of the ANN, the following considerations are made:

- **Input layer**: the input layer consists of 12 neurons corresponding to the 12 states elaborated in section 5.1.3.
- **Output layer (SARL)**: the ANN has 9 output neurons corresponding to the predicted Q-values of a specific battery action (see table 5.2).

- **Output layer (MARL)**: in this case, the output layer is larger and contains 15 neurons: one for each combination of battery and curtailment actions. Some DQL researches have worked with an architecture where one ANN is used per desired output action, but since this approach requires a separate forward pass for each action the cost scales linearly with the action space size, increasing the computational burden [29].
- **Hidden layers**: layers between the in- and output layer of the ANN are the so-called hidden layers. The number of hidden layers and neurons are hyperparameters which require fine-tuning for optimal performance. The next section further elaborates this.

Some extra remarks can be made about the used activation functions in the ANN. According to Ramachandran et al. [49], the rectified linear function $f(x) = \max(0, x)$ is the most successful and widely-used activation function for deep learning applications. In the study of Henderson et al. [50], neurons with this activation function (consequently called rectified linear unit, or ReLU) also exhibited the best performance. It is for this reason that all-but the output neurons are given a rectified linear activation. Since the output of our ANN is a numerical, continuous value (i.e. it represents the Q-value associated with each agent's actions) it serves as a regressor. In this case, the usage of the linear activation function $f(x) = x$ for the output neurons is common practice [29, 51, 52].

Figure 5.4 indicates the position of these elements in the general structure of the ANN. Additionally, all the components in the network's input (i.e. the states) are normalized with the observed limit values (following from the data analysis or nature of the state) before being passed to the network. By doing so, none of the components outweighs the others which would be the case if the values do not have the same order of magnitude, speeding up the convergence rate [53].

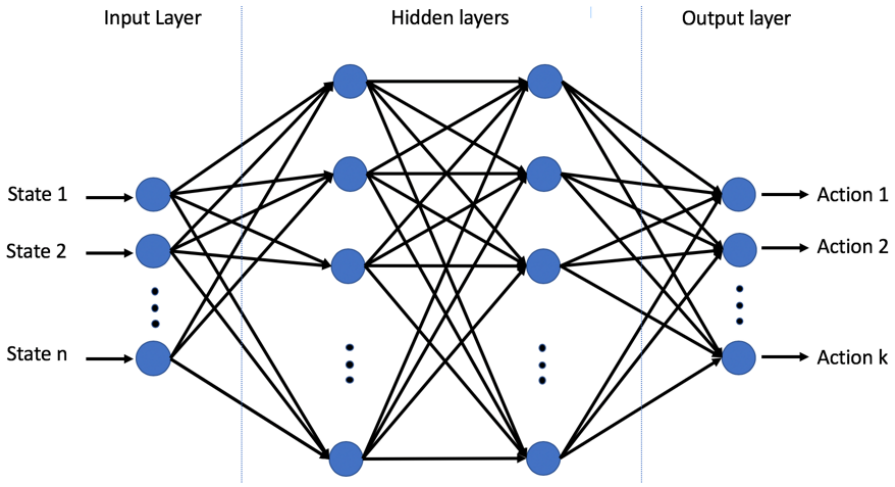

Figure 5.4: The general structure of a feed-forward ANN with the inputs (states), outputs (actions) and intermediate hidden layers.

5.2 Optimization of hyperparameters

5.2.1 General approach

Optimization of hyperparameters plays an important role in reinforcement learning algorithms to obtain optimal performance and speed up convergence (or avoid divergence altogether) [50, 54]. However, Henderson et al. [50] indicate that the value range of used hyperparameters is often not reported in published work. In this section, a description of the optimization process is given and substantiated.

Liessner et al. [54] present multiple strategies for the parameter selection, including grid and random search, Bayesian optimization and random forests. Both the Bayesian optimization and random forest methods require specialized coding approaches, which are beyond the scope of this thesis. Thus, we opted for a one-dimensional grid search primarily based on the unfeasibly high computational cost of a full systematic hyperparameter grid search. First, an informal search was performed on the most relevant hyperparameters (see the left-side column of table 5.3). This method is similar to the ones reported by Mnih et al. in [29].

After reaching the point of convergence, the one-dimensional grid search was started with parameter values symmetrically distributed around the values of the informal search. Classical AI heuristics, such as varying the number of neurons in each layer with steps of 2^k, were applied. The different parameters were varied one at a time, whilst keeping the others constant. This method is identical to the approach described in [50]. Based on the performance, indicated by the total cumulative reward received by the agents over time, the optimal values were selected. Table 5.3 summarizes the findings. For a more in-depth review on the utilized procedures and meaning of the different parameters, the reader is referred to appendix A.5 and the optimization example on the next page.

Table 5.3: Results of the hyperparameter optimization.

Hyperparameters	Optimal value
ANN structure (neurons per hidden layer)	(64,64)
Adam optimizer learning rate	0.001
Discount factor	0.99
Replay memory size	134400 quarters (200 weeks)
Minimum replay memory start size	672 quarters (1 week)
Minibatch size	64 experience samples
Target network update frequency	2688 quarters (1 month)

The values presented here are simulated in the single-agent setting. Again, this decision was based on grounds of higher computational cost and limited resources. The parameters obtained from the SARL scenario are directly used in case of the multi-agent setting, so no separate hyperparameter optimization was performed.

The parameters described above are extrinsic factors, nevertheless Henderson et al. describe the importance of the influence of intrinsic factors (e.g. effect of random seeds) on performance [50]. They demonstrated that results could differ drastically just from varying the randomization process. Therefore, the optimization of parameter values is always executed with 5 different random seeds and statistically analyzed afterwards which is similar to the method in [50].

5.2.2 Optimization example: exploration vs. exploitation

An important dilemma faced by all learning control methods is the trade-off between exploiting the current knowledge to perform optimally with respect to the learned policy, and non-optimal exploratory behaviour which might unearth a more favorable state-action mapping. This trade-off is known as the exploration-exploitation dilemma, and remains an open problem until this day [1]. To highlight the principles of the hyperparameter optimization process, a closer examination towards this issue in context of our grid control problem is given.

Through interaction with the environment over the course of 1 year - the time frame for which the REnnovates data is available - the agent will train and update its deep Q-network and strive to learn an optimal policy which maximizes the reward in the long-term. In order to ensure adequate discovery of optimal control actions, an ϵ-greedy policy is used, with ϵ the probability of taking a random action in a given time step. If epsilon is 1, the agent takes all of his actions at random, leading to maximal exploration but poor performance. A value of 0 implies that the agent will "exploit" and take actions that are optimal with respect to the current policy, but therefore miss the opportunity to explore new, possibly more valuable actions.

For the SARL case, the left-hand side of figure 5.5 shows the results of a grid-search over the indicated range of ϵ-values. It is readily verified that a higher exploration rate is not beneficial. The reason for this is twofold. When working with a single grid battery at the end of the feeder, a random battery action has a high chance of leading to a malign situation (e.g. discharging at full power at moderate $P_{aggregate}$ will likely lead to an overvoltage). In other words, the sensitivity of the grid voltage with respect to random SARL battery actions is large. Secondly, battery losses increase near linearly with ϵ (subfigure C2).

For the MARL case, similar dynamics can be observed. However, note the much lower sensitivity of the grid violations with respect to random MARL agent actions (compare the scales of figure 5.5 B2 SARL vs. MARL). This is due to the inclusion of PV curtailment actions (which can only lead to a reduction of overvoltages) and the distributed approach (a single faulty random action has lesser effect on the network). Nonetheless, battery losses again increase strongly with ϵ. For this reason it is chosen to work with $\epsilon = 0$ when checking the performance of the controllers, only allowing exploration through the randomization of the initial ANN.[1]

[1] This procedure differs from the used method in chapter 6, where a controller is first trained offline using an epsilon-decay scheme and transferring the knowledge afterwards to an online controller.

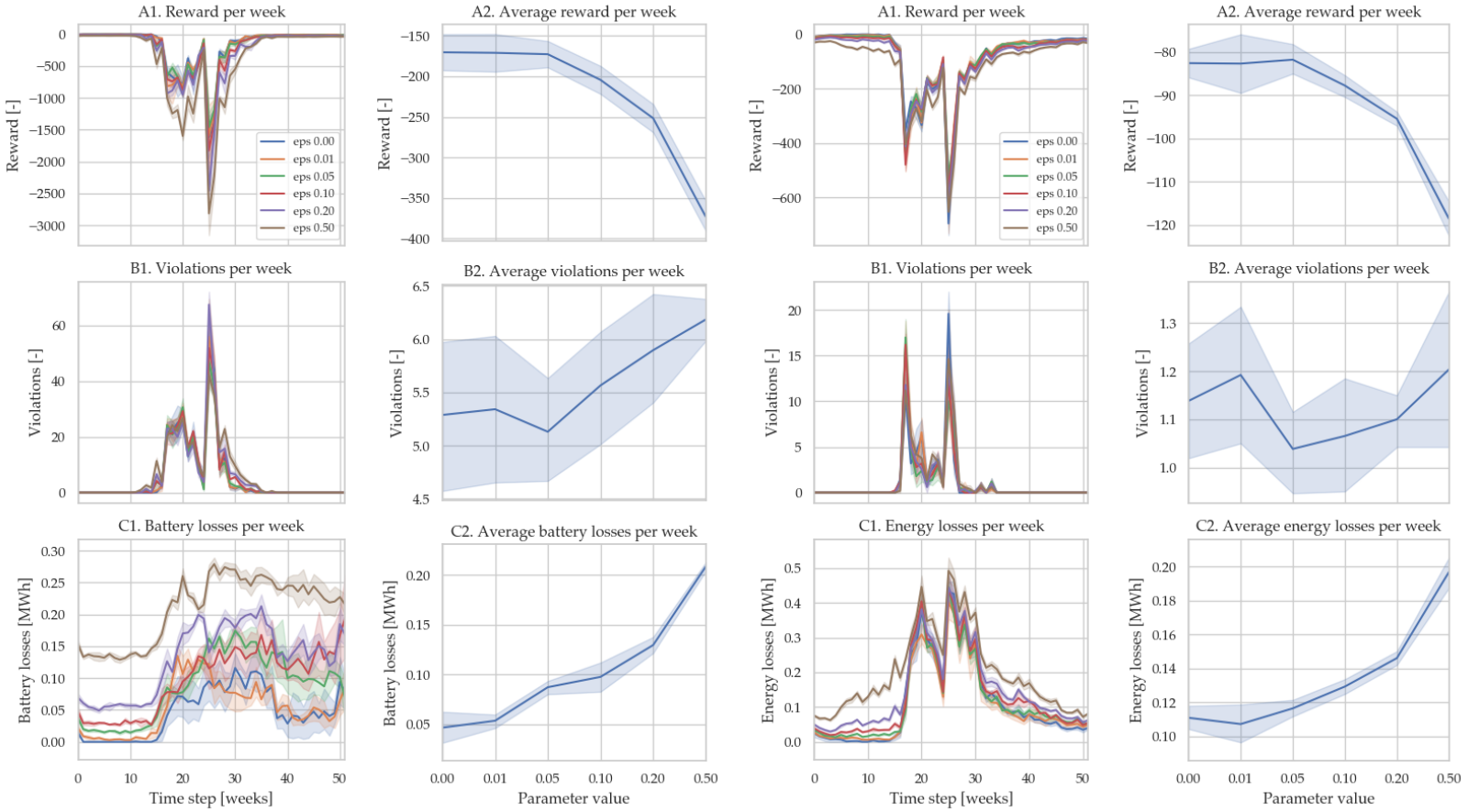

Figure 5.5: Experimental analysis on the influence of different ϵ-values for the SARL (left) and MARL (right) scenarios.

5.3 Performance of the controller

5.3.1 General procedure

In this section the performance of the RL based controller in the REnnovates-Linear setup is studied. We consider the MDP formulated in section 5.1. At the beginning of each simulation the ANN for each agent is initialized with random weights and biases. Additionally, the distribution process of the data IDs to the 29 available house nodes is performed (see section 3.1), resulting in a unique environment. Setting the seed at the start of each simulation ensures that all random processes (ANN initialization, house ID distribution, and random action selection through the ϵ-greedy policy) are reproducible. The performance is analyzed by comparing the number of violated quarter hours and incurred energy losses of the RL controller to the no-controllable resources scenario. A detailed comparison with the rule-based controllers follows in chapter 6, where the concept of transfer learning will be used to greatly enhance sample efficiency. Since quarter hourly data is available for one year, this is the considered time frame for both the SARL and MARL scenarios.

5.3.2 Single-agent: district battery

In the single-agent scenario, the agent controls the district battery placed at the end of the feeder in the grid. To get statistically significant results a set of 100 simulations is performed with the same randomized house ID distributions as utilized in chapter 3 and chapter 4 for the no-controllable resources scenario and rule-based controller performance analysis respectively.

An overview of the findings is given in figure 5.6 for both the SARL and MARL cases. Focusing on the former, it can be seen that given only one year of data the DQL controller performs poorly in comparison with the no-controllable resources scenario. On average, the performance is 10% worse compared to the no controllable resource scenario. Similarly, the total amount of violated weeks is increased. Figure 5.7 adds to the equation by indicating a negative amount of prevented violated quarter hours per kWh lost for almost 50% of the considered simulations. However, these results should come as no surprise given the known data-inefficiency of RL based controllers; giving solely a year of training data is simply insufficient for adequately training the agent.

In addition, the trends in the data are not very homogeneous, changing from season to season. The agent has to learn an optimal policy in winter that is very different from the one in summer, therefore making the learning process of the overall policy more complex. Learning seasonal trends when observing only one year of data is hardly possible. We could expect better results if the differences between the seasons were smaller, because in this case the agent could train on one particular policy a whole year long. As a result of the fluctuating climate in the Netherlands (REnnovates data), the agent has to perform a more complicated task.

As stated at the beginning of this chapter, the agent has to take actions in order to learn the optimal policy. At first, the random initialized agent charges and discharges his battery whether these actions are beneficial to prevent overvoltages or not. This random action taking process leads to a high amount of energy losses, without resolving almost no violations. Consequently, the controller efficiency is low (and sometimes negative), indicating that the RL controller performs badly. In chapter 6 it will be seen that the performance can be greatly enhanced by utilizing transfer learning.

To illustrate the performance of the controller, figure 5.8 highlights the evolution per week of the reward, the violations and the battery losses for a representative simulation. It is clear that only a small fraction of the problems on the grid is solved. In the summer, the controller has learned to eliminate some of the overvoltages, but no long-term policy has been adopted to regulate the SoC towards more favorable values to proactively counter impending quarter hours of high solar generation. This is indicated by the "overvoltages at full battery" part of stack plot B. The agent needs more data to realise these types of policies.

Furthermore, plots of the evolution of the learned Q-values over time can give more insight in the temporal policy. The Q-values represent the expected return from a given state for each action the agent can take. Nine battery actions, discussed in section 5.1.4, are possible in the SARL case, as a result 9 Q-values (outputs of neural network) are studied. Figure 5.9 looks at the interesting case where the agent is given a state in summer at noon when there is a lot of PV generation and thus the possibility for overvoltages on the grid.

The evolution of the graphs only changes after 18 weeks of training. This result is expected since there are no overvoltages in the first months of the year. The experiences gathered by the agent in winter simply do not describe cases with a lot of PV generation and thus the agent does not learn a policy for these kinds of states. After encountering overvoltages for the first time in the beginning of the summer, the agent clearly learns that charging the battery is the best option. After 30 weeks of learning the agent considers charging at 50% as the optimal action for the given state. As we will see in chapter 6, the actual optimal action is 75% or 100% in an (approximate) optimal policy scenario.

It can be seen that the Q-values of the studied state keep evolving over time, not having reached a state of convergence yet. This indicates the data-inefficiency of the RL controller: despite interacting with the environment for over a year, no clear policy has been established. Furthermore, the comparison of these results with figure 5.8 reveals that the agent has (partially) forgotten its pre-summer policy (keep battery losses low) in the post-summer period. That is, after receiving large penalties for overvoltages in the summer the agent mistakenly interprets post-summer states as potentially leading to voltage violations, leading to charging of the battery when not needed.

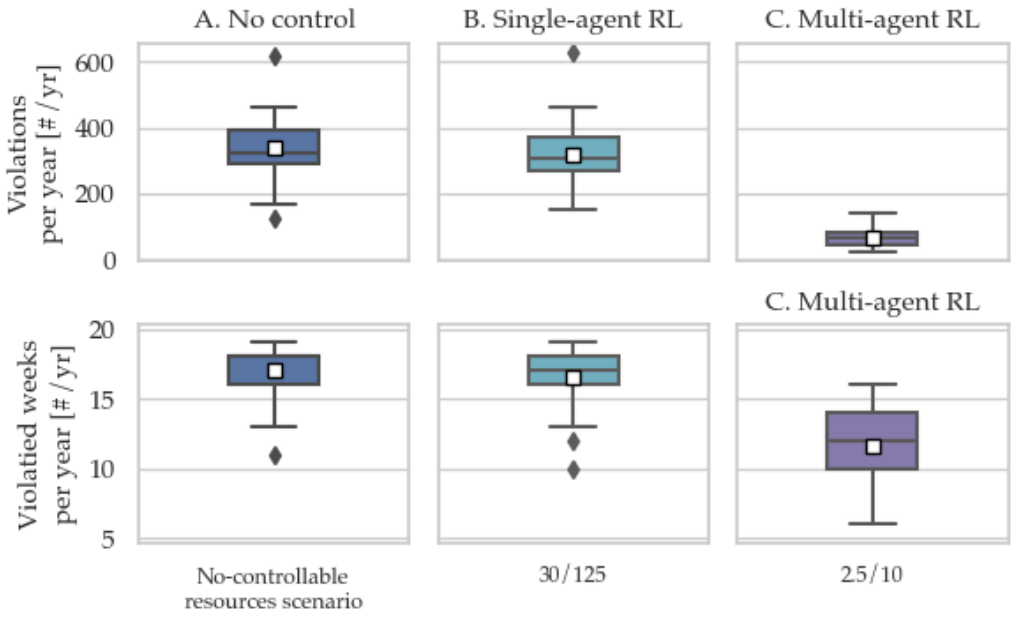

Figure 5.6: Comparison of the randomly initialized SARL and MARL controllers violations with the no-controllable resources scenario. Battery sizes indicated as $P_{b,max}/E_{b,max}$ (kW/kWh).

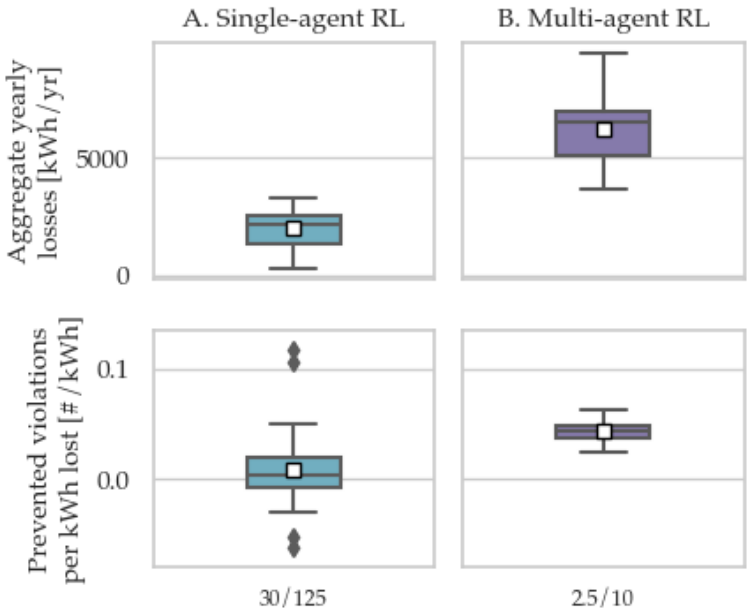

Figure 5.7: Comparison of the randomly initialized SARL and MARL controllers losses and controller efficiencies. Battery sizes indicated as $P_{b,max}/E_{b,max}$ (kW/kWh).

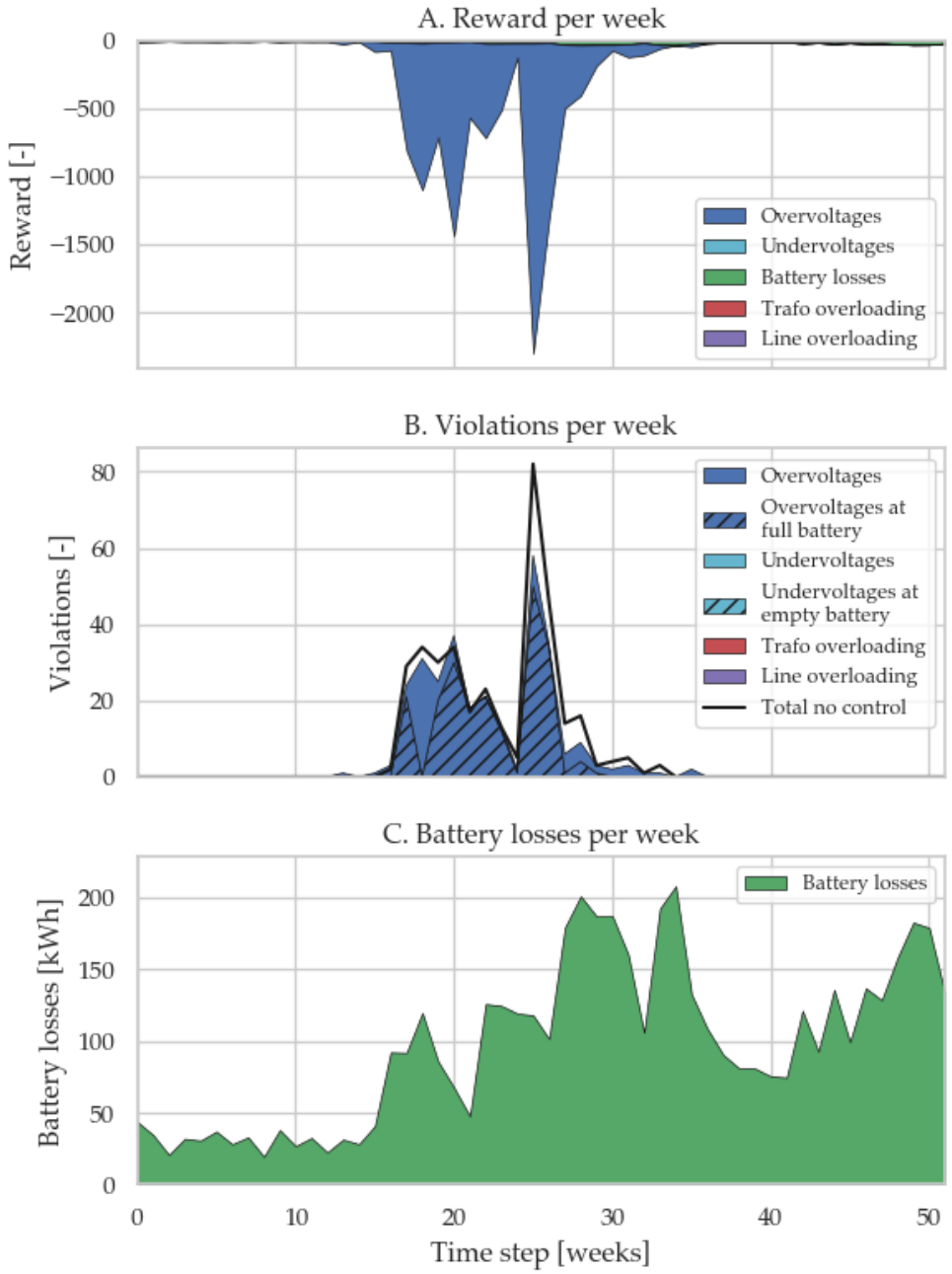

Figure 5.8: The randomly initialized agent is trained for one year long and performs poorly: almost no voltage violations are solved and the losses at the end of the year are high despite the absence of overvoltages. The analysis here is performed for one of the 100 house ID distributions which showed an around average number of violations in the no-controllable resources scenario.

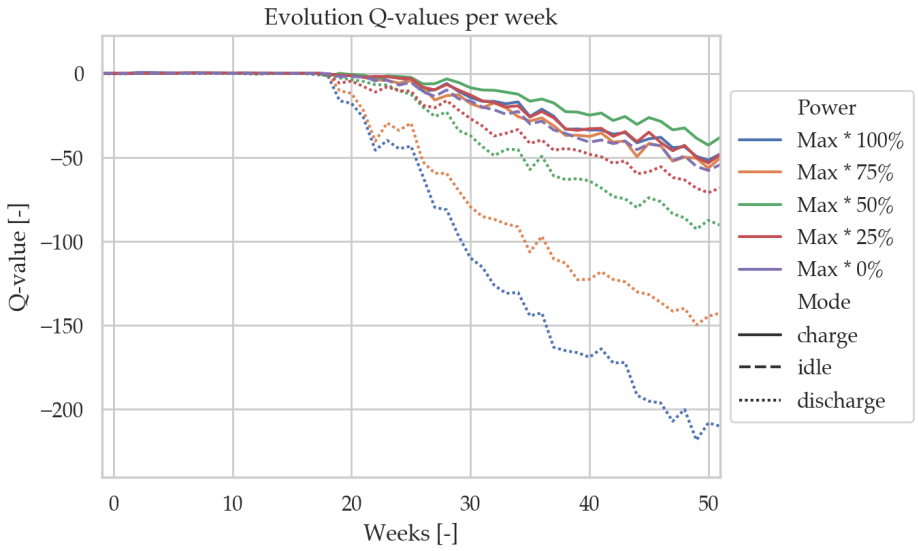

Figure 5.9: The Q-values at noon in summer after a year of training suggests that the agent has learned to charge its battery at large aggregated $P_{PV} - P_{load} - P_{hp}$ ($SoC = 60\%$ presumed). The difference between the charging stages is not very clear, showing a small preference for 50% charging.

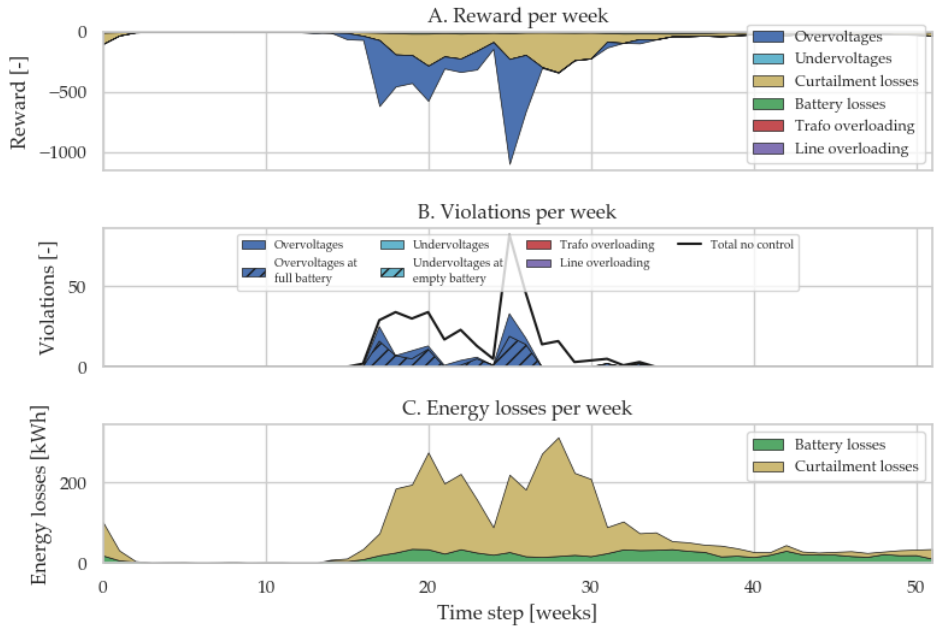

Figure 5.10: The randomly initialized agents are trained for one year long and perform much better than the SARL: a great part of the violations is solved but the losses are still abundant. The analysis here is performed for the same house ID distribution as in the SARL scenario.

5.3.3 Multi-agent: house batteries and curtailment

To analyze the multi-agent setting the same procedure is followed. Again, the performance of the controllers is observed over a time frame of one year for 100 randomized data ID distributions. The major differences with the SARL case lie in the distributed approach: now there are 3 independently acting agents at the end of the grid feeder, which in addition have control over their PV installations.

The results are presented in figure 5.6. It is clear that the amount of violations per year is much lower in case of MARL than the no control and SARL case. On average, a reduction of 80% in violated quarter hours is realized. One would think that therefore the MARL controller performance is superior to that of the SARL controller but when looking to figure 5.7 it is observed that the incurred losses are significantly larger, which causes the MARL controller's efficiency (prevented violations per kWh of losses incurred) to be almost identical to that of the SARL controller. Similar to the SARL scenario the learned policy is far from optimal. As will be seen in chapter 6 the performance can be greatly enhanced through the concept of transfer learning.

Figure 5.10 shows the evolution of the reward, the violations and energy losses per week for a simulation of one year. As opposed to the SARL agent, a large part of the grid violations is effectively resolved. The graph containing the energy losses clearly indicates that the agents have learned to mitigate these violations through curtailment of their respective PV installations. However, these curtailment actions do not resolve all violations and are used excessively leading to abundant losses.

Finally, the uniform distribution of the battery losses indicates that battery actions do not play an important role in the control process. The agents quickly learn the - for us - obvious relation between their curtailment actions on one end and effectively removing an overvoltage from the grid on the other end by receiving an immediate reward (or rather a reduced penalty) after performing those curtailment actions. Not all battery actions, however, give rise to immediate rewards. For example, discharging the battery in order to be able to charge the battery at upcoming moments of high PV generation only gives rise to long-term rewards, therefore making the learning process concerning battery actions more complex for the agent.

5.4 Conclusion

In this chapter an RL based controller was designed for a single-agent and multi-agent scenario with the primary objective of mitigating grid violations, whilst minimizing energy losses as a secondary target. Batteries and PV curtailment were used as flexibility resources. The agents in the multi-agent scenario act completely independent from each other and have to learn that there are other agents which act in the same, therefore non-stationary, environment. In the SARL setting, a single agent has control over the operations of a large grid battery placed at the end of the feeder.

The implemented RL algorithm, deep Q-learning with experience replay and a target update model, required fine-tuning of multiple parameters for optimal performance and avoiding convergence altogether. The optimization of these hyperparameters was done through a one-dimensional grid search for the single-agent case, the results of which were subsequently used for the multi-agent setting.

The overall results of the simulations lead to the conclusion that the performance of the controllers is inadequate after one year of training. Especially, the SARL controller performs poorly compared to the rule based controllers. Whereas the baseline battery controllers from chapter 4 resolved on average 85% of the overvoltages, the SARL controller decreased the average overall performance by 10% in comparison with the no-controllable resources scenario. Also, the generated energy losses are much higher compared to the baseline cases. In order to learn the optimal policy, the agent has to interact with the environment for many more years. The Q-values did not converge after one year of training which confirms the data inefficiency of deep reinforcement learning methods. As will be seen in the next chapter, the concept of transfer learning can greatly enhance the performance.

Finally, it was observed that the MARL controller outperforms the SARL controller by resolving on average 80% of all possible overvoltages. However, the MARL does create more energy losses in the process of doing so. This discrepancy in performance between the MARL and SARL cases was explained through the different reward-dynamics driving each of these control methods, which is one of the key findings in this chapter. Whereas, the SARL agent requires a focus on long-term rewards to discover an optimal planning policy for adequate regulation of its SoC, the MARL agent needs only a superficial insight, since PV curtailment does not require such long-term optimization scheme. Despite the better performance, the learned policy of the MARL agent is still not optimal and (similar to the SARL) more enhanced control strategies are necessary.

Chapter 6

Transfer learning

The analysis of the RL controller's performance developed in the previous chapter confirmed a well-known shortcoming of data-driven methods: their significant data-inefficiency. In order to design an accurate controller in real life, an abundant amount of data is necessary before adequate operational performance is reached. In most cases, such amounts of data are either not available or collecting a sufficiently large dataset is too time-consuming. The concept of transfer learning helps to circumvent these disadvantages: a controller is trained with available data in one domain and the gained knowledge is subsequently transferred to a different but related control task. In this chapter it is shown through two example cases that performance can be greatly enhanced by utilizing these principles.

6.1 Overview

To examine the applicability of transfer learning to the low-voltage grid optimization problem studied in this work, two cases are considered:

- **A well described distribution grid**: in this setting the physical aspects of the distribution grid in which the control task needs to be performed are known in advance by the control engineer. In our case, this is the Linear grid topology. The agents are trained ahead of time with available data in a simulated model of the environment. Once the grid is digitized, profiles from any project can be taken, including open source data. Subsequently, the gained knowledge is transferred to the agents acting in the "real" environment by using the trained DQNs as a starting-point instead of randomly initializing their ANNs.
- **An unknown distribution grid**: now the grid data of the "real" environment is not available, meaning no simulations of this topology can be made in advance. To enable transfer learning, the agents are trained similarly to the first case in a known environment (Linear), which physically differs from the actual target distribution grid. After the offline training, the experienced DQNs are used to initialize the agents in the "unknown" environment in order to bootstrap the online training.

The quotes around the "real" environment indicate that in this work no in-situ tests are done, but the real-life distribution grid is modeled as well. Since we use a model to represent this "real" distribution grid, we also need data to simulate the behaviour of the "real" agents acting in this environment. Therefore, a random sampling of REnnovates data IDs (from the pool of 82 collected datasets) is assigned to the 29 households. During training different random samplings are utilized, ensuring the transferred models never see the same environment twice. Summarized, this means the loads on each node is different in offline and online phase. In what follows, the two cases are compared with the randomly initialized DQN learner from chapter 5.3. In addition, the performance is equated to the baseline controllers from chapter 4.

6.2 Case study 1: a well described distribution grid

6.2.1 Setup

In the first case we study the potential of transfer learning where real world data is available for a given environment. Here, this is the Linear topology. Firstly, the agents are trained offline with data from the REnnovates project. Secondly, the information gained from the first training stage is transferred to the agents in the "real" environment by initializing the weights and biases of the target DQN with the same weights and biases obtained from the offline training. Additionally, the replay memory containing the agent's experiences $e_t = (s_t, a_t, r_t, s_{t+1})$ for a certain amount of time steps is carried over. Lastly, the agents are trained online for 1 year in the "real" environment, which has the same topology as the simulated environment. The transfer learning process is applied for both the single- and multi-agent cases.

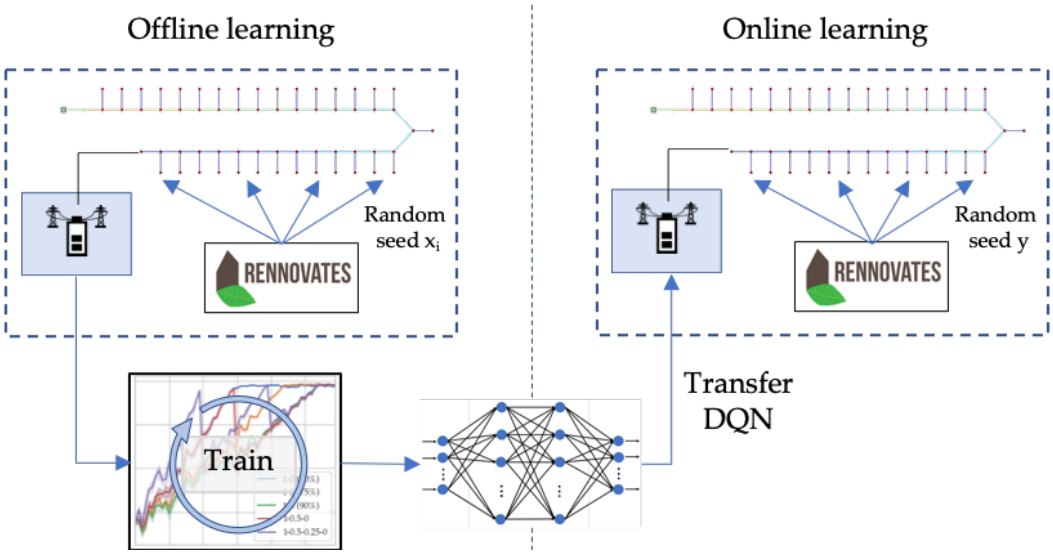

Figure 6.1: Single-agent transfer learning in a well described distribution grid. The random seeds determining the REnnovates data allocation differ: $\forall i : x_i \neq y$. This ensures the loads on each node is different in offline and online phase.

6.2.2 Precedent offline training with available real world data

The offline training process is similar to the online training process of the randomly initialized DQL controller, as stated in section 5.3. At the beginning of the first episode, the environment and all agents are randomly initialized, i.e. the weights and biases of their deep Q-networks are randomly chosen. In contrast to the normal DQL controller, the agents are trained over multiple episodes covering one year of data each, but with different distributions of REnnovates data IDs over the network. The latter is done by "resetting" the environment at the beginning of each new episode. To retain the knowledge obtained during the previous episode(s), the trained deep Q-networks of the agents are saved and passed on to the next simulation. The left-hand side of figure 6.1 summarizes the procedure.

By following this approach, the agent can learn and interact with the environment for multiple consecutive simulations effectively overcoming the issue of data-inefficiency: available data spanning only a limited time-horizon is intelligently employed. The DQL algorithm used to train the agents is given in section 2.3.3. For an overview of the utilized hyperparameters and the optimization process for finding these values, the reader is referred to appendix A.5.2.

Besides the difference in number of episodes, another key difference with the normal DQL controller is the allowable exploration rate in the ϵ-greedy policy. Section 5.2.2 dwelled upon the exploration-exploitation dilemma, indicating a trade-off between sacrificing the highest expected short-term rewards for the possibility of discovering a more favorable policy. It was established that in online training ϵ cannot be too large since this leads to high numbers of violations and battery losses. A randomly acting grid battery is not operationally justifiable. At the same time, a small ϵ limits the agent in exploring the new environment.

In an offline learning process in a simulated environment, random actions do not lead to potentially dangerous outcomes. This is one of the main advantages of offline learning: the exploration of the agent can be much larger than with the online variant, allowing the agent to more thoroughly search the policy space. This leads to a speed up of the learning process. Typically, an epsilon decay scheme is used where ϵ is chosen large at first (e.g. $\epsilon = 1$) and is reduced linearly following the reset of each episode. In doing so, the exploration is large at the beginning and the exploitation of optimal actions increases towards the end of the learning process, allowing verification of the performance.

The process described above is performed for both the single- and multi-agent scenarios, as seen in chapter 5. The most important results are described in the following sections. At the end of these training simulations, the weights of the neural networks are saved and are ready to be transferred to the agents facing the online learning task. It is emphasized that the training of the MARL agents is completely independent. Therefore, multiple DQNs (one per agent) are saved and transferred to the corresponding agents in the "real" environment.

Single-agent: district battery

The SARL agent, which controls the district battery, is first trained offline for 60 episodes - which equals 60 years of training data. The ϵ-value linearly decays from 1 in the first episode to 0 in the 45th episode. Afterwards, ϵ is kept constant at 0. The latter is the same as the online training ϵ-value of the randomly initialized DQL based controller used in section 5.3. To keep a fair comparison, $\epsilon = 0$ will also be used for the online learning with DQN transfer further in this chapter. The results of a representative offline training run are presented in figure 6.2, indicating the evolution per episode of the rewards, grid violations and battery losses. It can be seen that the rewards converge to a final asymptotic value. With respect to the primary objective of solving grid violations, the SARL agent has clearly adopted a working policy: almost no residual grid violations remain in the last episodes.

We also see the necessity for working with a 60-episodic task, as the battery losses seem to stagnate only during the final episodes. This highlights the importance of a correctly fine-tuned reward function: it should be the agent's main objective to solve grid violations. To achieve this, a long-term policy with adequate planning to regulate the SoC optimally over time is needed. When penalizing battery losses too harshly, such learning behaviour might be discouraged. With the current scaling factors, it can be seen that reduction of the battery losses is kept as a secondary target, starting convergence well after an optimal violations-policy was found.

Similar to the evaluation of the randomly initialized DQL controller, we study the evolution of the Q-values from the DQN of the SARL agent (see figure 6.3 and 6.4). First, we consider noon in the summer at a large aggregated power injection in the network. In contrast to the findings in figure 5.9, the offline learner does find the optimal battery action: charging at 75% of $P_{b,max}$. The difference with charging at full capacity, however, is almost indistinguishable. A second, very interesting dynamic, is given in figure 6.4. Here we look at a morning in summer, a little less than two hours before large PV generation is forecasted and at high SoC of the battery. A clear planning approach is noticed: despite the short-term penalty for battery losses, the agent decides on discharging its battery to proactively counter the impeding overvoltages. The interested reader is referred to appendix A.7 for a detailed view of the grid battery's operations over time following the learned policy.

Multi-agent: house batteries and curtailment

In the offline learning process for the multi-agent case, the agents control their own house battery and curtailment of their own PV installations. Similarly to the SARL approach, the agents are trained offline for 60 episodes with a linearly decaying ϵ-scheme from 1 in the first episode to 0 in episode 45 after which they fully exploit the learned policy. Since we opted to work with independent learners, the agents have to estimate the influence of other agents on the grid tracking actions throughout a non-stationary environment, whilst trying to figure out the individual best action to resolve voltage violations.

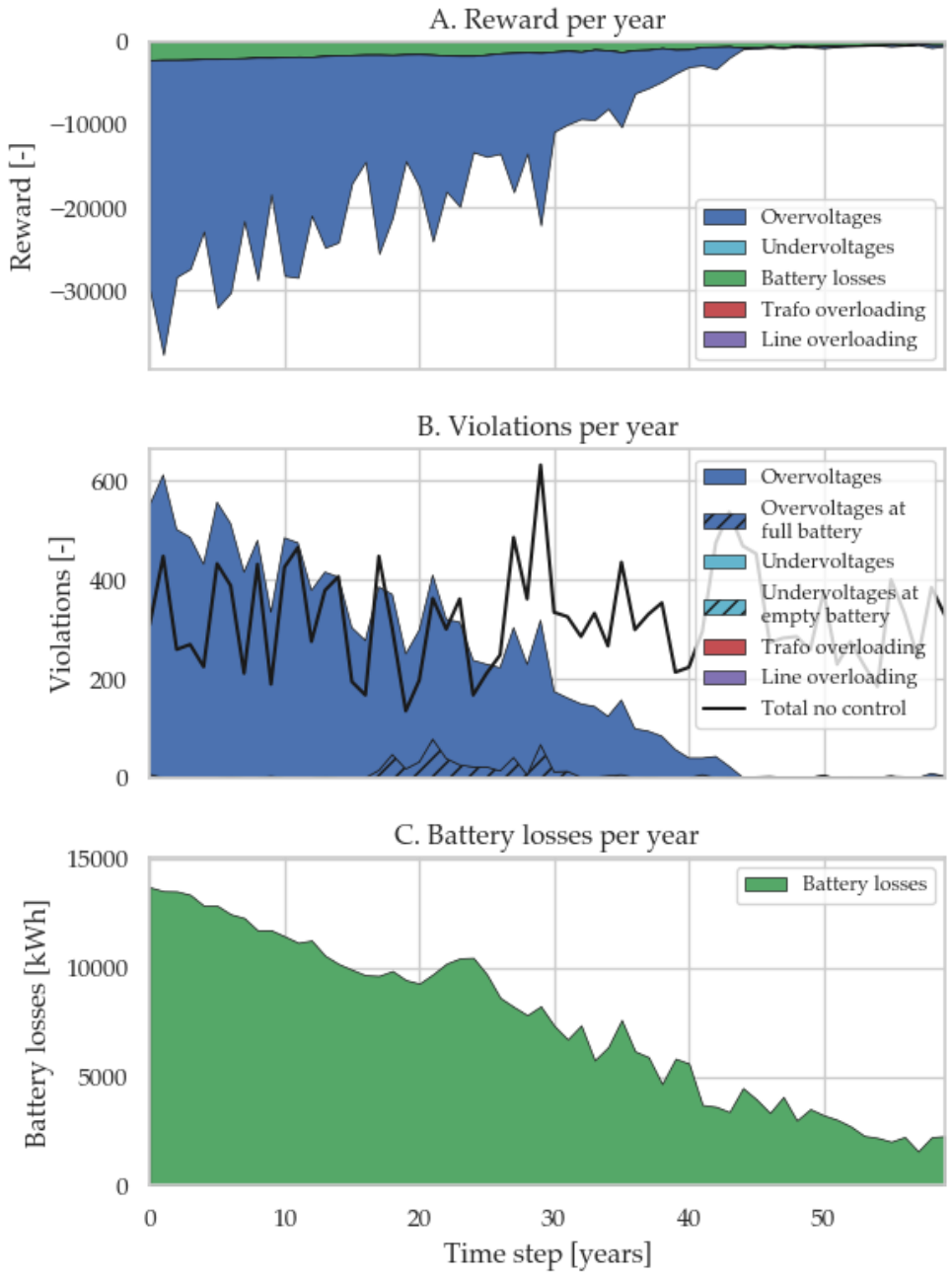

Figure 6.2: Offline training of the SARL agent. The agent successfully learns solving almost all grid violations after 45 episodes, after which the secondary objective of minimizing battery losses is initiated. Note the strongly varying violations per year in the reference no-controllable resources scenario (black line): this indicates the high variance between the different REnnovates data allocations in each simulation.

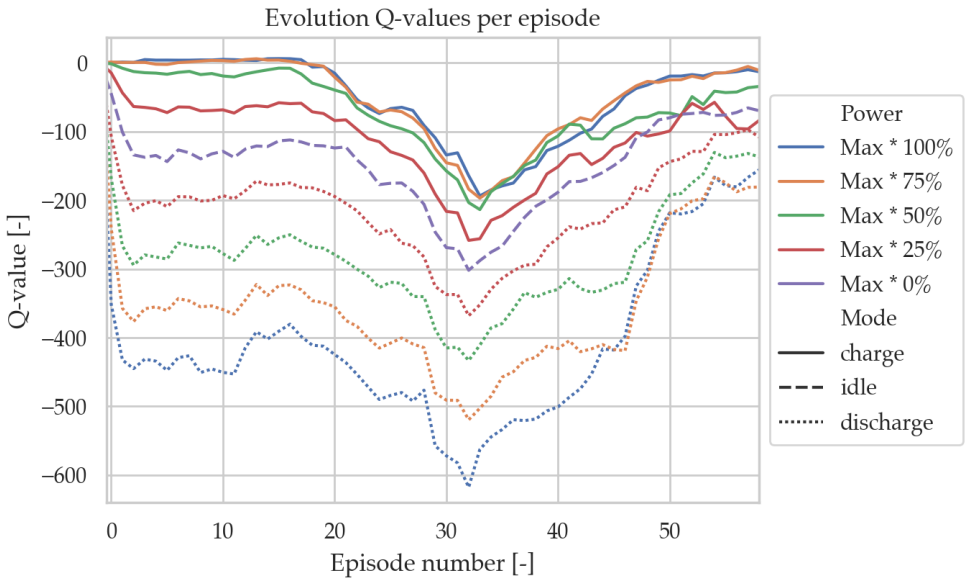

Figure 6.3: The Q-values at noon in the summer indicate the agent has correctly learned to charge the battery at high forecasted aggregate power injection into the network. There is a clear preference for charging the battery at 75% (which was found to be the optimal action in this case) or 100% of $P_{b,max}$.

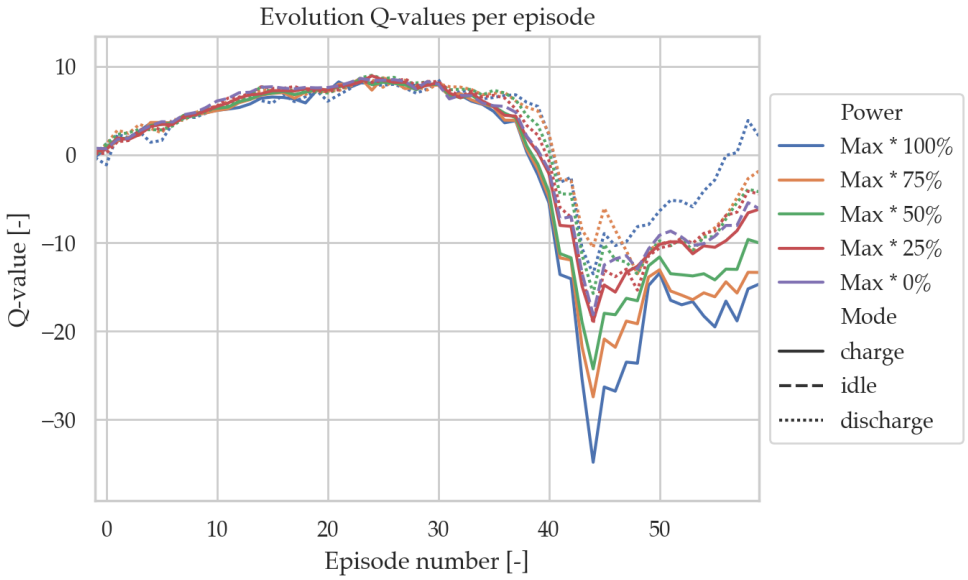

Figure 6.4: The Q-values at morning in the summer a little less than two hours before large PV generation is forecasted and at high SoC of the battery. The agent starts adopting a correct SoC planning policy after ±40 episodes, deciding on discharging its battery towards more optimal values.

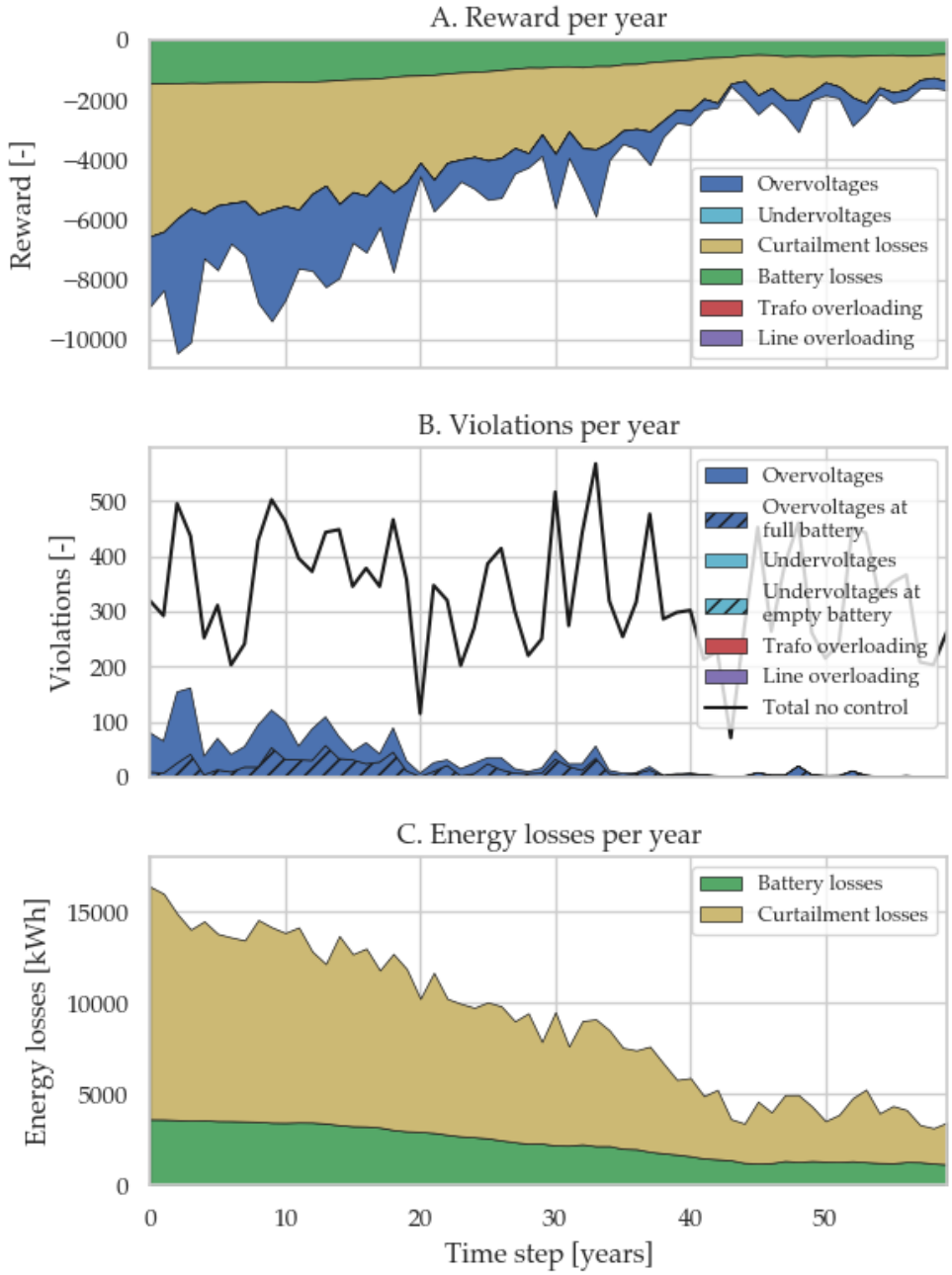

Figure 6.5: Offline training of the MARL agents. The energy losses are summed over all agents. A shift in learning dynamics can be observed compared to the SARL offline learner: the agent quickly understands that curtailment solves upcoming overvoltages, but starts off by curtailing at unnecessary times. The major objective seems to be a reduction in energy losses, an asymptotic value of ±3800 kWh per year is reached.

Figure 6.5 shows the results of the offline MARL learning process. Even during the first episode ($\epsilon = 1$, completely random actions) around 70% of overvoltages are solved. This is due to lower sensitivity of the grid to faulty actions of a sole agent: PV curtailment can never cause overvoltages and due to the distributed approach, agents can (by accident) correct each others bad decisions. This leads to the same dynamics as found in section 5.3.2 when comparing the regular SARL and MARL controllers. The clear two-stage learning process observed for the offline SARL scenario - minimizing violations and thereafter reducing battery losses - is therefore no longer visible. Rather, a completely new learning dynamic is established. The losses are abundant during the first episodes, mainly due to the high curtailment losses and a smaller part from battery losses. A large quantity of training data is needed and the major objective is shifted towards reducing the incurred losses.

6.2.3 Online training of the agent in the real environment

After having performed the 60-episodic offline learning task, the obtained DQN models can finally be transferred to the online agents in the "real" environment. For this, the DQN starting-point method is used, in which the "real" agents ANNs are initialized with the transferred models. To follow the same procedure as employed during the offline learning, the replay memories are transferred as well. The concept is shown earlier in the right-hand side of figure 6.1. The attentive reader might have noticed that this transfer of offline-to-online model is completely equivalent to just adding a 61th episode to the learning path described in the previous section. The main difference, at least in this work where no in-situ tests are done and the "real" environment has to be modelled, lies in the perception of the transfer: one should imagine this 61th episode as a real-life one-year field test of the controller in practice.

Single agent: district battery

In figure 6.6 an example of an online training process is given for the SARL scenario. The same results as during the last episodes of the offline learning process are, logically, observed. Once again, almost all violations are solved. If we compare this figure to figure 5.8 of the RL controller without transfer learning, we can clearly see a substantive improvement. In addition, it can be observed that the losses in winter are just a fraction of these in summer, but sporadic peaks are still observed.

Multi-agent: house batteries and curtailment

Figure 6.7 similarly shows an online example for the multi-agent case. It is clear that also in this case the agents have learned a violation-solving policy. Two things catch the eye: i) the near perfect mapping between resolved violations and PV curtailment losses, and ii) there are almost no battery losses present. The agents have learned the strong correlation between their curtailment actions and the mitigation of overvoltages, i.e. they have established an internal mapping between the aggregate power balance and calendar features on one end and the possibility to avoid an overvoltage through curtailment at the other end.

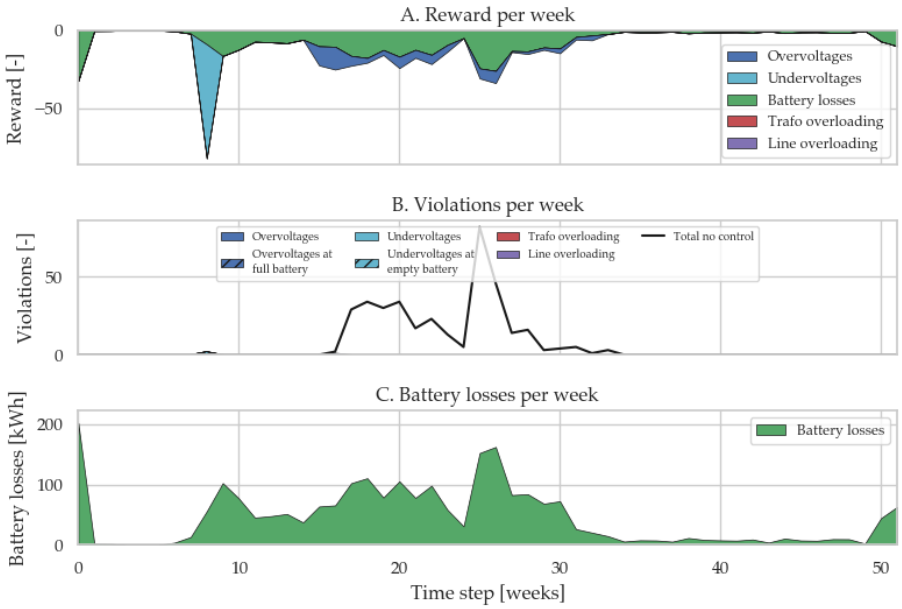

Figure 6.6: Online training of the SARL agent, initialized with transferred knowledge from the offline training process (DQN and replay memory). The results show great performance, solving nearly all violations whilst keeping the battery losses low during the winter. Only during spring some unnecessary battery losses are observed.

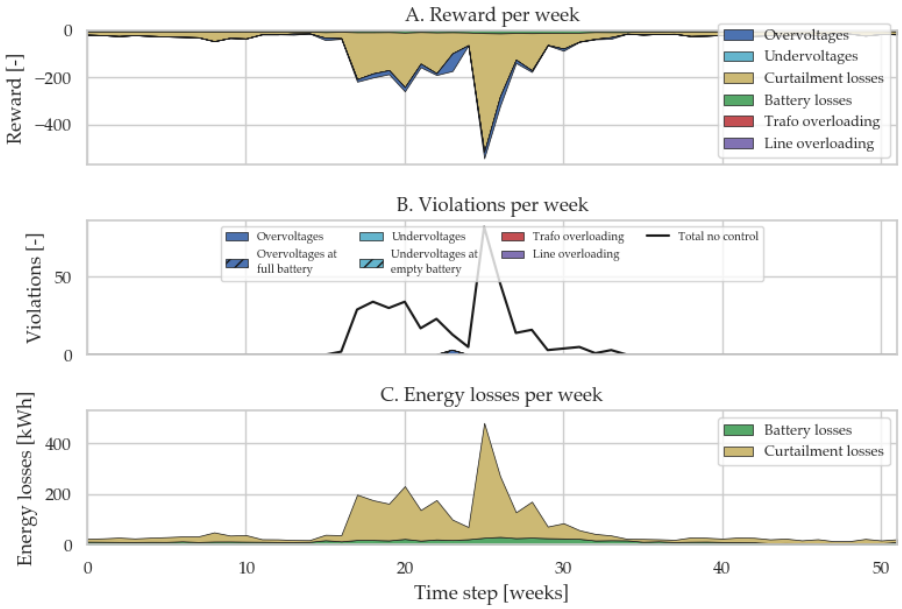

Figure 6.7: Online training of the MARL agents, each of them bootstrapped with the DQNs of the corresponding agents in the offline learning task. A clear policy is established with primarily curtailment as method to resolve violations.

Importance of the replay memory

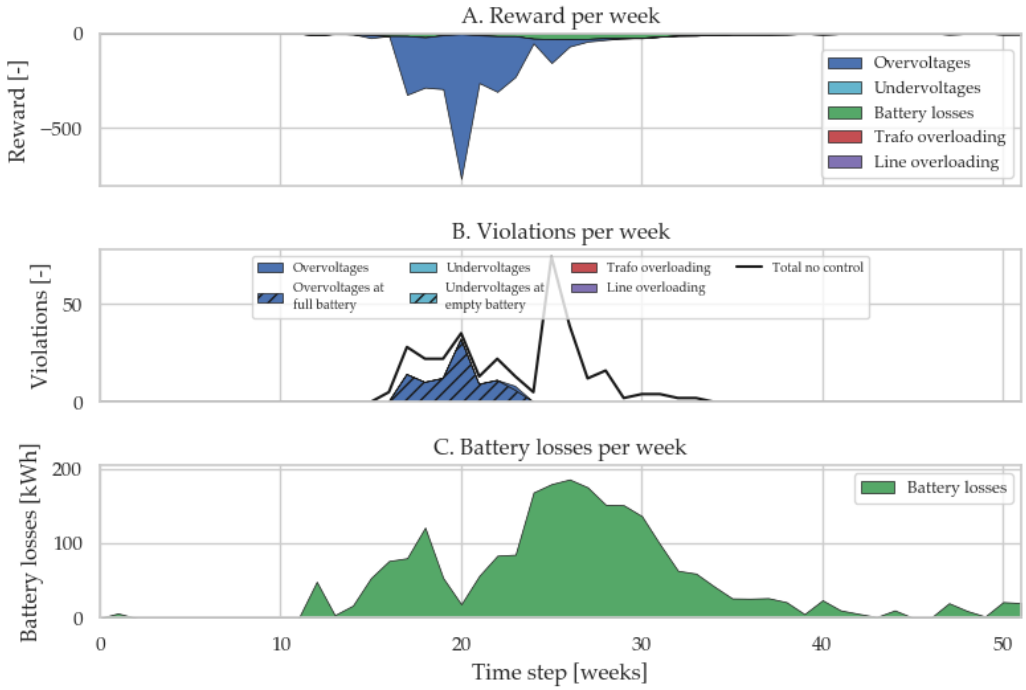

Figure 6.8: Online training of the SARL agent without transfer of replay memory. The same randomized allocation of data IDs to the households as in figure 6.7 is used. The performance worsens during the first summer months.

In the transfer learning procedure considered in the previous sections, the combination of DQNs and replay memories was passed on to the online learning agents. An interesting side-note towards a better understanding of the actual dynamics underlying the learning task, is found when considering the case where the replay memories are not transferred. Towards this end, compare figure 6.8 (no replay memory carried over) and figure 6.7 (replay memory transferred).

If the agent is initialized with an empty replay memory, the only experiences $e_t = (s_t, a_t, r_t, s_{t+1})$ gathered in the first months of training are from winter data. This abrupt change from a full replay memory (containing 4 years of training experiences, thus entailing a mixture of seasonal transitions) to a limited memory of only winter data in the online learning alters the learned policy of the agent notably. Moreover, at the beginning of the summer the agent has forgotten the optimal policy with large PV generation as only "winter states" have been seen. Only after mid summer, the agent has "recovered" and is able to resolve overvoltages again. This supports the discussion in section 5.3.2, where similar remarks were made about the importance of seasonal dynamics for the regular DQL controller. The agents there face the same issue of having a non-representatively filled replay memory, but are even worse off since their ANNs are randomly initialized.

6.2.4 Validation of the DQL controllers

In the previous section we established the qualitative performance improvement gained from offline-to-online transfer learning. To quantify these findings, the RL controllers with randomly initialized DQN agents developed in chapter 5 are equated with these enhanced learning agents. The same structure is followed as in the rest of this chapter, focusing on the SARL case first after which the MARL scenario is elaborated. To conclude the first case studied in this chapter, we bring all elements from chapters 4, 5 and 6 together in a comprehensive control strategy comparison for the REnnovates-Linear setup.

Single agent: district battery

The results for the no-controllable resources scenario and the RL controllers with and without transfer learning are presented in figure 6.9. Focusing on the SARL case, the adequacy of the transfer learning agent (with replay memory carried over) regarding the mitigation of grid violations is readily verified. In all of the performed simulations, more than 99% of violations are prevented, with some runs even showing a perfect improvement of 100%. Compared to the regular RL agent, which showed a 10% worsening compared to the no-controllable resources scenario, this is an increase of ±95 percentage points - a more than noticeable difference.

With respect to the incurred battery losses, the RL controller after transfer learning achieves an average yearly energy loss of ±1000 kWh. Compared to the 2000 kWh of the regular DRL agent, this translates into a 50% improvement. Keeping into consideration the accompanying reduction in grid violations, this is a remarkable feat. After all, to resolve more grid violations the agent needs to learn an optimal battery scheduling policy, including substantive ahead-of-time charge and discharge planning for optimal SoC regulation. In the regular RL process, the agent clearly fails to find such policy, but merely succeeds in generating a limited state-action mapping focusing on short-term battery operations.

These results highlight the usefulness of transfer learning in practical control tasks on low-voltage distribution grids with known topology data. An interesting element to note is the real-life simulation time needed to perform the offline SARL training procedure: approximately 48 hours where needed to train the final models, but the fine-tuning of the reward function and other hyperparameters preceding these runs increased the total computational burden. Nonetheless, the remarkable performance improvement achieved through solely two days of simulated training are a key result of this work indicating the usefulness of transfer learning in DR settings.

Finally, the results in figure 6.9 formalize the discussion in the previous section regarding the importance of carrying over both the DQN and replay memory. That is, it can be seen that the performance (both violations solved and battery losses incurred) of a MARL agent with carried over experiences outperforms the agent with sole ANN transfer over each of the 100 randomized simulations.

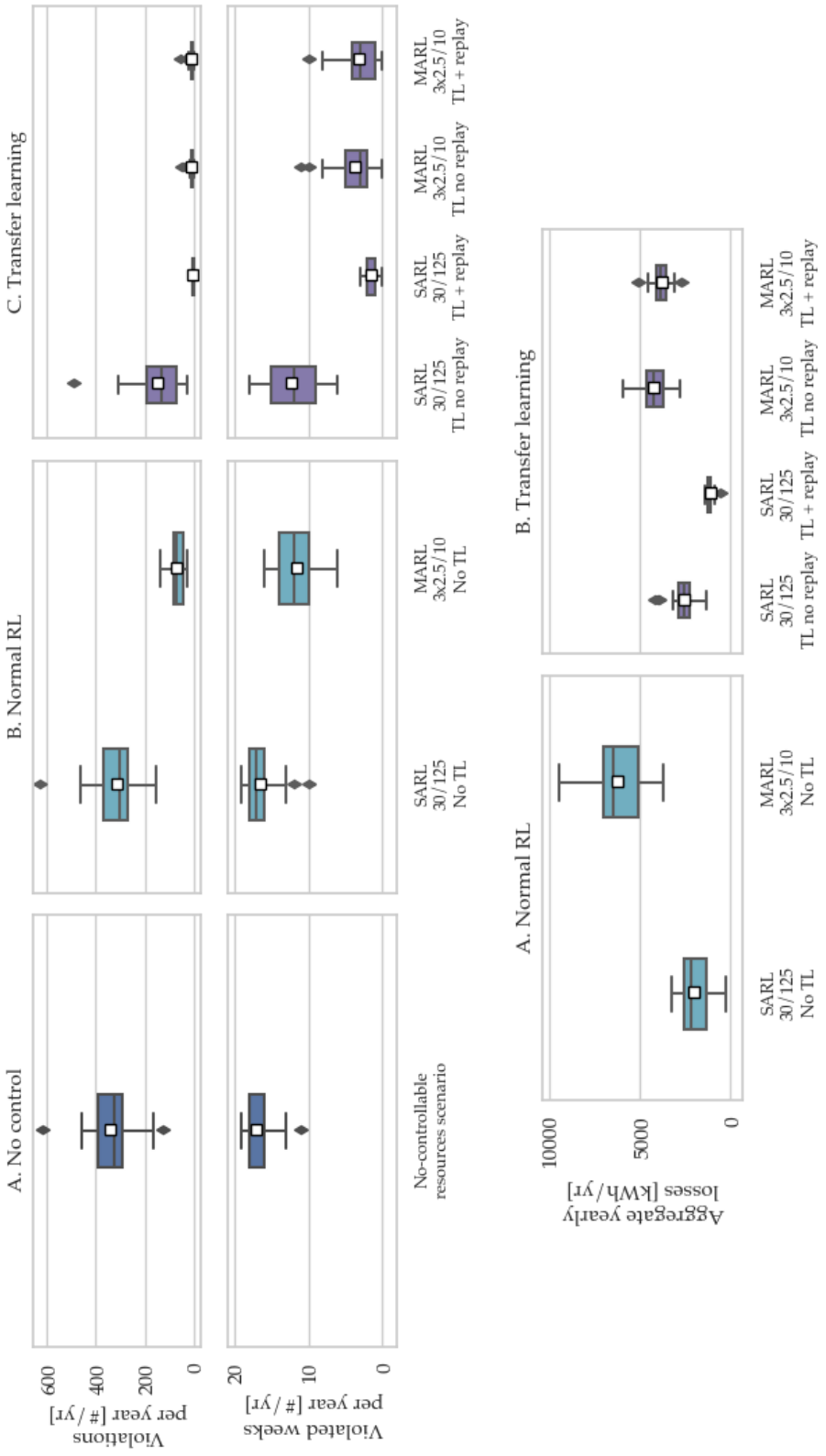

Figure 6.9: Comparing violations and losses between the RL controllers for 100 randomized house ID distributions.

Multi-agent: house batteries and curtailment

Next to the SARL results, figure 6.9 additionally shows the simulated findings for the no-controllable resources scenario, the randomly initialized MARL controller and MARL controller with transfer learning. The latter once again shows superior performance in comparison with the reference scenario where no control is implemented. However, the difference with the normal RL control strategy is less pronounced, a phenomenon which was attributed to the lower grid sensitivity towards faulty MARL actions and the "easier" policy these agents can learn to resolve grid violations, i.e. curtailment of their PV units without the need to develop a more complex planning approach needed when only battery control is entailed. Overall, the MARL controller with transfer learning resolves on average 98% of the violations, reiterating the success of their learned policies as discussed in the previous sections.

Whereas the difference towards grid violations is less pronounced between the regular RL and transfer learning cases, the results of the losses in figure 6.9 indicate a more pronounced distinction: the MARL agents with transferred knowledge clearly manage their operations under a more energy efficient state-action mapping. This supports the findings presented in figure 6.5, where the offline training procedure of an exemplary MARL agent was visualized. It was found that the losses converged to an asymptotic value of ±3800 kWh, the same value is found in the experimental analysis presented in this section. Compared to the yearly energy losses of 6200 kWh for the non-transfer learned MARL case, this is an improvement of approximately 40%, again demonstrating the benefits of transfer learning.

An interesting observation is made when MARL transfer learning with and without passing over the replay memory. Not passing on the experiences pooled in the agents memory has less influence on the MARL agents. In the SARL scenario, the early winter experiences fill up the replay buffer and subsequently overwriting the more complex SoC-planning policy. This has a bigger impact on overall performance than similar alterations of the less complex MARL policy - myopic curtailment - which is more easily recovered when the first overvoltages are observed.

Finally, to keep in line with the discussions given in the SARL section, the offline simulation time for the MARL transfer learning based controller amounted to ±4 days. The higher computational cost of training a multi-agent system is an important disadvantage which the control engineer should always keep into consideration.

6.2.5 Comparing all controllers

Throughout this work two main control strategies with aim of mitigating grid issues have been developed: on one end the rule-based controllers, on the other end the reinforcement learning based approaches with or without transfer learning. Each method has its strengths and weaknesses, both on a conceptual and purely performance level. To assess the latter, all of the analysis performed on the different types of controllers in this thesis are summarized in figure 6.10 and figure 6.11.

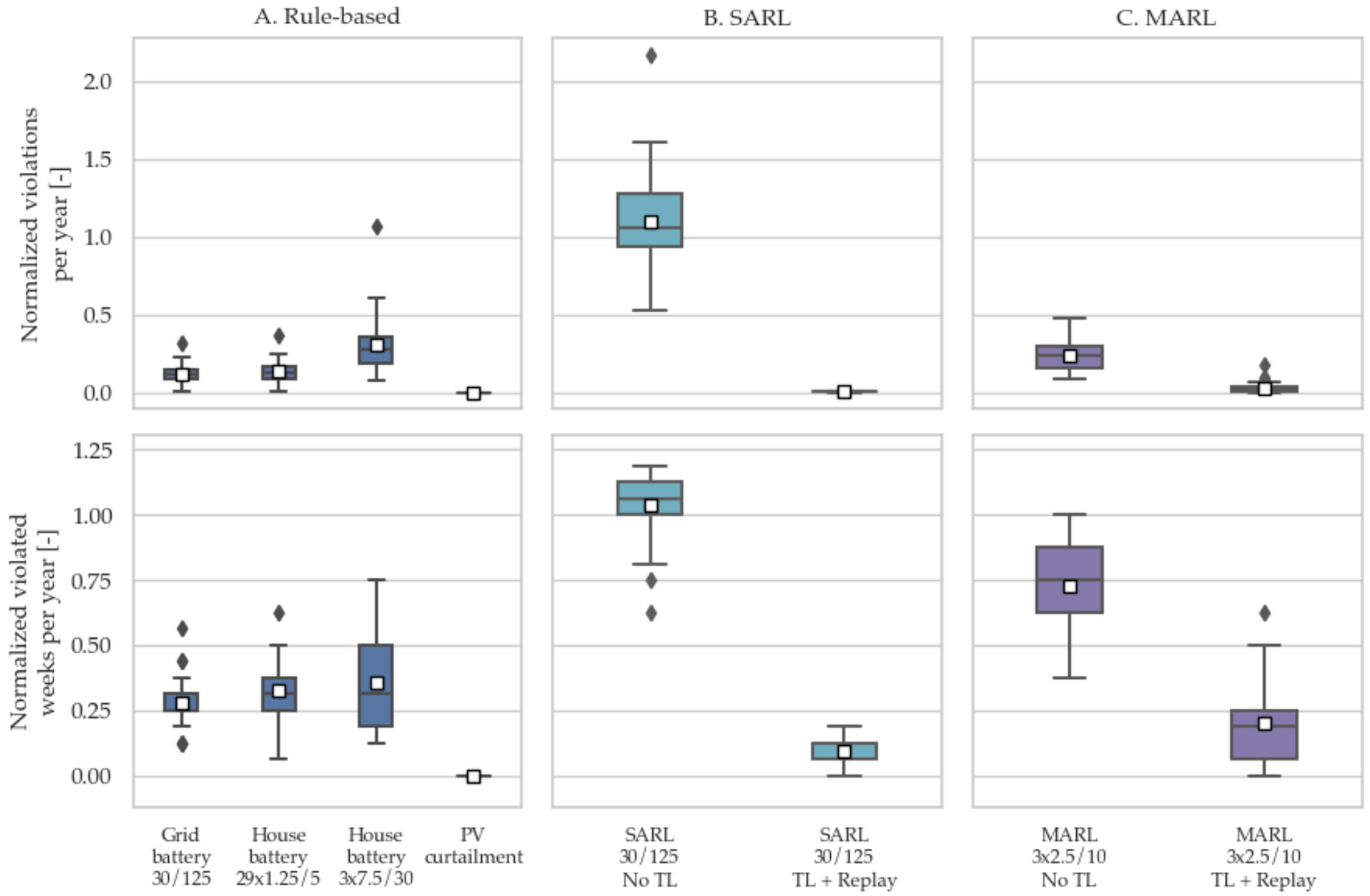

Figure 6.10: Comparing violations amongst all controllers. Battery sizes indicated as $P_{b,max}/E_{b,max}$ (kW/kWh).

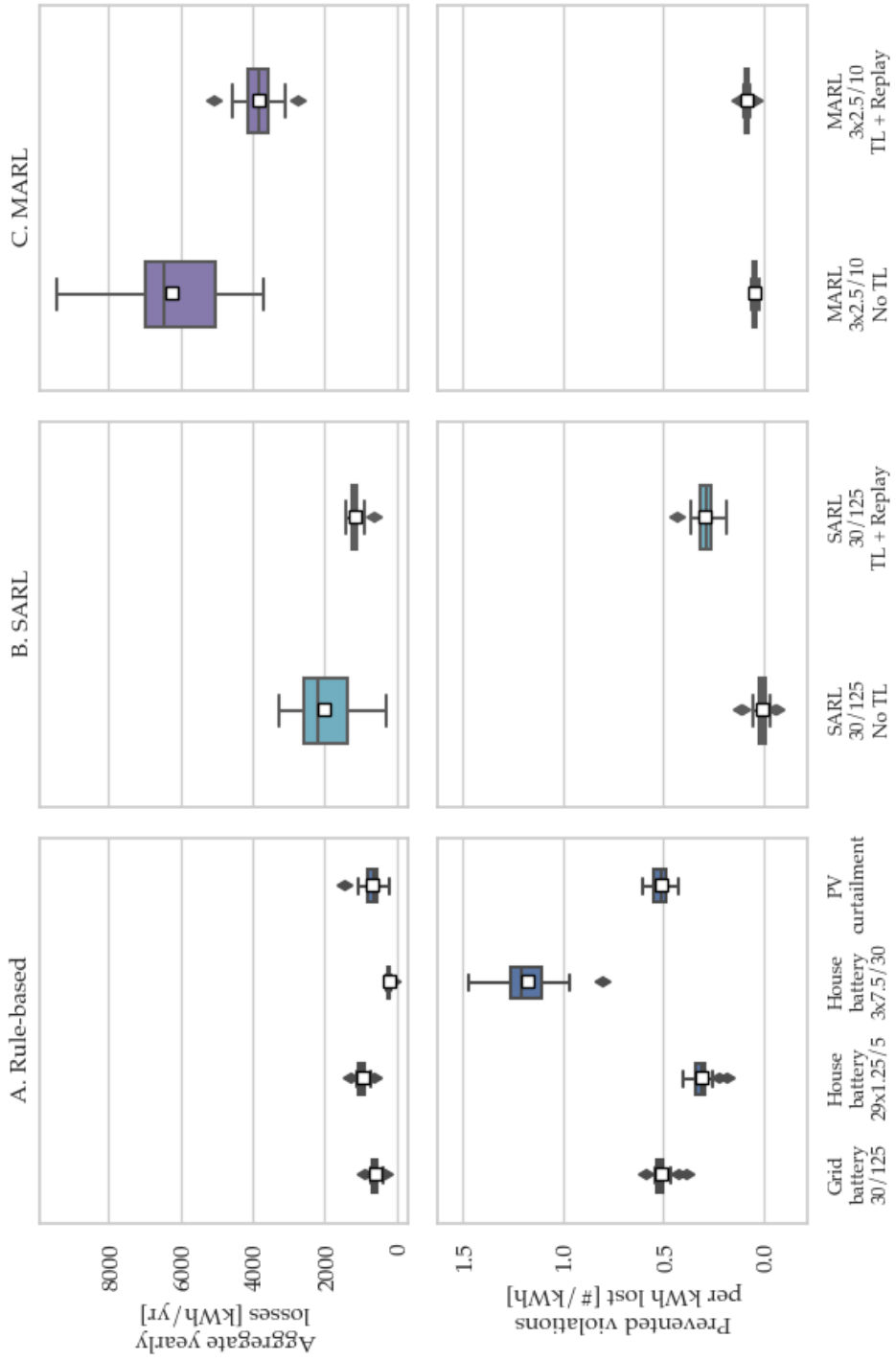

Figure 6.11: Comparing losses amongst all controllers. Battery sizes indicated as $P_{b,max}/E_{b,max}$ (kW/kWh).

Before jumping into the results, we would like to add an extra rule-based controller to the comparison. In chapter 4 the most broad scenario for the distributed house controller was considered where all 29 houses were assigned a battery unit. For the MARL case, however, due to computational limitations this amount of agents is infeasible. Nonetheless, a reference with three houses at the end of the feeder, controlled through a rule-based scheme, would be an interesting comparison for the multi-agent performance. For this reason the decentralized algorithm presented in section 4.2.2 is reconsidered, but now only the last three houses on the feeder are controlled through this rule-based approach. For the battery sizing, the exact same procedure as presented in section 4.3 is followed, leading to $P_{b,max} = 7.5$ kW and $E_{b,max} = 30$ kWh. Notice that, since no PV curtailment is available in this novel control method, a larger battery size is needed in comparison with the MARL case.

In what follows, a brief overview of the most important observations is given, substantiated by the different analysis performed throughout this work:

- Within the group of rule-based controllers, only the PV curtailment strategy is capable of resolving all grid violations. The biggest limitation of the battery based methods is their incapability to intelligently regulate P_b, leading to situations in which the maximum energy content is prematurely reached when further charging to avoid overvoltages is needed. To obtain a more intelligent battery control, diverse RL techniques were developed.
- The regular RL strategies created in chapter 5 perform poorly in comparison with the rule-based controllers. The large data-inefficiency of such methods lies at the origin of these issues.
- Through transfer learning with carry-over of the replay memory the performance of the randomly initialized RL agents with respect to violation-solving capabilities and incurred energy losses can be greatly enhanced. These TL controllers outperform all but the rule-based PV curtailment controllers when considering resolved grid violations.
- Without transfer learning, the randomly initialized MARL controller outperforms the SARL counterpart in terms of resolved violations, but suffers more battery losses in the process of doing so. This difference was explained through the different reward-dynamics driving each of these control methods: the SARL agent requiring a focus on long-term rewards to discover an optimal planning policy for adequate regulation of its SoC, versus the MARL agent for which a more myopic view suffices since PV curtailment does not require such long-term optimization scheme. The latter allows much faster learning for the MARL agents, which translates into the observed results in figure 6.10 and 6.11.
- When comparing the house-level control strategies (MARL with and without transfer learning, and rule-based battery control with 29 or 3 demand response enabled households), our MARL agent with transfer learning outperforms all other methods, even the rule-based approach where all 29 households are actively trying to balance the network. This is a particularly interesting result, indicating the capabilities of our RL method.

- Despite their great performance with respect to solving overvoltages, the RL controllers are characterised by substantial energy losses. In the end, it is the DSO who needs to decide to what (economic) extend the prevention of grid violations is justifiable with respect to the suffered energy losses.
- Finally, the two controllers with the best overall performance (high reduction in grid violations at the lowest energy losses) are the rule-based PV curtailment and SARL grid battery controllers. The latter has the advantage of also being capable of solving undervoltages, but this issue was not observed in the studied REnnovates-Linear setup. The PV curtailment technique, however, requires supervisory control over multiple house-level PV installations in the network, but does not entail the high fixed costs linked to large-scale grid battery storage.

In practice, rule-based controllers are often used as back-up systems for RL controllers, especially during the initial learning phase. Therefore, different combinations of rule-based and RL controllers developed in this thesis could be considered.

6.2.6 Centralized vs. decentralized control

The independent MARL method has some advantages over the rule based controllers concerning pure control aspects. First of all, compared to the baseline controllers, the MARL strategy removes a single point of failure: the central controller. This results in a more robust system architecture. Only the forecasts of the aggregated power balance and the monitored voltages on the grid are communicated to the independent agents. Additionally, if data-communication to one agent fails in the multi-agent scenario, the other agents are not influenced and can mitigate the effects. Similarly, failure of one agent would lead to a less catastrophic outcome compared to a malfunctioning central power flow calculation in one of the rule-based scenarios or to a wrong action of the SARL controller (higher maximum power). Lastly, in contrast to the SARL case, the MARL controller still requires the presence of energy communities to justify the unbalanced costs, which applies also for the rule-based controllers on house level.

6.3 Case study 2: an unknown distribution grid

6.3.1 Setup

In the second case, we study the potential of transfer learning if real world data is available, but in contrast to the first case the physical aspects of the distribution grid are unknown. Therefore, the agent is first trained offline with data from the REnnovates project in the known environment from the Linear project. The obtained knowledge is subsequently transferred to the agent in the "real", unknown environment by carrying over the DQN. Finally, the agent is trained online for one year in the "real" environment. Due to limited computational resources, this case is only researched for a single-agent scenario.

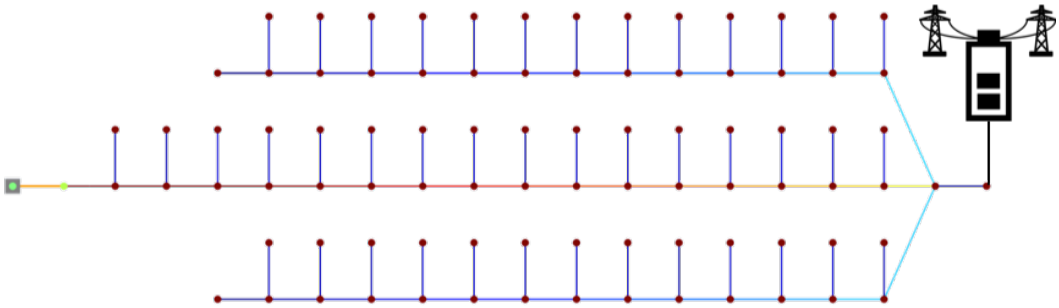

Figure 6.12: The modified grid used to study the potential of transfer learning when the environment is different in the offline and online training phases. It is presumed that the grid topology data from this distribution network is not known in advance.

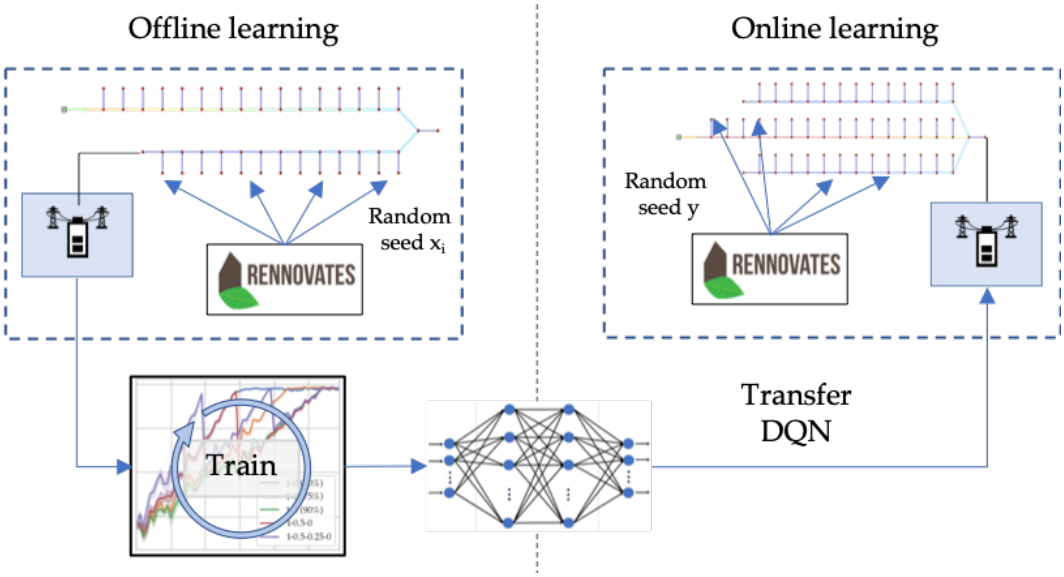

Figure 6.13: Single-agent transfer learning in an unknown distribution grid. The agent is trained offline in the Linear topology after which the experienced DQN is transferred to the "real", unknown environment.

6.3.2 Precedent training in a well-known environment

Similarly to the first case, the agent is trained offline on the Linear grid topology following the exact same procedure as described in section 6.2.2 of this chapter. Because of this approach, the reader is referred to the discussions in this section about the observed offline learning process.

6.3.3 The "unknown" environment

To establish the environment in which the online control task has to be performed, a fictitious, "unknown" grid was designed. The topology for this network is derived from the Linear setup: the bottom feeder was copied and symmetrically attached to the top-side of the network. Thus, a larger distribution system with radial branching is obtained. Figure 6.12 gives a graphical representation while indicating the position of the SARL controlled grid battery. The used converter power and battery energy content, respectively 45 kW and 190 kWh, are larger compared to case 1 because of the increased number of total houses connected to the network. A linear scaling approach based on the ratio of the number of houses in case 1 and case 2 is applied to obtain the proposed battery sizing. The choice for the indicated placement stems from the findings in section 4.3, indicating the battery can most efficiently resolve violations when placed near the end of the network. Therefore, the furthest common node between the two splitting up branches is chosen. It is assumed the control engineer has no practical information on the physical data of this network, and thus has to reside on the transfer learning process elaborated in this case study.

6.3.4 Online training of the agent in the "real" environment

To transfer the gained knowledge from the offline learning process to the online agent facing the control task on this enlarged network, the same starting-point method as in case study 1 is applied: the DQN is used to initialize the ANN of the RL based grid battery controller in the "real" environment. However, the replay memory is not transferred since the experiences gathered by the offline agents can drastically differ from the experiences collected in the new environment. Some informal test simulations were performed with transfer of replay memory, but no notable performance improvement was observed.

At the beginning of the online simulation, a random sampling of REnnovates load profiles (from the pool of 82 collected datasets) is assigned to the 42 households. The ϵ-value is kept constant at 0.1, a higher value compared to findings in section 5.2.2 for the Linear topology. It was experimentally observed that this higher exploration rate led to enhanced performance. A higher ϵ-value ensures more exploration and adjustment of the learned policy. This online training procedure is repeated multiple times (100 simulations with randomized REnnovates load profile distributions) for both a randomly initialized RL agent and RL agent with transferred DQN. Figure 6.13 summarizes. The findings are presented in the next section.

6.3.5 Validation of the DQL controller

The aim of case study 2 is to research the applicability of transfer learning in the context of the described unknown distribution network. This new environment differs strongly from the online training environment: an increased number of houses, a different grid topology and different battery sizing are implemented. Therefore, we consider solely the TL aspect, focusing on the differences between the performance of the randomly initialized agent and agent with transferred DQN.

Figure 6.14 shows the performance of the controllers with and without transfer learning over the 100 randomized simulations. The controller without offline training resolves only a small amount of few grid violations. A clear performance improvement is observed however for the SARL transfer learning agent, with an average reduction in the number of remaining grid violations of 8% (and up to 16% in some of the simulations) compared to the randomly initialized controller. As one might expect, the improvement is not as pronounced as in case study 1, but the clear advantage of transfer learning is established nonetheless.

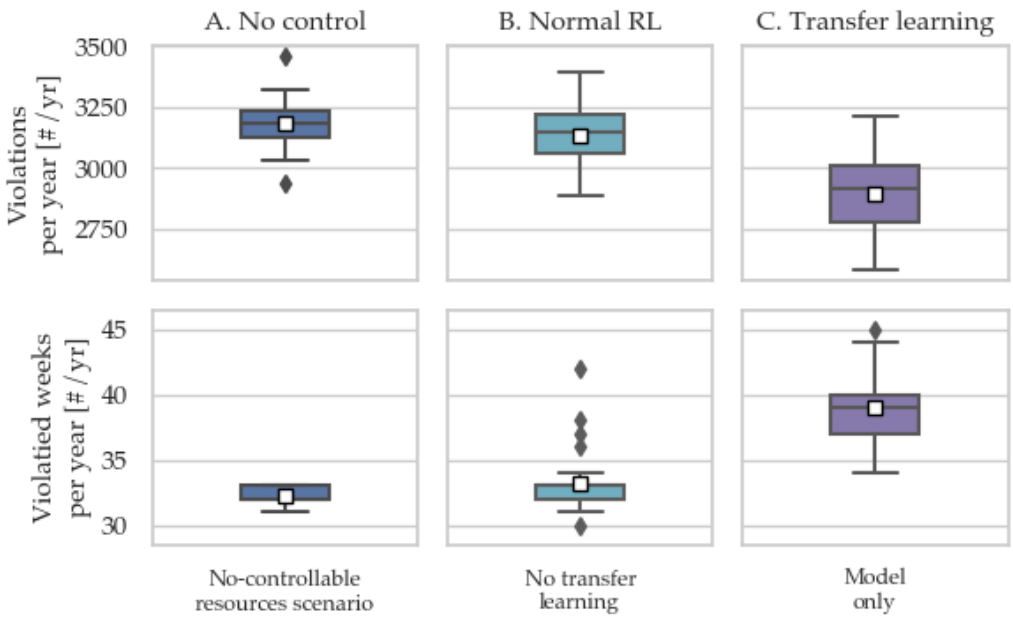

Figure 6.14: A controller with transfer learning performs better than the controller without transfer learning. Not all violations are resolved, but the result shows that transfer learning is even useful when the topology in online and offline training differs.

To gain some more insight in these results, figure 6.15 and figure 6.16 show an example of the online training in the new grid for one of the randomized simulations (the same random seed was applied to compare the transfer and no-transfer learning cases). A clear improvement can be seen when comparing both cases: the TL agent adopts more quickly a (limited) violation-solving policy.

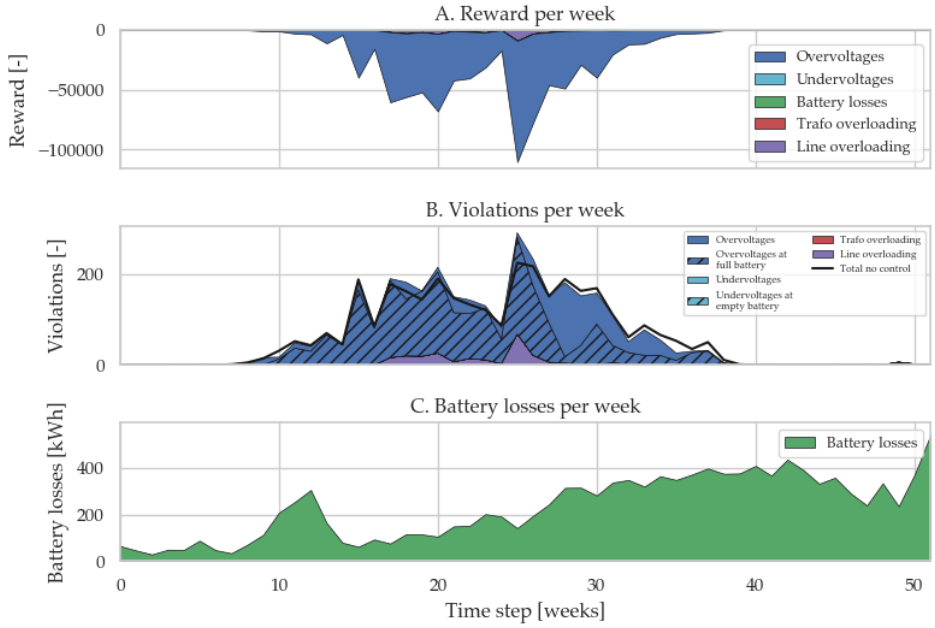

Figure 6.15: Online performance of the randomly initialized SARL agent. Almost no grid violations are solved, indicating the poor data-efficiency of this method.

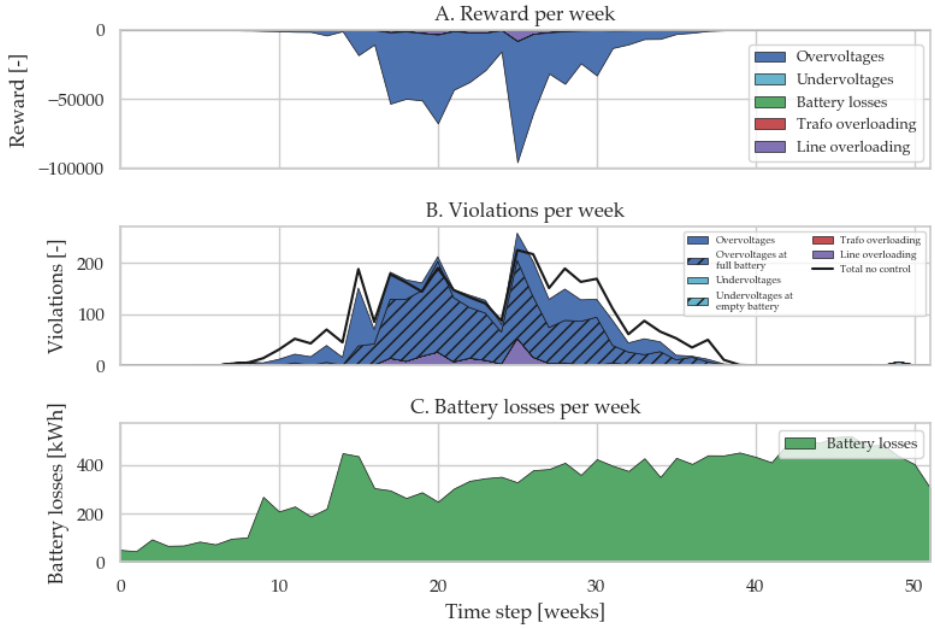

Figure 6.16: Online performance of the SARL agent when initialized through transfer learning. A clear improvement can be observed with respect to resolved violations, especially towards the end of the summer when the agent has gained more experience in the new environment

6.4 Conclusion

In this final chapter the most important elements elaborated throughout the rest of this work were brought together in the context of a transfer learning based RL controller. The latter had the aim of increasing the poor performance of the randomly initialized RL agent(s) from chapter 5. It was shown that the proposed controllers using transfer learning are capable of mitigating the bulk of the grid violations in the REnnovates-Linear setup over a multitude of randomized simulations, confirming the applicability of sample-efficient RL controllers for low-voltage grid optimization.

In the first case study, the applicability of transfer learning in situations, where the control engineer has perfect knowledge of the physical distribution grid data, was shown. Through a precedent offline learning process a trained DQN network was obtained and subsequently used for the initialization of the "real" agent, facing the control task in the "real" environment. By sampling from a large dataset and ensuring the same load profiles are never assigned to the same position in the network, the available data was efficiently utilized in this training and verification process without compromising the validity of the obtained results.

A clear methodological reasoning was established by first elaborating the utilized procedures and then discussing the SARL and MARL results, both from a statistically significant point of view over multiple randomized simulations and more in-depth analysis of the trends observed in each of these runs. A benefit of working with an offline training procedure found in this way is the possibility for larger state-space exploration through implementation of an ϵ-decay scheme. Additionally, the greatly enhanced practical learning rate and the possibility to use any source of consumer data for the offline training processes - even from open source databases - are key advantages of the proposed methods.

All of the controllers developed in this work were subsequently benchmarked against each other in a statistically significant way for the REnnovates-Linear setup. It was found that the rule-based PV curtailment controllers and the offline trained SARL and MARL controllers using transfer learning entailed the most successfully policies towards resolving grid violations. When additionally taking into consideration the incurred battery losses, the rule-based PV curtailment controller showed to be the first-best option, not taking into consideration other than purely control based indicators.

Finally, in the second case study the agent was presented a control task in a distribution network for which the physical grid topology data was not available. Even though the topological differences were significant, the results with transfer learning were noticeably better. Therefore, we point out that this extreme case demonstrates that transfer learning is also useful when the grid topology differs in the offline and online learning process. Using a different grid with a similar topology in both learning processes would be more realistic in practical control settings and it is expected to result in even better performance enhancements.

Chapter 7

Conclusion

This thesis presents a sample-efficient RL based controller designed for demand response applications in a single and multi-agent setting. More specifically, a deep Q-learning algorithm in combination with transfer learning is used to overcome the data-inefficiency typically entailed by these data-driven methods. Battery and PV installations in residential buildings were used as energy flexibility resources. The main objective was to mitigate the grid impact of air source heat pumps and PV installations in net-zero energy buildings on the low-voltage grid. As a secondary target, battery cycling and PV curtailment losses are minimized. We used real world data from large-scale pilots in Belgium and The Netherlands, more specifically grid data from the Linear project and consumer profiles from the the REnnovates project. The performance of the SARL and MARL controllers is evaluated using rule-based controllers and an RL based controller which does not employ transfer learning.

An extensive data analysis showed that a considerable amount of overvoltages occurred in summer as a result of the PV installations. Despite the heating electrification, there were no undervoltages observed. It was found that the proposed RL based controller with transfer learning canceled 99% of the grid violations in the single agent case and 98% in the multi-agent case, which demonstrates that the grid impact of RES and HP can be mitigated adequately with the suggested controller.

Furthermore, we showed that transfer learning improved the performance of the controller significantly with respect to the baseline RL controller. This baseline RL controller, which does not employ transfer learning, performed inadequately after 1 year of training, which confirms the extremely high sample complexity of DQL and renders such controllers infeasible in practice. It was found that the transfer learning controller outperformed the baseline RL controller when the topology of the low-voltage grid was the same in the offline and online training process. Even when simulating an extreme scenario in which the grid topologies differed substantially, the transfer learning controller exceeded the performance of the baseline RL controller. The results demonstrate that transfer learning methods provide a solution for optimal control strategies in limited data domains and can be used to accelerate the learning process of RL based systems in real world problems.

Table 7.1: The SARL and MARL controllers using transfer learning outperform the different RL and rule-based controllers. The multi-agent configuration using transfer learning has a lower investment cost compared to SARL and is much more realistic, because only a few small batteries are needed to mitigate almost all violations.

Controller	Average violations resolved
Best rule-based battery	90%
Rule-based PV curtailment	100%
SARL	-10%
SARL + transfer learning	99%
MARL	80%
MARL + transfer learning	98%

In addition, the results of the benchmarking test of the rule-based controllers and the RL based controller with transfer learning exposed the shortcomings of rule-based algorithms. Man-made policies require a great deal of expertise and modeling is very expensive, because of the variability in residential energy systems. The proposed controller resolved more violations compared to the baseline controllers, therefore providing a viable alternative to these classic rule-based control methods.

This study has made innovative contributions to research governing independent learning multi-agent systems. Whereas the majority of prior work describing multi-agent problems included only 2 agents, our work presents a method with 3 agents. In addition, most previous research focused on the implementation of collaborative agents instead of employing a complete independent learning process. Moreover, the presented independent multi-agent control strategy improves robustness compared to the single agent controller or the rule-based controllers used in this study as a baseline.

However, it should be acknowledged that the found policies are sub-optimal with respect to the secondary objective of reducing energy losses. The contradictory objectives, mitigating violations and minimizing losses, result in a complex learning process. Fine tuning of the energy losses scaling factor in the reward function is a painstaking and sensitive procedure. Future research can build upon the findings presented here and use more refined methods, in which they can employ the policies learned in our SARL and MARL settings and push them towards even greater energy efficiencies. Furthermore, the effectiveness of more advanced algorithms, such as double Q-learning, on the performance and convergence rate of the proposed methods can be considered. We emphasize that the reward function used in our MDP can also be utilized from an economical point of view. The costs of grid violations for the DSO are not only related to technical grid issues caused by those grid violations, but also to the operational reliability and ensuring security of supply towards their customers. Interested parties can use our results as an input for a more thorough economic analysis.

While this study focused on RL based control systems in zero energy buildings using data from specific projects in the Netherlands and Belgium, the presented framework is generalizable to other control energy applications. It can be noted that many RL researches lack the availability of real-life data and employ custom datasets with completely fictitious information. Therefore, the matching of these projects could well represent a universal scenario.

To conclude, transfer learning has proved to be a valuable asset in this study. We believe that transfer learning is going to play an important role in future real-life reinforcement learning DR applications. While transfer learning research within this setting is still in its infancy, this study contributes to the first steps towards a more extensive foundation in this research domain.

Appendices

Appendix A

Further considerations

A.1 Selection power flow program: pandapower

Different energy and power system modeling tools are reviewed by [55]. We selected pandapower [56] based on its simplicity in modeling and parametrizing of electric components such as transformers, lines and switches. pandapower is an open source Python package, making use of data analysis library pandas and was built on PYPOWER in order to focus on the modelling of distribution networks. Different papers [57, 58] use pandapower for the simulation of smart power grids on low voltage level, e.g. for designing optimal energy management strategies.

A.2 Battery sizing issue for the rule-based controllers

An interesting observation can be made when comparing the worst-case and moderate battery sizing for both the centralized and decentralized battery controllers: the larger battery performs (slightly) worse than the smaller storage unit - a rather non-intuitive perception. Figure A.1 highlights these findings by looking at the battery controllers performances for a range of battery sizes. Two observations can be made: i) at a given $P_{b,max}$ performance improves for bigger $E_{b,max}$ and ii) at a given $E_{b,max}$ performance decreases for bigger $P_{b,max}$.

The explanation for this phenomena is given in figure A.2, where the rule-based operations of the grid battery over time are more closely examined. The major issue with this control strategy lies in its selection of battery power: it always charges or discharges at $P_b = P_{PV} - P_{load} - P_{hp}$. At moments of expected overvoltages, this effectively pulls down the voltage to an allowable magnitude below 1.1 p.u., but at higher $P_{b,max}$ this excessively "discharges" the network resulting in a lower voltage than is necessary. However, the SoC increases rapidly and the battery prematurely reaches $E_{b,max}$ at times where further charging is needed to prevent impending overvoltages. Additionally, when the battery discharges to get the SoC to lower values again, a higher discharge power (which is available but not used) in the evening periods could be beneficial, since now the aimed value of 50% is not reached before the next overvoltage occurs. The same issues apply to the rule-based house battery controller.

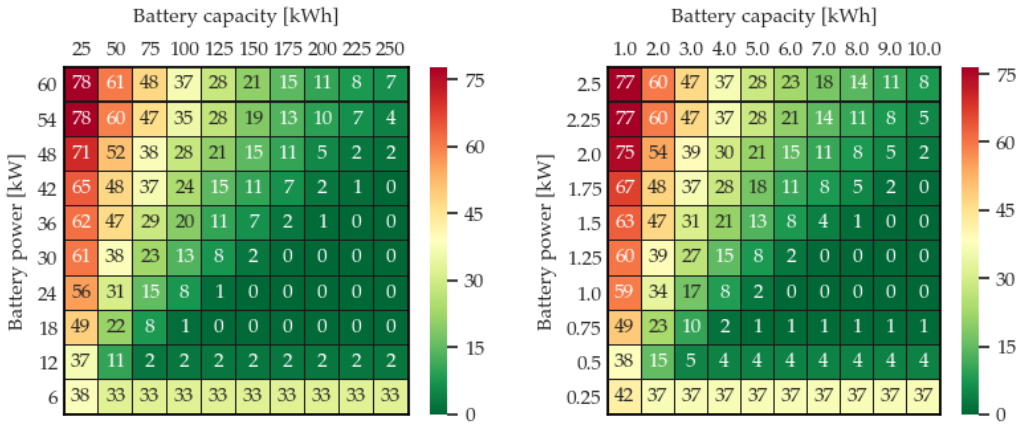

Figure A.1: Heat map for the rule-based grid battery controller (left) and house battery controller (right) showing the percentage of violations per year compared to the no-controllable resources scenario for different battery sizes. The analysis here is performed for one of the 100 house ID distributions which showed an around average number of violations without battery control.

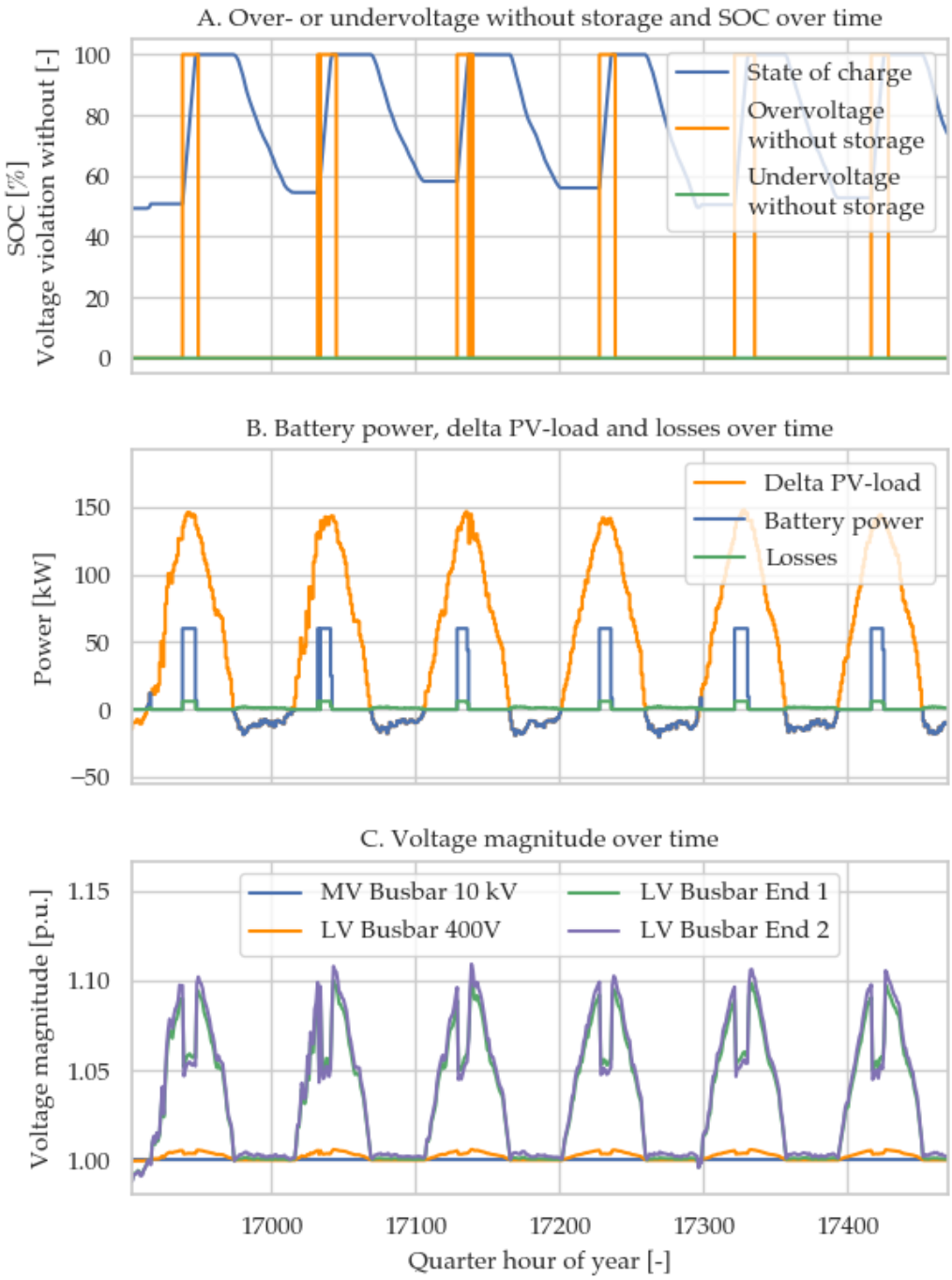

Figure A.2: A detailed view of the rule-based grid battery controller's operation in time. Utilized batter sizing $P_{b,max} = 60$ kW, $E_{b,max} = 250$ kWh. The 'violations without"parameter in subplot (A) indicates a boolean value (0: false, 100: true).

A.3 Clarifying the agent's states

A.3.1 Aggregate power balance

Figure A.3 indicates the clear linear relation between the instantaneous aggregate power generation $P_{agg} = \sum_{houses} (P_{PV} - P_{load} - P_{hp})$ and maximum grid voltage experimentally observed in the no-controllable resources scenario simulations. It is clear that a hysteresis is present with respect to overvoltages: below ±130 kW aggregate net power injection it is almost certain no overvoltage will occur; likewise $P_{agg} \geq \pm 133$ kW will most likely result in an overvoltage. Compared to the present order of magnitude this translates to a $(133 - 130)/130 = 2.31\%$ error margin in which it is unclear whether a given P_{agg} will lead to a voltage violation or not. Compared to the extra cost of refining this state, the aforementioned makes it more than justifiable to use the community level forecast.

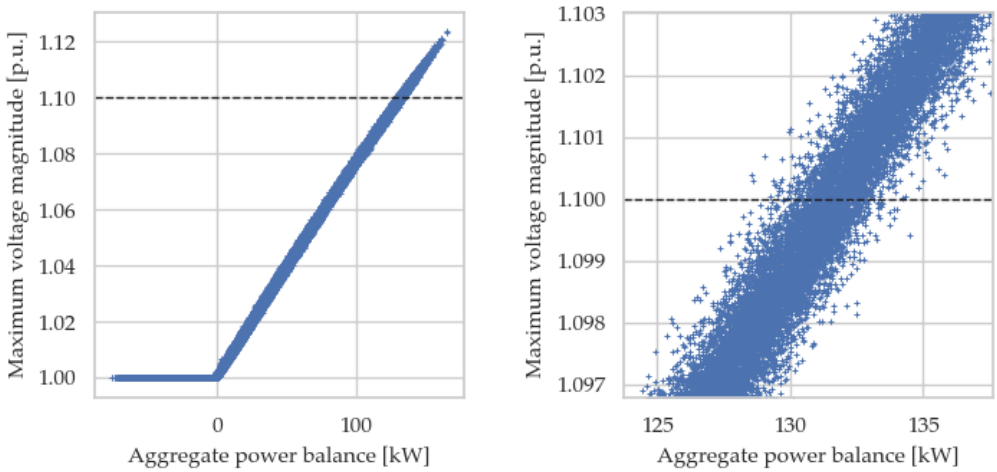

Figure A.3: Scatter plot of the aggregate power balance and maximum system voltage in the no-controllable resources scenario for all simulated quarter hours in the 100 randomized simulations.

A.3.2 Length of the forecast

To establish the optimal number of forecasted quarter hours to include in the agent's state, an experimental parameter analysis was performed. Here, we look at a completely myopic agent ($n_q = 1$) and more farsighted agents ($n_q = 8$, $n_q = 24$, and $n_q = 48$). Figure A.4 shows the results of this analysis for the SARL scenario, figure A.5 for the MARL case. In the former, it is found that more quarters lead to a reduction in battery losses, but a clear local optimum towards resolved violations is observed for the $n_q = 8$ case. Since it is the primary objective of the agent to solve grid violations, this translates into a maximum reward for the agent with the 2 hours ahead forecast. The results for the MARL scenario support these findings, also indicating an optimum at $n_q = 8$.

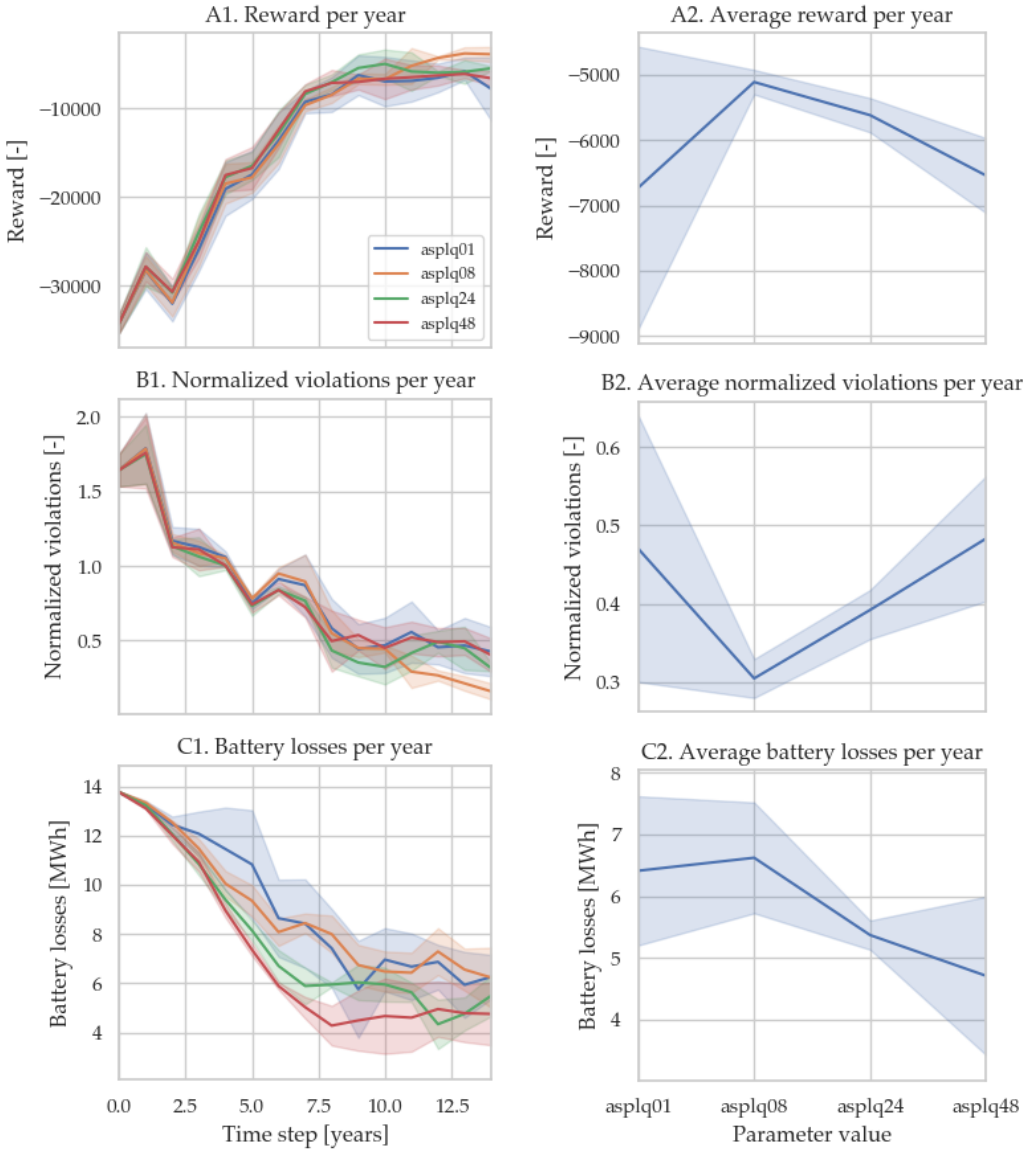

Figure A.4: Analysis of the optimal number of upcoming quarter hours to include in the SARL state. The right-hand side figure shows the indicated metrics averaged over the last five simulated episodes (see section 6.2 for the meaning of an episode). An epsilon-decay policy with $\epsilon = 1$ at the start and linear reduction to $\epsilon = 0$ over the first 10 episodes is used, afterwards epsilon is kept at zero. Each parameter is tested over five randomized simulations, with the same set of random seeds used for comparing the different parameter values. Violations are normalized with the no-controllable resources scenario.

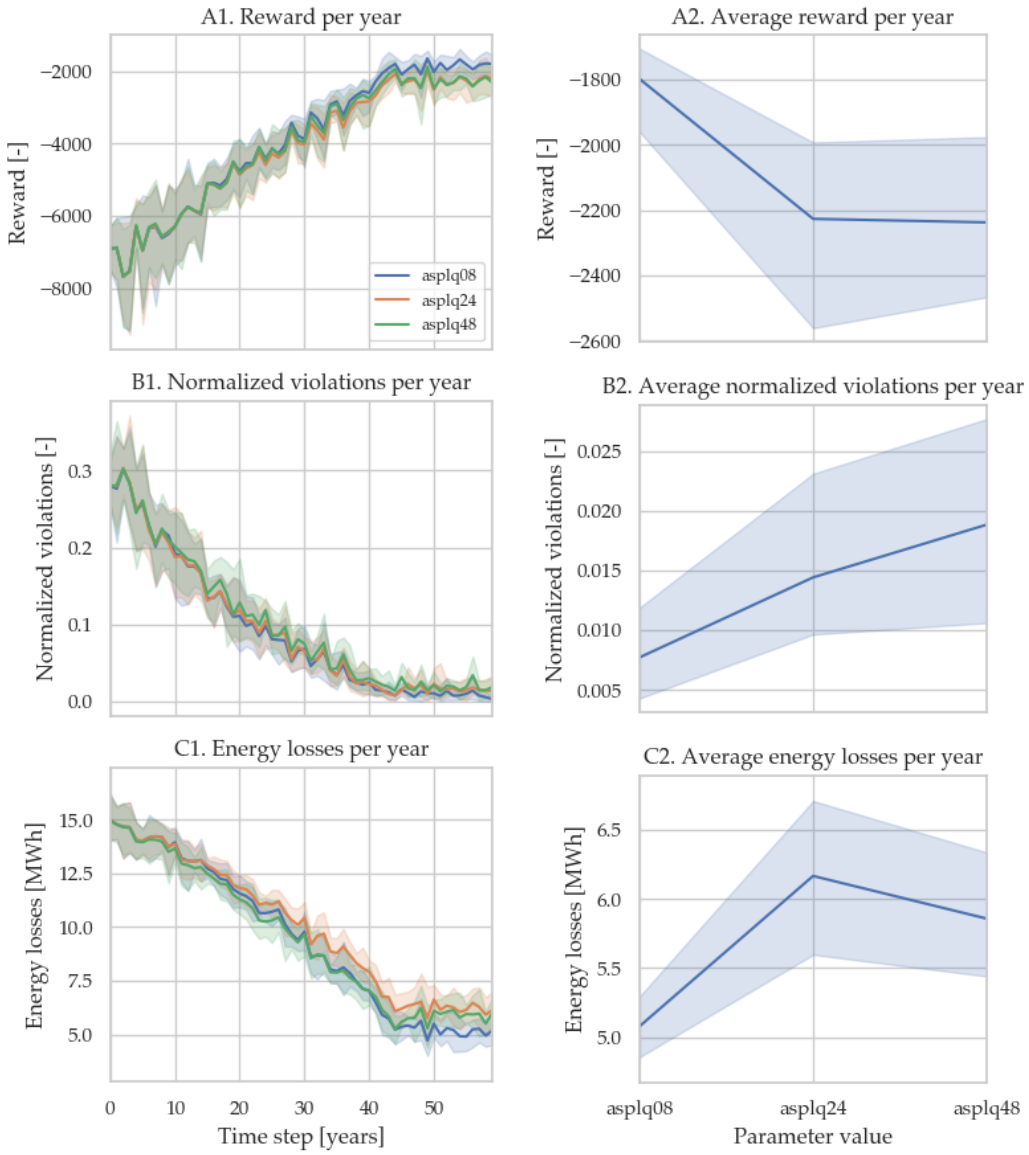

Figure A.5: Analysis of the optimal number of upcoming quarter hours to include in the MARL state. The same remarks as the caption for figure A.4 (SARL) are valid, only in this case 60 episodes were considered ($\epsilon = 0$ at episode 45). Due to the poor results in the SARL case, the completely myopic agent was not considered.

A.3.3 Time driven states

Three time driven states are used to describe the seasonal and daily trends in the data profiles as described in 5.1.3:

- **Quarter of day**: indicating the number of quarter hour in 1 day, so this state varies from 0-95 (96 quarter hours in one day). The initial quarter hour of day is set at 0.
- **Day of the week**: indicating the number of the day in the week, starting at 0 at the beginning of the simulation. This state varies from 0-6 (7 days in 1 week).
- **Season of year**: indicating the number of the season in one year, so varying from 0-3 (4 seasons in year), starting from 0.

Due to the design of these states, the simulations can start in every quarter hour, day or season. It is only important that the agent can learn the tendencies in time.

A.4 MARL battery sizing

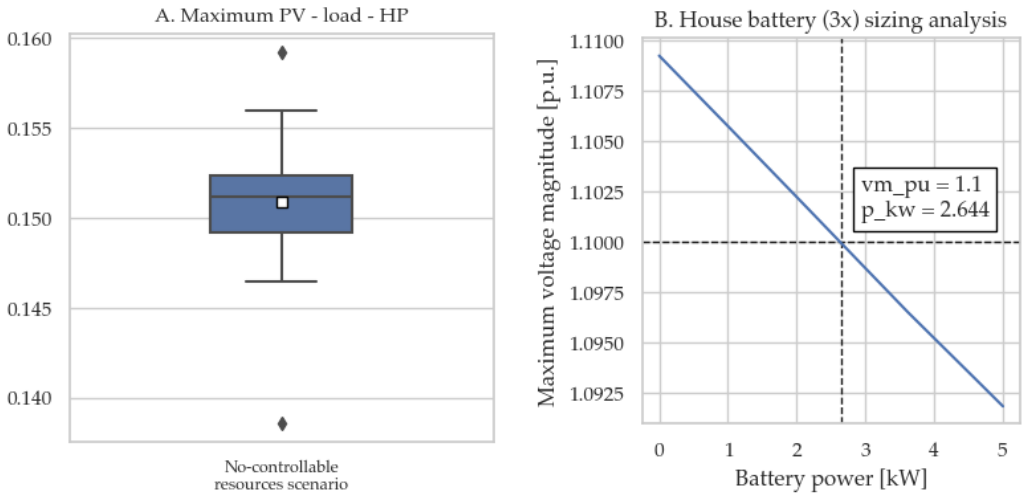

Figure A.6: The MARL battery sizing analysis is based on two considerations: the maximum instantaneous aggregate power balance in the 100 randomized no-controllable resources scenario simulations (A), and the iterative minimum battery power analysis with three houses placed at the end of the feeder (B).

The worst-case battery sizing analysis presented in section 4.3 gives an interesting approach to determine an upper bound for both $P_{b,max}$ and $E_{b,max}$. Nonetheless, it was found that these values are a strong exaggeration in comparison with the experimentally found suitable battery sizes. The bigger battery sizes even led to poorer rule-based performance, an issue discussed in section 4.4.2.

Since it is not in the scope of this work to analyse the performance of the RL controller over different ranges of battery sizes, solely a single, realistic combination of $P_{b,max}$ and $E_{b,max}$ is studied. For the SARL agent, this choice is rather obvious: we take the moderate battery sizing determined in the rule-based district storage unit scenario ($P_{b,max} = 30$ kW and $E_{b,max} = 125$ kWh). However, for the MARL case with three independent learners this design choice is less clear since both PV curtailment and battery control actions are possible. Therefore, we will not size the battery in the same way as for the SARL controller. The battery size is selected based on the following reasoning: the agent has to be able to resolve most of the overvoltages, but not all of them. In this way, we research if the agents would take only curtail actions in extreme cases when not all of the overvoltages can be canceled with battery actions.

This battery sizing for MARL is done in a similar way as in chapter 4, but instead of assuming a maximum production of all the PV installations (worst case) at the beginning of the battery analysis, we calculate the maximum aggregate power generation, since this is a less extreme upper boundary, $\max_t \left(\sum_{i=1}^{29} \left(P_{PV,i}^{(t)} - P_{load,i}^{(t)} - P_{hp,i}^{(t)} \right) \right)$. This is determined in each of the 100 performed no-controllable resources simulations. Figure A.6A shows an average value of $\pm$150 kW, or $150/29 = \pm 5$ kW per household. So, instead of using a maximum (worst-case) PV production of 7kW, we use the value of $P_{b,max}$=5kW in the battery analysis. We then perform the same iterative $P_{b,max}$ analysis as presented in section 4.3, but now only the last three houses on the feeder are assigned a storage unit. Figure 4.3B highlights the result, indicating an optimal battery sizing of approximately 2.5 kW. This value presents the needed battery size in order to be able to mitigate all the violations on the grid encountered in the 100 simulation. Overvoltages which cannot be canceled by the battery, should be canceled with curtailment.

The energy content is calculated in the same way as in section 4.3, following the reasoning that the battery has to charge at maximum power during 16 quarters to prevent an overvoltage, with 16 the maximum number of consecutive quarter hours observed in the data analysis with an overvoltage. A battery energy content of 10kWh is calculated using the battery power of 2.5kW for each battery.

A.5 Hyperparameters

A.5.1 Hyperparameters definitions

In chapter 5 the optimization of the hyperparameters was performed for the online RL controller. For each simulation 1 year of data was used in order to choose the right parameters in the same setting as for the actual controller. The following table shows the influence of different parameter values on the performance [29, 59].

Table A.1: Explanation of the hyperparameters.

Hyperparameters	Explanation
ANN structure (neurons per hidden layer)	A large amount of hidden layers can lead to overfitting and the inability to generalize data. Working with a small amount of hidden layers is maybe not enough to represent the complexity of the problem.
Adam optimizer learning rate	Each time the neural network is trained, the estimated error is calculated and the weights of the network are updated. The learning rate indicates how much the agent changes his weights according to the value of the estimated error. A small learning rate will lead to slow learning processes. A high learning rate will fasten up the learning but can lead to unstable processes where the agent forgets his learned policies from the past.
Discount factor	The discount factor, a value between 0 and 1 is used to discount the reward and calculate the cumulative discounted future reward. The discount factor determines the horizon of the agent. Using a large discount factors means that the agent will attach more importance to future rewards. On the other hand small discount factors represent immediate rewards.
Replay memory size	The amount of experiences $e_t = (s_t, a_t, r_t, s_{t+1})$ stored in the replay memory. If the memory size is small, the agent learns mostly (undesirable) temporal correlations. Bigger memories allow the agent to also learn from earlier experiences which speed up the learning.

Table A.1: Explanation of the hyperparameters (continued).

Hyperparameters	Explanation
Minimum replay memory start size	Only when the agent has reached enough experience, expressed as a minimum amount of experiences (minimum replay memory size) stored in the memory, the agent starts training. Starting with a small replay memory size has the same effects as explained in the definition of the replay memory size above. Starting with a too large memory means that the agent performs bad for a long time (because there is no training) or that the agent has already forgotten some experience (it is not anymore in the memory).
Minibatch size	At the beginning of each training process, samples from the replay memory are randomly drawn in order to use these former experiences to train the neural network. The amount of samples is called the minibatch size. The neural network is trained using a stochastic gradient method. For this method, larger batch sizes lead to a degradation in the generalization performance and optimization convergence, but smaller minibatch sizes lead to a less accurate estimate of the error gradient.
Target network update frequency	The frequency, expressed as the amount of time steps, at which the target network is updated. During the update, the weights of target are set equally to the weights of the main neural network used in the training process. Updating the target network very frequently, can lead to oscillations (the generated targets for the Q-values change frequently). Updating the target less frequently, leads to better convergence, but the value cannot be to low in order to update the future Q-values enough.

A.5.2 Hyperparameters for offline learning

The hyperparameters used in the simulations of the offline training process in chapter 6 are presented in table A.2. These parameters are obtained with a one dimensional grid search, similar to the online learning parameters. Instead of using only one year of data, we simulated 15 episodes in order to evaluate the values in function of a offline setting. The ϵ-value was set at 1 and decayed linearly to 0 in the 10th episode. In the last five episodes ϵ was kept constant at 0. Five seeds were used per parameter value in order to get a statistically significantly results.

Table A.2: Results of the hyperparameter optimization.

Hyperparameters	Optimal value
ANN structure (neurons per hidden layer)	(64,64)
Adam optimizer learning rate	0.001
Discount factor	0.99
Replay memory size	134400 quarters (200 weeks)
Minimum replay memory start size	672 quarters (1 week)
Minibatch size	64 experience samples
Target network update frequency	1 episode (1 year)

A.6 Scaling factors in the reward function

An important factor in the considered MDP is the design of the reward function. It is the agent's objective to maximize the discounted, cumulative reward over time. This effectively means the emerging control behaviour is steered by the definition of the reward signal. In our case, the four components (voltage magnitude, energy losses, line loading and transformer loading) described in section 5.1.5 constitute the reward function. The scaling factors in the different components should be interpreted as measures to set the relative importance of each component with respect to each other. In practice, an economical motivation could be used for this, but it is up to the DSO to decide the relative importance of each component with respect to costs and being able to ensure security of supply. Following reasoning is followed:

- Since the data analysis in chapter 3 revealed the dominant grid issue to be overvoltages, we take this parameter as the reference case. It is chosen to work with a scaling factor $\alpha_1 = 100000$, leading to $R_{overvoltage}(\max(U_i) = 1.1 \text{ p.u.}) = -10$. Since undervoltages are treated similarly to overvoltages, the same scaling factor is applied here: $\alpha_1 = \alpha_2$.
- Common sense tells us a single quarter hour with a line overloading or transformer overloading is more harmful for safe operations of the distribution network than a quarter hour with an over- or undervoltage. Thus, it is logic to penalize such violations more severely. Since in our reward reward function we always consider the highest loaded line or transformer in the system, it is

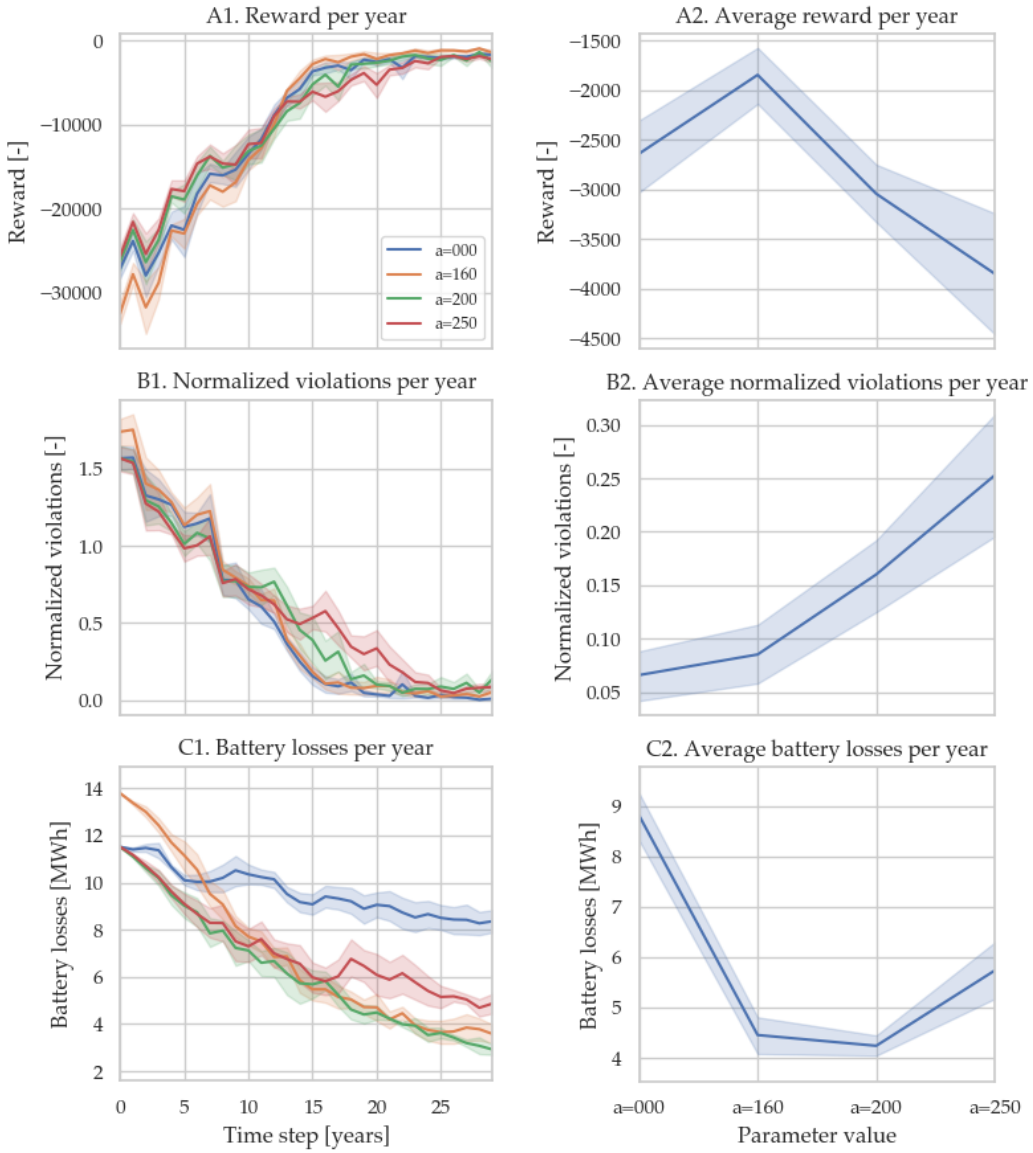

Figure A.7: Experimental analysis for the fine-tuning of the SARL scaling factor for losses in the reward function. At low α the agent focuses on solving violations, but the incurred losses are abundant. With increasing α the focus first shifts towards reducing battery losses, but the amount of resolved violations worsens. A local optimum of $\alpha = 160$ is found.

assumed that both situations are equally bad: $\alpha_3 = \alpha_4$. An arbitrary choice is made here: a line- or transformer loading of 100% is considered to be 2.5 times more disadvantageous as an overvoltage of exactly 1.1 p.u. From 5.4 it then follows that $\alpha_2 = \alpha_3 = 1$. It is difficult to verify the influence of this parameter on the MDP since the Linear grid is adequately designed for the REnnovates load profiles, meaning no overloading is observed in the simulations.

- The most crucial part in the reward design is the relative importance of energy losses. Since the source of these losses does not matter, i.e. a kWh battery losses is equally bad as a kWh PV curtailment, it logically follows that $\alpha_5 = \alpha_6$. The scaling should be done in such a way that solving voltage violations is the primary objective and that minimization of the losses remains a secondary target. A similar experimental approach is used as for the optimal number of quarters to include in the agent's forecast (see section A.3.2) and the offline hyperparameter optimization (see section A.5.2). The main difference is that now not 15, but 30 episodes were considered since the dynamics for the battery losses requires a long-term view. The results for the SARL case are indicated in figure A.7. For the MARL scenario, a similar approach was followed, leading to an optimal scaling factor of 400.

A.7 A closer look at the single-agent transfer learned policy

Figure A.8 shows the operations of the SARL agent during a week in summer. Two clear observations can be made:

- **SoC planning policy**: the agent has clearly learned a farsighted policy by proactively discharging its battery some hours before high PV generation is expected in the forecasts. This ensures that the SoC will be sufficiently low to start charging the battery at moments of impeding overvoltages.
- **Reduced overcharging**: compare the operations of our RL agent to that of the rule-based grid battery controller in figure A.2. The issues with the latter were highlighted in the accompanying section, reiterating the major disadvantage of the rule-based controllers: their inability to intelligently regulate the battery power. The approach of our RL agent creates a much smoother voltage profile (see subfigure 2 in both cases) and ensures the SoC of the battery does not prematurely reach 100%.

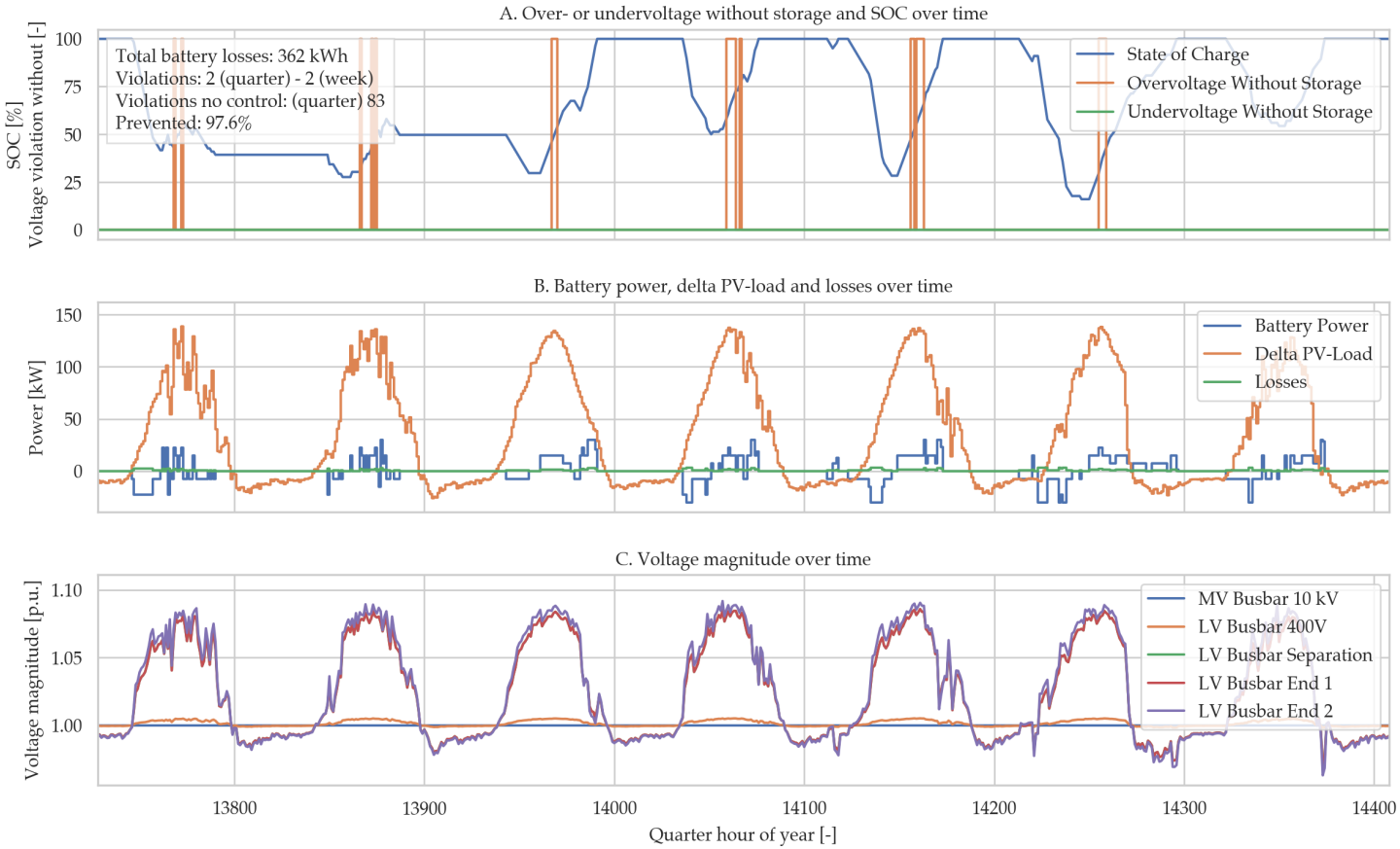

Figure A.8: A detailed view of the transfer learned SARL agent's operation in time. Utilized batter sizing $P_{b,max} = 30$ kW, $E_{b,max} = 125$ kWh. The 'violations without"parameter in subplot (A) indicates a boolean value (0: false, 100: true).

Bibliography

[1] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT press, second ed., 2018.

[2] A. Soares, D. Geysen, F. Spiessens, D. Ectors, O. De Somer, and K. Vanthournout, "Using reinforcement learning for maximizing residential self-consumption - Results from a field test," *Energy and Buildings*, 2019.

[3] B. Dean, J. Dulac, K. Petrichenko, and P. Graham, "Towards a zero-emission, efficient, and resilient buildings and construction sector," 2016. [Online]. Available: https://www.worldgbc.org/sites/default/files/UNEP 188_GABC_en (web).pdf. [Accessed: 2019-12-01].

[4] J. R. Vázquez-Canteli and Z. Nagy, "Reinforcement learning for demand response: A review of algorithms and modeling techniques," *Applied Energy*, vol. 235, no. October 2018, pp. 1072–1089, 2019.

[5] C. Protopapadaki and D. Saelens, "Heat pump and PV impact on residential low-voltage distribution grids as a function of building and district properties," *Applied Energy*, vol. 192, pp. 268–281, 2017.

[6] C. Finck, J. Clauß, P. Vogler-Finck, P. Beagon, K. Zhang, and H. Kazmi, "Review of applied and tested control possibilities for energy flexibility in buildings," tech. rep., 2018.

[7] H. Kazmi, J. Suykens, A. Balint, and J. Driesen, "Multi-agent reinforcement learning for modeling and control of thermostatically controlled loads," *Applied Energy*, vol. 238, no. June 2018, pp. 1022–1035, 2019.

[8] M. Ribeiro, K. Grolinger, H. F. ElYamany, W. A. Higashino, and M. A. Capretz, "Transfer learning with seasonal and trend adjustment for cross-building energy forecasting," *Energy and Buildings*, vol. 165, pp. 352–363, 2018.

[9] F. Qian, W. Gao, Y. Yang, and D. Yu, "Potential analysis of the transfer learning model in short and medium-term forecasting of building HVAC energy consumption," *Energy*, vol. 193, p. 116724, 2020.

[10] Q. Hu, R. Zhang, and Y. Zhou, "Transfer learning for short-term wind speed prediction with deep neural networks," *Renewable Energy*, vol. 85, pp. 83–95, 2016.

[11] "i.LECO," 2019. [Online]. Available: https://ileco.energy/. [Accessed: 2019-12-01].

[12] "REnnovates," 2019. [Online]. Available: https://rennovates.eu/. [Accessed: 2019-10-01].

[13] IEA, "World Energy Outlook 2018," 2018. [Online]. Available: https://www.iea.org/reports/world-energy-outlook-2018. [Accessed: 2019-12-01].

[14] CEER, "Electricity - voltage quality," *6th CEER Benchmarking Report on the Quality of Electricity and Gas Supply - 2016*, pp. 80–137, 2016.

[15] VREG, "Energy market," 2019. [Online]. Available: https://www.vreg.be/en/energy-market. [Accessed: 2019-12-11].

[16] R. Belmans, G. Deconinck, and J. Driesen, *Elektrische Energie Deel 2*. ACCO, 2011.

[17] Y. Liu, W. Qin, X. Han, P. Wang, Y. Wang, L. Wang, and F. Li, "Distribution network voltage control by active power/reactive power injection from PV inverters," 2018.

[18] T. Xu and P. Taylor, "Voltage Control Techniques for Electrical Distribution Networks Including Distributed Generation," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 11967–11971, 2008.

[19] A. P. Kenneth and K. Folly, "Voltage rise issue with high penetration of grid connected PV," in *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol. 19, pp. 4959–4966, IFAC, 2014.

[20] European Commission (EC), "Third energy package," 2019. [Online]. Available: https://ec.europa.eu/energy/en/topics/markets-and-consumers/market-legislation/third-energy-package. [Accessed: 2019-12-13].

[21] European Commission (EC), "Energy Efficiency Directive," 2019. [Online]. Available: https://ec.europa.eu/energy/en/topics/energy-efficiency/targets-directive-and-rules/energy-efficiency-directive. [Accessed: 2019-12-13].

[22] A. Afram and F. Janabi-Sharifi, "Review of modeling methods for HVAC systems," *Applied Thermal Engineering*, vol. 67, no. 1-2, pp. 507–519, 2014.

[23] K. J. Hunt, D. Sbarbaro, R. Zbikowski, and P. J. Gawthrop, "Neural networks for control systems-A survey," *Automatica*, vol. 28, no. 6, pp. 1083–1112, 1992.

[24] P. Kofinas, A. I. Dounis, and G. A. Vouros, "Fuzzy Q-Learning for multi-agent decentralized energy management in microgrids," *Applied Energy*, vol. 219, no. March, pp. 53–67, 2018.

[25] L. Torrey and J. Shavlik, "Handbook of Research on Machine Learning Applications," pp. 242–264, IGI Global, 2009.

[26] H. Kazmi and J. A. K. Suykens, "Large-scale transfer learning for data-driven modelling of hot water systems," no. August, 2019.

[27] M. R. Tousi, S. H. Hosseinian, and M. B. Menhaj, "A Multi-agent-based voltage control in power systems using distributed reinforcement learning," *Simulation*, vol. 87, no. 7, pp. 581–599, 2011.

[28] Zou, Jinming, Y. Han, and S.-S. So, "Overview of artificial neural networks," in *Artificial Neural Networks: Methods and Applications* (D. J. Livingstone, ed.), pp. 14–22, Sandown: Humana Press, 2008.

[29] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[30] G. Shi, D. Liu, and Q. Wei, "Echo state network-based Q-learning method for optimal battery control of offices combined with renewable energy," *IET Control Theory and Applications*, vol. 11, no. 7, pp. 915–922, 2017.

[31] B. Jiang and Y. Fei, "Dynamic residential demand response and distributed generation management in smart microgrid with hierarchical agents," in *Energy Procedia*, vol. 12, pp. 76–90, 2011.

[32] D. Lee and W. B. Powell, "An intelligent battery controller using bias-corrected Q-learning," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 1, pp. 316–322, 2012.

[33] D. Li and S. K. Jayaweera, "Reinforcement learning aided smart-home decision-making in an interactive smart grid," in *2014 IEEE Green Energy and Systems Conference, IGESC 2014*, pp. 1–6, IEEE, 2015.

[34] Y. Wang, X. Lin, and M. Pedram, "A near-optimal model-based control algorithm for households equipped with residential photovoltaic power generation and energy storage systems," *IEEE Transactions on Sustainable Energy*, vol. 7, no. 1, pp. 77–86, 2016.

[35] A. Sheikhi, M. Rayati, and A. M. Ranjbar, "Demand side management for a residential customer in multi-energy systems," *Sustainable Cities and Society*, vol. 22, pp. 63–77, 2016.

[36] L. Raju, S. Sankar, and R. S. Milton, "Distributed optimization of solar microgrid using multi agent reinforcement learning," in *Procedia Computer Science*, vol. 46, pp. 231–239, Elsevier Masson SAS, 2015.

[37] S. Sekizaki, T. Hayashida, and I. Nishizaki, "An intelligent Home Energy Management System with classifier system," in *2015 IEEE 8th International Workshop on Computational Intelligence and Applications, IWCIA 2015 - Proceedings*, pp. 9–14, IEEE, 2016.

[38] B. V. Mbuwir, F. Ruelens, F. Spiessens, and G. Deconinck, "Battery energy management in a microgrid using batch reinforcement learning," *Energies*, vol. 10, no. 11, pp. 1–19, 2017.

[39] T. Navidi, "Coordination of Distributed Energy Resources without Power Grid Models using Reinforcement Learning," tech. rep., Standford University, 2018.

[40] X. Qiu, T. A. Nguyen, and M. L. Crow, "Heterogeneous Energy Storage Optimization for Microgrids," *IEEE Transactions on Smart Grid*, vol. 7, no. 3, pp. 1453–1461, 2016.

[41] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 2, no. 10, pp. 1345–1359, 2010.

[42] "Linear," 2019. [Online]. Available: http://www.linear-smartgrid.be/. [Accessed: 2019-10-08].

[43] "Keras: The Python Deep Learning library. [accessed: 2019-10-08]," 2019. [Online]. Available: https://keras.io/.

[44] "Tensorflow," 2019. [Online]. Available: https://www.tensorflow.org/. [Accessed: 2019-10-08].

[45] L. Thurner, A. Scheidler, F. Schafer, J. H. Menke, J. Dollichon, F. Meier, S. Meinecke, and M. Braun, "Pandapower - An Open-Source Python Tool for Convenient Modeling, Analysis, and Optimization of Electric Power Systems," *IEEE Transactions on Power Systems*, vol. 33, pp. 6510–6521, nov 2018.

[46] W. McKinney, "Data Structures for Statistical Computing in Python," *Proceedings of the 9th Python in Science Conference*, vol. 445, pp. 51–56, 2010.

[47] "PYPOWER 5.1.4," 2019. [Online]. Available: https://pypi.org/project/PYPOWER/. [Accessed: 2019-10-08].

[48] R. Sevlian and R. Rajagopal, "A scaling law for short term load forecasting on varying levels of aggregation," *International Journal of Electrical Power and Energy Systems*, vol. 98, no. August 2017, pp. 350–361, 2018.

[49] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," in *6th International Conference on Learning Representations, ICLR 2018 - Workshop Track Proceedings*, pp. 1–13, 2018.

[50] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," in *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pp. 3207–3214, 2018.

[51] B. Q. Huang, G. Y. Cao, and M. Guo, "Reinforcement learning neural network to the problem of autonomous mobile robot obstacle avoidance," *2005 International Conference on Machine Learning and Cybernetics, ICMLC 2005*, no. August, pp. 85–89, 2005.

[52] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Networks*, vol. 107, pp. 3–11, 2018.

[53] P. Covington, J. Adams, and E. Sargin, *Deep neural networks for youtube recommendations*. New York: Association for Computing Machinery, 2016.

[54] R. Liessner, J. Schmitt, A. Dietermann, and B. Bäker, "Hyperparameter optimization for deep reinforcement learning in vehicle energy management," in *ICAART 2019 - Proceedings of the 11th International Conference on Agents and Artificial Intelligence*, vol. 2, pp. 134–144, 2019.

[55] M. Groissböck, "Are open source energy system optimization tools mature enough for serious use?," 2019.

[56] Energy Management and Power System Operation. University of Kassel and the Department for Distribution System Operation at the Fraunhofer Institute for Energy Economics and Energy System Technology (IEE), "pandapower," 2019. [Online]. Available: https://www.pandapower.org/. [Accessed: 2019-12-01].

[57] J. H. Menke, N. Bornhorst, and M. Braun, "Distribution system monitoring for smart power grids with distributed generation using artificial neural networks," *International Journal of Electrical Power and Energy Systems*, vol. 113, no. July 2018, pp. 472–480, 2019.

[58] H. Hua, Y. Qin, C. Hao, and J. Cao, "Optimal energy management strategies for energy Internet via deep reinforcement learning approach," *Applied Energy*, vol. 239, no. June 2018, pp. 598–609, 2019.

[59] E. Hoffer, I. Hubara, and D. Soudry, "Train longer, generalize better: closing the generalization gap in large batch training of neural networks," in *Advances in Neural Information Processing Systems 30*, pp. 1731–1741, Curran Associates, Inc., 2017.