Deep Reinforcement Learning for Optimal Building Energy System Operation

Deep Reinforcement Learning für den optimalen Betrieb von Gebäudeenergiesystemen

Von der Fakultät für Maschinenwesen der Rheinisch-Westfälischen Technischen Hochschule Aachen zur Erlangung des akademischen Grades eines Doktors der Ingenieurwissenschaften genehmigte Dissertation

vorgelegt von

Thomas Bruno Schreiber

Berichter: Univ.- Prof. Dr.-Ing. Dirk Müller
Univ.- Prof. Dr.-Ing. Veit Hagenmeyer

Tag der mündlichen Prüfung: 24. Mai 2024

Diese Dissertation ist auf den Internetseiten der Universitätsbibliothek online verfügbar.

Vorwort

Die vorliegende Dissertationsschrift entstand im Rahmen meiner Tätigkeit am Lehrstuhl für Gebäude- und Raumklimatechnik des E.ON Energieforschungszentrums der RWTH Aachen University. Während meiner Arbeit in öffentlich geförderten Forschungsprojekten wurde ich wiederkehrend vor die Herausforderung gestellt, den Betrieb von Gebäudeenergiesystemen unter dynamischen Randbedingungen zu optimieren. Gleichzeitig zeichnete sich in der Literatur der Trend ab, dass die KI-Algorithmenfamilie Deep Reinforcement Learning immer vielversprechender für die Anwendung im Bereich der Betriebsoptimierung technischer Systeme wurde. Diese Entwicklung ergab sich im Wesentlichen getrieben durch einige beeindruckende Erfolge in der Grundlagenforschung. So entstand die Idee für diese Dissertation.

Ich möchte mich ganz herzlich bei meinem Doktorvater, Univ.- Prof. Dr.-Ing. Dirk Müller, für die Freiheit dieses Dissertationsthema voranzutreiben sowie für die produktive und gehaltvolle Forschungs- und Arbeitsatmosphäre am Lehrstuhl bedanken. Mein besonderer Dank gilt außerdem Univ.- Prof. Dr.-Ing. Veit Hagenmeyer für die angeregten Diskussionen und die wertvollen Einblicke im Rahmen des Promotionsprozesses sowie Univ.- Prof. Dr.-Ing. Andrea Benigni für den Prüfungsvorsitz im Rahmen der mündlichen Prüfung.

Des Weiteren möchte ich mich bei meinen Kolleginnen und Kollegen bedanken, die mich bei meiner Arbeit immer wieder unterstützt haben, mit mir über Ideen und Forschungsfragen diskutiert haben und die Zeit am Institut auch über die reine Arbeit hinaus zu einem unvergesslichen Lebensabschnitt gemacht haben. Besonders Marc Baranski, Phillip Stoffel, Laura Maier, meinen lieben Kolleginnen und Kollegen im Kreis der Teamleiter und meinen Teammitgliedern im Team Urbane Energiesysteme möchte ich von Herzen danken. Ein weiterer Dank ist meinen vielen studentischen Abschlussarbeitern gewidmet, die mich durch ihre exzellente Forschungsarbeit vielseitig unterstützt haben. Ein besonderer Dank gilt hier Sören Eschweiler, Aron Schwartz, Christoph Netsch und Vincent Evenschor, unter deren Betreuung die Journal- und Konferenzartikel entstanden sind, die in das Anwendungskapitel dieser Dissertationsschrift eingeflossen sind.

Ein ganz persönlicher Dank gilt meiner Ehefrau Carla Schreiber, die meine Entscheidung, noch einmal fünf Jahre in Aachen zu arbeiten, mitgetragen und mir die Freiräume geschenkt hat, mich auch neben der Arbeit mit meiner Dissertation zu beschäftigen. Meinen beiden Kindern Carlotta und Kasimir möchte ich für die Stunden danken, die sie geduldig auf mich verzichtet haben und für einen ganz neuen Blick auf Leben und Arbeit, den ich ohne sie wohl nicht erlangt hätte. Schließlich möchte ich meiner Familie und meinen Freunden für die vielen

unterstützenden Worte und spannenden Diskussionen über meine Forschungsarbeit danken. Auch da sie mir ein Umfeld geschaffen haben, in dem ich bis zu diesem Punkt wachsen konnte.

Aachen, Mai 2024

Thomas Schreiber

Abstract

International climate protection targets make the development of scalable methods for increasing the efficiency and sustainability of energy systems a necessity. At the same time, the digital revolution continues to advance and algorithms from the field of artificial intelligence are finding their way into more and more areas of life. Deep Reinforcement Learning (DRL) is a class of algorithms designed to solve sequential decision-making problems using neural networks, which has been successfully demonstrated for a variety of problem classes in recent years. Motivated by this, the applicability for optimization problems in building engineering is investigated in this dissertation. Relevant research questions are identified and the advantages and disadvantages compared to the popular Model Predictive Control approach are discussed. Based on the introduction, a workflow for the development of DRL-supported building automation is presented. The workflow chapter concludes with the presentation of a collaborative work in which a DRL algorithm is compared and discussed with other novel control methods. The application scenarios in the application chapter are selected from current research projects on the operation optimization of building energy systems. The proposed workflow is demonstrated, addressing questions of algorithm comparisons, practicable training times, use of monitoring data for training, selection of hyper-parameters, data infrastructures, and applicability of simulation-based trained algorithms to real control problems. In the subsequent discussion chapter, the opportunities and challenges of the algorithms and the experiences gained in the course of the work are discussed in detail. The results of the work can be summarized as follows. DRL algorithms have promising properties and can compete with other novel approaches in terms of performance, but the technical effort required for implementation should not be underestimated. The advantages lie in the inherent adaptability to variable environmental conditions, the low computational costs after training, and the potential to process large stochastic problems with high performance. The disadvantages lie in the difficult interpretability, the data-intensive training, and the unavoidability of stochastic actions. Nevertheless, great potential is seen in publishing pre-trained algorithms for recurring optimization tasks and using them as expert systems in product development. In particular, since development, implementation, and training are one-time processes, the effort is worthwhile. Ultimately, the result can be a self-optimizing software module that continues to improve along with the technical system in its operational environment.

Kurzfassung

Internationale Klimaschutzziele machen die Entwicklung skalierbarer Methoden notwendig, um Effizienz und Nachhaltigkeit von Energiesystemen zu erhöhen. Gleichzeitig schreitet die Digitale Revolution weiter voran und Algorithmen aus dem Bereich der Künstlichen Intelligenz halten Einzug in immer mehr Lebensbereichen. Deep Reinforcement Learning (DRL) ist eine Klasse von Algorithmen für die Lösung sequenzieller Entscheidungsprobleme unter Verwendung Neuronaler Netze, welche in den vergangenen Jahren für vielfältige Problemklassen erfolgreich demonstriert wurde. Hierdurch motiviert, wird in dieser Arbeit die Anwendbarkeit für Optimierungsprobleme aus der Gebäudetechnik untersucht. Es werden relevante Fragestellungen herausgearbeitet und die Vor- und Nachteile gegenüber der populären Modellprädiktiven Regelung diskutiert. Auf Basis der Einführung wird ein Workflow für die Entwicklung DRL unterstützter Gebäudeautomation vorgestellt. Das Workflow-Kapitel schließt mit der Vorstellung einer kollaborativen Arbeit ab, in der ein DRL-Algorithmus gegen weitere neuartige Regelungsverfahren verglichen und diskutiert wird. Die Anwendungsszenarien des Anwendungskapitels wurden aus aktuellen Forschungsprojekten zur Betriebsoptimierung von Gebäudeenergiesystemen ausgewählt. Der vorgeschlagene Workflow wird demonstriert; hierbei werden Fragen zu Algorithmusvergleichen, praktikablen Trainingszeiten, Nutzung von Monitoringdaten beim Training, Auswahl von Hyper-Parametern, Dateninfrastrukturen und Anwendbarkeit simulationsbasiert trainierter Algorithmen auf reale Regelungsprobleme, behandelt. Im anschließenden Diskussionskapitel werden die Chancen und Herausforderungen der Algorithmen sowie die im Zuge der Arbeit gemachten Erfahrungen detailliert diskutiert. Die Ergebnisse der Arbeit lassen sich wie folgt zusammenfassen. DRL-Algorithmen besitzen vielversprechende Eigenschaften und können in ihrer Leistungsfähigkeit mit konkurrierenden Verfahren mithalten, allerdings ist der technische Aufwand für die Implementierung nicht zu unterschätzen. Die Vorteile liegen in der inhärenten Anpassungsfähigkeit an variable Umweltbedingungen, dem geringen Rechenaufwand nach dem Training und dem Potenzial, große stochastische Probleme performant zu verarbeiten. Die Nachteile liegen in der schwierigen Interpretierbarkeit, dem datenintensiven Training und der Unvermeidbarkeit von stochastischen Aktionen. Dennoch wird ein großes Potenzial darin gesehen, für wiederkehrende Optimierungsaufgaben vortrainierte Algorithmen zu veröffentlichen und in der Produktentwicklung als Expertensysteme einzusetzen. Insbesondere, da Entwicklung, Implementierung und Training dann einmalige Prozesse sind, lohnt sich der Aufwand. Schließlich kann das Ergebnis ein sich selbst optimierendes Softwaremodul sein, das sich zusammen mit dem technischen System in seiner Einsatzumgebung immer weiter verbessert.

Contents

Nomenclature

Formula symbols and units

Symbol	Meaning	Unit
a	Action	-
A	Action-space	-
B	Batch size	-
C	Operation costs	€
c_p	Specific heat capacity at constant pressure	J/kg K
d	Diameter	m
D	Replay buffer size	-
EER	Energy Efficiency Ratio	-
J	Online-savings (control performance)	€
k_{el}	Electricity price	€/kWh
k_{3WV}	Total valve position (three-way-valves)	%
K_{el}	Electricity costs	€
L	Loss	-
$\dot{m}$	Mass flow	kg/s
m	Mass	kg
n	Training episodes	-
P	Power	W
Pr	Transition probabilities	-
$\dot{Q}$	Heat flow	W
Q	State-action-value	-
r	Reward	-
s	State	-
S	State-space	-
SOC	State of charge	%
T	Temperature	K
W_{el}	Electrical work	kWh
X	Relative valve opening or closing	%

Greek symbols

Symbol	Meaning	Unit
α	Learning rate	—
β	Probability of interference	—
γ	Discount factor	—
ε	Exploration rate	—
η	Efficiency	—
θ	Trainable parameters	—
π	Policy	—
ρ	Polyak update	—

Indices and abbreviations

Symbol	Meaning
abs	Absolute
AHU	Air Handling Unit
amb	Ambient
ANN	Artificial Neural Network
API	Application Programming Interface
BACS	Building Automation and Control System
BEMS	Building Energy Management System
BES	Building Energy System
CNN	Convolutional Neural Network
DRL	Deep Reinforcement Learning
DQN	Deep Q-Network
DDPG	Deep Deterministic Policy Gradient
EER	Energy Efficiency Ratio
el	Electric
eps	Episode
FMU	Functional Mock-up Unit
gl	Glycol
HEMS	Home Energy Management System
HP	Hyper-parameter
HTTPS	HyperText Transfer Protocol Secure
HVAC	Heating, Ventilation, and Air Conditioning
IoT	Internet of Things
IT	Information technology
LSTM	Long Short-Term Memory
LR	Linear Regression
MDP	Markov Decision Process
MPC	Model Predictive Control
MQTT	Message Queuing Telemetry Transport
net	Network
OM	Operation mode
pen	Penalty
PID	Proportional–Integral–Derivative
PPO	Proximal Policy Optimization

Continues on the next page

Indices and abbreviations

Symbol	Meaning
PC	Personal Computer
PS	Pre-training Strategy
PV	Photovoltaic
RBC	Rule-based control
RF	Random Forrest
ReLU	Rectified Linear Unit
REST	Representational State Transfer
ref	Reference
RL	Reinforcement Learning
RMSE	Root Mean Square Error
s	Site
SAC	Soft-Actor-Critic
SVR	Support Vector Regression
t	Total
w	Water

List of figures

List of tables

1 Introduction

In this chapter, the presented dissertation is motivated. For this purpose, the need for research is discussed first. This is followed by a discussion of one of the most promising methods in the field of optimal control for building energy systems, along with its advantages and disadvantages. It is described how data-driven methods are changing the research field, and finally an introduction is given regarding the algorithms that are investigated, discussed, and validated via experiments in the course of this work.

1.1 Motivation and background

We are living in times of great transitions. Climate change is putting pressure on governments around the globe to set ambitious targets for reducing greenhouse gas emissions [IPCC, 2018] and current geopolitical instabilities make the necessitate reducing the share of fossil gas from leading energy sources. The optimization of building energy systems (BES) is one key challenge as buildings account for 36 % of the energy use and almost 40 % of the CO_2 emissions globally [IEA - International energy Agency, 2018] and for around 40 % of the energy use and around 33 % of the CO_2 emissions in Germany [Deutsche Energie-Agentur, 2018]. In order to meet the political targets regarding the reduction of greenhouse gases, ambitious measures are necessary [IEA - International energy Agency, 2013].

Besides the modernization of the building envelope, the modernization of the technical equipment, and the influence of the user behavior, a high potential lies in an optimized energy system operation. Optimized operation strategies aim to optimally use the local BES under dynamic boundary conditions, such as weather forecasts, internal thermal loads of the buildings, and dynamic energy prices. Also, contemporary challenges can only be addressed by increased interconnection of the energy sectors electricity, heating, and cooling [Wietschel et al., 2018]. Demand response management is a key concept here that has been present in the literature for decades [Kilkis, 1999]. The reason is that it enables reduction of CO_2 emissions, while reducing operation costs and increasing electrical grid stability [Leibowicz et al., 2018; Palensky and Dietrich, 2011]. In particular, the optimal use of energy storage systems with respect to dynamic boundary conditions can increase the flexibility of local energy systems [Santos et al., 2017]. Driven by this prospect, the requirements for automation infrastructure are increasing [Han and Lim, 2010] due to the more complex processing of internal and external influencing signals for energy management applications - a trend that will continue

in the coming years [Shah et al., 2019]. Therefore, the implementation of robust, predictive, and self-adapting BES control algorithms that increase optimization capabilities, based on the available infrastructure, is one of the long-term goals of scientific work in this field.

Through operation optimization of BES, savings between 20 % and 50 % are usually published in the scientific literature [Afram and Janabi-Sharifi, 2014; Álvarez et al., 2013; Baranski et al., 2018; Deng et al., 2015; Gruber et al., 2014; Kolokotsa et al., 2009; Oldewurtel et al., 2012; Wang and Ma, 2008]. As a benchmark, a rule-based control (RBC) mode is usually used, which corresponds to the state-of-the-art in BES [Fütterer et al., 2017]. Here, a set of hard-coded rules is used to react to current sensor signals such as the outside temperature or the heating and/or cooling demands of the building [Bode et al., 2018]. RBC as the current state-of-the-art even in modern BES is generally neither predictive nor adaptive due to its limitation to pre-defined thresholds and static integration only of current sensor signals. RBC is therefore unsuitable for dealing with the increasing complexity in BES. The promising results from the literature are achieved via optimized and predictive operation, based on forecasts of the influencing boundary conditions and from automatic adaptation of the control logic to changing conditions.

Despite the high potential, outside of the scientific research, operation optimization of BES does not play a major role in the BES industry. However, a few companies have already entered this market segment in Germany and also promise energy saving potentials between 20 % and 44 % to their customers [*Kieback&Peter*, 2020; *MeteoViva GmbH*, 2020; *Recogizer Group GmbH*, 2020]. To determine the reasons for the low dissemination of structured operation optimization and the use of optimization methods in practice, the Institute for Energy Efficient Buildings and Indoor Climate in Aachen, Germany conducted a survey among BES planners, operators, and system integrators [Schild et al., 2019]. The survey showed that while respondents see great potential in operation optimization, there are many barriers to its widespread application in practice. One of the reasons for this is that the constructor of the building is seldom the operator, and the energetic efficiency of the later operation plays a subordinate role during the planning and construction process. While the goal during these early phases is to save investment costs, the later operator would benefit from higher investments in BES and the implementation of optimal control algorithms, through lower operation costs. This makes optimization methods particularly interesting, which can be implemented after completion of the construction phase, without high engineering efforts. Subsequently, these optimization methods adapt to and improve existing BES autonomously. However, the most significant reasons are lack of knowledge, unwillingness to invest (due to relative low energy prices compared to engineering services), and also low available computing power in local automation systems. Especially the latter is a particular obstacle for optimization-based methods.

In the following, Model Predictive Control (MPC), which is an optimization-based control

approach widely used in scientific research, is presented and discussed, focusing on how data-driven methods have changed the methodology in the literature in recent years. According to the literature, a particularly high potential lies especially in the combination of data-driven and optimization-based approaches to overcome the obstacles from practice [Afram et al., 2017].

1.2 Model Predictive Control and data-driven methods

After introducing the motivation for BES operation optimization and the difficulties in applying the methods in practice in the last section, this section presents one of the most promising methods from the literature. It also discusses how data-driven methods have shaped and changed the research field in recent years.

Some of the most promising results in operation optimization of BES are achieved by applying MPC to the control problems [Lee et al., 2009; Lu et al., 2015; Ooka and Ikeda, 2015; Sanaye and Shirazi, 2013]. As illustrated in figure 1.1, MPC is based on a model of the system to be controlled, which is combined with actual sensor measurements to represent the current state of the system. Based on the current state and the forecasts of the influencing environment conditions, the model is used to calculate the optimal sequence of control signals for a defined number of time-steps in the future. Algorithms to convert state and forecasts into optimal actions usually come from the field of mathematical optimization and aim to obtain the optimal sequence of actions for the forecast horizon with as few iterations as possible. In most optimization studies, the aim is to achieve a balance between energy consumption and user comfort, but there is also a number of studies where different optimization objectives, like demand response control, are investigated. A comprehensive summary of MPC for BES, including implementation guidelines, is provided to the literature in [Drgoňa et al., 2020].

In [Deng et al., 2015], the authors formulated a MPC problem for the optimal control of a cooling supply system with a chiller bank and a thermal storage, supplying a large campus site at the University of California. Aside from the uncertainties in the predictions, they achieved nearly optimal operation. In [Kolokotsa et al., 2009], a real-world application of MPC integrated into a BES was demonstrated. The controller obtained near-optimal setpoints by balancing energy consumption and indoor air quality. In addition to the formulation of the optimization problem, the handling strategy of uncertainties in the model and forecasts is of decisive importance. The authors of [Oldewurtel et al., 2012] integrated weather forecasts successfully into their MPC controller by integrating probabilistic constraints into the problem formulation. Reviews of further applications of MPC for BES can be found in [Afram and Janabi-Sharifi, 2014; Afram et al., 2017] and in [Wang and Ma, 2008].

One obstacle of MPC that has not yet been sufficiently solved in the literature arises from the optimization loop that must be continuously executed during operation, resulting in high

computational power requirements to solve the MPC problem in real-time. This follows especially from the optimization procedure which must execute the model until the optimum control actions (with respect to the objective function and the forecasts) are found. The required computing power thus depends in particular on the complexity of the model, the efficiency of the optimization procedure, and the number of time-steps considered in the future. In [Álvarez et al., 2013] and [Gruber et al., 2014] the authors addressed this problem by introducing parallelisation in computation for their MPC and a handling strategy for simplified models. Another promising approach is to divide the optimization problem into many smaller problems that can be solved more quickly and easily. The authors of [Baranski et al., 2018] presented an approach for distributed exergy-aware MPC and yielded promising results for a typical BES.

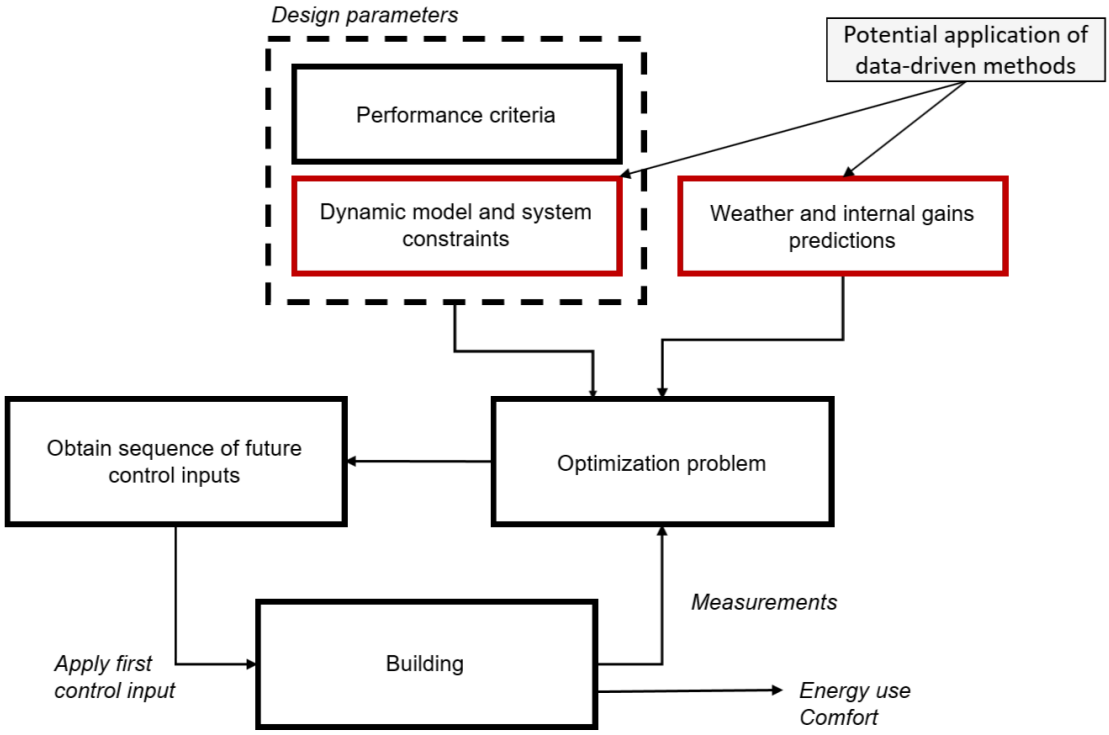

Figure 1.1: The conventional MPC framework for BES [Oldewurtel et al., 2012]. The red rectangles mark where data-driven methods have been already proposed in the literature. The optimization problem is shown, consisting of the performance criteria, the dynamic system model, and the disturbance (weather and internal gains) prediction. Based on the prediction of the disturbance variables and the current system state (sensor measurements in combination with the system model), the optimal sequence of actions is generated via an optimization problem. The first action is written to the BES actuators and the loop starts again.

Besides forecasting the boundary conditions reliably, the accuracy of the used model is a decisive factor for the control quality with MPC. Three modeling approaches are distinguished

in the literature, differing mainly in whether they are based on expert knowledge about the system to be controlled, monitoring data from the system, or a combination of both. One major challenge with MPC for BES is the lack of resources (monetary and domain experts) and detailed system information that would be required to create a detailed physical model (white-box model). The approach to increase model accuracy and at the same time reduce the labor intensive physical modeling is to create grey-box models of the system. Here, comparatively simple physical or statistical equations are calibrated with monitoring data [Wang and Ma, 2008]. The purely data-driven method is referred to as black-box modeling [Afram et al., 2017].

The high effort in the model creation especially for white- and grey-box models is considered an obstacle, as the modeling can only be economically realized for systems with high energy demands and where the operator is motivated to improve the efficiency. Another issue is that the modeling is error-prone and time-consuming, even for domain experts. Furthermore, the models do not automatically adapt to changes in the systems, which leads to repetitions of the time-consuming procedure [Ahmad et al., 2017]. While for a long time these obstacles in the implementation of BES models could only be overcome in the context of research projects, some of the difficulties could be overcome in the future by applying data-driven methods also in the context of practical applications.

We are living in the age of data and data-driven methods are penetrating more and more areas of industry and research. Driven by the ever-improving availability of data infrastructure and low-cost wireless sensors, the availability of algorithms to learn functional relationships from data is also increasing [Minoli et al., 2017]. This applies to traditional statistics, but increasingly also to algorithms from the field of machine-learning [Alfred, 2016]. As shown in figure 1.1, there are also potential applications for data-driven methods for MPC, both in the forecast of boundary conditions as well as in the modeling of the dynamics of the system to be controlled.

In recent years, numerous studies have repeatedly proven that machine-learning and deep-learning techniques allow to learn highly non-linear relationships and therefore are promising tools for forecasting boundary conditions [Ahmad et al., 2017; Hassan et al., 2019; Mocanu et al., 2018]. In [Ahmad et al., 2017], the authors compared two promising architectures, namely artificial neural networks (ANN) and random forests (RF) for the prediction of the hourly HVAC energy consumption of a hotel in Madrid, Spain. Although the neural networks showed slightly better results, both algorithms were well suited for this application. The authors of [Hassan et al., 2019] applied support vector machines for the purpose of predicting the energy consumption of an office building in the Mediterranean region. After parameter tuning, they reached very low deviations from actual measurements and recommended their approach for further application. An application of deep-learning techniques, in particular recurrent neural networks for occupancy prediction in a smart home application can be found

in [Javed et al., 2017]. The interested reader is also referred to the following successful applications [Piotr Żymełka, 2019; Ryu et al., 2017] and to these comprehensive review papers [Mocanu et al., 2018; Sun et al., 2020; Wang and Srinivasan, 2015].

From the different studies, is also is evident that the optimal algorithm, as well as its parameters, are highly dependent on the application scenario and the data. Therefore, efforts are also being made to completely automate this process. For the purpose of energy system modeling, the authors of [Rätz et al., 2019] developed a tool which finds the best combination of algorithms, parameters, and time windows with respect to validation data. A similar approach but with a focus on linear piecewise regression models was presented in [Kämper et al., 2021]

Data-driven models are also successfully used for system identification (thus for the modeling of the technical equipment of the BES) in the following studies [Stepancic et al., 2015; Yang et al., 2020]. In one published approach, often referred to as data-predictive control, the authors successfully combined the advantages of MPC and a machine-learning algorithm called random trees and demonstrated how peak power consumption can be reduced by following their method [Jain et al., 2017, 2016; Smarra et al., 2018]. They also emphasize that black-box modeling of the system dynamics is always an alternative if the costs of creating white- or grey-box models are high. Comprehensive reviews on the application of data-driven models for BES control applications can be found in [Kathirgamanathan et al., 2021; Maddalena et al., 2020].

In this section, successful MPC applications for BES were presented. Furthermore, the trend to integrate data-driven methods for the environment condition forecasts and for the BES models itself has been introduced. Both applications show a high potential to reduce the necessary engineering efforts during the implementation of optimal BES control. The next section focuses on the unsolved obstacles and how another family of artificial intelligence algorithms called Reinforcement Learning could also be a promising tool in the future of BES control.

1.3 Problems with Model Predictive Control and Reinforcement Learning as a possible alternative

As promising as the results achieved and published using MPC for BES control are, as briefly discussed in the last section, there are still some obstacles in the way of widespread practical application. This is mainly because MPC still requires much greater engineering effort compared to standardd RBC, which is often uneconomical. The effort results mainly from the fact that the modeling of the system to be controlled is not trivial, often time-consuming, and error-prone even for experts [Haji Hosseinloo et al., 2020]. Further, the choice of the appropriate cost function and forecast horizon is also crucial, since it directly influences the

behavior of the controller and thus of the system to be controlled. In addition, the modeling process must be repeated after each change of the system [Dowling and Haridi, 2008]. If a component is replaced, parts of the building are refurbished or the type of usage changes, the models must be adapted to the new system behavior manually.

While there is great potential here in automated modeling through data-driven methods, some obstacles remain even with continuously re-calibrated black-box models. A major challenge in these approaches is the poor ability of non-linear data-driven models to extrapolate beyond the limits of training data [McCartney et al., 2020; Rätz et al., 2019; Zhao et al., 2019]. This means that the model integrated into the MPC algorithm can reliably represent the system behavior only within operating states that have already been approached. In addition, the appropriate time windows for re-calibrating the models still have to be set manually, which again leads to a high engineering effort. Another issue arises from the design feature that with MPC, the results of already executed computing operations are not stored. This results in high computational costs during the entire operation and thus high demands on the hardware of the building automation system [Marantos et al., 2020].

Therefore, driven by these obstacles, another promising control approach has gained more attention in the scientific literature in recent years called Reinforcement Learning (RL). RL algorithms are inherently adaptive to their environment and some are also referred to as model-free. Therefore, there is the potential to avoid (or at least reduce) the labor-intensive and error-prone modeling of the system and at the same time ensure adaption to changing environments. A more detailed differentiation between the methods, as well as a discussion of the term *model* and what functionalities it fulfills in the different algorithms, is provided in chapter 2. Generally, with RL, a software agent learns a tailored control policy from the interaction with an environment. For the agent, it is mandatory to collect data trough exploration of the environment dynamics. When it comes to BES, this exploration can potentially be carried out in time windows when there are no occupants in the building or within boundaries where no efficiency and comfort related constraints are violated. Trained RL algorithms encode the optimal action in certain system states and therefore map states directly to actions, a feature that leads to significant lower computational costs compared to MPC [Görges, 2017].

In contrast to MPC, the exploration with RL is inherent and the degree of exploration can be determined by a single parameter. As outlined, the computational costs of trained RL algorithms are comparatively low, as the results of operations already executed are stored and encoded in the algorithm. In the field of RL, promising results have been published regarding the control of complex environments like Atari games or the Chinese board game Go in recent years [Li, 2017; Mnih et al., 2015; Silver et al., 2017].

In summary, according to the literature, RL has three main advantageous areas over classic control approaches:

- **Effort**: For systems where the creation of a model is too complicated or costly, a policy

can be learned with comparatively little effort.

- **Computation**: Even high-resolution state-spaces can be processed with relatively little computational costs.
- **Adaptiveness**: The controller can easily adapt to changing environmental conditions, like aging BES or changing user behavior.

However, these advantages are also contrasted by some challenges. The exploration of large state-spaces cannot be performed in real-time interaction and requires pre-training with monitoring data or simulations. Further, since RL as a subgroup of machine-learning is a statistical method, inefficient or even system-damaging actions cannot be completely excluded, which makes a rule-based security layer necessary.

Therefore, in this dissertation, the aim is to investigate the potential of RL for BES by reviewing the scientific literature, proposing a promising design workflow, and performing several experiments, to address some of the most urgent current research questions. The experiments are performed considering real-world energy systems, taken from finished and ongoing research projects that are kindly supported by the German Federal Ministry for Economic Affairs and Climate Action. Inspired by these real-world application scenarios, the conducted experiments make use of data-driven models of the real systems, simulation models written in the modeling language Modelica [Müller et al., 2016], and a control application in a real-world BES. For the implementation of the used RL algorithms, the Python library Tensorflow [Agarwal et al., 2015] is used.

After elaborating on the open challenges in the use of MPC for BES and introducing RL as an alternative with potential, the next section focuses on the goals and structure of this dissertation. A structure is developed, which contains a workflow for the implementation of RL-supported BES control systems, based on the motivation and the state-of-the-art review. The presented procedure is then applied to the three application examples, which have been inspired from the scientific work in research projects with real BES. The structure concludes with a concluding discussion of the results and additional aspects addressing the experiences made during the investigations, a conclusion, and an outlook for future research pathways and possible products.

1.4 Goal and structure of the presented work

The structure of the presented work is visualized in figure 1.2. The objective of the work is to investigate the applicability of novel RL algorithms for optimal control applications in the context of BES control. For this purpose, the scientific literature is reviewed, a promising RL control design workflow is proposed, several questions currently discussed in the scientific literature are addressed, and associated experiments are conducted. The objectives of the application case studies are:

- to investigate the plug-and-play capabilities of RL algorithms,
- to investigate novel methods of monitoring data-driven pre-training of the algorithms,
- and finally to investigate the transferability of pre-trained algorithms from a simulation into a real system.

The previous two sections motivate the presented work by introducing the challenges regarding the optimal control of BES, the difficulties that come with the promising MPC approach, and the potential of data-driven methods and RL.

First, in the next chapter, the fundamentals of RL are introduced and a detailed literature review is conducted covering the fundamental research as well as the application of the algorithms for BES. Based on this, the following chapter elaborates on the necessary steps when developing a RL-supported BES automation and proposes a promising workflow. The experiment framework developed in the course of this work as well as the tools and implementations used are presented in the following. Furthermore, an efficient method for the optimal hyper-parameter identification of a RL controller is introduced. The chapter concludes with a summary section of a comparison paper that has been published collaboratively with colleges from the institute, in which a RL algorithm has been compared, evaluated, and discussed against other novel BES control approaches. This is followed by an application chapter in which three different application scenarios with different control objectives are presented. The three application case studies are inspired by real-world energy systems investigated in the course of finished and ongoing research projects. From case study to case study, lessons learned are extracted, which are particularly relevant and therefore discussed for future work. Each case study contains its own results and discussion section. The subsequent chapter provides a consolidated discussion of the case studies covering also the lessons learned between the case studies, and relevant aspects in future research and practical application. In the final chapter, general conclusions are drawn and possible future research and product development pathways are presented.

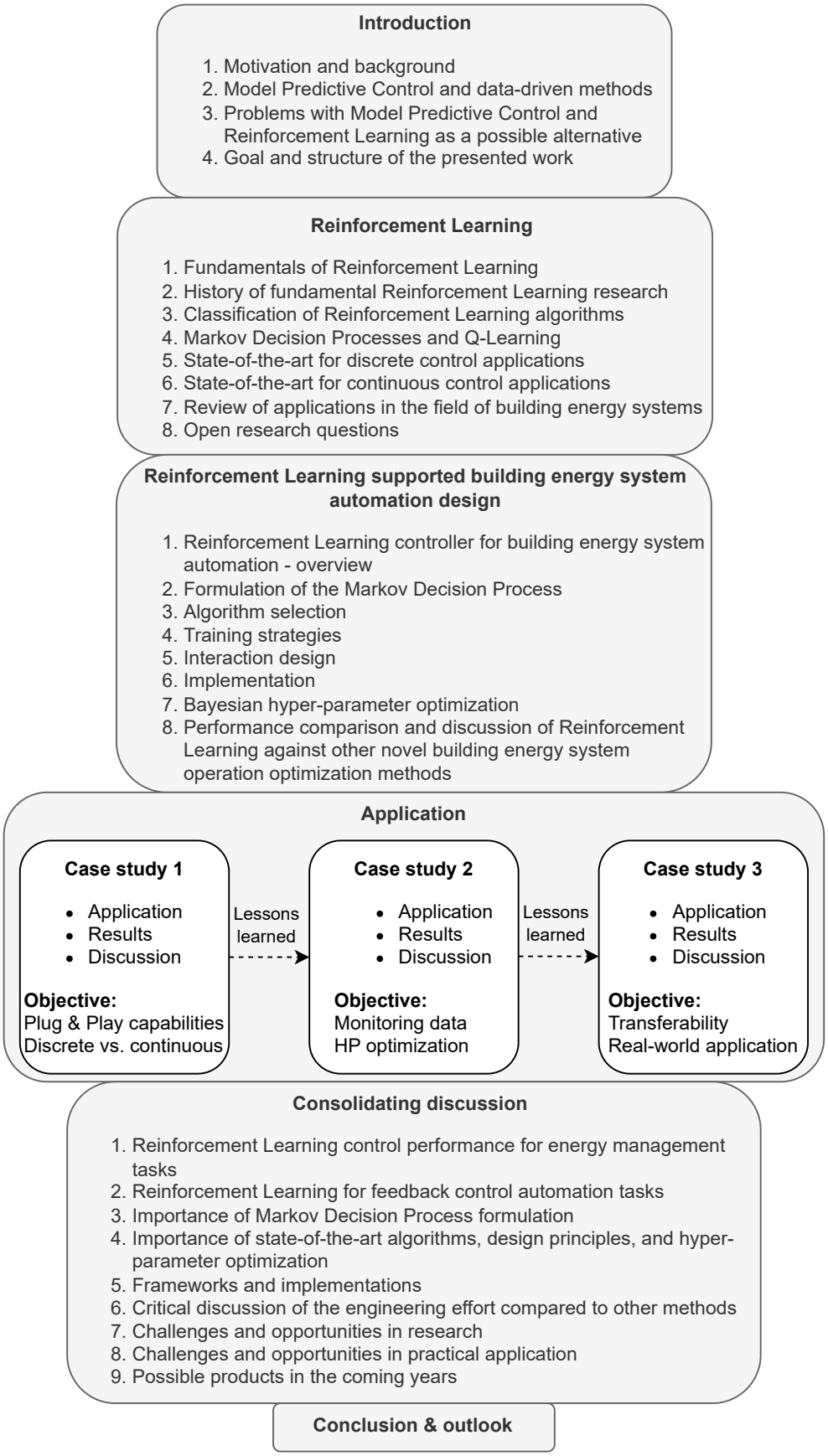

Figure 1.2: The structure of the presented work.

2 Reinforcement Learning

The last chapter introduced the general motivation to investigate optimal control methods and data-driven methods in the field of BES. The promising MPC method with its advantages and disadvantages has been described, and it has been elaborated how novel RL algorithms can be a promising alternative. In this chapter, the key concepts of RL are introduced and the potential for applying this promising approach to BES control applications is elaborated. First, an introduction to the basic principles is presented, followed by a review of the fundamental research, and a classification of available algorithms in the field. The classification of the algorithms includes a discussion of their characterizing design principles and their potential for the application in BES. After the presentation of the state-of-the-art of RL, a literature review and discussion of the successful applications of the algorithms in BES control applications follows. The available application and review papers considered as particularly relevant in the current scientific discussion are presented and discussed in more detail in a designated section.

2.1 Fundamentals of Reinforcement Learning

As a subfamily of machine-learning algorithms, RL algorithms can be distinguished from the other subfamilies by the type of feedback the algorithm receives during training. With unsupervised learning algorithms, similarities are learned and the algorithm receives no feedback at all. Contrarily, supervised learning algorithms receive immediate performance feedback for the generated output. In RL, the algorithm receives delayed feedback and adjusts its actions in order to maximize the numerical performance feedback (reward) from its environment over time. Therefore, RL algorithms are particularly promising for optimal control applications, where actions are performed to maximize a reward signal over time and sometimes actions are rewarded with delay. Unless otherwise indicated, the introduced definitions refer to the corresponding handbooks [Francois-Lavet et al., 2018; Sutton and Barto, 2018].

Figure 2.1 shows the basic framework of any RL process on the one hand and the loop between pre-training and online interaction on the other hand. The algorithm receives state observations and rewards from an environment, in our case a BES, and executes actions based on these observations. In RL terminology (or, as introduced in the next section, Markov Decision Process terminology), the state describes the current condition of the environment and the state-space describes the space of all possible states an environment can have. The

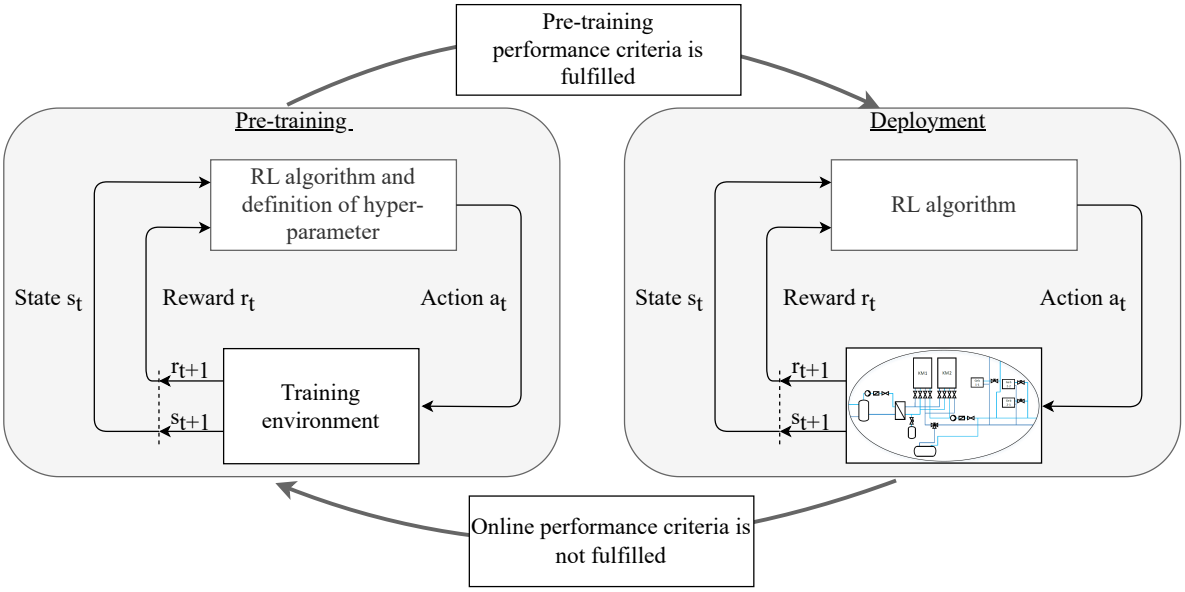

Figure 2.1: The basic concept: A RL algorithm interacts with an environment. It performs actions, based on state observations, rewards, and resulting next states. In addition, the pre-training and deployment loop is illustrated, which should be added for technical systems.

same applies to the executed action, where the space of all possible actions is referred to as the action-space. The reward is a numerical signal from the environment, which evaluates a state usually via a mathematical function. When it comes to policy learning, for example in the case of the RL subcategory Q-Learning, the experience of the algorithm is stored in a learned state-action-value relationship (called Q-values) that maps state-action pairs to action-values. The Q-values, i.e. the value of performing a certain action in a certain state, result from a designed reward function in combination with the historical actions selected. In its simplest and original form, Q-values are stored in a lookup table as exemplarily shown in figure 2.2. Here, each possible action in a given state is assigned a value based on the immediate and future possible reward. More on Q-Learning and its further developments in recent years in section 2.4. The characteristics of the algorithm is influenced by several parameters (called hyper-parameters). Fundamental hyper-parameters are the learning rate, the exploration rate, and the discount factor. By manipulating these numerical values, it is possible to determine multiple aspects: how quickly the algorithm adapts to new observations, whether the algorithm performs the action that appears to be the optimal one in a given state, or whether it takes new paths through the environment to enhance the system experience, and how strongly the algorithm weights large rewards in the future against small rewards from the current state. If the information from the environment is complete and the hyper-parameters and reward signal are well chosen, the algorithm improves its policy in order to maximize the reward from the environment over time.

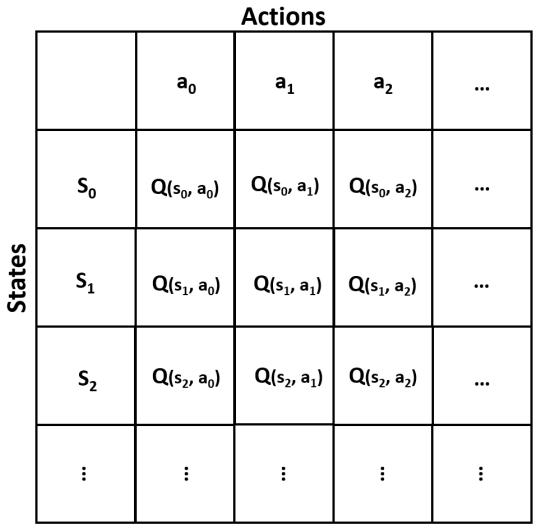

Figure 2.2: Exemplary representation of a Q-value lookup table. Each possible action, of each possible system state receives a value, which indicates the preferable action decision. A Q-value is composed of the immediate expected reward and the maximum Q-value of the resulting next state.

For technical systems where training of the RL algorithm on the real system is not possible, e.g. because the training time until optimal actions are learned is impractical, a pre-training phase is necessary. In this pre-training phase, the algorithm is trained either offline, by means of monitoring data, by an appropriate simulation, or on a test bench. If the defined performance criterion is fulfilled, the algorithm can be used on the real system. If, on the other hand, performance criteria are violated during application on the real system, for example because temperature limits are exceeded or not reached, the algorithm must return to pre-training after the problem has been evaluated by an expert. Possible reasons that an algorithm shows a different policy when applied to a real system after pre-training are, for example, data quality obtained via sensors or inaccuracies of used forecasts. These aspects will be further addressed in chapter 3.

2.2 History of fundamental Reinforcement Learning research

After giving an overview over the basic concepts of RL in the last section, a summary of the fundamental research in the field is presented in this section.

Like MPC, RL has its roots in optimal control theory for dynamic systems, which was further developed in 1957 through Richard E. Bellman's dynamic programming and is based on earlier work (early 19th century) done in the context of the Hamilton-Jacobi theory [Bryson, 1996]. The idea behind any optimal control approach is always to compute the optimal control actions for a controlled system. This requires some kind of description of the system dynamics and a

cost function that assigns a certain value to certain system states. On this basis, the optimal combination of actions for a system can be calculated, for example, by means of dynamic programming.

Traditionally, and as still done today in white-box MPC, the system description is given through a mathematical description of the system to be controlled, which can be used for mathematical optimization. The mathematical description fulfills the functionality of a system model, which returns for different actions correspondingly different resulting system states. These are then evaluated with respect to the cost (or objective) functions. The term model is particularly important here, since it represents the essential distinguishing criterion between the different optimal control families and thus also the different MPC and RL families. It is not clearly described what exactly fulfills the properties of a model and the definitions differ in different areas of science. Generally speaking, scientific models always have the purpose to describe a certain part of the world in a simplified way and to generate useful outputs for defined inputs [Frigg and Hartmann, 2020]. The question of which part of the discussed MPC and RL algorithms fulfills this property will be discussed again in the following. In most RL families, stochastic models are used, which approximate certain equations from the optimal control theory. However, this is also the case for black-box MPC, which already shows the difficulty of a sharp separation of these two control approaches.

As illustrated in figure 2.3, there has been a strong increase in the number of publications in the last decade. One reason is that some fundamental stability problems in the training of neural networks as function approximators were solved and published in 2013. This led to a significant increase in generalizability of the algorithms and the potential to process high-dimensional inputs.

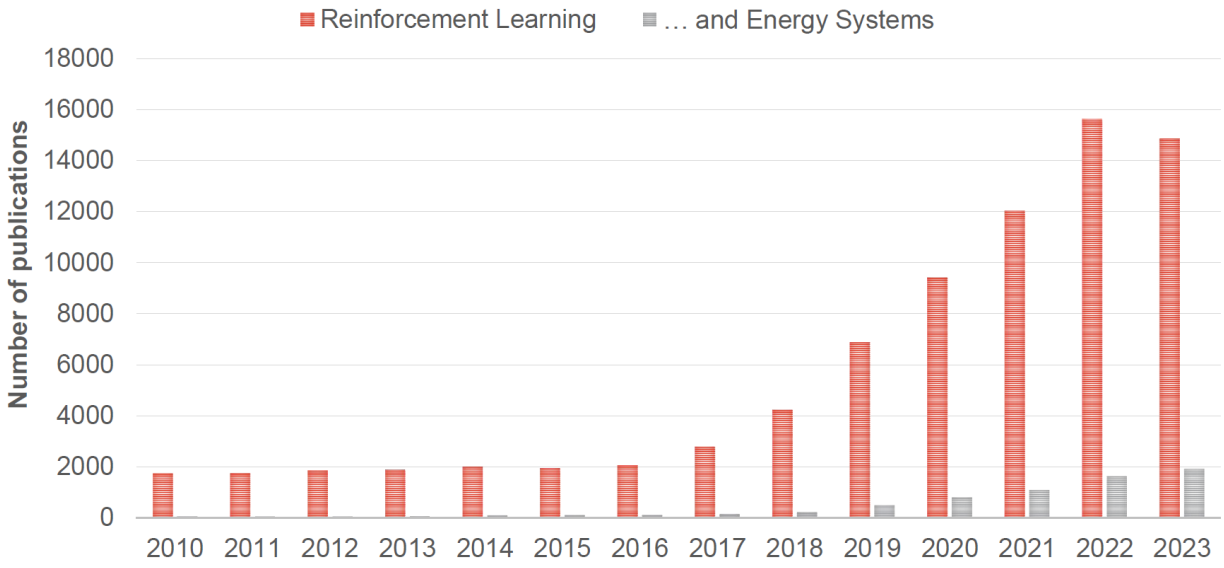

Figure 2.3: Research in the field of RL: Number of publications, accessible via Scopus (https://www.scopus.com (Accessed: 24.11.2023)).

As visualized in figure 2.4, the basic concept of a value function, which not just includes the immediate reward of an action but provides a numerical signal of how valuable an action in a certain state is, goes back to the early years of optimal control theory [Bellman, 1954]. In addition to the basics from this field, the Markov Decision Process (MDP) represents another key concept for RL; It was proposed in 1957 by the Russian mathematician Andrei Andreyevich Markov and represents a form of his state chains, extended by actions and reward signals, and has become the standardized framework for the description of sequential decision problems.

The promising results published since 2013 in this research field, however, have been achieved by using neural networks [Rosenblatt, 1958] as function approximators. The research group DeepMind demonstrated how RL algorithms with neural networks-based state-space approximation can be trained to play Atari games on a super human performance level [Mnih et al., 2015], not only in discrete action-spaces (finite number of actions) but also in continuous action-spaces (actions in the form of a float number within defined bounds) [Lillicrap et al., 2015]. These RL algorithms showed promising results by learning the value function not only by performing actions but also by learning through passive observation of an applied control policy [Hester et al., 2018]. The last publication is particularly interesting because model-free RL shows its full potential to reduce manual engineering if the modeling of a training environment can be avoided. In 2017, a publication led to learning an optimal policy for the Asian board game Go using a RL algorithm. [Silver et al., 2017]. The algorithm beat the current champions of the game in a competition. The game was previously considered impossible to solve by machines because of its almost infinite state- and action-space and its stochastic nature. Annually, numerous publications appear, in which the presented RL algorithms are trained to be more data-efficient and to interact more and more robustly with highly dynamic and hard-to-predict environments.

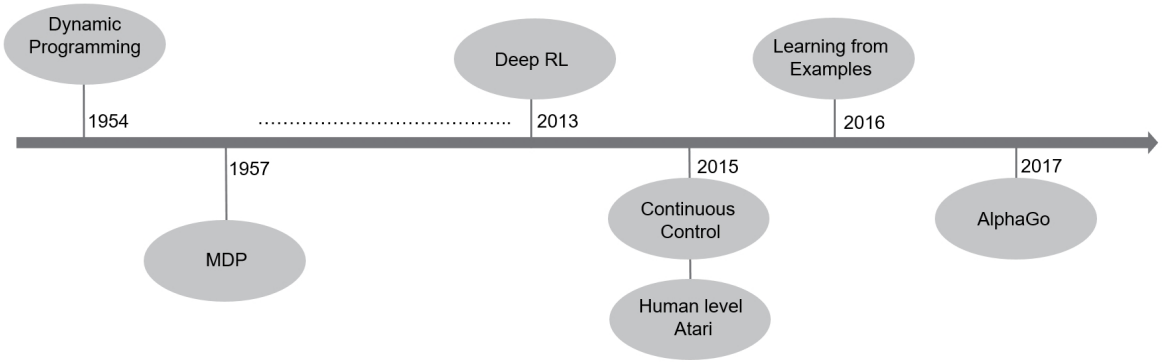

Figure 2.4: Time-line of selected research in the field of RL.

2.3 Classification of Reinforcement Learning algorithms

In this section, a classification of published/available RL algorithms is given. As shown in figure 2.5, RL algorithms are initially divided into two families: algorithms in which the control actions are performed on the basis of a model of the controlled environment and algorithms where the control actions are performed without a model. "Without model" in this context refers to the absence of a black-box or white-box model representing the physical dynamics of the system only in combination with a planner (optimizer), which executes the model to plan the optimal action pathway. Nevertheless, in the course of learning, (in the light of different definitions of the term model) model building takes place in the form of learning to forecast the expected future rewards that result from a certain action in a certain state.

While model-based algorithms typically use data from interactions to learn a model of the environment, which is then used for stochastic or deterministic planning, model-free algorithms use the data to learn other functional relationships from optimal control theory, to evaluate and select the control actions. In the case of model-based approaches, the distinction from data-driven MPC is not fully determined in the literature. An attempt of a clear classification was made in [Nagy et al., 2018]. The authors elaborate that only the exploration strategy, i.e. the structured increase of system experience (in the form of interaction data) as well as the adaptivity of the policy itself, are unique for RL. While in MPC typically a strict mathematical optimization is performed based on the model of the environment, model-based RL learns the best practice for the controlled environment using a combination of an adaptive model and an adaptive policy.

However, algorithms referred to as model-free have some highly promising advantages for the use in BES control applications. With comparable good control performance, the resulting computational costs (after training) are significantly lower, which is particularly relevant for the use in existing buildings, where the computing power is often a limiting factor for the implementation of energy management and contol logics. Further, the adaptation to changes in the controlled environment is much faster, as long as the overall relationship between states, actions, and rewards of the control task does not change. This is supported by the results also published in [Nagy et al., 2018], where the authors compared white-box MPC, black-box MPC, model-based RL, and model-free RL for space heating control of a simulated building model [Ruelens, 2016]. They elaborate that with comparable performances, the computational costs of model-based RL are more than 20 times higher than those of the model-free RL algorithm. They also underline the stability of the model-free RL approach against sudden changes in the environment. At the same time, they emphasize the higher sample efficiency and overall performance of the model-based approach. However, the sample efficiency in particular has been significantly increased for model-free algorithms since then [Haarnoja et al., 2018; Hessel et al., 2017]. Even though model-based algorithms can achieve a higher maximum control performance in theory, the model-free algorithms are evaluated being

more promising for real-world BES control applications due to their lower required computing power and higher robustness against changes in the system (which with the model-based family lead to instant model errors and thus critical losses in control performance). The model-free family of algorithms is therefore considered in the course of this work.

The model-free family is further divided into algorithms where the control policy is directly learned, for example using gradient descent methods with neural networks, and those where a state-action-value function is learned. The latter, referred to as Q-Learning, in turn have some inherent advantages for use in BES control applications. Here the state-action-value function is learned from environment transitions (in form of: [state, action, next state, reward] samples), thus the experience can be extracted from data fragments from the system. Monitoring data can be used for training, and the algorithms can therefore be trained by passive observation of other control policies.

In the other group (called policy optimization) the policy is learned directly on the basis of the reward history during fixed episodes of interaction. A new policy must always be applied for a fixed episode (period of time, for example weeks) before the algorithm computes the series of rewards received during that episode and updates the policy according to the gradients. In contrast to Q-Learning, the algorithm evaluates not only the benefits of the immediately following states, but also the reward history of the entire episode. Every update to the policy must be tested by interacting with the system, under the constraint that the policy is not updated during the episodes. Until the optimal policy is reached, many test weeks with suboptimal policies may have to be carried out. The data-efficiency of the policy optimization-based family is therefore lower. This significantly limits the potential for use in real systems. Emphasizing this, the authors of [Biemann et al., 2021] found that, with a state-of-the-art Q-Learning-based algorithm for continuous control (the SAC algorithm), a typical data center cooling control problem was solved with ten times less data compared to a state-of-the-art policy optimization-based algorithm.

Therefore, taking the comparable final control performance, the data-efficiency, and the offline training ability into account, the focus of most publications in the field and of the investigations carried out in the course of this work consider Q-Learning and hybrid algorithms (combining Q-Learning and policy optimization techniques). Algorithms for continuous as well as discrete control actions are available, and stability and performance improvements for the algorithm architectures are published almost monthly. This is also clearly reflected in the representation of these algorithms in the discussed papers in section 2.7.

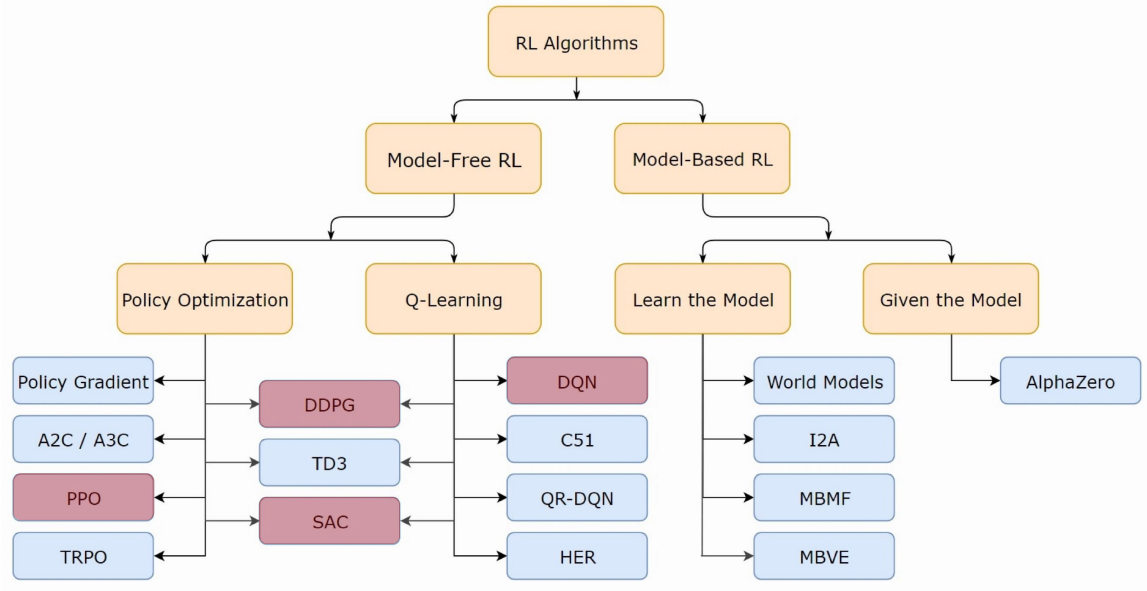

Figure 2.5: Families of RL algorithms according to the popular Deep RL OpenAI repository Spinning Up [*OpenAI Spinning Up*, 2020]. The algorithms relevant for this work are marked in red. PPO: Proximal Policy Optimization, DDPG: Deep Deterministic Policy Gradient, SAC: Soft-Actor-Critic, and DQN: Deep Q-Network.

2.4 Markov Decision Processes and Q-Learning

The established framework for the description of RL problems is the Markov Decision Process (MDP), which represents a formalized description for sequential decision problems [Sutton and Barto, 2018]. A MDP is characterized by its state-space S, action-space A, transition probabilities (from one state to a next state S') $Pr : S \times A \times S'$, a reward function $R : S \times A$, and a discount factor γ. If the MDP is well formulated, the interacting RL algorithm learns the optimal actions in the possible MDP states, encoded in the optimal policy (π^*) and maximizes the expected total reward over time.

If the transition probabilities are known, the MDP is considered as a planning problem [Sammut and Webb, 2017]. In this case, the MDP can be solved via dynamic programming: since the transition probabilities encode the dynamic behavior of the environment to be controlled, an optimizer can test its control actions on this model and calculate the optimal set of actions with respect to the reward function. The white- or black-box models used for traditional and data-driven MPC, as well as the model in model-based RL, therefore encode the transition probabilities of the environment in MDP terminology.

However, since in most real-world applications a detailed model of the controlled environment is not available (or only with high engineering effort), methods have been developed to solve the MDP without prior knowledge of the system dynamics [Wooldridge, 2001]. The optimal

policy ($\pi^* : S \mapsto A$) is the functional relationship which maps states to actions and represents the relationship to be learned in those approaches. Among this family of algorithms referred to as model-free, a further distinction is made between policy optimization-based approaches and value-based approaches. As outlined in section 2.3, value-based approaches are particularly interesting for BES as the algorithms can learn from incomplete episodes, can be trained with historical data, and are more robust against changes in the controlled environment while reaching a similar control performance as the policy-based algorithms.

A well researched family of algorithms encoding the optimal policy in form of the Q-values (state-action-values) is called Q-Learning. With Q-Learning, the optimal policy is learned implicitly in the form of the values of states and the actions in states. Those combinations are called Q-values, and therefore the learning process is referred to as Q-Learning. In the basic form, the Q-values (which encode the direct and future rewards from an action in a state) are stored in a lookup table, as illustrated in figure 2.2. During learning, the updates of the Q-values ($Q(s,a)$) are then propagated back through the table using the Bellmann equation [Bellman, 1956]:

$$\underbrace{Q(s,a)}_{new\,value} \leftarrow (1-\alpha) \cdot \underbrace{Q(s,a)}_{old\,value} + \underbrace{\alpha}_{learning\ rate} \cdot (\underbrace{r(s,a)}_{reward} + \underbrace{\gamma}_{discount\ factor} \cdot \underbrace{\max_{a'} Q(s',a')}_{\text{expected reward}}) \quad (2.1)$$

According to equation 2.1, $Q(s,a)$ is an estimation of the future discounted (γ) rewards, expected when selecting a certain action (a) in a given state (s). The learning rate α represents the sensitivity against new experiences over past experiences, and the discount factor γ determines the weighting between immediate and future rewards. Therefore, the maximum (max) Q-value of the next state (which is expected to result from the current action) encodes how beneficial the action will be for future rewards. The discount factor determines (γ) how much the algorithm should plan, which means how much future possible rewards (encoded in the maximum Q-value of the next state) are weighed against immediate rewards.

During training, random actions are chosen with a decreasing probability. The ε-greedy exploration strategy is one approach to balance exploration and exploitation throughout the training process: ε specifies the probability for the controller to select a random action against exploiting the learned policy. During training, ε is progressively decreased in each training episode (n): $\varepsilon \to 0$ for $n \to n_{max}$.

After training, the action is selected from the table via:

$$a = \arg\max Q(s,a) \quad (2.2)$$

Therefore, this approach does not require a separate optimizer which tests control actions on a model of the system dynamics, hence the classification as model-free. Nevertheless, a kind of model building takes place by learning to predict the expected immediate and future rewards that result from a particular action in a particular state. The Q-values therefore encode the result of an optimization leading to the optimal actions in certain states.

2.5 State-of-the-art for discrete control applications

Discrete control applications are characterized by the actions only being selected from a given and countable set of possibilities. For large and/or stochastic state-spaces, tabular Q-Learning is not suitable due to the "curse of dimensionality", which referrers to the issue that with each added state-space feature, the dimension of the space and therefore the complexity of the MDP increases over-proportionally. Therefore, for complex applications (as most real-world applications are) function approximation of the state-action-values ($Q^{\pi}(s, a)$) has been established in the literature [Buchanan, 2005].

While early implementations were characterized by poor generalizability and performance, the promising results published in recent years have been achieved by using neural networks as approximators. A solution for the stable training of neural networks as function approximators was first published in 2013 [Mnih et al., 2013]. With the proposed design principles, much more complex and highly non-linear problems have become solvable, as it was demonstrated on 49 Atari 2600 games [Mnih et al., 2015]. Since then, deep neural networks have become an established method for the approximation of the Q-function.

The schematic structure of a Deep Q-Network (DQN) for state-action-value approximation is shown in figure 2.7. The input layer is the state-vector s with n entries. After a certain number of hidden layers, the output layer with m possible actions contains a value-entry for each action (a_{1-m}). This design is necessary for the integration of the neural network into the Q-Learning architecture. While after training a single output neuron would be sufficient, which outputs the selected action, the update function (Bellmann equation 2.1) processes the evaluation of all possible action values. Hence, the architecture of assigning a value to each action and then, after training, choosing the action according to the highest value. The tunable parameters of a DQN are iteratively updated via stochastic gradient descent and back-propagated through the network. After training, the DQN represents a data-driven model that does not only represent the dynamics of the environment but also a predictive model of future states and the resulting future expected reward value.

The authors of [Mnih et al., 2015] introduced a novel DQN architecture, as visualized in figure 2.6. The two effective improvements are: To reduce the correlations in the sequence of training samples resulting from direct training with data from online interaction, experience replay in the form of a data replay memory is used. Transitions (s, a, r, s') are stored, and for

each network update, a certain number is randomly selected for training the DQN parameters (θ). Further, to improve the stability of learning, a target-network (with own parameters θ^-) is introduced, whose trainable parameters remain constant for a fixed number of steps. The target-network is used to calculate the target-Q-values for the Q-network updates according to the Bellman equation (2.1). Thus, the modified loss function resulting from a set of randomly selected transitions from the experience memory is:

$$L_i(\theta_i) = \mathop{\mathbb{E}}_{(s,a,r,s')}[\underbrace{(r(s,a) + \gamma \max_{a'} Q(s', a'|\theta^-)}_{Target\,Q} - \underbrace{Q(s,a|\theta)}_{Predicted\,Q})^2] \tag{2.3}$$

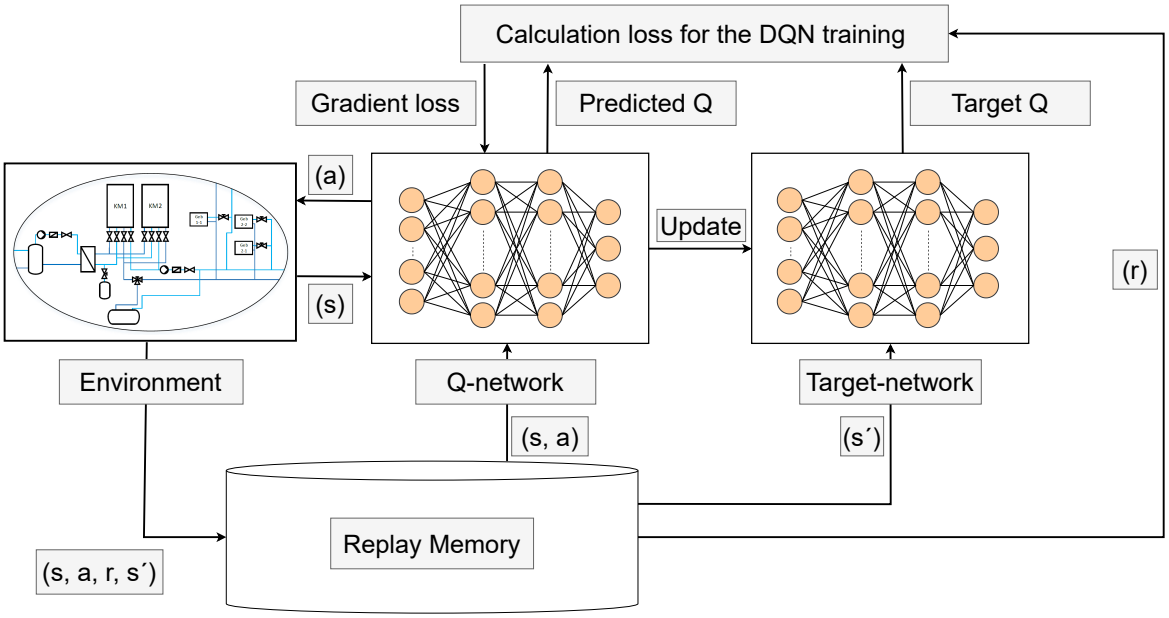

Figure 2.6: The DQN architecture proposed by the authors of [Mnih et al., 2015]. The target-network predicts the value of the next state for training the Q-network. The Q-network interacts with the environment. The loss function gives the gradient for training the Q-network. Training is not performed with correlated data from the immediate interaction but with random data from the replay memory. The target-network is updated from the Q-network regularly but not in every training cycle, which stabilizes the training.

A well-tested implementation of the algorithm is provided within the Python package Stable-Baseline [Hill et al., 2018]. The implementation includes the state-of-the-art improvements to the DQN training process: A replay memory serves as an internal sample storage and breaks correlations within the training samples and further increases sample efficiency by selecting and reusing random training samples from the storage [Schaul et al., 2015]. Prioritized replay accelerates the training speed by selecting training samples from the replay memory, which lead to a higher loss and therefore to faster training of the neural network [Schaul et al.,

2015]. Double Q-Learning improves numerical stability by alternatingly using one network (i.e. the current network) to predict the current Q-values while the other one is updated with the training samples (i.e. the target-network) [Hasselt et al., 2016]. The "dueling" DQN architecture divides the Q-value estimation in two separate sequences of fully-connected layers after the entry layer [Wang et al., 2016]. The streams estimate a state-value and an action advantage, which are combined at the end for the Q-value estimation of the state-action pair. This extension restricts the action-space to actions that are more promising in the state, thus speeding up the training. In the default settings, two hidden layers with 64 neurons, using rectified linear units (ReLU) activation functions, are implemented.

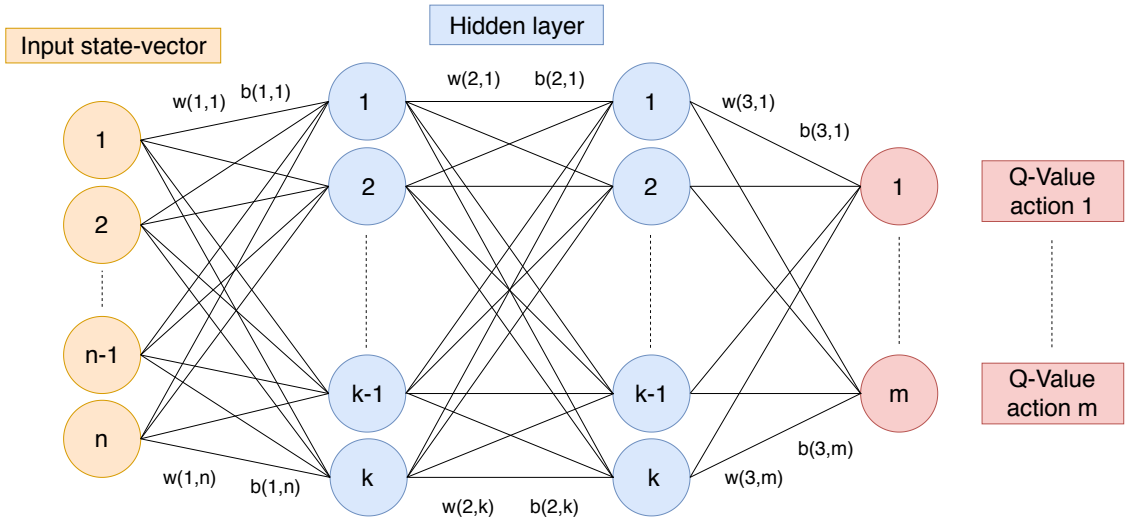

Figure 2.7: State-action-value approximation with a neural network. The network receives the state and returns the Q-values for m discrete actions possible in that state. While after training a single output neuron would be sufficient (which outputs the selected action) the update function (Bellmann equation 2.1) processes the evaluation of all possible action-values.

2.6 State-of-the-art for continuous control applications

Unlike discrete control tasks, continuous tasks do not limit the number of possible actions. Rather, a floating number within defined limits is calculated. The corresponding family of algorithms is therefore more suitable when the action-space is limited but the optimal outputs cannot be defined in advance by countable actions. A possible action-space would be, for example, a valve position, which can only be between 0 % and 100 %, and can have any value in between. For discrete algorithms, 100 possible actions would have to be given, which would be completely independent of each other for the algorithm and would all have to be tested for each state of the MDP.

The state-of-the-art for continuous control tasks in continuous action-spaces follows the actor-

critic architecture, a hybrid between policy-based algorithms and value-based algorithms. In the actor-critic approach, the actor-network learns a control policy $\pi_\theta(s, a)$ (stochastic or deterministic) by updating the policy parameters θ via gradient ascent [Sutton et al., 2000]. The critic-network represents the approximation of the state-action-value function $Q(s, a|a_t = \pi_\theta(s))$, as described in section 2.5. $Q(s, a|a_t = \pi_\theta(s))$ is used to calculate the gradients for the actor-network updates and thereby iteratively optimizes the policy π_θ. The schematic structure of the architecture is shown in figure 2.8. For each interaction with the environment, the reward from the environment is used together with the current state and the action performed by the actor-network to train the critic-network. This way it is possible to learn the value of an action in a state even without pre-defining the number of actions. The continuous output of the actor-network serves as input for the critic-network along with the observation (current state) from the environment. The Q-value output from the critic-network, on the other hand, is used to evaluate the action performed by the actor-network and is used to determine the gradients for the gradient ascent-based update of the critic-network. A closed loop of improving the policy (through an updated actor-network) and improving the action evaluation (through an updated critic-network) results. After training, the critic-network is not needed anymore and the trained actor-network can be deployed independently.

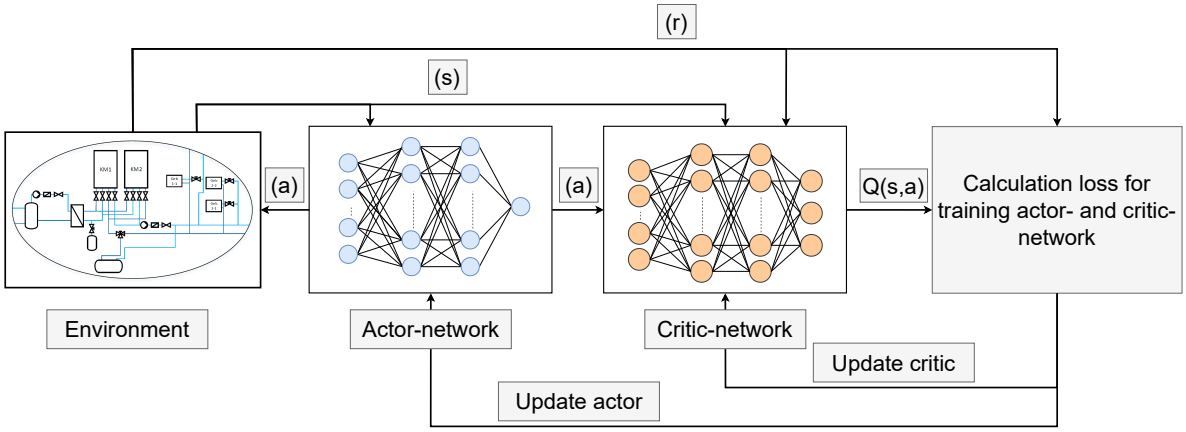

Figure 2.8: Schematic structure of the actor-critic approach: The critic-network is used to calculate the value (Q) of state-action pairs (s,a). Q is used to calculate the gradient for the training of the actor-network, which is via states (s) and actions (a) in direct interaction with the environment. The reward is used to evaluate the action of the actor-network by the critic-network on the one hand and on the other to calculate the gradient for the update of the critic-network. A closed loop of improving the policy (through an updated actor-network) and improving the action evaluation (through an updated critic-network) results.

For DDPG (Deep Deterministic Policy Gradient), a well-tested implementation of the approach is also provided within the Python package Stable-Baseline [Hill et al., 2018]. Both networks are updated with target-networks and the critic has a replay memory (like the DQN

introduced in section 2.5). The DDPG can therefore be understood as an extension of the DQN for continuous action-spaces. For a comprehensive description of the underlying mathematics, the following literature is recommended: [Silver et al., 2014; Sutton and Barto, 2018].

The best results in terms of convergence speed, stability, and final performance in literature are achieved by the so-called Soft-Actor-Critic (SAC) algorithm. SAC is a slightly modified version of the DDPG algorithm [Lillicrap et al., 2015] that uses entropy[1] regularization for the exploration-exploitation trade-off and optimizes a stochastic policy that allows to incorporate uncertainties in the environment better [Haarnoja et al., 2018]. With entropy regularization, the policy is optimized by maximizing a trade-off between expected reward and entropy (regulating the exploration-exploitation process). Within the context of entropy-regularized RL, entropy is an indicator for the information density of samples from a stochastic source (the environment). The idea is to incorporate the predictability of the effects that actions (selected by an algorithm) have on the environment. Entropy-regularized RL therefore changes the standard objective function by adding an entropy term. This way, the algorithm receives a bonus reward at each time-step that is proportional to the entropy of the policy in the given state.

2.7 Review of applications in the field of building energy systems

In this section, a general overview of the current literature in the field of RL for BES is presented first. This is followed by a more detailed discussion of selected publications. Based on this, in the next section, open research questions are elaborated, which are addressed in this work.

2.7.1 Literature overview

RL has been studied in recent years for many different optimal control problems in energy systems in general and also for BES. In the literature, the investigated systems can be divided into four categories: Optimal HVAC and BES system operation [Wan et al., 2018; Wang et al., 2017; Yang et al., 2015], smart grids and energy systems [Bahrami et al., 2018; Biemann et al., 2021; Rayati et al., 2015], distributed energy systems [Kofinas et al., 2018; Pinto, Piscitelli, Vázquez-Canteli, Nagy and Capozzoli, 2021; Touzani et al., 2021], and optimal electric vehicle operation [Dorokhova et al., 2021; Vandael et al., 2015]. The recurring control objectives are: Energy efficiency and occupant comfort in buildings [Nagy et al., 2018; Zhang et al., 2018], reduction of temperature deviations from setpoints in energy systems [Al-jabery et al., 2017]

[1] Entropy originates from thermodynamics and describes the increasing stochastic disorder of the particles of a closed system in the context of the second law, when heat or matter is added. In contrast to this, entropy in computer science is also a measure of disorder, but it refers to interference signals that occur with the actual signal and to the computational effort required to expose the actual signal.

or adaptivity to external signals in demand response applications [Vázquez-Canteli, Ulyanin, Kämpf and Nagy, 2019; Zhong et al., 2021]. As with other optimal control methods for BES, the use of thermal energy storages under dynamic boundary conditions has been investigated in many studies. The applicability of RL for the optimal use of thermal storage systems was already investigated with a simulated laboratory environment in 2006. The authors demonstrated that although RL does not reach the performance of MPC, RBC was clearly outperformed [Liu and Henze, 2006]. Three years later, the authors of a comparative study between MPC and RL in a BES control application concluded that the control performance of RL can reach the control performance of MPC, even in cases where a deterministic system model is available [Ernst et al., 2009].

Since then, RL has been studied for the control of low exergy BES [Yang et al., 2015], energy management in microgrids [Kofinas et al., 2018], process energy control [Dong et al., 2020; Filipe et al., 2019], for optimal control of HVAC systems [Brandi et al., 2020; Chen et al., 2018; Jia et al., 2019], battery storages [Shang et al., 2020], and the coordination of thermostat controllable loads [Kazmi et al., 2019]. In [Yang et al., 2015] the authors demonstrated how RL can find a control policy for a low-exergy building equipped with PV and geothermal heat pumps for which MPC would have been unnecessarily complex. They also demonstrated how to overcome the potentially data-inefficient controller training by linking training data generation to a simple guiding RBC. In [Vázquez-Canteli et al., 2017], the authors used Q-Learning to optimize the heating and cooling of a heat pump and storage equipped building. In [Chen et al., 2018], the authors used tabular Q-Learning to control a HVAC system combined with controllable windows for natural ventilation. Compared to a classical RBC strategy, up to 23 % energy savings were achieved with significant increase in occupant comfort. A drawback is that with tabular Q-Learning, the learning process can be unstable with large time gaps between actions and associated rewards.

Driven by the reasons introduced in section 2.3, in the literature on RL for energy systems, there is a strong trend towards model-free, value-based approaches, such as Q-Learning. Deep Q-Learning is the most widely used RL technique [Han et al., 2019] and energy savings of 10-20 % (compared to RBC) are typically published [Mason and Grijalva, 2019]. However, although offline training with monitoring data is a promising option for such algorithms, comparatively few studies have focused on the possibility of training the RL controller in this way [Vázquez-Canteli, Ulyanin, Kämpf and Nagy, 2019; Yang et al., 2015], and even fewer studies have investigated the potential of using monitoring data to learn a data-driven training environment for the algorithm. [Di Natale et al., 2021; Lork et al., 2020; Schubnel et al., 2018].

Some studies also investigated control applications with continuous action-spaces. In [Li, 2017], an actor-critic algorithm was used to control the HVAC system of a data-center. With data-calibrated EnergyPlus models, the authors saved 15 % energy costs. The authors of

[Biemann et al., 2021] investigated a similar use case and highlighted the robustness, performance, and training data efficiency of the used SAC algorithm. A real-world system was controlled in [Ruelens et al., 2018]. The authors used Fitted Q-Iteration (an earlier version of machine-learning based Q-Learning) for electric water heater control in a laboratory and reduced energy consumption over 40 days by 15 % compared to classic thermostatic control. In [Kofinas et al., 2018], the authors proposed a cooperative multi-agent RL technique with fuzzy state-space approximation in a continuous action-space. The controlled microgrid included PV, a fuel cell, and a diesel generator as well as a battery bank and was efficiently controlled by the coordinated agents.

Rather few studies provide critical views on RL algorithms. One is provided in [Khayatian et al., 2021]. The authors investigated the robustness of RL algorithms against uncertainties. For this purpose, artificial load profiles of buildings were generated and used as a basis for decision making for both a RBC algorithm and a trained RL algorithm. They found that the RL algorithm, as long as it is trained with little data, is very sensitive to uncertainties and unseen system states. They conclude that the control performance of RL algorithms for real-world applications might be over-estimated.

In recent years, some comprehensive review articles have been published on the topic. A comprehensive review, with the focus on RL in demand response applications, can be found in [Vázquez-Canteli and Nagy, 2019]. More recently, another comprehensive review about the current state of research has been published in [Wang and Hong, 2020]. Based on 77 evaluated studies, a wide variety of BES, algorithms, and control objectives are discussed. The authors conclude that, due to the inherent trial-and-error process, RL online training can be very data- and time-demanding. They also address the question of how to avoid undesirable actions during training and conclude that pre-training with monitoring data and with a simulation environment as well as the use of expert knowledge are particularly promising. Further published reviews for BES control RL application with a focus on autonomous BES control and controlling occupant comfort can be found in [Mason and Grijalva, 2019] and in [Han et al., 2019]. As well as one review with a broader focus on the application of RL to different energy systems, not only BES [Perera and Kamalaruban, 2021]. This study also highlights how the popularity of the method continues to grow, whereas the popularity of MPC has plateaued at a high level.

2.7.2 Discussion of selected publications

After providing an overview of the literature on RL for BES in the last subsection, this subsection now discusses selected publications in more detail.

Important groundwork for the use of RL for demand response with residential buildings was carried out in the course of a dissertation at the KU Leuven [Ruelens, 2016]. As well as

in the course of the from this dissertation extracted and through it inspired publications [Nagy et al., 2018; Ruelens et al., 2018]. The authors investigated the limits of buffer storage sensor encoding with an innovative neural network architecture (autoencoder) and tested their algorithm in a real-world application with the objective to exploit the storage capacities of electric water heaters for demand response. Besides this interesting case study, an essential contribution lies in the scientific comparison of data-driven MPC, model-based RL, and model-free RL. On the one hand, a distinction between data-driven MPC was elaborated, which has been discussed in section 2.3. On the other hand, a performance comparison between model-free and model-based RL for a typical demand response use case for a single family building was published here. While model-based RL can almost reach the performance of MPC and is slightly more data-efficient, the strengths of model-free RL are the much lower computational costs and the robustness of the algorithms against changes in the controlled system. This is especially relevant considering BES where computing power is a limiting factor and system changes (due to changes in occupant behavior or aging energy systems) are unavoidable.

Another effort to compare the performance of different RL algorithms against MPC was made in [Dorokhova et al., 2021]. The authors compared the continuous control DDPG, the discrete control DQN, stochastic MPC, deterministic MPC, and a RBC for the control task of maximizing the PV self-consumption for electric vehicle charging. The competing control objective has been defined as the state of charge of the battery of the vehicle when the vehicle is started. In their mathematically formulated experimental setup, the RL algorithms achieved slightly lower but comparable performance compared to the MPC algorithms, but with significantly shorter computation times. The MPC algorithms clearly achieved the highest control performance but were critically evaluated, in particular because of the complex model formulation and the long computation times. It is noteworthy that the RBC algorithm also performed very well. However, the authors argue that the potential is limited when systems become more complex, especially if vehicle-to-grid would become relevant for power grid stabilization in the future, the control rules would be too complex for such applications.

One of the most comprehensive reviews of RL for BES control was published in [Wang and Hong, 2020]. The authors discussed 77 studies with a special focus on the correct application of RL-specific design principles. Namely the consideration of the Markov Property, which declares that the future of a MDP must be predictable from its current state only [Sutton and Barto, 2018]. They found that this was hardly fulfilled in most of the investigated problem formulations, because:

- 91 % did not include historical data, which is problematic, considering the slow building thermal dynamics, and
- 83 % did not include predicted states, such as weather forecasts.

Both aspects are critical with respect to the achievable performance of the algorithms. The first because the current dynamics of the system are hardly assessable purely on the basis

of current sensor values and the second because the algorithm is not provided with important information for forward-planning actions and thus tends to operate the system overly cautiously. Further, the authors stated that even most of the very recent publications use comparatively simple algorithms and do not make use of recent RL innovations in terms of stability and convergence speed. They also found that very few studies make it into practical application, just 11 % of the controllers were implemented in actual buildings.

The authors concluded with three main challenges that need to be addressed to exploit the full potential of RL for BES:

- To increase the applicability in real buildings, the training speed and robustness must be improved.
- To establish comparability between RL controllers, standardized open-source testbeds and datasets are required.
- To use trained RL controllers for different tasks, the generalization capabilities need to be improved.

One of the most visible projects in RL for demand response for BES is the CityLean challenge [Nagy et al., 2021]. The challenge was originally started to address the problem of lacking comparability of different RL algorithms and methods by publishing a documented and standardized training and testing environment on GitHub. The task of the RL application consists in the coordinated electricity supply of 10 buildings (residential and non-residential) with the aim to avoid peaks in demand and to keep the total energy demand as low as possible. For this purpose, the thermal demands of the buildings in four different climate zones were simulated upfront and added to the environment. The electricity demand results from the operation of air-to-water heat pumps and chillers which can be controlled together with the temperature setpoints for thermal storages in the buildings. The buildings are equipped with different systems, for example, some have PV panels on the roof. This leads to different state-spaces for the individual buildings. Unpublished test demand time-series were used to evaluate the solutions proposed by the participants, and the submitted RL algorithms and problem formulations were compared in terms of their performance. The challenge entered its second phase in 2021 and several creative solutions were published for the problem [Pinto, Deltetto and Capozzoli, 2021; Pinto, Piscitelli, Vázquez-Canteli, Nagy and Capozzoli, 2021; Vázquez-Canteli, Detjeen, Henze, Kämpf and Nagy, 2019; Vázquez-Canteli et al., 2020]. One of the most promising spproaches comes from one of the challenge developers himself [Vázquez-Canteli et al., 2020]. It involves controlling each building with a SAC algorithm that calculates the actions. At the same time, each building has a simple prediction algorithm of the future energy demand of the building. The RL algorithm receives the current state of its building, along with this prediction and the demands of its direct neighbors. By formulating the problem in this way and rewarding coordinated actions, coordinated agent

actions are generated; this approach theoretically works for a very large number of buildings, as in the case of a large city.

Two studies addressing the problems of applying RL in practice were published in [Touzani et al., 2021] and in [Zhang et al., 2021]. In [Zhang et al., 2021], the authors implemented a testbed for RL-based energy management for demand response. The authors describe the necessary hard- and software to translate signals from a cloud-based RL energy management system into real actuator signals. The developed testbed, including monitoring of the technical equipment, is presented and the interaction with a Q-Learning algorithm is described. Although the published energy savings after training the algorithm are very promising, the main contribution is the description of the interaction between the RL algorithm and the real sensors and actuators. An application of RL for a real BES was also published in [Touzani et al., 2021]. The authors use a DDPG algorithm to control the BES of the so-called FLEXLAB, a well-equipped experimental test facility at the Lawrence Berkeley National Laboratory in Berkeley, California, United States. The HVAC systems of the building are controlled in combination with a battery storage and a PV system. The DDPG was previously trained with a simulation and was subsequently evaluated against a state-of-the-art RBC algorithm. Under dynamic price signals, load shifting saved up to 39 % costs while maintaining thermal comfort.

In [Brandi et al., 2020], the authors demonstrated Deep Q-Learning for the optimization of indoor temperature control and heating energy consumption in a simulated office building. The control objective was to optimize the energy consumption of the environment, designed to meet the dynamic behavior of an office building case study located in Torino, Italy. The authors emphasize that activated learning (learning rate > 0) can lead to instable interaction, if the problem is not well formulated. After a sensitivity analysis of the hyper-parameters, they published energy savings between 5 % and 12 %. Although the analysis was carried out carefully, the authors did not aim for an optimal hyper-parameter selection for this problem. A question raised by this work is whether the process of optimal hyper-parameter determination can be automated and improved by available stochastic methods such as Bayesian hyper-parameter search.

The authors of [Kotevska et al., 2020] identified the interpretability and the trust of operators in the quality of the actions performed by RL algorithms as a core challenge of RL research in the BES domain. To address this challenge, methods are presented to compute correlations between the input features from the state-space and the probability that the algorithm performs a certain action. For demonstration purposes, data from a real building is used and coupled with a simulation environment. The algorithm used (DQN) is trained to control a heat pump and temperature setpoints from two rooms under dynamic outdoor temperatures and price signals. After training, the action probabilities of the algorithm, the feature importance for certain actions, and other metrics are calculated for each system state. The

presented approach by means of visualization and the interpretation of the actions via expert knowledge enables an interpretation of the RL actions. Thus, confidence in the reliability and performance of the systems can be generated by the proposed procedure.

2.8 Open research questions

The literature review conducted shows that despite the large number of successful demonstrations, there is still a significant need for research before RL can be widely used for BES control application. In many still not investigated application scenarios, the challenge will be to exploit the adaptivity and control performance of RL algorithms for widespread use, without having to repeatedly undertake the same high degree of training and engineering effort for each individual energy system. Since, after training, RL algorithms map system states directly to optimal actions without much computing power requirements, there is great potential for the implementation directly with the building automation infrastructure. However, the plug-and-play capabilities of the algorithms to solve BES control problems (when used as published and just connected to sensors) are not yet sufficient to actually reduce the engineering effort. Also, the issue of selecting the optimal hyper-parameters, which have a significant impact on the final controller performance and the necessary training times, is not considered at all in most studies, or only through sensitivity analyzes. In addition, there is a need for further research on the transferability of trained algorithms to other control problems (transfer learning) and on the question of how monitoring data can be used appropriately for training the algorithms. To support the scientific work on these challenges, this dissertation defines a workflow for the development of RL-based BES control systems and presents three application examples in which various current issues in the scientific literature are addressed using RL-based BES control for applications inspired by current BES related research projects. Discrete and continuous algorithms for load shifting tasks are compared, different monitoring data-based RL controller training strategies are evaluated, innovative state-space descriptions are investigated and the transferability of trained controllers to new systems is explored. The question of the selection of the optimal hyper-parameters is considered in all case studies, which is addressed by a method introduced within the workflow chapter (chapter 3).

3 Reinforcement Learning-supported building energy system automation design

After deriving the general motivation and the potential of RL for BES from the available scientific literature as well as introducing the state-of-the-art of the most promising RL algorithms, in the last chapter; this chapter presents the methods and tools used in this work. This includes the implementations, design principles, frameworks, and experiment workflows. Furthermore, an efficient method for the design and parameterization of the algorithms is introduced. The chapter concludes with a summary section of a comparison paper that has been published collaboratively with colleges from the institute, in which a RL algorithm (designed following the proposed workflow) has been compared, evaluated, and discussed against other novel BES control approaches. Thus, a complete workflow is presented (and benchmarked) with which RL-supported control of BES can be developed.

First, section 3.1 gives a high-level overview of the general workflow and the training and interaction variations for RL-supported control design. All aspects are briefly introduced before providing a more in-depth discussion in the fallowing sections. Section 3.2 introduces the most important considerations to be taken into account when formulating the control task (or in RL terminology: the MDP). This is followed by a discussion of the criteria that should be considered when selecting a suitable algorithm and a training strategy for the MDP, in sections 3.3 and 3.4 respectively. Section 3.5 subsequently focuses on the design of safe interaction between the RL algorithm and real BES. Sections 3.6 and 3.7 deal with the concrete implementation of the RL-supported BES control and the optimization of the algorithm-specific hyper-parameters in the programming language Python. Finally, in section 3.8 the workflow is benchmarked against other novel BES control approaches. The presented workflow is based on RL-specific handbooks [Francois-Lavet et al., 2018; Sutton and Barto, 2018], best practices from RL-related publications (with a focus on BES application and a broader focus on RL), and the authors own experiences through working with BES and RL algorithms. A more detailed discussion, addressing the experiences made through the application described in chapter 4, is provided in chapter 5.

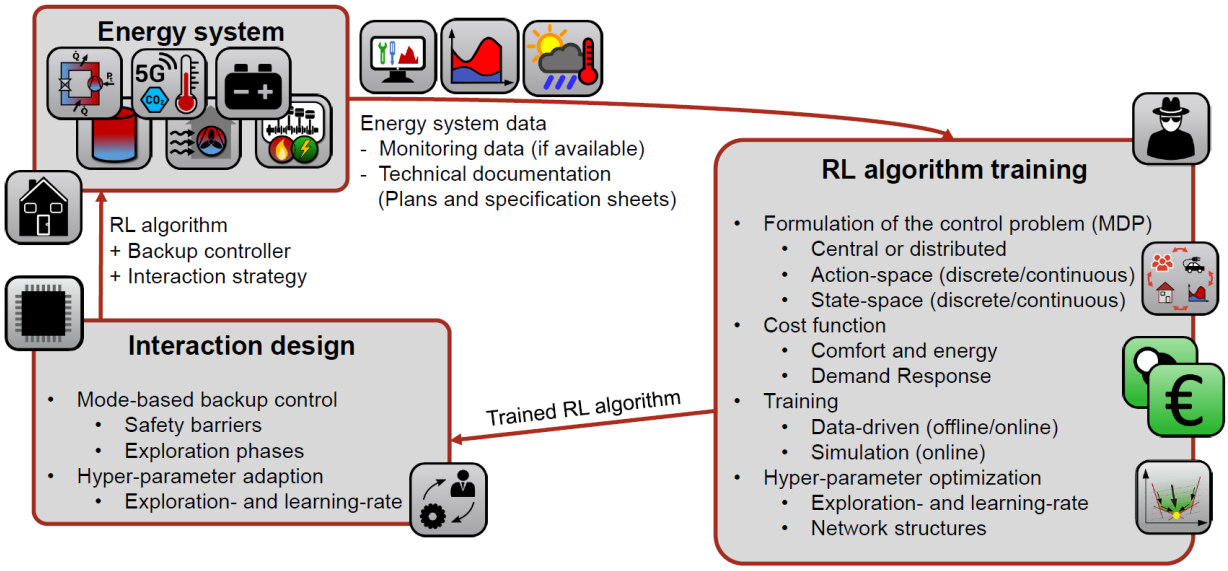

Figure 3.1: Basic workflow of RL-supported BES automation development.

3.1 Reinforcement Learning controller for building energy system automation - overview

Figure 3.1 shows the steps necessary for the design of a tailored, RL-supported BES automation system. As with all machine-learning algorithms, the problem formulation and the training of the algorithm are the most critical elements. The behavior of the algorithm, after training, is significantly influenced by the type of training and the input data used.

When formulating the control problem (MDP formulation), it is important to ensure that the state-space contains all the features needed to predict the future behavior of the MDP and the associated reward function. Otherwise, they are considered partly observable MDPs, and the control problem becomes much more complicated, which also limits the applicable algorithms drastically. For continuous control actions, the choice of algorithms is again limited. Traditionally, the algorithms for continuous actions also tended to demand more training data until they learn an optimal policy. This changed with the SAC algorithm, which (compared to the DQN with all its published extensions) generated similarly high, and in some cases even higher, convergence rates in the associated literature. At the same time, the convergence speed is especially dependent on the dimension of the problem, therefore the state-space should be as compact as possible (while including all necessary information). The action-space should also be compact, but at the same time it has to provide the necessary interactions to control the system. In other words, action- and state-space must satisfy the two most important RL design criteria: the future of the MDP must depend exclusively on its current state and the evolution of the reward function must be well predictable by the current state in combination with the performed action. This should be fulfilled, while the action- and state-space should

contain as little unnecessary or redundant information as possible, because this must be filtered by the algorithm through additional computational operations to identify the features that are essential for the control problem.

The training strategy depends crucially on the available information about the BES to be controlled. The effort to build a customized, simulation-based training environment based on plans and technical documentation is not economical for most individual BES. Moreover, this overhead limits the advantage of RL over white-box MPC, since such a model could also be used directly in a MPC application. On the other hand, training by means of an extensive simulation or even a real system in a test bench can be a promising option for a component manufacturer. For products which are sold many times, it is worth the effort, since the trained algorithm can represent an optimal and adaptive control for the individual system. If a RL algorithm should be trained for an existing system, the training with monitoring data can be beneficial; either by direct offline training of the algorithm, in which case the problem of over-adaptation to the observed policy must be taken into account; or via the intermediate step of learning a data-driven training environment based on the monitoring data. In this approach, the algorithm can then use the available exploration strategies to learn the policy during interactions with the data-driven model representing the system dynamics. If the training is performed via a simulation or a data-driven training environment, the hyper-prameter (e.g. learning rate, exploration rate, or the discount factor) can be optimized for the MDP. Since the training of RL algorithms is very time-consuming, efficient methods are needed to find the optimum. In section 3.7, the selection process of the proposed Bayesian hyper-parameter search is presented.

RL algorithms are statistical methods that lead to optimized system control after a large number of interactions. In order to integrate them into robust and secure BES automation, a concept for the actual operation is required in addition to the pure design and training of the algorithm. The concept must include safety barriers in the form of threshold monitoring, above which the actions of the algorithm are overwritten by a conventional, robust system control. In order to adapt the algorithm to changing systems, exploration phases can be defined in which random actions in a defined range increase the size of the system training dataset, for example, while there are no occupants in a building. An advanced concept could also include a targeted manipulation of exploration and learning rates during operation; for example, after a change in system dynamics, both could be adjusted to adapt to the new dynamics faster.

3.2 Formulation of the Markov Decision Process

The MDP represents the problem formulation to be solved by the RL algorithm. Even the most novel and best-performing algorithm from literature cannot solve a poorly formulated

problem. In this context, poorly refers to: that the algorithm does not have all the information that determine the future of the MDP, its actions do not or only slightly affect the course of the rewards, or the reward function is not formulated in such a way that the desired policy is rewarded. Finally, also the step-size (how often the algorithm interacts with the environment) and the prediction horizon (how many steps in the future the algorithm takes into account) are crucial. The algorithm must be able to interact with the MDP sufficiently often to control it stable. At the same time, the training is stabilized by the observability of the effect of an action in the state before the next action is executed. Predictions of the boundary conditions (whose horizon match the flexible elements in the system and the possible actions) complete the state-space and make it possible not only to learn a statistical best practice, but actually to plan ahead.

The state thus defines what numerical inputs the algorithm receives in each time-step from the environment, or the technical system with which it interacts. In BES applications, these can be, for example, storage states, temperature setpoints, actual temperatures, outdoor temperatures, solar radiation, electricity prices, and other internal variables from the technical systems such as volume flows, pressures, or rotation speeds. At the same time, a particular strength of RL is that not all internal quantities from the technical systems have to appear in the state-space, but just exactly those that make the future of the reward function and the dynamic next states of the MDP predictable with respect to the selected actions. A simple example: If a RL algorithm is used to control the air handling unit (AHU) of a building, the AHU itself is a component of the technical system, which (assuming no malfunction) always shows the same dynamics. However, the boundary conditions from the environment (temperatures and solar radiation), the presence of occupants, and the use of appliances resulting in internal loads are different. Since the dynamic of the AHU remains the same, the information from the air setpoint temperatures, actual air temperature, presence sensors (or time signals such as day of the week or hour of the day), the status of mechanical shading, and the boundary conditions from the environment are sufficient to predict the effects of an action on the evolution of the reward function (for example, maintaining a temperature band around the setpoint), while many internal measurements from the AHU do not represent additional information for the RL algorithm. Internal measurements from the AHU are thus no longer required for the control of the AHU alone, while the sensors in the target system (temperature in the heated rooms) become more important. If, in addition to the target temperature, the energy consumption is also to be optimized, the required energy of the AHU in certain operation states is needed as a data point, although here as well, no sensors from the internal control of the AHU are required; the energy consumption of the AHU as a function of its influencing ambient conditions is sufficient, as long as no malfunction strongly alters its dynamic behavior. This is one feature of RL that is particularly suitable for saving sensors in BES. At the same time, as already pointed out in the discussion of the literature in subsection 2.7.2, the consideration of historical sensor values is of high importance for the RL algorithm

to learn the dynamics of the system to be controlled. This is also addressed again in the demonstration chapter 4. Since heating and cooling effects on the system can be learned not only from the current sensor value but from the history of the sensor values over time, at least one historical value (or the gradient of the measurement) must be added for short-term effects, or a series of historical values for long-term effects. Also the historical actions of the algorithm should be taken into account. Since RL algorithms do not necessarily have a memory, to process actions from the last time-steps within the training, this is important for technical systems. The algorithm can learn from this information, in the state-space the difference of the influence from its own actions to the influence of the boundary conditions on the environment. Without this information, the training can be unstable and, in particular, drastic changes in the actions can hardly be avoided.

In addition to the definition of the state-vector (what the algorithm receives from the MDP in each time-step), it is important to define the temporal component. This involves the time-step width (how much time passes between two actions) as well as the prediction horizon (how far in the future does the algorithm plan). Both aspects must match the possible actions that the algorithm can execute. For example, if the algorithm can load a storage with a defined or limited thermal power, the step-size and prediction horizons should be chosen in such a way that the algorithm can load and unload sufficiently often (in the sense of a planning sequence of actions) in the prediction horizon to take advantage of the flexibility and at the same time react to drastic changes at the end of the prediction horizon. The first aspect, the time-step width, is furthermore influenced by a RL specific feature. Namely, that the training and the interaction with the MDP can be significantly stabilized if the effect of an action is already observable in the next time-step. While RL algorithms can also solve MDPs where rewards are rarely returned, the MDP is more straightforward to solve by the algorithm if the next state already represents the effect of the last action. As a basis here, the time constants (especially the delay between an action and the observable effect of the action in the state-vector) of the system to be controlled must be considered. Time-step widths from the literature for BES control applications are: 1 minute for actuator level control tasks, 5 to 15 minutes for BES-level energy management, and 15 minutes to hours, or even days, for district-level energy management. An additional aspect concerns the resolution of available data APIs for the boundary conditions such as weather services or electricity price predictions, which are mostly provided in a resolution of 15 minutes. Another important aspect is that the prediction horizon must essentially fit the control task. If a prediction of one day is made available to the algorithm in an hourly resolution, the state-vector already increases by 24 entries. Considering that in most BES applications this already exceeds the number of available sensors in the system, it is clear how each additional day increases the problem complexity and thus the training time required. Twenty-four hours have therefore become widely used in BES applications. However, if the flexible elements in the system have sufficient capacities to allow for load shifting over several days, longer prediction periods can

be efficiently included in the form of coarser resolutions or statistical features of the prediction time-series.

In addition to the interaction rate and the state-space, the action-space has to be defined. The action-space can be continuous or discrete, as described in the sections 2.5 and 2.6. Continuous action-spaces are particularly suitable for control tasks where the algorithm has to reach a setpoint through its actions. Discrete action-space algorithms, on the other hand, are well suited for energy management tasks or tasks where the algorithm has to select from a pre-defined number of system states. As with the state-space, the goal should be to keep the action-space as compact as possible. Five possible actions in a discrete control task already imply that the algorithm has to approach all possible states of the MDP five times during training in order to learn all following system responses from the MDP. The same is applicable to the dimension of a continuously operating algorithm. For example, a pump speed and a valve position to be combined and continuously controlled by an algorithm corresponds to an action-space dimension of two. A key advantage here, especially for actuator level control tasks is that continuous control algorithms can learn the relation between the gradient of the reward function and the gradient of the action-space and, unlike discrete algorithms, interpolate not only between states but also between actions. For energy management with a limited number of decisions in each time-step, discrete algorithms should nevertheless be chosen, since continuous interventions are usually not provided by the BES automation (except setpoint optimization which can be designed as a continuous energy management task).

The last element of the MDP to be defined is the reward function. In policy-based algorithms, unlike value-based algorithms, rewards are only evaluated at the end of training epochs, which leads to more training intervals with suboptimal actions and is mostly impractical in the BES domain. Value-based algorithms, on the other hand, calculate the reward for the current and subsequent time-steps. Historization and the calculation of cumulative rewards are processed internally in the algorithm and therefore do not need to be specified in the reward function. The reward function should encode the desired policy that the algorithm should show after training, with respect to state-space and action-space. For example, if there is a trade-off between energy consumption and user comfort, both aspects must be provided in the reward function, for instance in the form of the room temperature and the current energy demand. Appropriate weighting factors have to be used to balance the two elements of the reward function. In order to smooth the actions, additional quantities can be implemented, which slightly penalize action changes or take temperature gradients into account. For training, it is advantageous if there are not infrequent large rewards but rather if the gradient of the reward function increases over many small rewards up to the highest rewards, which is easier to learn. The trade-off between temporarily lower rewards for loading a storage at a favorable time against unloading it later to avoid even lower rewards is then the task that the algorithm learns.

3.3 Algorithm selection

As outlined in section 2.3, there is a wide range of available RL algorithms that can be classified into different families. This work focuses on the model-free RL algorithms, as discussed in chapter 2. While the model-based algorithms come with the strengths and weaknesses of data-driven MPC, in the model-free algorithms lies a particular potential for optimal BES control with low demands on the available computing power. The most important constraints for selecting the model-free RL algorithm for the BES to be controlled are the following:

- Is the control problem discrete or continuous according to the definitions provided in the sections 2.5 and 2.6?
- Does efficient handling of training data matter? Should the algorithm be trained offline or is a simulation available as a training environment?
- Can the control quality of the algorithm be determined for each time-step or can it only be evaluated at the end of episodes when complete control trajectories are available?

Overall, for any RL application, care should be taken to select the most current algorithm for which stable implementations are available according to the respective selection criteria. RL research is a very active field and, as it has been elaborated in [Wang and Hong, 2020], many publications in the field use older algorithms for their studies and therefore do not take advantage of recent RL research.

For discrete control tasks via Q-Learning, the most common algorithm is the DQN with the extensions published in recent years. Through various design paradigms such as the use of a replay buffer, a target network, or prioritized training sample selection, the stability, performance, and sample efficiency has been improved and the latest implementations outperform the original implementations in all categories by several orders of magnitude.

For continuous control tasks where a stochastic policy is explicable, the SAC algorithm should be chosen, which (like its predecessor, the DDPG) is a hybrid of Q-Learning and policy optimization. However, while with the DDPG (due to its comparatively simple exploration strategy) good policies are found only after very intensive training times and long phases with suboptimal policies, this subfamily of algorithms was significantly improved by extending the value function by an entropy term that determines the exploration, which led to the proposed SAC algorithm becoming the state-of-the-art in this subfamily.

Another algorithm that can be considered for selected problems is the Proximal Policy Optimization (PPO) algorithm. This is a purely policy optimization-based algorithm, which means that it cannot be trained with historical data, unlike SAC and DQN. Another disadvantage is the comparatively inefficient data usage, since each new policy has to be tried on the real system or a simulation. Thus, the PPO (as discussed in chapter 2 on policy-based algorithms) is only of interest if a high quality training environment in the form of a simulator

is available and the system to be controlled is not subject to large changes. Advantages of the PPO are its comparatively stable convergence towards the optimal policy, its parallelization capability during training in newer implementations, and its usability for discrete as well as for continuous control problems.

3.4 Training strategies

As with all machine-learning methods, the training of the algorithms is one of the major factors influencing the functional relationship which is later represented by it. However, unlike supervised learning, the training dataset in RL is not static, but is generated with a dynamic training environment during the interaction with the algorithm. An exception is offline training of value-based RL algorithms. Here, algorithms such as SAC or DQN are trained with static datasets, for example using monitoring data of a system.

All other training methods require a dynamic training environment. There are two options for creating such a training environment. On the one hand, a physics-based simulation environment of the system to be controlled can be created. Here, it should be considered that the effort has to be weighed against MPC. If a simulation model of a high quality is available, it can be used directly within a MPC application. The advantage of RL is then reduced to the lower computing power required during operation, the inherent adaptivity to changing system behavior, and the performant computing of large, stochastic states. However, the latter can still be of interest when a model is available but the algorithm is to be used under highly varying boundary conditions in practice.

The second way to generate a training environment is to learn a data-driven training environment from operational data. This method like offline training, is also based on monitoring data, but it increases the training potential significantly, since different policies can be tested by the algorithm on the learned models. Like in a simulation environment, not only is an observed policy learned, but the policy is also optimized in interaction with the dynamic environment. However, there are also some critical aspects that have to be considered. On the one hand, the dataset must cover as large a range of operating states as possible, which the algorithm will also reach in later operation. Furthermore, the data-driven training environment must be tested. If it shows physically implausible behavior in some operating states, a policy can be learned that, for example, exploits sensor noise or other model properties to maximize the reward function rather than solving the physical control problem. Nevertheless, this training option holds promise when local computing power does not meet the requirements for MPC, sufficient monitoring data is available, and later adaptation in operation is desired.

The aforementioned training through monitoring data (offline learning) is a promising method if a large amount of monitoring data for a certain system is available, at best controlled with

different policies. From such a dataset, a good policy can be learned by offline training with a RL algorithm, which can serve as a starting point for later use with the real system. However, if only monitoring data for a certain system controlled with a certain policy are available, there is the risk of over-fitting to a suboptimal policy; this effect can only be compensated by extensive exploration on the real system and thus reduces the potential for offline training.

The last variant is the training on the real system. If the MDP formulation is compact and the relationships in the system are simple, training on the real system can be considered. Here, it is important to have a comprehensive interaction strategy, including a backup RBC, since random actions are chosen during the exploration and this may cause comfort losses and inefficient operation. In contrast, if a policy is learned for a specific component, training in a testbench can also be considered. In the process, different boundary conditions can be learned in a targeted manner, thus ensuring free exploration of the algorithm. After training, an algorithm trained in this way is a lightweight software component that can be delivered together with the BES; furthermore, this algorithm has learned the optimal system behavior, and quickly adapts to deviating environmental conditions.

3.5 Interaction design

Another important issue to be addressed when using RL in interaction with real BES is the one of safe interaction. A concept is needed with which the promising properties of RL algorithms, such as their tendency towards the optimal control policy, their adaptivity, and their low hardware requirements, can be exploited without their stochastic action selection during training negatively affecting BES operation. As described in the last section, pre-trained algorithms are recommended for most applications, but even if the algorithm has been pre-trained extensively, suboptimal actions cannot be completely excluded.

In principle, there are two possible variants to address this aspect in a RL-based BES automation. The first variant is to allow the algorithm to select only from pre-defined operation states (for example during the course of energy management application) in which the supply security of the consumer systems is not affected by the actions but only the BES mode is affected. The second variant, if the algorithm also imposes actions that affect the quality of supply, such as comfort (for example, measured via the deviation of a room temperature from a setpoint), is to implement a state-of-the-art rule-based automation as backup control. As illustrated in figure 2.1, after training, the algorithm can interact with the real system until a defined performance criterion is violated. In this case, the state-of-the-art automation implemented using RBC and conventional feedback control takes over. Using room control as an example, a temperature limit could be implemented before the actual comfort violating temperature limit is reached, which is learned via high penalty signals as a barrier for the algorithm and activates the takeover by a conventional PID controller. If the BES is then

again in a stable and safe operation, the RL automation can be reactivated. The data generated during the control by the backup automation can be used again for the offline training if value-based algorithms (such as DQN or SAC) are used. However, to avoid the risk of over-fitting to the conventional automation, the takeover by the backup automation should also be accompanied by high penalties, which over time will lead to the avoidance of these operation states. Over the duration of BES operation, the conventional backup automation will then be activated less and less frequently and eventually become redundant.

In addition to implementing safety mechanisms, there are further aspects to consider if RL algorithms are connected to data from real-world BES. Sensor raw data can contain noise on the one hand and sensors can fail on the other hand. There are two ways to cope with sensor failures. On the one hand, sensors have to be monitored extensively to detect a failure early, to replace faulty sensors early, and to avoid an adaptation of the RL algorithm to the no-longer-deterministic system behavior. At the same time, regular backups of the algorithm must be stored frequently in order to avoid that the algorithm has to be trained again from scratch after an operation phase with faulty sensors. Rather, it should always be possible to backtrack and revert the training status of the algorithm to a certain (stable) algorithm state in the application history.

If a safe interaction with the real system and a strategy for the failure of sensors are given, the adaptation of the hyper-parameters of the algorithm can be considered during operation. While RL algorithms have an inherent adaptivity that continuously learns slowly changing system dynamics (such as valve characteristics) or changing building usage, this process can be positively influenced by choosing appropriate hyper-parameters during operation. For major changes in the BES, the learning rate and exploration factor parameters can be adjusted to accelerate the adaptation of RL-based building automation to the new system behavior.

3.6 Implementation

There are numerous frameworks that provide implemented RL algorithms. Python is by far the most widely used programming language in this field and the neural networks used in the context of Deep RL are typically implemented using the highly specialized libraries, Tensorflow (Google) [Agarwal et al., 2015] or PyTorch (Facebook) [Paszke et al., 2019]. While it is indeed possible to implement RL algorithms directly from the respective publications, frameworks are available here that make the published algorithms available via implemented and tested APIs. Some of the most popular are: Tensorflow Agents, Google Dopamine, Keras-RL, and OpenAI (Baselines and Spinning Up). Which framework should be used depends mainly on how active the community is, how well the algorithms are tested, and finally, how intuitive the usability is. OpenAI Baseline sets a standard here with its community. Since the OpenAI company also provides the OpenAI Gym collection of training environments and the GitHub

projects made available constantly incorporate the latest innovations from the field into the respective algorithms, they set a standard also in the industry. However, the implementations are often not intuitive and can only be used by specialized developers with a high level of effort. This is where Stable-Baselines can be useful. As a fork of OpenAI Baselines, Stable-Baselines provides further developed, improved, and tested RL algorithms based on OpenAI Baselines. With an API inspired by the comparatively user-friendly Scikit-Learn Python machine-learning library, the usability of the algorithms are high and RL experiments can be implemented in a user-friendly and comprehensible way. Stable-Baselines also provides methods that make the training concepts described in section 3.4, such as hyper-parameter adaptation, and offline training available in a comparatively user-friendly way. Therefore, in the course of this work, implementations from this framework are used.

For the modeling of thermal systems, the open-source Modelica library AixLib [Müller et al., 2016] is used. The models are integrated and simulated as functional mock-up units (FMU) [Blockwitz et al., 2012]. Interfaces (action-, state-space, and rewards) are implemented according to the standards defined by the OpenAI Gym [Brockman et al., 2016]. Figure 3.2 shows the schematic structure of the RL algorithm when interacting with a FMU. In each iteration, a FMU is simulated for one time-step and a state-vector is generated. Both the state-vector and the reward signal are processed by the algorithm, then the control signal is forwarded to the FMU. Subsequently, a new state-vector is generated then a new iteration begins. Additionally, external state variables, like monitoring data, are passed to the FMU. To obtain standardized and RL-processable interfaces, all control actions and state-vectors are normalized to the range from -1 to 1.

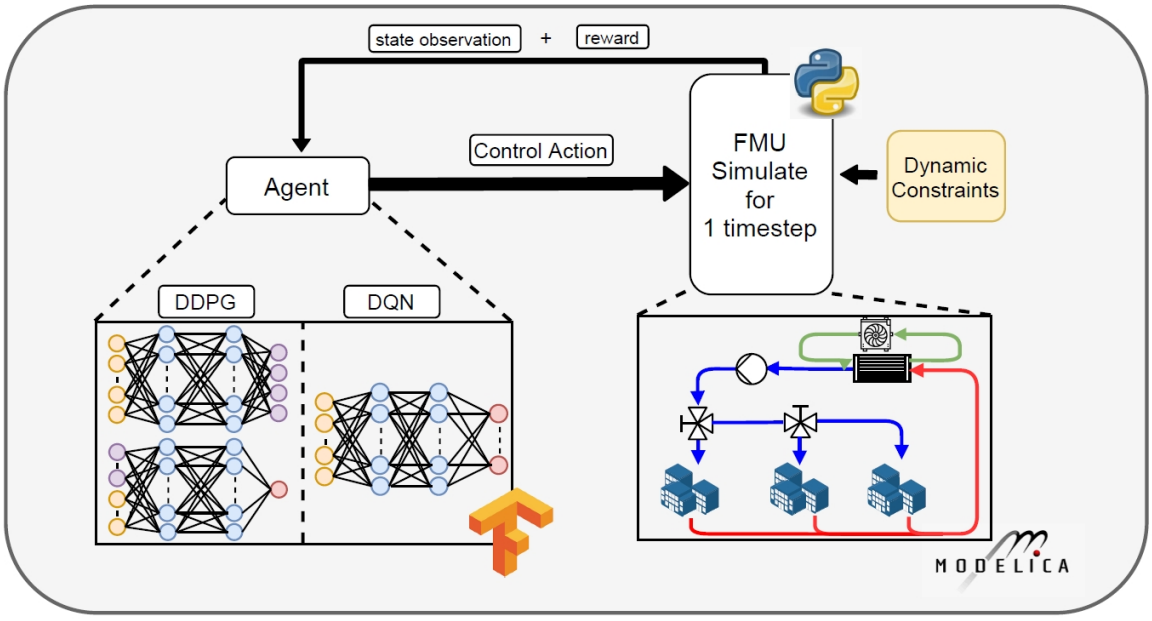

Figure 3.2: Exemplary illustration of a FMU-based training framework with an interface for continuous and discrete controlling RL algorithms.

For data-driven modeling and the implementation of data-driven training environments, Scikit-Learn is used. As the most successful library for data-driven models, Scikit-Learn provides classic machine-learning algorithms from the areas of supervised learning, unsupervised learning, and regression with comprehensible implementations and supported by an active community. Using Scikit-Learn, machine-learning algorithms can be applied to individual problems, tested, and graphically evaluated with a few lines of code. The Scikit-Learn library also provides training routines and evaluation metrics. Therefore, in the course of this work, implementations from this library are used.

3.7 Bayesian hyper-parameter optimization

The configuration of the hyper-parameters significantly influences the convergence speed, as well as the final policy [Liessner et al., 2019]. Finding the optimal set of hyper-parameters is not trivial. Setting the hyper-parameters manually is time-consuming and error-prone even for domain experts [Chollet, 2018]. On the other hand, unlike with simple supervised learning tasks, such as classification or regression with lower complexity, it is not feasible to try all combinations of hyper-parameters. This is not practical for RL because of the comparatively long training times. Moreover, the convergence of Deep RL is not guaranteed even for well-formulated MDPs and depends largely on the choice of hyper-parameters [Liessner et al., 2019]. A well-chosen set of hyper-parameters should quickly lead to a policy that results in high rewards for a given MDP.

In many publications, different methods to determine the optimal hyper-parameters have been proposed and compared. The authors of [Putatunda and Rama, 2018] compare Bayesian hyper-parameter search with random search, and grid search for XGBoost classification tasks on six real-world datasets. They find that Bayesian hyper-parameter search yields the best results when speed and accuracy are considered equally. In a comparable study the authors of [Shekhar et al., 2021] highlight the performance of Bayesian search for the optimization of hyper-parameters of neural networks. In a comprehensive study, the authors of [Bischl et al., 2023] compare a variety of different hyper-parameter search methods for different use cases. In the analysis of accuracy, efficiency, and other practical aspects of applicability, a recommendation for the use of model-based search methods such as Bayesian search is given for deep-learning and RL. In particular, the computationally expensive training (compared to simpler machine-learning applications) is given here as the mayor reason.

In the following, the most common methods for hyper-parameter search from the literature are analyzed and discussed for their applicability to RL. Figure 3.3 shows a qualitative evaluation of the procedures according to multiple aspects: usability, computational cost, data efficiency, robustness against local optima, and the expert knowledge required for the application.

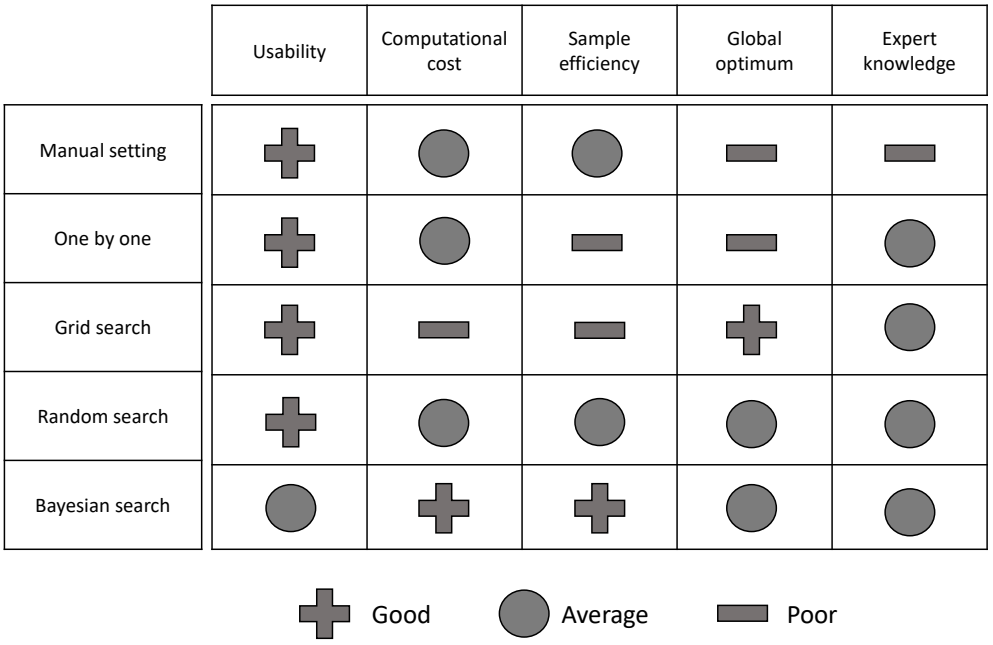

Figure 3.3: Strengths and weaknesses of different methods for hyper-parameter optimization.

Manual setting of the hyper-parameters does not lead to an optimal selection in most cases, since it is error-prone and time-consuming even for experts. The iterative procedure here is not practical for the long training times of RL and is also not well-suited to address the interdependencies of hyper-parameters. Nevertheless, this method is still one of the most widespread in all areas of machine-learning [Chollet, 2018; Liessner et al., 2019], which is due to the very individual and therefore diverse procedure also problematic from a reproducibility perspective [Bergstra and Bengio, 2012]. The procedure to automate the manual setting of hyper-parameters, referred to as "one by one", comes with similar difficulties. Although the required expert knowledge can be reduced slightly, the hyper-parameters are still optimized without considering their interdependencies.

These methods are contrasted by the grid search method, a brute-force procedure in which the optimal combination of hyper-parameters is determined by trying all possible combinations [Liessner et al., 2019; Pedregosa et al., 2011]. Although this method is the only one that guarantees finding the global optimum, it is highly inefficient. It iterates through a high number of combinations without major improvements to the performance [Bergstra and Bengio, 2012]. Thus, grid search is not suitable even for many supervised learning tasks and is not applicable for performing RL experiments in an adequate time.

The random search method, in which the hyper-parameters are selected randomly, is often much more efficient. Although the method is based on luck, it has been shown that comparably good combinations as with the grid search method can be found usually with significantly fewer iterations. [Bergstra and Bengio, 2012].

According to recent literature, the method considered as the most promising for RL control problems is the model-based method called Bayesian hyper-parameter search. Here, the idea is to create a simple statistical model (Tree Parzen Estimator for example) of the hyper-parameter space and to select the next combination to be tested based on this model [Barsce et al., 2017; Bergstra et al., 2015; Liessner et al., 2019]. In other words, during the Bayesian hyper-parameter search process, a statistical surrogate model is trained to predict the performance of a hyper-parameter combination. This model is much less expensive to compute compared to the objective function (the performance of the trained RL algorithm) and serves as a basis for the selection of the next hyper-parameter combination to be tested. This is a very important feature when it comes to the applicability of hyper-parameter optimization methods to RL experiments. The method represents the best trade-off between applicability and reliability and is therefore recommended and used in this work.

3.8 Performance comparison and discussion of Reinforcement Learning against other novel building energy system operation optimization methods

This section summarizes a collaborative paper, which has been published in the course of this work, in which RL (represented by a SAC algorithm) has been compared against other novel control approaches [Stoffel et al., 2023]. The previously described workflow for the design of RL-supported BES automation was applied here as well. The approaches against which the SAC algorithm has been compared were: White-box MPC, Adaptive Grey-box MPC, Adaptive Black-box MPC, and Approximate MPC. With these methods, more and more relationships of the system are learned by the algorithms for the control. The SAC algorithm is in parallel with Approximate MPC in this sense, because in both cases, state observations are directly mapped to actions. As a benchmark problem, the air conditioning of a thermal zone was chosen, which includes thermal inertia, internal thermal loads (stochastically present people and appliances), and other disturbance variables in the form of outdoor temperature and solar radiation. A carefully implemented rule-based strategy was selected as the baseline for comparison. The possible continuous actions were the setpoint temperature of the water circuit of a HVAC system and the thermal power to be supplied to a concrete core activation system. Thus, the algorithms had to find a trade-off between a highly sluggish but persistent heating system and a highly responsive but more energy-intensive heating system. For this purpose, weather forecasts and time programs of the setpoint indoor temperatures were provided. The performance indicators chosen were the Kelvin hours of discomfort over

one year and the total heating energy required. In addition, softer criteria, such as the computing power required to calculate the setpoints, were also taken into account, as well as an assessment of the approaches in the following categories: pre-operation data need, data quality requirements, model developing effort, MIMO (Multiple Input Multiple Output) handling, adaptability, IT requirements, know-how dependence, interpretability, transferability, and scalability.

As can be seen in figure 3.4, all methods can achieve significant energy savings compared to the rule-based (RB) approach with improved thermal comfort. The SAC algorithm achieved nearly the same energy savings as the MPC variants, but the thermal comfort achieved does not quite reach the level of the MPC controllers.

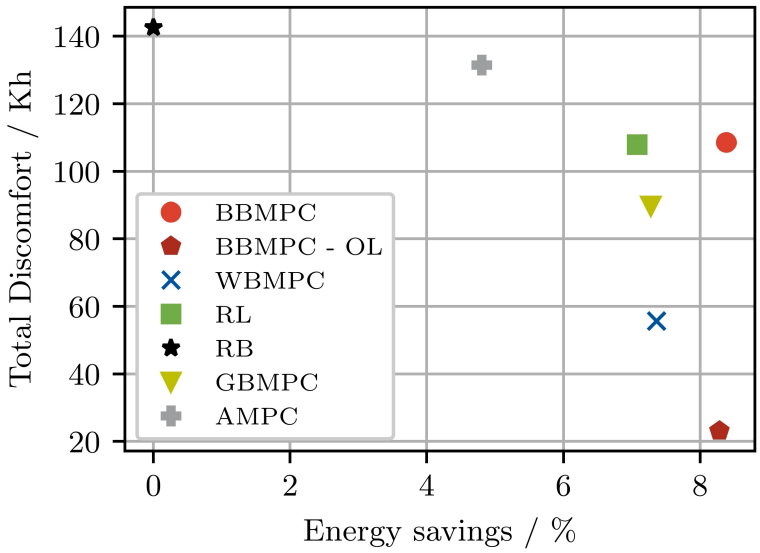

Figure 3.4: The performance of different novel control approaches for a benchmark problem. On the x-axis, the percentage energy savings compared to a rule-based (RB) control are plotted. On the y-axis, the annual Kelvin hours of discomfort. The methods studied are: White-box MPC (WBMPC), Grey-box MPC (GBMPC), Black-box MPC (BBMPC), Black-box MPC with online learning (BBMPC-OL), Approximate MPC (AMPC), and Reinforcement Learning (RL).

On the other hand, AMPC and RL require significantly less computation time for calculating the next setpoint, which is achieved by direct mapping of states to actions, whereas in the MPC approaches the optimizer iterates through several solution alternatives. For the other soft evaluation criteria, all methods have strengths and weaknesses and specific application scenarios for each method are conceivable. The SAC algorithm has been evaluated positively in the criteria: Transferability, MIMO handling, and adaptability. Transferability in this context refers to the transferability of trained algorithms to related problems, MIMO handling describes the ability to handle higher-dimensional state- and action-spaces, and adaptability refers to the ability of the approaches to adapt to changes in the system to be controlled or in

the boundary conditions. A slightly positive rating was given for the criteria scalability and model developing effort. Scalability, in contrast to transferability, describes the increase in the size (state-vector) of the problem with the same general system interdependencies, and model developing effort describes the effort required to design the control strategy by an expert. A neutral rating was given in the categories IT requirements and know-how dependence, which address the need to execute higher programming languages common to all methods as well as the need for expert knowledge in the respective methodology. A negative evaluation was given to the SAC algorithm in the categories: Pre-operation data need, data quality requirements, and interpretability. This results especially from the fact that the algorithm has to learn by random actions, which requires pre-training on a simulation or at least supported by monitoring data. Furthermore, erroneous data influence the algorithm more than other methods, since it requires filtering the disturbances in addition to learning the action-reaction relationships of the system. Finally, the interpretability of the trained algorithm is difficult, as it can only be achieved by additional stochastic analyses as described in [Kotevska et al., 2020].

In summary, the SAC algorithm and RL in general, like the other methods considered, have strengths and weaknesses that have advantages and disadvantages for the respective application scenarios. Its performance for the benchmark problem can compete with the other methods investigated.

4 Application

After the previous chapters introduced the motivation for the application of RL for BES (through relevant literature), and presented a workflow for the design of RL-supported BES automation systems, this chapter presents a selection of three case studies. The control problems are selected from systems which are the subject of German research projects funded by the Federal Ministry for Economic Affairs and Climate Action. The introduced algorithms and concepts are applied in the three case studies to different application scenarios, targeting different current research questions.

In the first case study, two different algorithms for load shifting in a simulated cooling supply system are applied. Here the focus is on three different aspects. Firstly, the out-of-the-box use of two state-of-the-art algorithms is evaluated; secondly, an algorithm which allows discrete control only (DQN) is compared with an algorithm that allows continuous control (DDPG); and thirdly, the applicability of RL algorithms to the problem of load shifting in a cooling network is investigated.

In the second case study, a further investigation is carried out based on a specific supply site of the same cooling network. The aspects that are considered here are DQN as an expert system for energy management of an ice storage system under dynamic constraints, Bayesian hyper-parameter search as a method for optimal selection of training hyper-parameters, and monitoring data-driven training of RL algorithms.

Finally, in the third case study, RL is used to control a real valve of an AHU. Here, the transferability of algorithms, pre-trained through a simulation, to a real system is investigated. Furthermore, aspects of the necessary data infrastructure and an innovative state-space description are addressed.

In addition to the problem formulation, all case studies have a detailed presentation and discussion of the results. Lessons learned for future work are derived. In addition, all insights from the application and the observed results are finally incorporated into a summarizing discussion of all results in the next chapter.

4.1 Case study one

In the first case study, two RL algorithms (DQN [Mnih et al., 2015] and DDPG [Lillicrap et al., 2015]) are investigated, with the aim to learn an improved operation policy for load

shifting in a simulated cooling network. Improvements have been proposed for both algorithms since their first publication, but the DDPG, in its applied form, can be understood as an extension of the DQN for continuous action-spaces. This allows, comparing the advantage of continuous control actions with discretized actions for the use case. In a subsequent section, it is also demonstrated how training and performance are improved by choosing optimal hyper-parameters and using state-of-the-art algorithms for the use case problem.

The investigated cooling supply system is part of a hydronic cooling network of a business and research facility located in Berlin, Germany. The campus comprises seven buildings, divided into three sites. The cooling demand results from process cooling, equipment cooling, and cooling energy for HVAC systems. The total amount of energy, purchased for one year, is about 1800 MWh.

In the considered operation mode, illustrated in figure 4.1, all locations are supplied by one central compression chiller ($\dot{Q}_{max} = 600$ kW), which is operated with power from the electric grid. The chiller feeds into a glycol-circuit, which transfers the cooling energy to the network via a heat exchanger. While the chiller receives the setpoint value for the temperature in the glycol-circuit as control input, the mass-flow can be controlled by a separate pump. Due to the relatively constant operation, the heat exchanger is modeled with an efficiency of $\eta = 0.9$. The supply of the consumers is carried out via a pipe network, which is operated with water as the medium. The dynamic pipe model from the AixLib [Müller et al., 2016] is used and parameterized as a steel pipe of $d = 200$ mm. The total length of the pipes (flow and return) is 644 m [Bschorer et al., 2019]. For the efficiency of the chiller (EER), the following ambient temperature (T_{amb}) dependency is derived from the monitoring data:

$$EER = 6 - \frac{T_{amb} + 2\,^{\circ}\mathrm{C}}{32\,^{\circ}\mathrm{C} + 2\,^{\circ}\mathrm{C}} \cdot 3 \cdot \frac{\dot{Q}_{chiller}}{\dot{Q}_{chiller\,max}} \tag{4.1}$$

This equation results, together with the cooling load ($\dot{Q}_{chiller}$), in the current electricity consumption of the chiller:

$$P_{el} = \frac{\dot{Q}_{chiller}}{EER} \tag{4.2}$$

The loads of the buildings are aggregated to the three consumer sites. The potential, which the algorithms can use for load shifting, results from the thermal mass of the concrete of the supplied buildings, as well as from the temperature tolerance of the linked manufacturing processes within the buildings. From monitoring data, the capacity for all three locations is estimated with a capacity of 500 kWh each, with $m \cdot c_p = 77\ \frac{kWh}{K}$, in the range from 5.5 °C to 12 °C. This range corresponds to the minimum and maximum temperature for the connected cooling appliances in the real system.

The dynamic price signal is determined by the historical data of the day-ahead electricity prices that are traded on the European Energy Exchange spot market, for the market area in Germany Luxembourg (DELU). The data is downloaded via the API of the European Network of Transmission System Operators for Electricity (short: ENTSO-E [Hirth et al., 2018]). It is assumed that a dynamic electricity price contract, which would be offered to an industrial customer, would have an offset, but would generally follow the dynamics observed on the spot market.

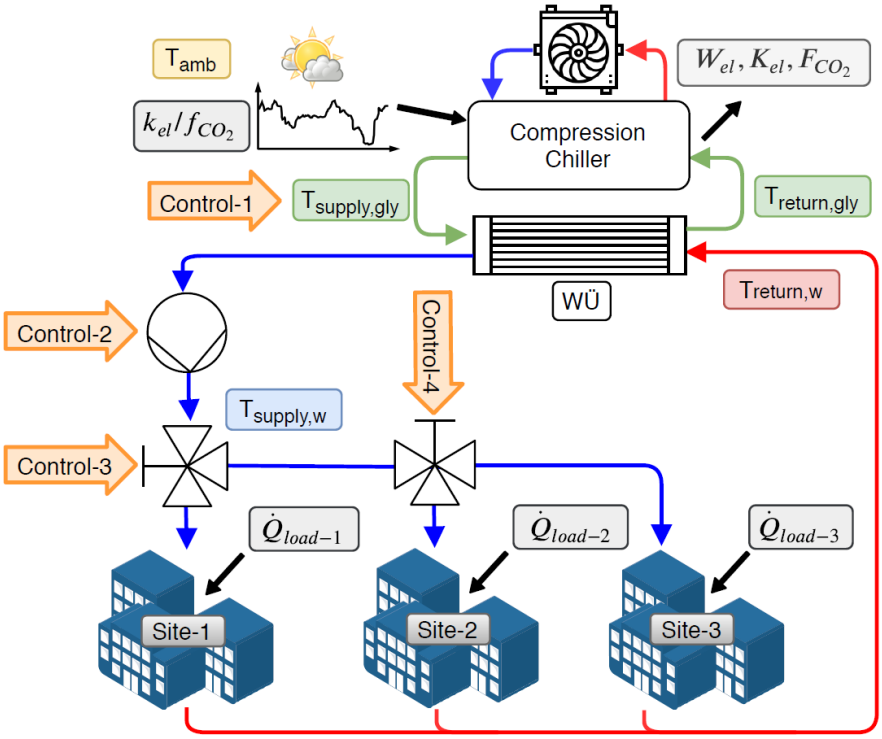

Figure 4.1: Schematic structure of the simulated energy system and data interfaces. Monitoring data of the cooling loads at the consumer sites are included together with the electricity price and the ambient temperature as exogenous inputs to the system.

4.1.1 The environment

The interaction between the algorithm and the environment for one week of operation is visualized in figure 4.2. The dotted line marks the current time-step. The interaction with the environment as well as the simulation of the cooling network is carried out with a 15-minute resolution. This reflects the available resolution of the price signal and weather data on the one hand and represents a feasible trade-off between the granularity of control and observability of the effects of an action in the next state. The state-vector is composed of 75 entries. The observation contains electricity prices (k_{el}) and weather data (T_{amb}) for the next 12 hours in a hourly resolution (first and second graph).

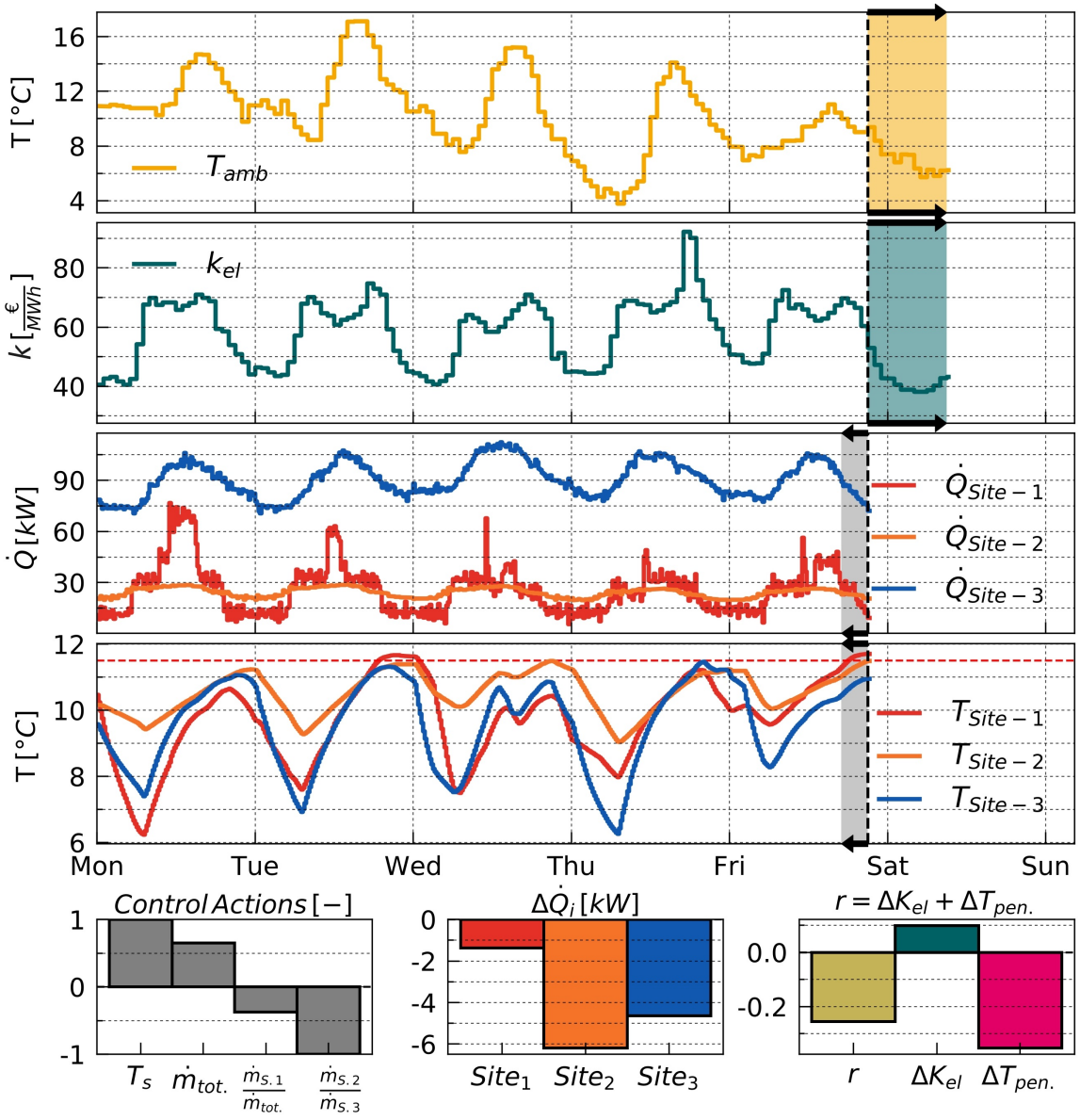

Figure 4.2: Visualization of the state-space, the action-space, and the reward signal. The algorithm receives at a given time-step (dotted line) current and historical consumer loads and temperatures, as well as price and ambient temperature forecasts.

According to the Markov Property, the future of a MDP must depend on its current state only. In this case, this is fulfilled by including the consumer loads ($\dot{Q}_{Site\text{-}i}\, i \in [1,2,3]$) and temperatures ($T_{Site\text{-}i}\, i \in [1,2,3]$) for the last 4 hours (third graph and fourth graph), also in a hourly resolution . Thus, the algorithm is able to learn, based on the gradients, whether the consumption or the temperature of the thermal mass is rising or falling and how fast. In addition, for the algorithm to learn relationships such as working hours and weekdays, the state-vector includes information about the weekday and the quarter of the day (one-hot-

encoded) as well as the last 4 control actions. The latter serves as an indication about own influence of the algorithm on the system dynamics.

The baseline for the system is the currently in the real system applied purely demand-oriented policy: current cooling loads are provided by the chiller, regardless of the electricity price and without the use of flexibility at the consumption sites. The reward (equation 4.3), includes one term for the energy costs (compared to baseline) and one penalty for temperature range violations. The cost term ($\Delta W_{el} \cdot k_{el}$) is calculated from the difference between the current consumer load ($\dot{Q}_{load}$) and the current supply load ($\dot{Q}_{supplied}$), divided by the current ambient temperature-related EER of the chiller, multiplied by the electricity price (k_{el}). This reflects the difference in electricity purchases compared to the direct load coverage and rewards the shift to periods with lower electricity prices. The penalty term ($\Delta T_{penalty}$) for deviations from the temperature limit at the consumer sites (11.5 °C) is designed to balance it with the cost term. The exponential function is used to allow small temperature deviations as a trade-off to high cost savings, whereas larger deviations are penalized over-proportionally.

$$\begin{aligned} reward &= \Delta W_{el} \cdot k_{el} + \Delta T_{penalty} \\ \Delta W_{el} &= \frac{\dot{Q}_{supply} - \dot{Q}_{load}}{EER} \\ \Delta T_{penalty} &= -\frac{1}{3} \cdot \sum_{i} exp(max(0, (11.5\,°\mathrm{C} - T_{Site.\text{-}i}))) \end{aligned} \tag{4.3}$$

Monitoring data of the cooling loads at the consumer sites and the ambient temperature are available for 87 weeks of operation. The data is split according to figure 4.3; after four weeks, one week is removed from the dataset for testing.

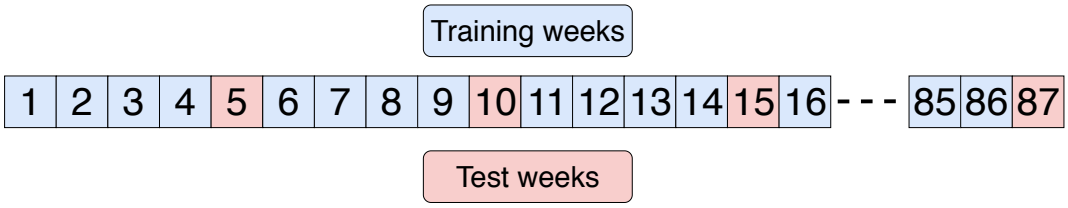

Figure 4.3: Division of the available weeks into training- and test-weeks.

Since the DQN is limited to discrete actions, the operation parameters of the chiller (temperature setpoint ($T_{supply,chiller}$) and mass-flow ($\dot{m}_{w,tot}$)) are devised in 6 operation modes, leading to 6 equidistant flow temperature levels in the cooling network:
$T_{net} \in [3.5, 4.5, 5.5, 6.5, 7.5, 8.5]$°C. Since the action-space is limited in size by this discretization, the valves at the consumer locations (figure 4.1) are controlled independently of the algorithm; the cooling energy is distributed to the consumers in proportion to their demand.

The DDPG on the other hand returns an action vector of 4 float numbers. The entries are summarized in table 4.1. The vector includes the parameters of the chiller ($T_{supply,chiller}$ and

$\dot{m}_{w,tot}$) as well as the position of the two valves ($k_{3WV\text{-}1}$ and $k_{3WV\text{-}2}$). The valves distribute the cooling water to the consumer sites and can be considered, due to the continuous action-space of the DDPG. $T_{supply,chiller}$ is controllable in the range from 2 °C to 7 °C, the water mass-flow ($\dot{m}_{w,tot}$) is calculated within the environment and results from the controllable target network temperature T_{net} in the range from 3.5 °C to 8.5 °C, and the two valves within the network can be opened between 0 % and 100 %. All control signals ($a_1 - a_4$) are scaled between between -1 and 1.

Table 4.1: List of the control signal ranges for the DDPG algorithm.

Action		Min		Max	
a_i	meaning	value	scaled	value	scaled
a_1	$T_{supply,chiller}$	2 °C	-1	7 °C	1
a_2	$\dot{m}_{w,tot} \rightarrow T_{net}$	3.5 °C	-1	8.5 °C	1
a_3	$k_{3WV\text{-}1}$	0	-1	1	1
a_4	$k_{3WV\text{-}2}$	0	-1	1	1

4.1.2 Results of case study one

This section presents the results of the first case study. The learning process is presented, the resulting control policies are compared, and finally the possible improvements through hyper-parameter optimization and the use of state-of-the art algorithms are shown.

Learning progress

Figure 4.4 shows the two averaged terms (ΔK_{el} and $\Delta T_{penalty}$) of the reward function for the 70 training episodes. Both algorithms were trained on a regular desktop computer (CPU: Intel Core i5- 8265U 1.60 GHz; RAM: 16 GB). The training of the DQN required $\sim$ 7 hours, and that of the DDPG $\sim$ 14 hours. Both algorithms show almost identical convergence behavior over 70 training episodes and the training processes for both are robust against the initialization of the neural networks. One episode is defined as one cycle through the training dataset. The exploration factor ε continuously decreases and asymptotically approaches 0. Both elements of the reward function increase and finally converge towards a threshold. At the end of the training, the flexibility use saves on average $\sim$ 40 € per week, which is about 14 % of the total operating costs. The average penalty received per week is $-5 < \Delta T_{penalty} < 0$, which corresponds to a deviation of the temperature of the three thermal masses of about 0.5 °C over a period of 3 hours.

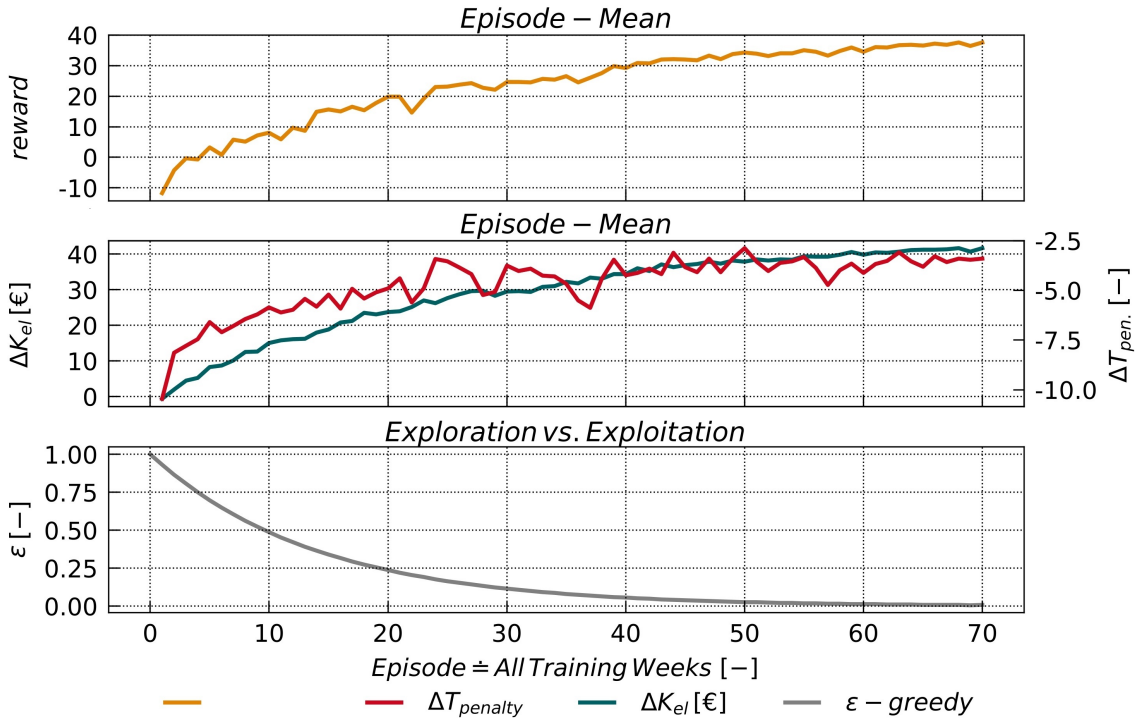

Figure 4.4: The average reward and penalty during the training of the DQN. The top plot shows the absolute averaged reward for the algorithm over 70 training cycles, the middle plot the evolution of the two parts of the reward function, the monetary cost term and the penalty term for temperature deviations from the set temperature, and the bottom plot, the asymptotically decreasing exploration factor.

Policy of the DQN

In figure 4.5, the learned operation policy for the investigated system, with boundary conditions of one week from the test dataset, is presented. The consumer loads and the cooling energy provided by the chiller are shown. With $T_{s,glykol}$ ($T_{supply,chiller}$) and $\dot{m}_{w,tot}$, the control actions of the algorithm are visualized. Below that, the time-series of the temperatures of the three thermal masses are added. The electricity price and the ambient temperature are also included. The accumulated operating costs over the operating time is shown below. The green line represents the costs resulting from the learned operation policy. The grey line represents the costs that would result from the baseline. By load shifting to times of low electricity prices and low ambient temperatures, the algorithm saves about 20 % of the operation costs for this week. It is also ensured that the temperature limit of 11.5 °C of the thermal masses is rarely violated and that all deviations are significantly below 1 °C ($\Delta T \leq 0.7$ °C), occuring only for a short period of a few hours. Large savings are achieved by maximum cooling of the thermal masses during a period with electricity prices $k_{el} \sim 0$ €/kWh.

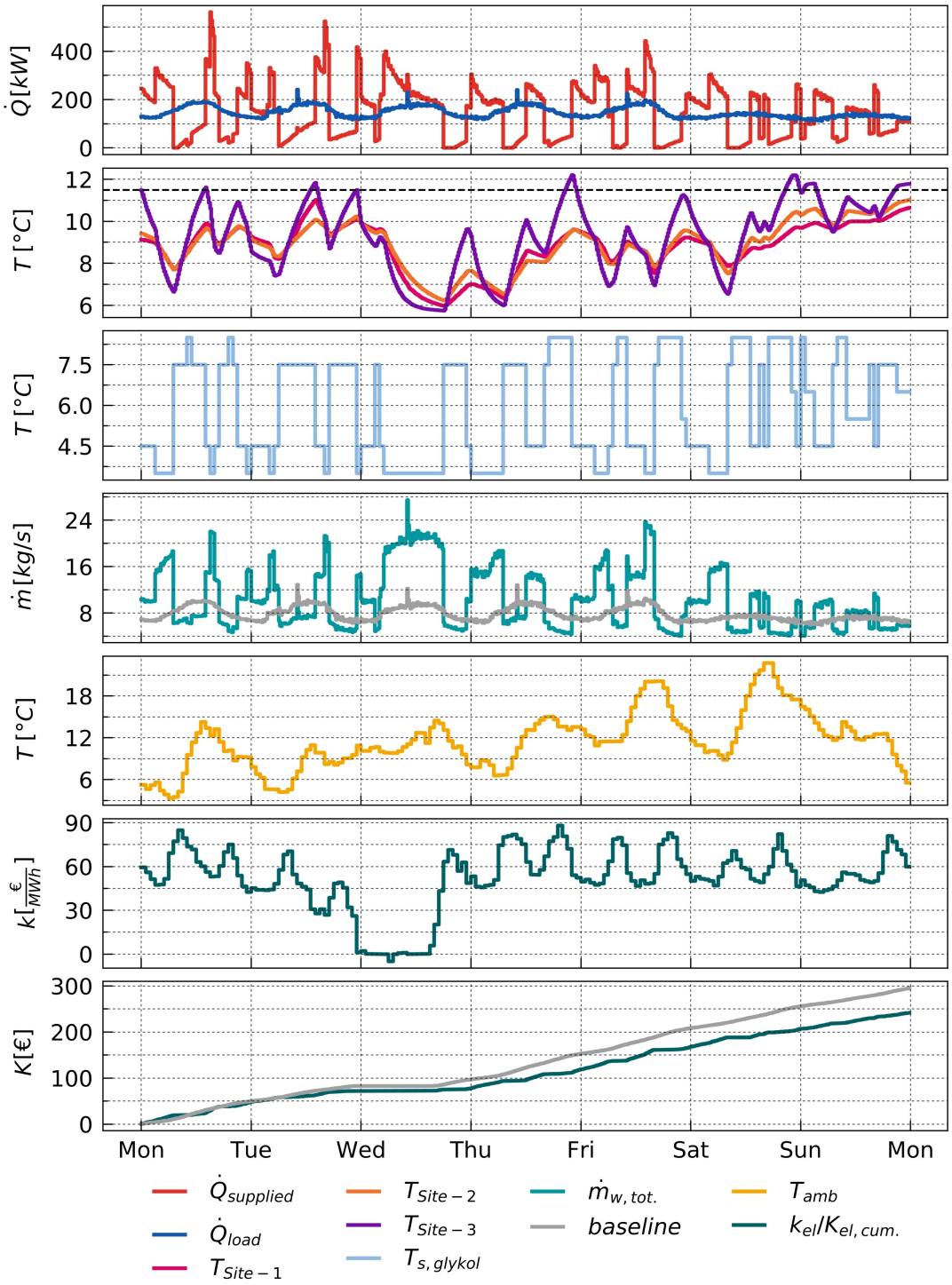

Figure 4.5: DQN operation of the cooling supply system with the learned policy for an exemplary week. The top plot shows the total cooling load at the consumer sites and the supplied cooling power. Plots two and three show the temperatures in the cooling circuit of the chiller and the temperatures of the thermal masses at the consumer sites. Plot four shows the mass flow in the cooling network with and without DQN control. The two plots below show the boundary conditions electricity price and outside temperature and the bottom plot shows the accumulated operation costs with and without DQN control.

Additionally, figure 4.6 shows the reward value that the algorithm receives during interaction with the system. The plot shows that the algorithm accepts small and negative rewards in the short- to mid-term in order to maximize long-term rewards. If the algorithm supplies more cooling energy than the current accumulated consumer loads, the rewards are negative, because the operation costs for this time-step are higher as they would be when following the baseline policy. In the long-term, the cooling energy is then used to supply correspondingly less energy when the electricity costs and/or outside temperatures are high.

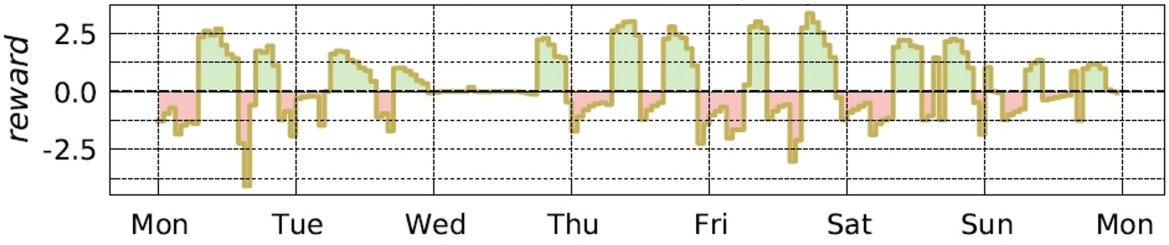

Figure 4.6: The algorithm receives temporarily small or negative rewards to achieve larger ones later.

Comparison of the algorithms

In figure 4.7, the performance of the two algorithms is compared. The results for training-weeks (left) and test-weeks (right) are very similar in all cases, which suggests that despite the many training cycles on the training-weeks, no over-adaptation to these boundary conditions has taken place.

A minimum of 4.7 % of the energy costs are saved during the 16 test-weeks. The average cost savings of 12.7 % for the test-weeks are slightly higher than for the train-weeks. The maximum value of the operation cost savings is 30.3 %. A similar trend can be observed in the evaluation of temperature limits. A deviation of 0.5 °C for the duration of one hour leads to a penalty of $\Delta T_{penalty} = -1.65$. The average penalty of $\Delta T_{penalty} = -3.5$ for the test-weeks is therefore considered very low. On average, the savings are close but the maximum value is slightly lower for the DDPG. The DDPG algorithm achieves 15 % higher savings compared to the DQN algorithm in the test-weeks with the lowest savings.

According to the observations, the temperature limit of the thermal masses is better-maintained by the DDPG. The penalty received with the DDPG algorithm is reduced by 70 - 80 % for test-weeks and training-weeks, compared to the DQN. The maximum penalty received in one week is 60 - 70 % lower for the DDPG. Because of the slight superiority of the DDPG, the following two evaluations regarding the adaptability and the influence of the thermal masses are carried out with this algorithm.

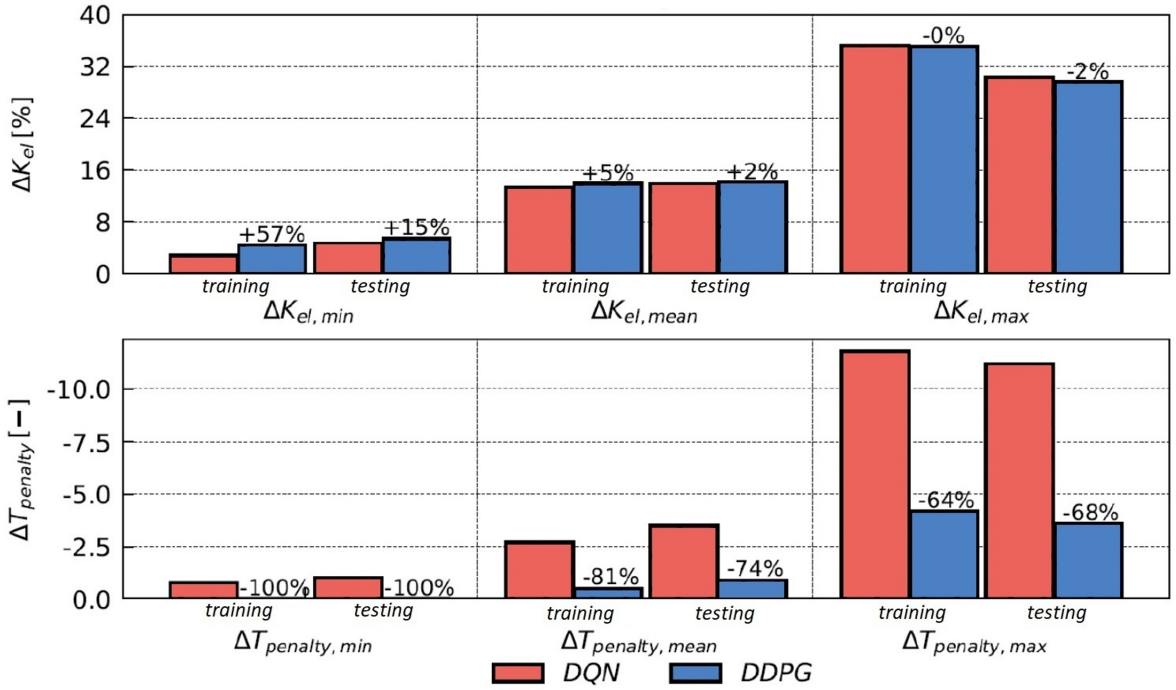

Figure 4.7: Comparison of the two investigated RL algorithms. Shown are the two terms of the reward function for the two algorithms compared for the three cases: Week with the smallest rewards (left), the rewards mean over all weeks (middle), and the week with the maximum rewards (right). Additionally in each case the performance on the training and the testing data is shown. The difference between the algorithms in each case is also given as percentages.

Flexibility potential

The size of the thermal masses at the consumer sites determines the available flexibility in the system, which the algorithm can exploit. To investigate their influence on the learned operation policy, three different capacities are studied here. Starting from the reference (storage capacity of 500 kWh over the temperature range from 5.5 °C to 12 °C), the size of the thermal mass is doubled or halved respectively. The results of the three modifications are presented in figure 4.8.

The operation cost savings can be increased by increasing the thermal mass of the consumers. An almost linear increase, when considering the minimum savings (left), is observed. In other words, a doubling of the thermal mass also leads to a doubling of the saved operating costs. This is not the case for average (middle) and maximum (right) savings. If the thermal mass is doubled from the lowest to the reference value, the operating cost savings for the test weeks are increased by 74 %. A further doubling of the thermal mass results in additional savings of 33 % for the test-weeks. In the first step, the maximum value increases by 53 %. In the second doubling of the capacity, the increase is only by 38 %. The reason for this is related to

the limited maximum cooling load that can be supplied to the cooling network by the chiller. This is therefore an environment-related limitation. The penalty received due to violations of the specified temperature limit increases over-proportionally for minimum, maximum, and average values. But even with the largest thermal mass, the average penalty received per week is just -3.2 for the test-weeks. This corresponds to a violation of the temperature limit of the 3 thermal masses of 0.5 °C over a period of less than 2 hours. However, the increased inertia due to the larger masses actually seems to result in a more difficult system to control. The temperature limits are still well-maintained, but violations become disproportionately longer and/or higher with the increase in mass at the consumer sites.

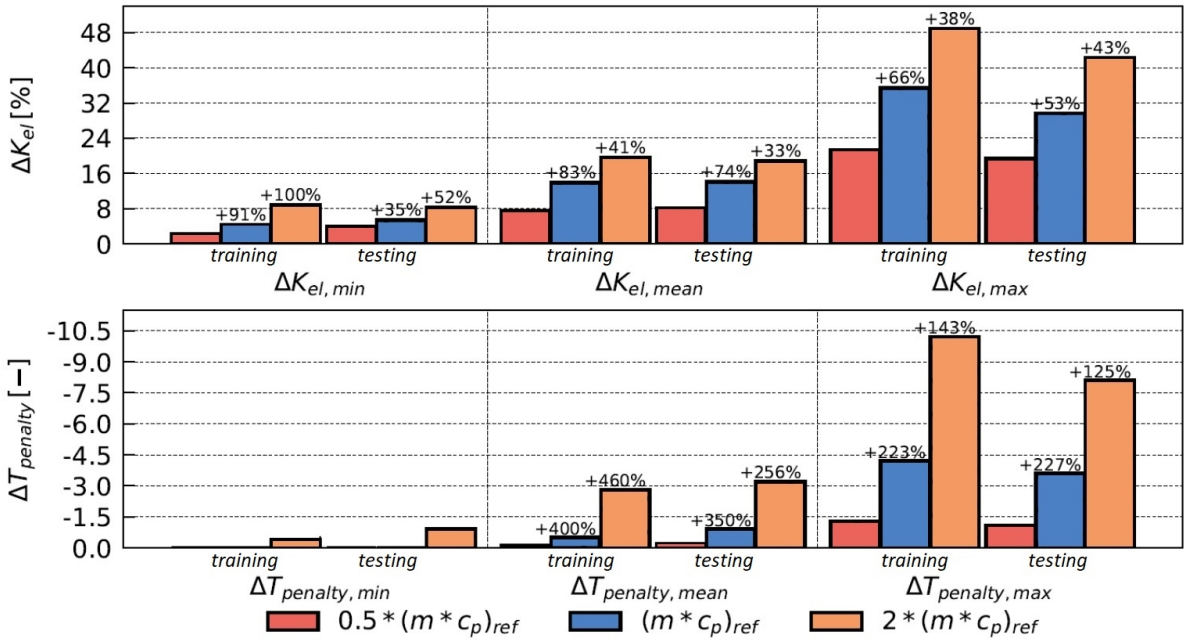

Figure 4.8: Investigation of the influence of the thermal masses on the learned policy. Shown are the two terms of the reward function for three different sizes of the thermal masses at the consumer sites and three cases: Week with the smallest rewards (left), the rewards mean over all weeks (middle), and the week with the maximum rewards (right). Additionally in each case the performance on the training and the testing data is shown. The difference from the change of the mass is given as percentages.

Adaptability

In this subsection the capability of the DDPG algorithm to adapt to changes in the controlled environment is investigated. The exploration is switched off with $\epsilon = 0$. For the experiment, it is assumed that the two three-way-valves have an offset value of - 15 %, which could occur after a faulty calibration process or through clogging over time, for example. The two valves can therefore only open to a maximum of 85 %, and valve positions between 0 % and 15

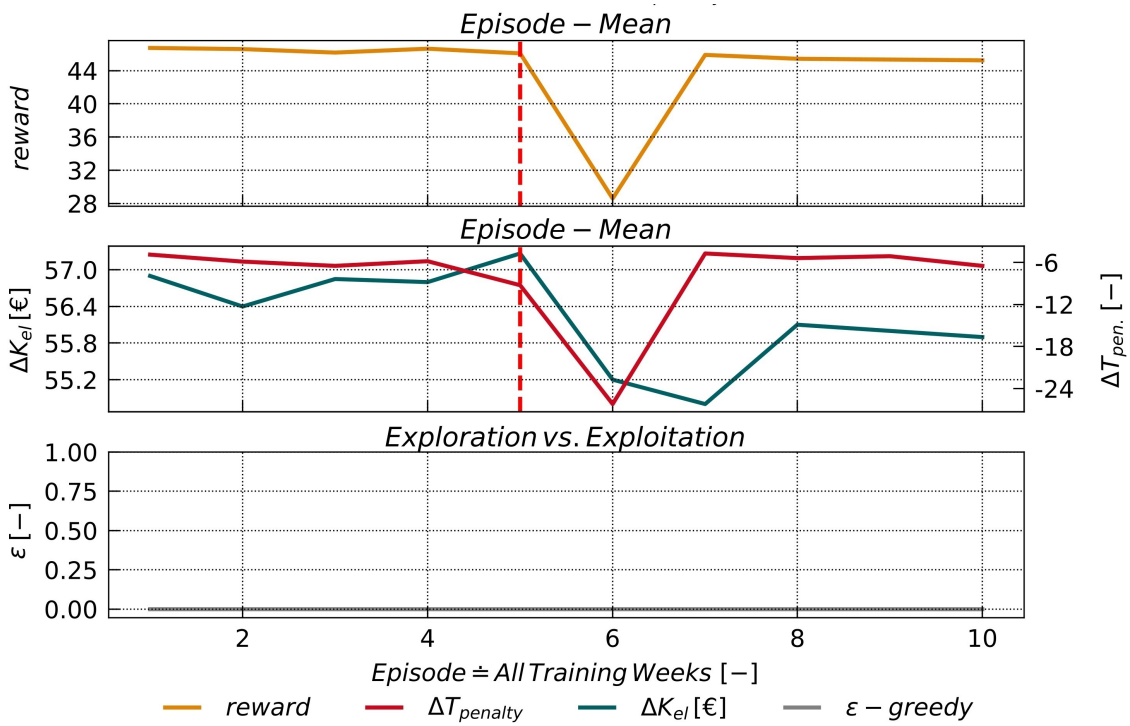

Figure 4.9: Investigation of the influence of a change in the system behavior. Like in figure 4.4 the top plot shows the absolute averaged reward for the algorithm, the middle plot the evolution of the two parts of the reward function, the monetary cost term and the penalty term for temperature deviations from the set temperature, and the bottom plot the exploration factor (set to zero). Here, a sudden change in the dynamic of the three-way-valves is implemented after the fifth episode which leads to a visible decrease of performance in episode six and an adaption of the policy (visible in the recovered performance of the algorithm) in episode seven.

% have no effect because they are already completely closed with the offset at 15 %. The algorithm must therefore learn to open the valves wider for the same mass flow and at the same time learn to handle the new limits.

The time-series of the reward values obtained for the described investigation are shown in figure 4.9. The red dotted line marks the time from which the offset value k_{offset} is subtracted in the simulation. In the following episode, the penalty increases significantly and is more than four times higher compared to the reference ($\Delta T_{penalty}$ drops from -6 to -24). The saved operation costs decrease, but remain on a similar level (less than 4 %). The average reward received decreases by about 40 %. It should be noted that the reward received for the next episode rises again to almost the level before the system change. Thus, the DDPG already reaches the old level of performance one training cycle after the occurrence of the change.

Improvements through state-of-the-art methods

In this section, it is investigated how the training process is influenced by the use of state-of-the-art modifications of the applied algorithms as well as by an optimized selection of the used hyper-parameters. As stated before, the DDPG in its used form can be viewed as an extension of the DQN for continuous action-spaces. However, both algorithms have been continuously improved in recent years.

Instead of the DDPG, the SAC algorithm is now used for continuous control. SAC adds an entropy term to the value function that returns what value an action has in a given state with respect to system exploration. Areas of the state-space in which the algorithm has little data on the system dynamics have high entropy and are thus more likely to be approached. The exploration and training becomes inherent in the calculation of the action values and the ϵ-greedy procedure, with which the exploration is controlled in the DDPG, is replaced. Two additional improvements are also added for the DQN. The first, called prioritized experience replay [Schaul et al., 2015], affects the selection of training samples from the replay buffer. While this was implemented purely randomly in the classic architecture, this implementation adds a prioritization of the samples. The priority is added to the state-action-next state-reward pairs already during storage and results from the deviation from the expected course of the value function to the actual course of the value function. A high deviation indicates a region where the algorithm needs more training. Therefore, such samples are selected with higher priority. The second extension is called dueling networks. Here, the state-action-value approximator is divided into two separate estimators: one for the state-value function and one for a state-dependent action advantage function. The advantage of this factorization is that state-action combinations are excluded from the possible actions in a state that would immediately have a strong negative effect on the rewards. The smaller dimension of the action-space thus speeds up the training. While in the classical approach the current state is processed together with all possible actions directly, the state-value as well as the advantage of a certain action is processed separately and combined afterward [Wang et al., 2016]. This architecture has proven effective for numerous applications and was therefore also included in the implementation used.

To find good hyper-parameters for both algorithms, 25 iterations of a Bayesian hyper-parameter optimization (section 3.7) are performed. The search-space and the considered hyper-parameters are shown for the DQN in table 4.2 and for the SAC algorithm in table 4.3. The distribution of the search-space is plausibly chosen based on the literature and the search distributions are divided into uniform and logarithmic uniform value ranges. The resulting hyper-parameters for the DQN algorithm are: $\alpha = 8.00e-5$, $\gamma = 0.9807$, $\mathcal{D} = 34263$, $B = 99$, $\epsilon_{\text{fin}} = 0.0673$, $expl.frac. = 0.4062$, and $targ.update = 2377$;
the ones for the SAC algorithm are: $\alpha = 7.49e-4$, $\gamma = 0.9926$, $\mathcal{D} = 64476$, $B = 118$, and $\rho = 3.11e-3$.

Table 4.2: Hyper-parameter search space for the DQN algorithm

Hyper-parameter	Search Distribution
learning rate α	loguniform(1e-5, 1e-3)
discount factor γ	loguniform(0.9, 0.995)
replay buffer size $\mathcal{D}$	loguniform(10e3, 10e4)
batch size B	uniform(64, 256)
final exploration rate ϵ_{fin}	uniform(0.02, 0.075)
exploration fraction	uniform(0.3, 0.7)
target network update interval	uniform(500, 5000)

Table 4.3: Hyper-parameter search space for the SAC algorithm

Hyper-parameter	Search Distribution
learning rate α	loguniform(1e-5, 1e-3)
discount factor γ	loguniform(0.9, 0.995)
replay buffer size $\mathcal{D}$	loguniform(10e3, 10e4)
batch size B	uniform(64, 256)
Polyak update ρ	loguniform(1e-4, 0.1)

The discount factor (γ) is chosen in a similar range for both algorithms, which means that the abilities of both algorithms to plan for the future appear similar to the optimization. It should also be noted that the reduction of the exploration factor selected for the DQN ($expl.frac.=$ 0.4062) is much larger than the value used in the original DQN and DDPG implementations (0.001). This indicates that the training process can be accelerated significantly by a faster reduction of the exploration factor.

Figure 4.10 shows the training processes of both algorithms after hyper-parameter optimization. It can be seen that the training processes have been accelerated by several orders of magnitude. With the original setup, 70 training epochs with all available training weeks were necessary, but now both algorithms show a positive performance already after 50 weeks and stabilize already after 200 weeks. For the DQN, the decisive factor here is in particular the faster lowering of the exploration factor, which leads to a clear acceleration of the training. Interestingly, the entropy-based exploration mechanism of the SAC algorithm still shows accelerated training even compared to the accelerated exploration of the DQN. After 250 weeks of training, however, the effect is over and both algorithms stabilize at an almost constant performance.

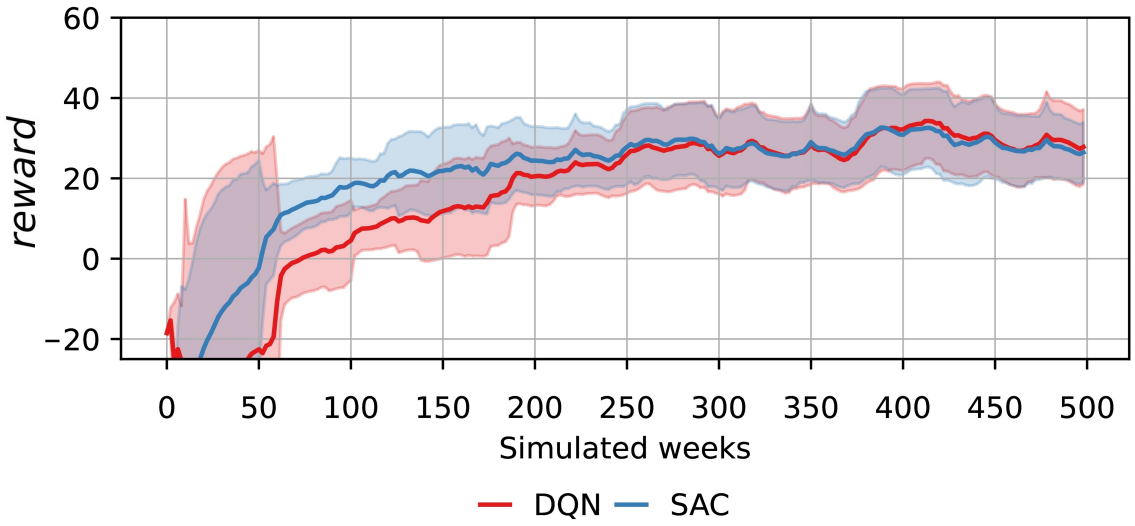

Figure 4.10: Training process of the DQN and the SAC algorithm after hyper-parameter optimization.

4.1.3 Discussion and lessons learned from case study one

This subsection summarizes the key results of the case study, discusses them, and also derives lessons learned for future work. This is based on the result plots presented in the previous subsection and the experience gained in the implementation process of the case study.

The application of the model-free algorithms for load shifting in a simulated cooling network was evaluated. The results show that the implemented algorithms are capable of learning an operation policy for the simulated cooling supply system that leads to a more economical operation for all training- and test-weeks. In this combined approach, 70 weeks of boundary condition data are enough to learn a superior policy, and although extensive training cycles were applied on the training data, no over-fitting (lower performance on the test-data) has been observed. The learned policies of both algorithms do not lead to large or long-lasting violations of the temperature limits.

The DDPG shows a slight superiority, but since the deviations from the temperature limits are already low with the DQN, the superiority of the DDPG is marginal. The ability of the DDPG of adjusting the policy to rapidly changing environment dynamics (offset in the controlled valves) was demonstrated. Exploration was disabled here and the networks control inputs were therefore adjusted based on the previously learned policy. Other hyper-parameters were not considered in the first five result subsections in order to evaluate the out-of-the-box use of the applied algorithms. It was also shown how the convergence behavior of the algorithms can be accelerated if state-of-the-art design principles are taken into account and optimized hyper-parameters are selected. For the DQN, the largest effect was observed by lowering the exploration factor faster. Using an entropy-based exploration strategy, the SAC algorithm

found the optimal policy significantly faster, compared to the DDPG and even to the DQN with optimized hyper-parameters.

The influence of the thermal masses in the system was also analyzed. The learned policies are able to comply with the given temperature limit almost at all times. The tendency that the received penalty increases with the increase of the thermal mass can be explained by the increasing cost saving potential: with larger capacity, the operation cost savings potential dominates the reward signals and the algorithm allows longer violations of the temperature limit.

Based on the study, the following lessons learned can be derived for future work:

- RL is generally well-suited for energy management tasks with high-dimensional state-spaces. However, the effort for generating the training environment should be critically weighed against the effort for a MPC application.
- If the problem is well-formulated, a discrete control algorithm can achieve similarly good performance as a continuous controlling algorithm. The dimension of the action-space is crucial here. Discrete control should only be used if there are a limited number of pre-defined system interactions to choose from. The DQN processes each action separately and cannot learn a relationship between neighboring actions.
- The selection of algorithms should take into account the latest design principles from RL research. The ability to find the optimal policy for reference problems has improved by a factor of 10 in the recent literature. This was also evident here, through the high convergence speed of the SAC algorithm, which (with its entropy-based exploration mechanism) also outperformed the convergence speed of the DQN with optimized hyper-parameters. Also, the selection of optimal hyper-parameters is important. In this study, in particular, the adjusted lowering of the exploration factor led to a significant speed-up of the training process. If hyper-parameter optimization is not possible for the problem, it is recommended to use literature values for problems with a comparable complexity.

4.2 Case study two

In the previous case study section, the applicability of RL for load shifting in a cooling supply system has been demonstrated. The two investigated algorithms (DQN and DDPG) are generally both applicable for this purpose. However, it has been found that the out-of-box use of these two algorithms does not lead to sufficient convergence speeds. Therefore, in this case study, the focus is on strategies to reduce the necessary interactions with the real system by applying different pre-training strategies (PS). In most existing studies, the training of RL controllers has been mostly performed with physical simulations of the investigated systems

[Han et al., 2019; Mason and Grijalva, 2019]. This approach, however, makes the time-consuming and error-prone modeling of the system dynamics necessary, which reduces an advantage of RL over MPC. In this case study, therefore, a fully data-driven RL training approach is demonstrated, using data of a real-world cooling supply system. State-of-the-art methods from the fields of data-driven modeling, offline training, Deep Q-Learning, and Bayesian hyper-parameter optimization are used for this purpose. Five promising PS are investigated with the aim of evaluating the final control performance and convergence speed of the trained algorithm (DQN).

The rest of this section is organized as follows. In the next subsection, the experiment design is presented. All steps from the raw monitoring data through a data-driven training and evaluation environment to the trained and evaluated RL controller are described. In subsection 4.2.4, the results are presented, which are then discussed in subsection 4.2.5. First, the description of the data-driven training and evaluation environment is presented. Based on this environment, the control problem (MDP) to be solved by the RL controller is introduced. In order to learn the optimal control policy for the system, five different PS are considered, about which an overview is given in the following. Finally, additional concepts from offline training and expert guidance used in this case study are introduced. As introduced in section 3.6, all implementations are built upon freely available Python packages, namely Scikit-Learn [Pedregosa et al., 2011], Keras [Chollet and et al., 2015], Stable-Baselines [Hill et al., 2018], and HyperOpt [Bergstra et al., 2015].

As in the last case study section, the investigated energy system is part of the same business and research facility in Berlin, Germany. At the considered supply site, the cooling energy demand also results from indoor air-conditioning, process cooling, and device cooling. In total, the thermal cooling demand fluctuates between 50 kW and 650 kW. Figure 4.11 shows a photograph of the considered cooling energy supply system, and figure 4.15 shows the schematic system structure together with the representative operation modes of the subsystems.

The supply system consists of two compression chillers with approximately 500 kW cooling power each and an ice storage with a maximum cooling power of 900 kW and a total capacity of 8.040 kWh. The cooling network is operated with a supply temperature level of 6 °C. While both chillers are technically designed for this, only chiller 1 can be operated at a temperature level that is also sufficiently low to load the ice storage. The cooling energy demand, the ambient temperature, and the historical day-ahead spot market electricity prices (DE-LU), from the ENTSO-E platform [Hirth et al., 2018], are considered as dynamic constrains.

The entire system is equipped with sensors, which continuously monitor all relevant volume flows, flow temperatures, cooling powers, and electric powers. All data is recorded in an event-based way, submitted using the IoT MQTT protocol, and stored in a time-series database; the access is realized via a REST API. Monitoring data of the system is available from the time period July 13, 2017 to January 01, 2020. After pre-processing (NaN- and outlier-handling)

a dataset of 40 weeks with a high data quality is generated.

The dataset with 26,880 samples in a 15-minute resolution is split into a training and a testing dataset. This is done by dividing the dataset into six equally long segments with a training episode length of 40 days and a test episode length of 7 days, each. The training samples are used for offline training, online training, and hyper-parameter tuning, while the testing weeks are used for the final RL controller evaluation.

Figure 4.11: A photograph of the cooling supply system investigated in case study two. The two compression chillers and the ice storage tanks are visible [EnBA-M, 2018].

4.2.1 Data-driven training and evaluation environment

For this experiment a data-driven model of the aggregated energy system is developed. In order to learn data-driven models from monitoring data, a semi-automated, data-driven modeling workflow is applied, similar to the one published in [Rätz et al., 2019]. The workflow includes data pre-processing, model training, and hyper-parameter tuning, model selection as well as the initial system decomposition and the final system aggregation. Different machine-learning algorithms are trained and tuned, according to the workflow illustrated in figure 4.12, to represent the dynamics of all subsystems in their representative operation modes. Considered algorithms are linear regression (LR) [Hüttermann et al., 2019], support vector regression (SVR) [Jain et al., 2014; Zhang et al., 2016], random forests (RF) [Ahmad et al.,

2017; Jain et al., 2017, 2016], and ANN [Ahmad et al., 2017; Mocanu et al., 2018; Ryu et al., 2017].

In order to capture the thermal dynamics of the system, the data-driven models are designed with a resolution of 5 minutes, which represents a trade-off between system inertia, sensor accuracy, and capturing the system dynamics. In contrast, the control actions and dynamic constrains are implemented in a 15-minute resolution, which on the one hand addresses the available resolution of the constraints and on the other hand favors the observability of action effects in the next state. The model tuning is performed via Bayesian hyper-parameter optimization and all models are evaluated based on their root-mean-square-error (RMSE) and coefficient of determination (R2-score).

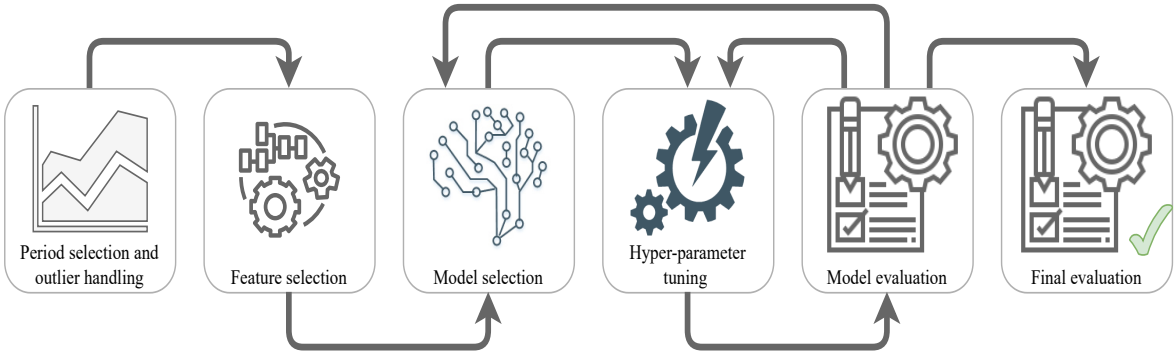

Figure 4.12: The data-driven modeling workflow. Starting with the selection of training data and the choice of suitable features; models and hyper-parameters are optimized with respect to validation data.

Modeling of the compression chillers

The models of the compression chillers are designed to approximate the electric power consumption P_{el}. Both the cooling power setpoint $\dot{Q}_{set}$ and the ambient temperature T_{amb}, influence the compression chiller's energy efficiency ratio (EER). The functional relationship to be learned is provided in equation 4.4.

$$EER = \frac{\dot{Q}_{set}}{P_{el}} = f\ (T_{amb},\ T_{return}) \rightarrow \quad P_{el} \approx\ f\ (T_{amb}, \dot{Q}_{set}) \tag{4.4}$$

It is derived from the monitoring data that chiller 1 operates on two different temperature levels, which results in two different operation modes. Therefore, two independent data-driven models are used here for the two operation modes; one for ice storage charging and one for direct supply of the cooling demand from the cooling network.

Figure 4.13 illustrates how data pre-processing reduces the feature values to those relevant for identifying the physical relationships. The input data for the data-driven model of chiller 2

are visualized in the form of the feature pair plot and the feature histogram before and after data pre-processing. The histograms show how pre-processing the data effectively increases the density of samples with physically plausible feature values. At the same time, irrelevant monitoring data, for example from inactive periods, are excluded.

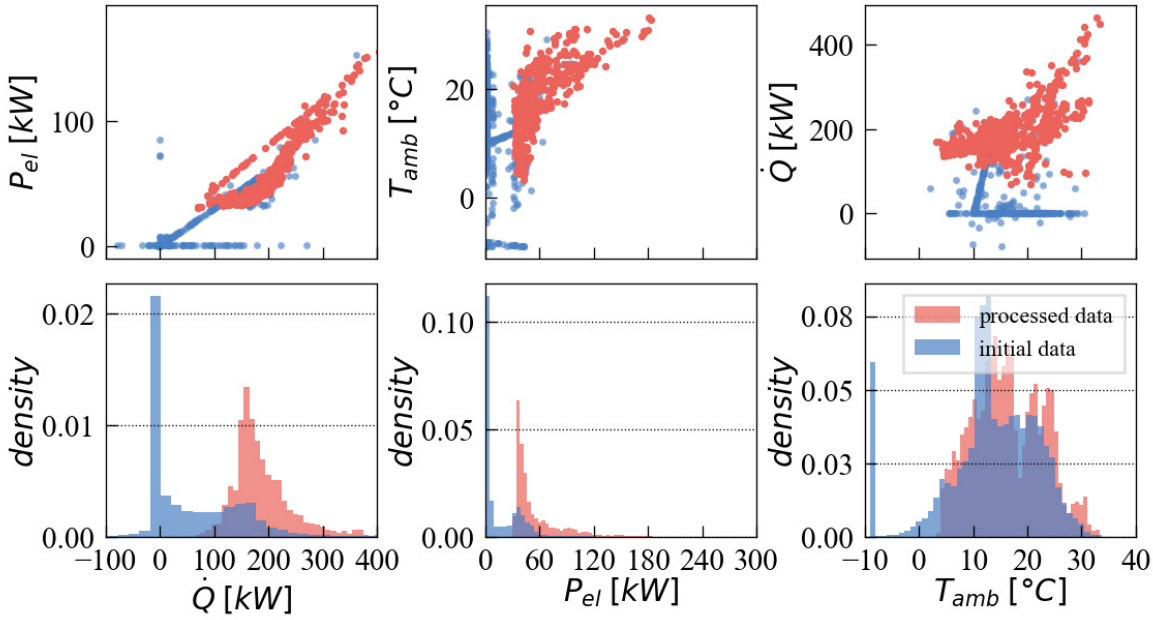

Figure 4.13: Data pre-processing of chiller 2. The top three plots show the correlations of the features cooling power, electrical power, and ambient temperature with each other (feature pair plots). The three lower plots show the uniform relative frequency of the features in the dataset before and after pre-processing (histogram).

By applying outlier- and NaN-handling, erroneous values, for example from sensor malfunction, are excluded. Values outside physically plausible limits, such as negative cooling loads, are eliminated as well. For example, a period in the dataset is identified and deleted from the training data where the outdoor temperature sensor recorded constant $T_{amb} = -9.1°C$ over four weeks in the spring of 2018.

Modeling of the ice storage

The ice storage system is designed with three operation modes, namely loading, unloading, and idle. During the idle mode, the ice storage does not interact with any other subsystem model. Nevertheless, a decrease in the state of charge (SOC in percentage of ice) results from heating from the environment, correlating with the ambient temperature (T_{amb}). The feature space includes one feature to provide the machine-learning model with information about the state of charge of the previous time-step SOC^{t-1}:

$$SOC_{idle} \approx f(\, T_{amb}, SOC^{t-1}) \tag{4.5}$$

$$SOC_i \approx f\left(\dot{Q}_i,\ T_{amb}, SOC^{t-1}\right) \qquad i\epsilon[loading, unloading] \tag{4.6}$$

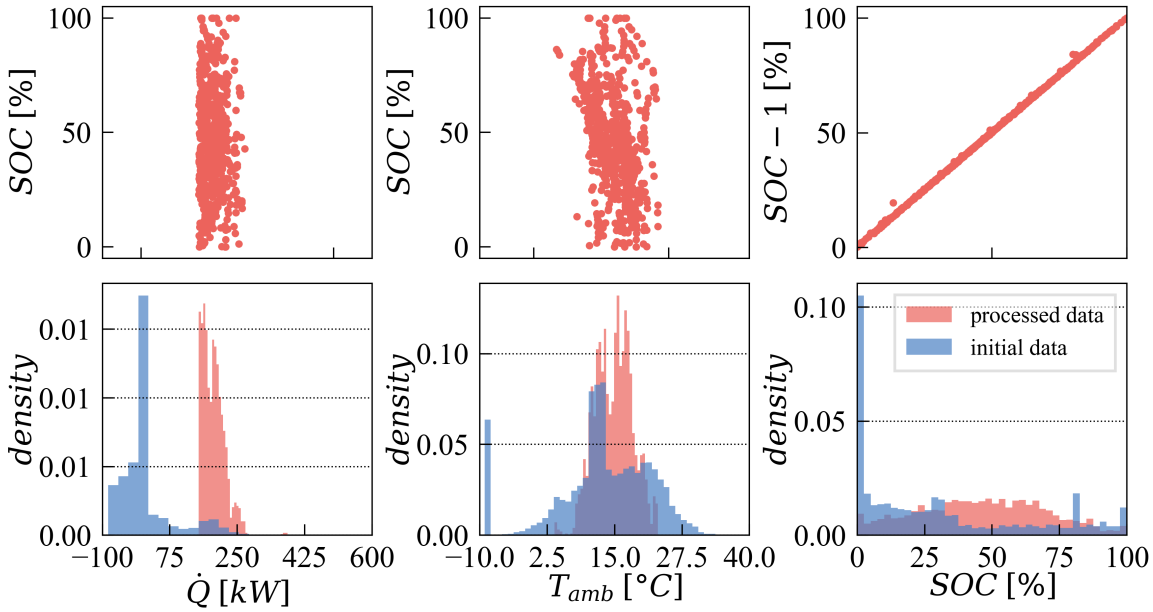

Figure 4.14: Data pre-processing of the ice storage in loading mode. The top three plots show the correlations of the features cooling power, ambient temperature, and state of charge with the state of charge and the state of charge in the last time-step (feature pair plots). The three lower plots show the uniform relative frequency of the features in the dataset before and after pre-processing (histogram).

For the operation modes loading and unloading, it is assumed that the state of the ice storage is only sufficiently characterized, if historical information is included in the form of a series of past state of charge values. In contrast to these consideration, the addition of more than one historical value did not show further model improvements. Therefore, only the state of charge of the last time-step is used as a feature for the data-driven model.

In figure 4.14, the input data for the loading model of the ice storage is visualized in the form of the feature pair plots and the feature histograms before and after data pre-processing. As the figure illustrates, pre-processing the data significantly reduces the number of outliers. In contrast to the data visualization of the input data of the compression chiller 2 in figure 4.13, the functional relationships between the features $\dot{Q}$, T_{amb}, and the resulting SOC are not directly visible by means of a linear one-to-one correlation. Naturally, a linear correlation between the SOC and its prior value SOC^{t-1} can be observed.

Modeling of the interconnected energy system

The data-driven models of each operation mode of each subsystem are aggregated to represent the dynamics of the whole cooling energy supply system. Therefore, the trained models are

interconnected to enable energy exchange between compression chiller 1 and the ice storage. The cooling energy supply system, consisting of the two compression chillers, the ice storage tanks, and a heat exchanger (between the glycol circuit and water circuit) is aggregated as shown in figure 4.15. The generation units are designed to meet a given cooling demand $\dot{Q}_{load}$, which is extracted at the heat exchanger. The cooling load $\dot{Q}_{load}$ along with the ambient temperature T_{amb} are exogenous boundary conditions for the aggregated model. Inspired by the operation modes implemented in the automation of the real system, three operation modes OM-0, OM-1, and OM-2 are implemented, as listed in table 4.4.

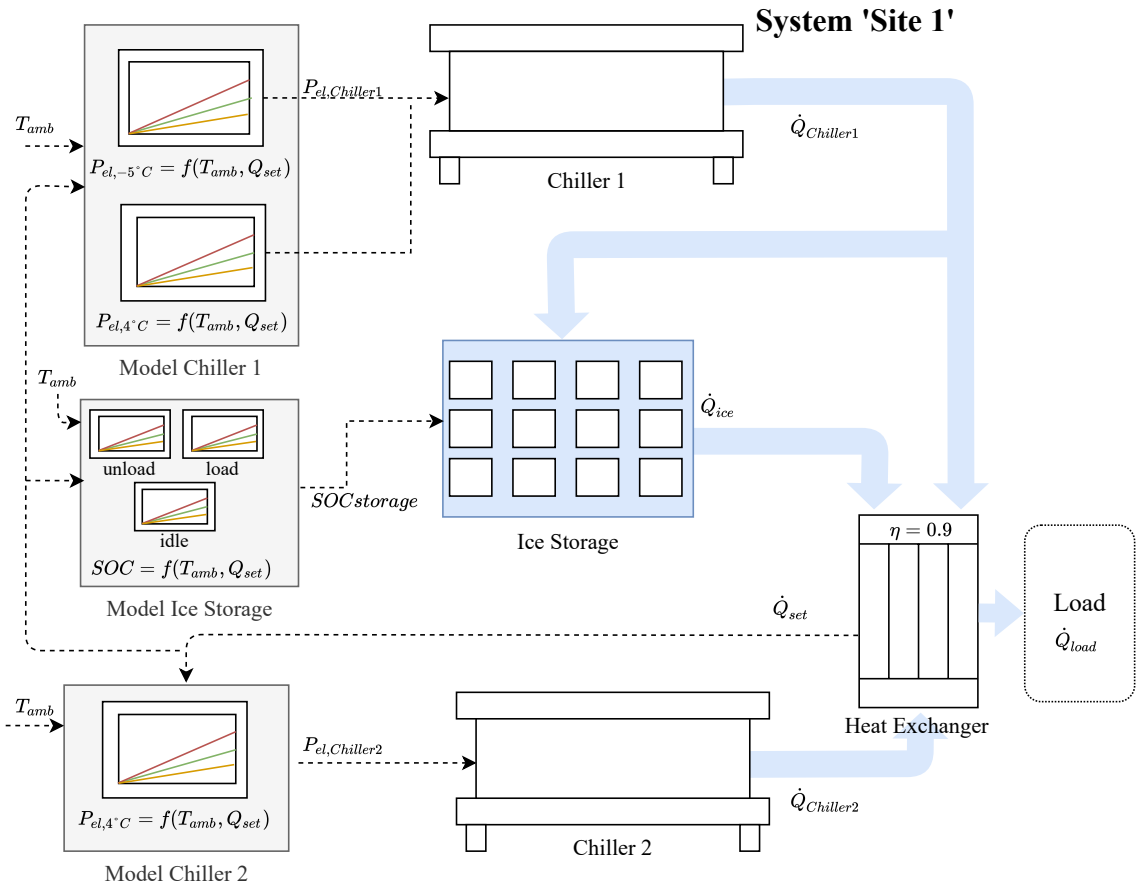

Figure 4.15: The schematic structure of the aggregated case study two cooling system, consisting of the six interconnected machine-learning models: three models to predict the SOC of the ice storage (loading, unloading, and idle); two to predict the electric power consumption of chiller 1 ($P_{el,\ chiller1}$: one for ice storage loading and one for direct network supply); and one to predict the electric power consumption of chiller 2 ($P_{el,\ chiller2}$).

Results of the data-driven modeling

In this section, the results of the data-driven modeling approach are presented and discussed. Figure 4.17 shows the RMSE and the R^2-scores, of all the applied algorithms for all 6 individual subsystem operation mode models, after hyper-parameter tuning. The LR baseline models generally provide less accurate scores compared to the other models. It is outperformed by RF, ANN, and SVR in all cases. RF reduces the RMSE down to 79.2 % in relation to the LR baseline. The same is observed for the R^2-score, which shows highly similar trends in all cases. Therefore, the algorithm for the corresponding subsystem model is selected based on the RMSE score on the test dataset, after hyper-parameter tuning.

RF and SVR provide the most accurate subsystem models. SVR outperforms RF and ANN for the chiller 1 subsystem models, for which fewer training data samples are available, and slightly outperforms RF and ANN for the ice storage charging model. The prediction of the electric power demand of the compression chillers is learned with R^2-scores between 0.94 and 0.99 and RMSE values between 2.02 kW and 3.51 kW. Also, the prediction of the percentage of ice formation within the ice storage is learned accurately with a R^2-score of near 1 and RMSE values between 0.08 % and 0.72 %. These results show that the dynamic behavior of the subsystems is successfully learned.

A few interesting observations were made in the course of the hyper-parameter tuning:

- The hyper-parameter tuning of the ANN models shows that small batch sizes (in this case, 5-15 samples per batch) lead to the best results. Naturally, the RMSE improves with more training epochs until it reaches a threshold value. If training continues after the threshold is reached, over-fitting occurs and the prediction accuracy on the test data decreases again. The maximum value is generally lower for models that converge after a relatively small number (5-10) of training epochs. Further, the score of most models does not significantly improve beyond 5 neurons within the hidden layers.
- Throughout all experiments conducted with SVR, the RMSE score declines with higher values of the regularization parameter C and the kernel parameter gamma. While the accuracy improvements become small with $C \geq 10$, variations of gamma have notable effects on the RMSE over the entire range of the search space. Apparently, the choice of a high regularization parameter C ($C > 1,000$), can compensate for a poor choice of gamma.
- The experiments confirm that RF is insensitive to changes in its hyper-parameters. Since the number of nodes (Decision Trees) has the biggest influence, an optimal number for each model is identified; improvements were insignificant beyond a certain threshold (once the number of nodes is higher than 7).

The experiments also confirm the poor extrapolation capabilities of non-linear data-driven models. This is observed especially for ambient temperatures (T_{amb}), beyond the bounds

in the training dataset. Nevertheless, it was possible to learn the relevant characteristics of all subsystems and operation modes based on the available data. Figure 4.16 shows the characteristic of chiller 1 in ice storage loading mode, learned with a SVR model. Although the model appears to be rather accurate, with a RMSE of 2.02 kW, the predictions become inaccurate in areas of the feature space where no monitoring data is available: electrical power P_{el} increases with decreasing cooling load $\dot{Q}$ at high T_{amb}.

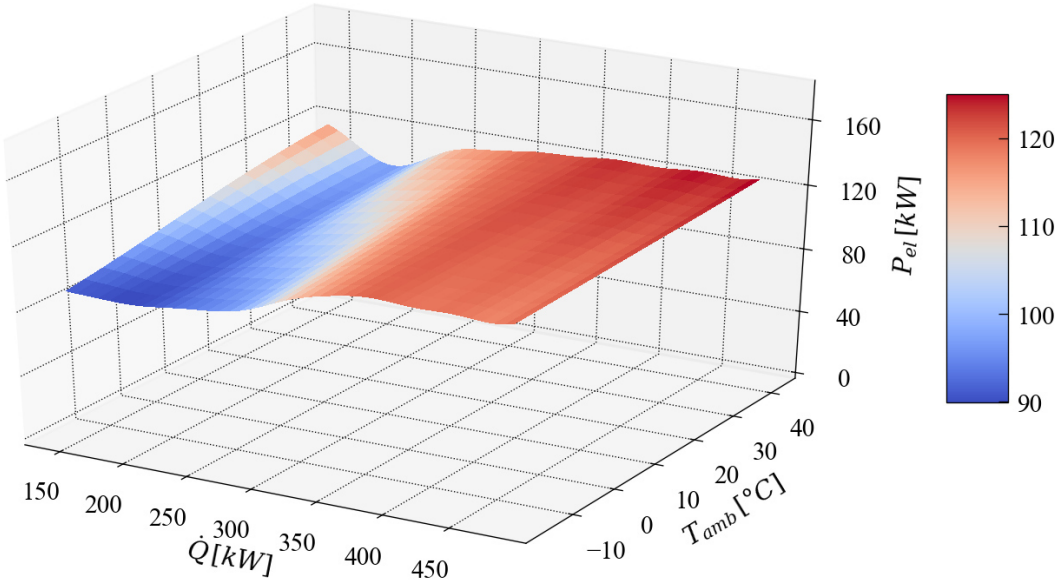

Figure 4.16: The dynamics of chiller 1, learned via a SVR model in the three dimensions: Cooling power, electrical power, and ambient temperature. The diagram shows the expected trend towards more electrical power at higher ambient temperatures and higher required cooling power. However, it also shows the limits of the model in regions which were not represented in the dataset (low cooling load at high outdoor temperatures: physically implausible drop in the required electrical power at the top left corner of the plot).

As indicated by the RMSE and R^2 scores in figure 4.17, all subsystem models show highly accurate operation characteristics. The ice storage model is the most accurate and coincides with the time-series from the monitoring data for almost all time periods. In general, it is shown that the interaction characteristics of the data-driven models represent the characteristics of all components in the supply site with high accuracy. The aggregated model is thus suitable for the use as a data-driven training and evaluation environment.

4.2.2 Markov Decision Process formulation

The formulation of the MDP, with its action-space, state-space, and reward function $r(a_t,\ s_t)$ is presented in this section. The possible actions of the MDP are the three energy management

operation modes from table 4.4. The possible actions are: OM-0, where the cooling demand of the network is provided by chiller 2, while chiller 1 loads the storage. In the operation modes OM-1 and OM-2, the cooling demand of the network is supplied by unloading the storage or by direct supply through chiller 2, respectively. The control action is performed every 15 minutes (every third simulated time-step). The state-vector consists of 15 variables, as listed in table 4.5.

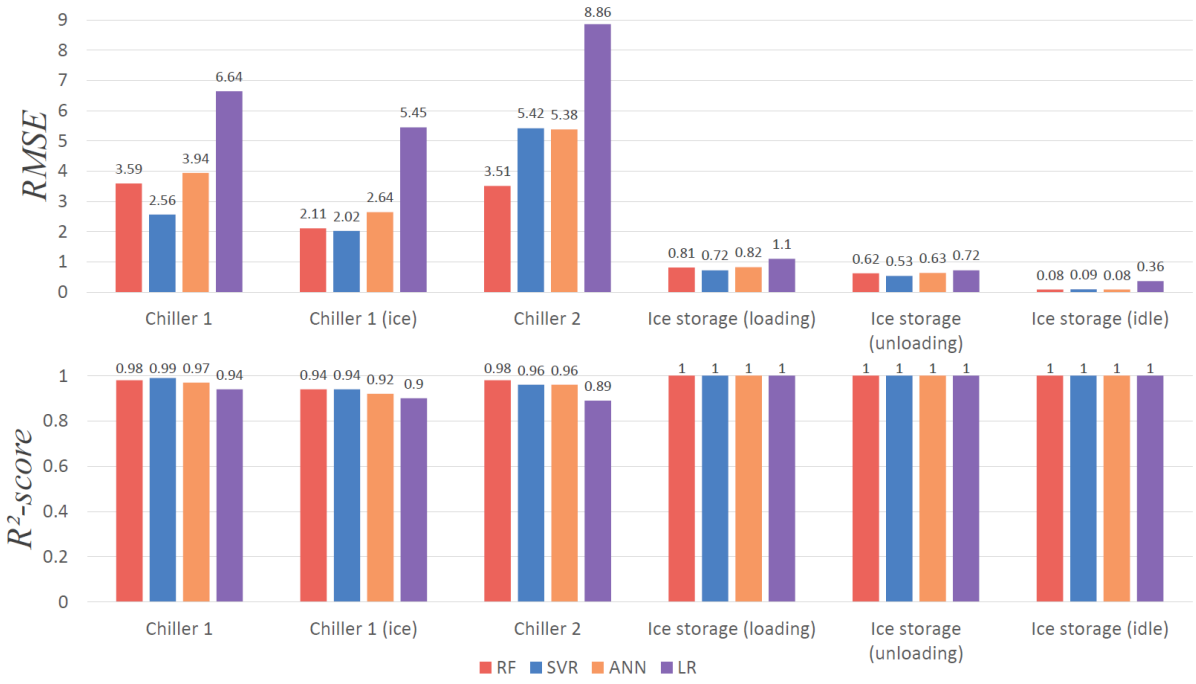

Figure 4.17: Comparison of the best data-driven models for the six operation modes of the thee components after hyper-parameter tuning, based on the RMSE and R^2-scores.

The authors of [Wang and Hong, 2020], highlighted that the control performance of a RL controller can benefit significantly from the inclusion of forecasts of the control problems constraints. In order to find a trade-off between including forecasts and the size of the state-vector, statistical features (minimum, maximum, and mean) are used here, instead of the complete time-series, with a rolling 16 h horizon. A perfect forecast is used based on historical data of the ambient temperature and the electricity price. Due to the rolling horizon, it can be assumed that the influence of the inaccuracy in real forecasts, as with MPC, is relatively small. Further, meta information regarding the hour of the day t_h, the day of the week t_d, and the month of the year t_M are included as sinusoidal signals. The electricity costs $C_{el,t}$ compared to the costs $C^b_{el,t}$, resulting from a **baseline** policy π^b (in €), represent the reward signal:

$$r_t\left(a_t, s_t\right) = \left(C^b_{el,t} - C_{el,t}\right) - 0.1(if: a_t \neq a_{t-1}) - 0.5\Delta SOC \tag{4.7}$$

The **baseline** policy π^b used to evaluate the savings, (realized with a RL controller, trained under one of the applied PS) reflects the case of direct load coverage. In this case, OM-2 is constantly active, chiller 2 supplies the network, and neither the electricity price nor the forecast of it are taken into account. In addition, two small terms are added in order to facilitate reasonable control behavior from an engineering perspective. Since an increasing number of chiller starts and shutdowns are undesirable from an engineering point of view, a small term (0.1) penalizes action changes. Further, penalizing the difference of the SOC between the first and current time-step of an episode leads to another term ($0.5\Delta SOC$). This guides the RL controller to learn policies that do not empty the ice storage at the end of each episode. Therefore, the policy of the RL controller is influenced towards a more system-friendly operation, from an engineering point of view. All RL controllers are evaluated based on their mean episode savings.

Table 4.4: The three possible operation modes of the aggregated energy system.

Action	Chiller 1 ($510kW$)	Chiller 2 ($500kW$)	Ice Storage ($900kW; 8,040kWh$)
OM-0	load ice storage	supply network	loading
OM-1	inactive	inactive	supply network
OM-2	inactive	supply network	inactive

Table 4.5: The state-space variables presented to the RL controller at each time-step an action is performed.

State-space variables	
$P_{el,\ chiller1}$: electrical power chiller 1	$P_{el,\ chiller2}$: electrical power chiller 2
SOC: ice storage SOC	$\dot{Q}_{dem}$: cooling energy demand
T_{amb}: amb. temperature	$T^{16}_{amb,max}$, $T^{16}_{amb,min}$, $\bar{T}^{16}_{amb}$: forecast
k_{el}: electricity price	$k^{16}_{el,max}$, $k^{16}_{el,min}$, $\bar{k}^{16}_{el}$: forecast
t_h: hour of day	t_M: month of year
t_d: day of week	

4.2.3 Overview of the investigated pre-training strategies

The different PS, evaluated in this case study, are visualized in figure 4.18 and summarized in table 4.6. These PS represent three different approaches: **PS-A** refers to online RL in interaction with the data-driven model of the energy system; in **PS-B**, an online training phase is combined with offline training, based on artificially generated expert trajectories; and with **PS-C**, training is realized, with monitoring data only. It is further differentiated

Table 4.6: The evaluated PS combinations, leading to five implemented workflows.

Concept	PS-A	PS-B	PS-C
Offline training	✗	✓	✓
Expert trajectories	✗	✓	✗
Monitoring data	✗	✗	✓
Guiding RBC	A1:✓ A2:✗	B1:✓ B2:✗	✓

between variation **1** and variation **2**, in which a guiding RBC controller is either implemented or not, respectively.

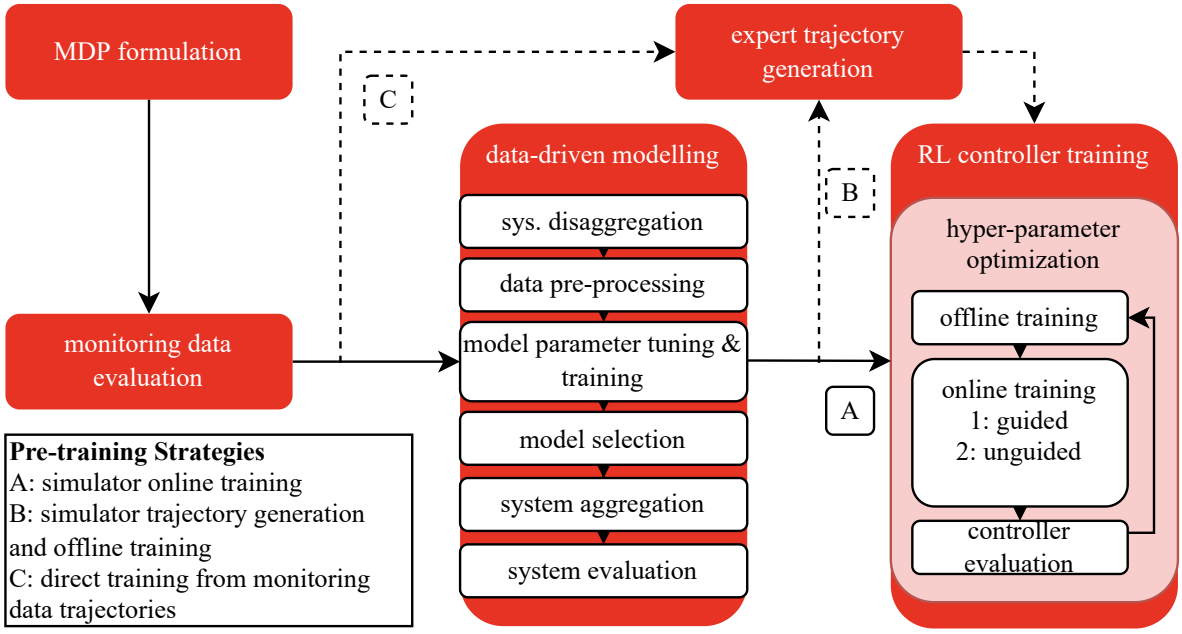

Figure 4.18: The evaluated PS, where different workflows are compared: direct online training with the data-driven model (PS-A) and offline training either with expert trajectories (PS-B) or with monitoring data (PS-C). For PS-A and PS-B, the performance is additionally compared with and without a guiding RBC controller (1: guided; 2: unguided).

The data-driven model is used in combination with the training dataset, first for expert trajectory generation, secondly for online controller training, and thirdly for controller evaluation, using the test dataset. The expert trajectories used for PS-B are generated by running simulations of the model using a computationally cheap RBC. Alternatively, the expert trajectories used for PS-C are directly extracted from monitoring data. The difference is, therefore, that in PS-B, expert trajectories are generated using the simulation, and for PS-C, only the realized control actions observed in the monitoring dataset from the real system are used.

Since an efficiency-based RBC is implemented for the real system, in which the ice storage is always loaded overnight, PS-C represents this case. Between 10 and 12 o'clock at night, the ice

storage is fully loaded, which is usually completed in the early morning hours. The strategy aims to exploit the lower outdoor temperatures at night for loading in order to increase the efficiency of chiller 1 during operation. All PS are evaluated in terms of their mean episodic savings over the baseline policy. Further, all PS are studied with their representative optimal hyper-parameter combinations.

Offline training and RBC guidance

In offline RL training, the RL algorithm goes through supervised training, to approximate the state-action-value function prior to online RL training. The aim is to reduce the number of necessary interactions with the real system till the optimal policy is learned. In analogy to online training, during offline training, batches of stored environment transition samples are backpropagated through the network of the DQN via stochastic gradient-descent. The number of training epochs is considered as a tunable hyper-parameter.

An expert trajectory, used for DQN training, contains a temporal sequence of (state, action, reward, next state) pairs. The corresponding trajectories are generated differently for PS-B and PS-C. While the expert trajectory for PS-C is extracted directly from the monitoring data (and thus encodes the currently applied RBC), the trajectory for PS-B is generated artificially by applying an electricity price-based RBC in the data-driven training environment.

In the case of PS-C (where expert trajectories for the offline training are generated directly from monitoring data) the historic control actions are detected and marked within the monitoring dataset by using a simple conditional action classifier. The trajectories therefore result from the "load at night" RBC, introduced along with the PS overview in the last section. As introduced, this RBC does not take the price signal into consideration but exploits the lower ambient temperatures at night to operate the chillers with higher EER values.

For the offline training with PS-B, a more economic, price-based RBC is implemented. This RBC takes the forecast of the electricity price for the next 16 hours into account when making the control action. It exploits fluctuations of the electricity price by loading the ice storage (OM-0) whenever the SOC is below 50 % and the current electricity price is lower as the mean value of the forecast ($k_{el} \leq \bar{k}_{el}^{16}$). Unloading of the ice storage (OM-1) is chosen if the price is higher ($k_{el} > \bar{k}_{el}^{16}$).

In addition to pre-training of the RL controller with different expert trajectories, the challenging transition from offline to online training is addressed by stabilizing the initial phase with a guiding RBC. The respective RBC follows the expert trajectory with which the RL controller is pre-trained offline and interferes with the action of the online RL controller under a decreasing probability β. β is initialized with $\beta_0 = 1$ (RBC) and is linearly reduced to $\beta_i \to 0$ (RL) over time. The number of training steps the RBC intervenes in the RL controller action is considered as a tuneable hyper-parameter. The PS are distinguished by the numbers

1 and 2; PS-A1, PS-B1, and PS-C are the variants with implemented RBC guidance, while PS-A2 and PS-B2 do not include RBC guidance.

4.2.4 Results of case study two

In this section, the results of case study two are presented. All experiments are conducted with an ordinary PC (CPU: Intel Core i5- 8265U 1.60 GHz; RAM: 16 GB). The training of one DQN under a certain PS took 1,093 seconds on average.

Hyper-parameter optimization

The hyper-parameters found via HyperOpt [Bergstra et al., 2015] are shown in table 4.7. The search-space is reasonably selected on the basis of the literature and each PS is optimized for 40 iterations. In the conducted experiments, hyper-parameters are the ones of the training process itself. For the DQN structure, on the other hand, the default settings from the Stable-Baseline implementation [Hill et al., 2018] are used. Throughout all PS, RL controller performance improvements between 105 % and 184 % are observed in relation to the average savings over the baseline policy (before hyper-parameter optimization). At the same time, the necessary interactions until convergence are reduced from ∼800,000 to ∼125,000. All following results are based on the best hyper-parameter sets for the respective PS.

Comparison of the PS

The learning curves of the RL controllers, trained with all introduced PS are shown in figure 4.19. The plot shows the mean episodic savings ($\bar{J}_{eps}^{train}$) and the deviation (shaded area) over a period of 30,000 simulated hours with the training dataset. In table 4.8, five performance indicators are additionally presented.

The standard deviations are constantly high for all PS because the training environment is randomly initialized repeatedly over the course of the simulated 30,000 hours. The reason is that only 6 * 40 days (5,760 hours) of data are available for the training. Therefore, it is necessary to extend the possible training epochs with the available data, by random initialization of the environment: In order to generate more data for the training and at the same time avoid over-fitting to known data, the training process repeatedly starts at a new, randomly selected time-step. Even a perfect policy could thus only act within the limits of its suboptimal starting conditions. However, the convergence of the RL controllers can be clearly observed in the time-series of the mean episodic savings.

The offline pre-trained DQN, with implemented RBC guidance (PS-B1) consistently yields positive savings and the performance is also slightly higher than the one of the "price based"

RBC, with which it is pre-trained. The guided DQN without offline pre-training (PS-A1) achieves the highest monetary savings after the 30,000 simulated hours. The other PS also yield consistently positive but lower savings at the end of the evaluation period. The first-year savings are used to compare the PS based on their costs of training caused by system exploring control actions during the initial phase of online interaction. Due to its rapid convergence, the unguided DQN (PS-A2) yields the highest savings within the first year (8,785.7 EUR).

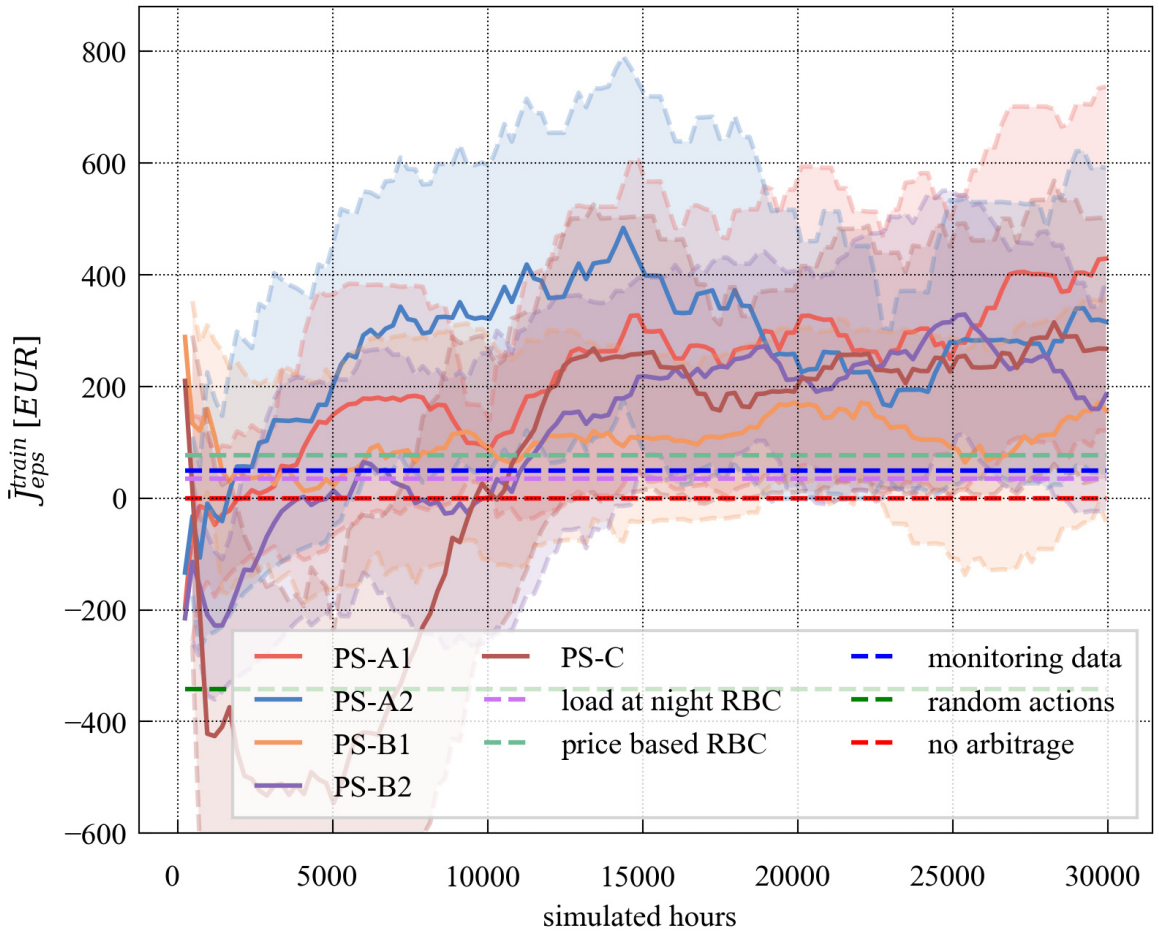

Figure 4.19: Convergence behavior of online training of all five differently designed RL controllers over 30,000 simulated hours on the training dataset. The baseline policies are highlighted with dashed lines. The solid lines represent the mean episodic rewards and the shaded area represents the spectrum between best performing episode and worst performing episode.

With PS-A1, where the RBC guidance is implemented with no prior offline pre-training, the training process is finished with the highest savings. In comparison, the DQN trained under PS-A2, where the interaction is unguided, converges more rapidly (break-even time and first-year savings).

Regarding those PS, where the DQN is initialized with offline pre-training (PS-B1, PS-B2,

and PS-C), it is shown that high exploration values in the initial phase of the interaction lead to a significant negative controller performance, as it can be seen in the learning curves of PS-B2 and PS-C. Once online training is initialized, both respective controllers underperform the control performance of the expert policies they are offline pre-trained with. In summary: In the initial phase of online interaction (at best), the pre-trained RL controllers reflect the trivial baseline RBC encoded in the expert trajectories they are trained with.

Table 4.7: Results of hyper-parameter optimization after 40 search iterations.

Hyper-parameter	PS-A1	PS-A2	PS-B1	PS-B2	PS-C
offline learning rate	n/a	n/a	9.9e-3	9.9e-3	9.9e-3
offline epochs	n/a	n/a	40	40	40
discount factor (γ)	0.93	0.97	0.91	0.93	0.95
learning rate (α)	5.01e-5	6.37e-5	4.61e-6	1.48e-5	5.43e-5
buffer size	94900	3400	3600	33400	36600
batch size	56	84	60	116	44
init. exploration rate (ε)	0.65	0.75	1.0	1.0	0.75
fin. exploration rate	0.04	0.01	0.02	0.01	0.03
init. RBC interference	0.7	n/a	0.75	n/a	0.9
time-steps with RBC	21000	n/a	34000	n/a	32000

RL controller performance on test data

In figure 4.20, the performance of all RL controllers trained with their respective PS for a three-week period of the test dataset is shown. The central subplot shows the electricity cost savings compared to the baseline policy, the lower subplot the electricity price for this time period, and the top subplot the storage loading and unloading cycles resulting from PS-A1 and PS-B1. The used colors for the time-series of the top and the central subplots are listed in the legend. All PS outperform the heuristic "price based" RBC, based on the dynamic electricity price forecast. The visualization of the ice storage SOC (top subplot) is used to compare the time-series of PS-A1, PS-B1, and the "price based" RBC. Therefore, the differences between two variations, one with and one without offline pre-training and both with guiding RBC, are compared. For these two policies, an additional analysis is conducted. The selected actions are analyzed, by extracting the action probabilities (the normalized Q-values), at each time-step. This determines the probability of the RL controller to select one of the possible actions (OM-0, OM-1, and OM-2).

Figure 4.20: Time-series plots of the trained RL controllers throughout a three-week test period. In the legend, the time-series of the storage SOC (top subplot) as well as the time-series of the savings, compared to the baseline policy (central subplot), are assigned to the respective PS. PS-B1 results in a policy that resembles the "price based" RBC policy, while PS-A1 is characterized by a higher frequency of storage loading cycles. The savings are highest with PS-A1 & 2.

Table 4.8: Performance metrics of the investigated PS. The PS are evaluated in terms of the final mean episode savings, using the training dataset ($\bar{J}_{eps}^{train}$) and by using the test dataset ($\bar{J}_{eps}^{test}$), as well as by the first-year savings and the break-even time.

savings		PS-A1	PS-A2	PS-B1	PS-B2	PS-C
mean (train)	[€]	429.1	315.9	156.9	185.3	267.5
first-year (train)	[€]	4,336.5	8,785.7	2,397.2	-525.3	-10,649.6
break-even (train)	[days]	89.9	67.4	0	164.6	862.6
mean (test)	[€]	102.9	107.7	35.6	75.0	112.9
10-day (test)	[%]	11.4	11.9	3.94	8.3	12.5

On this basis, the coefficient of determination (R^2-score) is calculated to evaluate the correlation between the action probabilities and the constraints (electricity price k_{el}, ambient temperature T_{amb}, and cooling demand $\dot{Q}_{dem}$). Among the three features analyzed, PS-B1

indicates being predominantly influenced by fluctuations of the electricity price ($R^2_{x_{el}} = 0.39$, $R^2_{T_{amb}} = 0.03$, $R^2_{\dot{Q}_{dem}} = 0.07$). In comparison, when following PS-A1, no state-space feature appears to be statistically more significant than the others ($R^2_{x_{el}} = 0.17$, $R^2_{T_{amb}} = 0.21$, $R^2_{\dot{Q}_{dem}} = 0.15$). PS-A1 therefore leads to a more complex policy in which the other constraints, in addition to the electricity price, have a significant influence on the control action.

4.2.5 Discussion and lessons learned from case study two

This subsection summarizes the key results of the case study, discusses them, and also derives lessons learned for future work. This is based on the result plots presented in the previous subsection and the experience gained in the implementation process of the case study.

The results show that with a guiding RBC, it is possible to implement a RL controller that outperforms this guiding RBC. Offline training of the RL controllers (PS-B and PS-C) leads to policies similar to those expert policies the RL controllers are offline pre-trained with. However, especially in the initial phase, the RL controllers show significant lower control performances. This observation underlines the findings published in [Ross and Bagnell, 2010], regarding a growth of deviations from the offline trained policy caused by the lacking ability of the RL controller to extrapolate to unseen states. The performance of the already sub-optimal policy is further decreased by random exploration [Hester et al., 2018]. Maintaining an online training process that outperforms the RBC is only possible when following PS-B1, in which the RL controller shows higher electricity cost savings compared to the "price based" RBC. Generally, RBC guidance stabilizes the learning process in exchange for slower convergence.

The use of monitoring data for offline training (PS-C) does not lead to a promising controller performance, especially directly after the offline pre-training. The trained RL controller seems to over-fit to the sub-optimal policy in the historic trajectories. This calls for an early stopping strategy for offline RL controller training, in order to exploit the information about the system dynamics encoded in the data but at the same time avoid over-fitting to the observed policy. Another possibility is the use of more diverse training trajectories, for example if different control trajectories in different monitoring datasets for a similar system are available. Furthermore, PS-C was the most costly during the online training process. In spite of the results on the test period (figure 4.20), PS-C performs best on the test dataset on average (table 4.6: mean (test)). It is concluded that due to the costly training process, a more diverse set of trajectories has been processed leading to the ability to exploit more diverse combinations of constraints. A temporarily negative cost savings term may be tolerable, in cases where user comfort and safety requirements remain unaffected by the selection of RL controller actions. As the results show, the initial financial burden of the trial-and-error learning process is quickly compensated when following both variations of PS-A and leads to higher accumulated savings throughout the entire training process.

In the SOC time-series subplot of figure 4.20, there is a visible discrepancy between the policies learned trough PS-A1 and trough PS-B1. PS-B1 closely resembles the expert trajectories of the implemented guiding RBC. This suggests that the combined influence of offline pre-training and RBC guidance influences the policy (learned via PS-B1) throughout the entire training process and afterwards. Based on the significant differences in the correlations between the state-space features and the Q-values, it is concluded that the policy resulting from PS-B1 strongly adapted and only slightly improved the "price based" RBC policy. By comparison, the RL controller trained under PS-A1 has also learned to exploit fluctuations in the ambient temperature (T_{amb}) and in the cooling energy demand ($\dot{Q}_{dem}$).

One controller (trained under PS-A2) outperformed the RBC based on price signal forecasts, after 67 days of online training. The controller was not pre-trained offline before the first interaction with the environment. This demonstrates (for this case study) that training under real-time conditions is possible if the state-space is compact and the set of hyper-parameters is well chosen. This is a promising result, as the approach of creating a data-driven pre-training environment for a system appears only reasonable if both the quality of the monitoring data and the potential savings justify the effort.

The intuitive approach of offline pre-training of the RL controller with historical data did not yield promising results. Future research should focus on how historical data can be used for offline pre-training and how this step affects the performance of the RL controller during exploration and online training.

An essential prerequisite for the approach and the use of RL in BES (in general) is the availability of the necessary IT infrastructure. Data-points, sensors, and actuators, as well as interfaces to external data sources must be available for the algorithm. In this study it was possible to show that if these prerequisites are met RL controllers can make promising control actions few weeks after deployment. Nevertheless, increasing digitalisation and the spread of building automation and control systems (BACS) are key when it comes to bringing advanced methods from the field of adaptive and predictive control from science into practice.

The problem of the error-prone and time-consuming physical modeling, which was carried out in most previous studies was addressed, by applying state-of-the-art machine-learning algorithms, using monitoring data of the investigated system. The learned models were designed to represent the dynamics of two compression chillers and an ice storage in an interconnected cooling energy system. The modeling process is automated to a large extent and can (under the premise that high quality monitoring data are available) be applied to diverse systems.

Bayesian hyper-parameter optimization led to control performance improvements of up to 184 % and reduced the necessary interactions with the training environment from $\sim$ 800,000 to $\sim$ 125,000. Future studies should investigate how transferable hyper-parameter selections for RL controllers are. Since some of the hyper-parameters are of similar magnitude for all PS, it seems plausible that they are transferable for similar applications. The main influencing

factors here are the completeness and size of the state-space, the size of the action-space, and the delay as well as the design of the reward signal. In future work, the issue of hyper-parameter selection should always be considered. In this case, Bayesian hyper-parameter optimization proved to be a good choice. However, a sensitivity analysis can also be used as a first estimation.

Based on the study, the following lessons learned can be derived for future work:

- If the quality of the monitoring data is ensured by a modern IT infrastructure, a complete workflow of data-driven system modeling and RL algorithm training can be applied.
- Training with monitoring data without the intermediate step of building a data-driven training environment comes with several problems. If only data for a certain training policy is available, a strong tendency to over-fit to this policy can be observed. The performance of the algorithm then (especially through additional exploration) does not reach the performance of the observed policy in the training data. This problem could be overcome if more diverse policies were available for offline training.
- If RL algorithms are used for energy management and the quality of supply is not influenced by the actions performed by the algorithm, training on the real system is feasible. With a compact state- and action-space, a RL algorithm can learn a superior policy in a few weeks and thus represent an expert system for energy management.

4.3 Case study three

After the previous two case studies focused on the training and performance of RL algorithms using dynamic simulations and data-driven models, the third case study now examines an application on a real system. While case study one dealt with the application of RL for load shifting in a cooling network and case study two with an energy management application for an ice storage under dynamic boundary conditions, now a feedback control application is investigated. Special focus is given to a generalizable problem formulation for the selected task as well as to the transferability of simulation-trained algorithms to real systems. It is demonstrated how a DQN algorithm can be used for self-improving valve control.

Valve control is a recurring problem in BES control. Often, poorly tuned PID controllers lead to poor control performance and aging components resulting in (for example) changing valve characteristics, further reduce the control performance over time. Due to its adaptive characteristics and low computational requirements (after training), RL has a particular potential to extend BES automation by self-learning and self-improving software components. Therefore, an adaptive valve control for HVAC systems by means of RL is investigated. One of the biggest energy consumers in buildings are HVAC system [Yuan et al., 2019]. Thus, optimal control strategies for those systems and the associated subsystems are becoming increasingly

important for sustainable building operation [Zhang et al., 2018]. Conventional HVAC control is based on On/Off rules and PID controllers, which are comparatively easy to implement and have low initial costs [Mirinejad et al., 2008]. However, in practice, the parameters of PID controllers are often not set optimally, which significantly reduces their control performance and results in inefficient operation [Wemhoff, 2012]. Also for HVAC systems, MPC promises a nearly optimal operation but demands computing power and an accurate system model [Afram and Janabi-Sharifi, 2014]. Especially the creation of the model is uneconomic in many cases and must be repeated after every change in the system. Due to its model-free nature and recent developments in deep-learning, optimal HVAC control with (model-free) RL is also considered within this context as a promising alternative [Wang and Hong, 2020]. The low computational costs (after training) and inherent adaptiveness promise significant advantages over other methods [Görges, 2017].

The investigated task is to control the outgoing air temperature of a cooler by adjusting the valve of the water/air heat exchanger with respect to the incoming air temperature and the setpoint temperature. Solving this problem via RL requires the definition of a MDP which should be as generic as possible, to allow the transfer from a simulation to different real-world heat exchangers.

Besides the application to different real-world scenarios with different hydraulic characteristics, transferability is also important because the simulation model generally reflects the reality only to a limited extent, therefore an adaption to the actual system dynamics is necessary. The hydraulic system determines the effect of a change of the valve position on the outgoing air temperature [Recknagel, 2017]. The underlying hydraulic system of the cooler is a throttle circuit [Recknagel, 2017]. The RL algorithm training as well as the final control script are implemented on an ordinary PC. The communication between the PC and the HVAC automation is realized via a REST API, a database, and a MQTT broker.

Figure 4.21 shows the hydraulic scheme of the considered AHU and figure 4.22 the available data points of the investigated subsystem. The AHU provides temperature-controlled fresh air to a workshop and a laboratory. In addition to the heat exchangers shown, the components (a to d) show flaps, fans, and filters along the air inflow and outflow. In the next section, the considered subsystem and the control task are discussed in more detail.

4.3.1 A generic problem formulation for AHU valve control

As shown in figure 4.21, the incoming air with temperature T_3 streams through the cooler before it enters the reheater. Both heat exchangers are connected to different supply water circuits. The task is to learn to control the cooling heat exchanger outgoing air temperature T_4 (like a PID controller) with respect to a setpoint signal under the dynamic environment conditions (incoming air temperature and cooling water temperature). A further influencing

factor is the humidity, but since this is not measured directly after the cooler, the combination of incoming air temperature and valve position leading to a certain outgoing air temperature must be somehow differently be encoded in the state-vector. In the cooling case, the setpoint temperature which can be set in the laboratory and the workshop directly corresponds to the setpoint temperature behind the cooler. In figure 4.22, the temperature with the highest influence are the temperature of the incoming air (T_3) and the temperature of the cooling water (cooler (T_{w1})). The reward signal is designed to solely depend on the deviation of the outgoing temperature from the setpoint: $r = f(|T_t^{set} - T_t^{out}|)$.

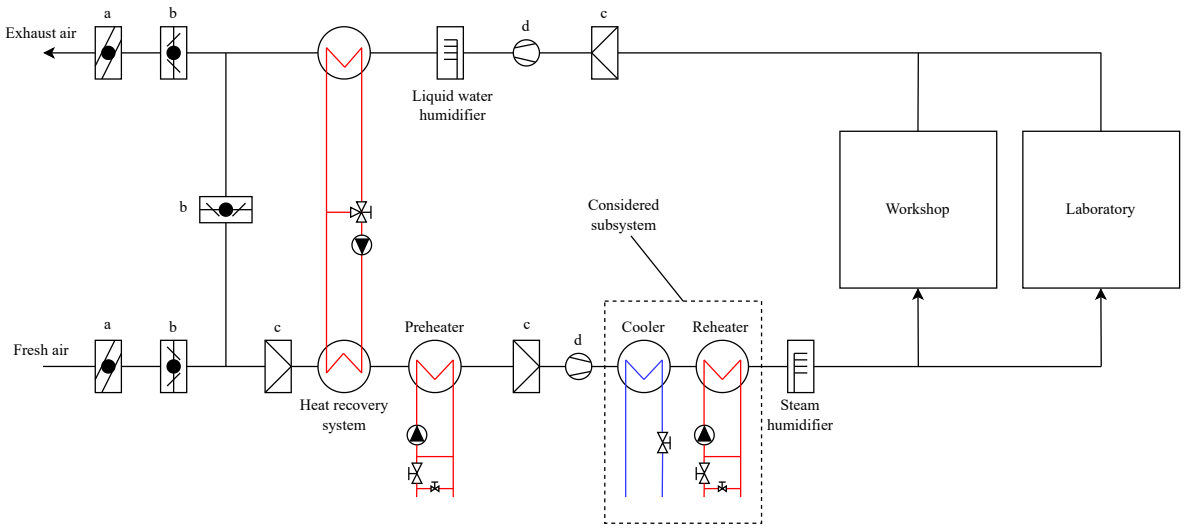

Figure 4.21: The hydraulic scheme of the considered AHU. The AHU supplies a laboratory and a workshop with conditioned fresh air and has a heat recovery system, a preheater as well as a cooler and a reheater. For this case study, the throttle valve at the cooler is controlled.

Figure 4.23 shows the reward function. Maintaining the temperature in a range of 1 °C around the setpoint leads to the highest rewards. All sections of the reward function have a slope higher than 0 to steer the algorithm in the direction of the smallest deviations. If the discount factor is higher than 0, the Q-values represent not only the immediate reward but also the estimated future rewards resulting from an action. Therefore, it is assumed that the DQN will learn a policy that also avoids overshooting. The algorithm can take three possible actions; it can open, hold, or close the valve: $a_t = [+X, \pm 0, -X]$. This has two advantages over the specification of fixed valve positions. Firstly, the online interaction with the real system is safer since the algorithm is not able to select extremely different valve positions from one time step to the next. Secondly, the MDP is of a much lower dimension, due to the smaller action-space, and therefore the training is less data-demanding. On the other hand, the selection of X requires a trade-off between strong intervention in extreme situations and a granular control.

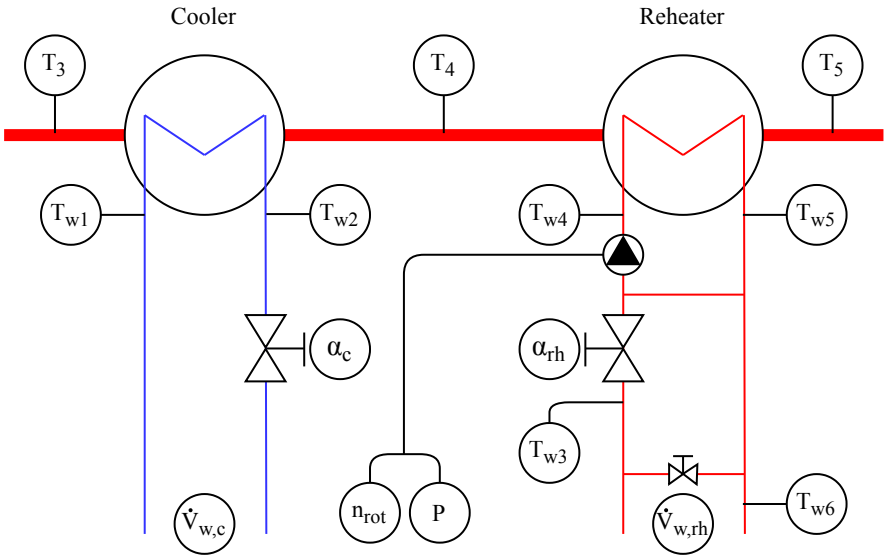

Figure 4.22: The hydraulic scheme of the cooler and the reheater subsystem with shown sensors and actuators.

The definition of the state-vector is a key challenge, since not only must it include all information necessary to solve the control problem, but it should also be as compact as possible (to reduce training times), and should be generic enough to transfer the applicability of the algorithm from the simulation to different heat exchangers. Including all available data-points, together with historical values (to learn the dynamic of the thermal inertia) would result in a very large and over-specified state-vector. Taking these requirements into account, the state-vector is defined as follows:

$$s_t = [T_t^{set} - T_t^{out}, T_{t-1}^{set} - T_{t-1}^{out}, T_{t-2}^{set} - T_{t-2}^{out}, T_{t-3}^{set} - T_{t-3}^{out}, T_{t-4}^{set} - T_{t-4}^{out}, a_{t-1}, a_{t-2}, a_{t-3}, a_{t-4}, a_{t-5}]$$

This state-vector incorporates the deviation from the setpoint in the current and the four steps before as well as the last actions, selected by the algorithm. This makes the state-vector generic because the deviation is not dependent on the temperature levels of the heat exchanger. The past selected actions provide the algorithm with information about the effect of its actions on the outgoing air temperature under unknown boundary conditions in a generic way.

Figure 4.24 visualizes the state-vector. The dotted line plots a setpoint of 21 °C and the black line plots an exemplary course of an outgoing air temperature. The red lines and the blue lines mark the deviations from the setpoint.

4.3.2 Deep Q-Network training

For the pre-training, a simple Modelica simulation model of a water/air heat exchanger with an admixing water circuit is used. The used model is freely available in the AixLib GitHub

repository [Müller et al., 2016] and provides the physical relationships sufficiently for this purpose. The simulation model is exported as a FMU [Blockwitz et al., 2012] and is connected to the algorithm following the standardized OpenAI Gym interfaces [Brockman et al., 2016]. The incoming air temperature is designed as a sinus function, which is initialized with random parameters, each training episode. The incoming air temperature varies between 1 and 10 °C below the setpoint with different frequencies. The setpoint is constantly set to 21 °C. One episode consists of 50 simulated minutes with one interaction every other minute. X in the action-space is set to 10.

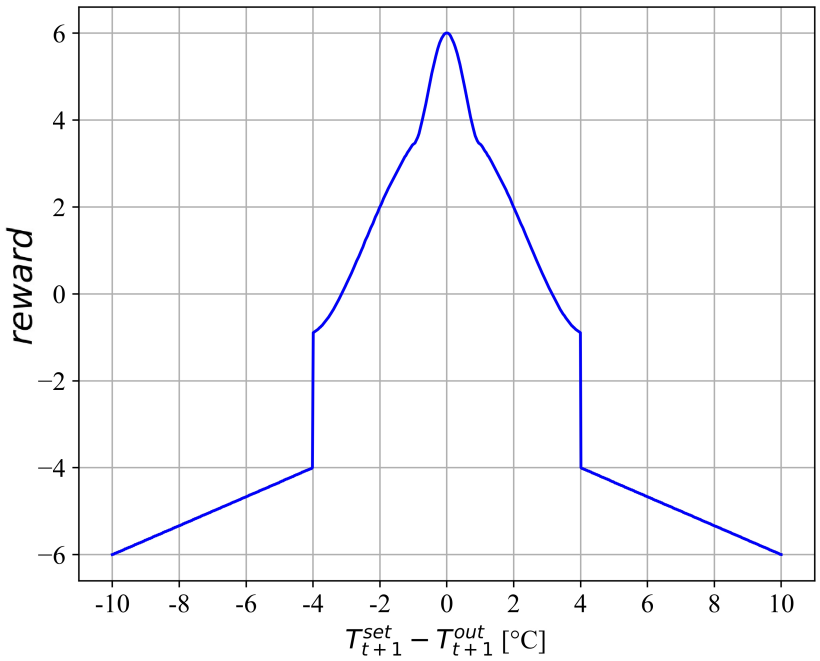

Figure 4.23: The reward function in this case study is designed to lead the algorithm with linear rewards in the direction of 4 °C deviation from the setpoint and leads the algorithm here with higher gradients in the area of minimal deviations.

For this setup, HyperOpt [Bergstra et al., 2015] is used to identify the optimal hyper-parameters for the used DQN via a Bayesian hyper-parameter search. The resulting hyper-parameters are:

- batch size (64),
- replay buffer size (10,000: approximately 167 hours of interaction),
- minimum replay buffer size (480: training starts after 8 hours),
- target-network update frequency (30),
- number of neurons (30),
- and discount factor γ (0.36).

The exploration versus exploitation trade-off is handled via ϵ-greedy: The probability to select a random action after every interaction is reduced by 0.001.

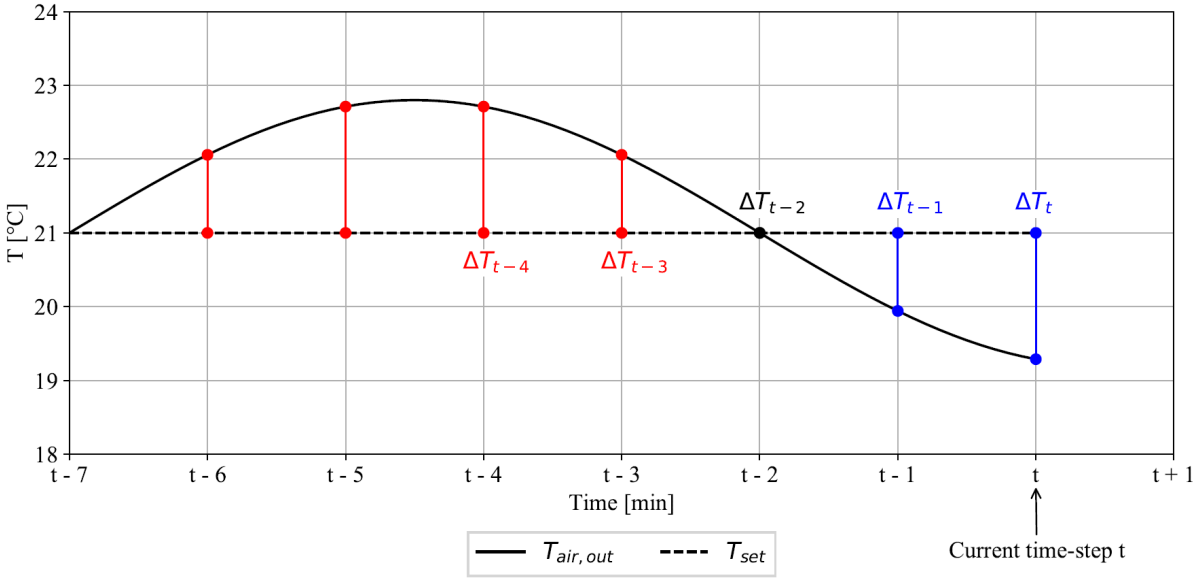

Figure 4.24: Visualization of the state-vector, consisting of the temperature deviations from the setpoint. Here exemplarily represented by the positive deviations from the setpoint (21 °C) for the time-steps t-7 to t-3 and the negative deviation from the setpoint for the time-steps t-1 to t.

Figure 4.25 shows a positive trend in the rewards in the course of 10.000 training episodes. Considering the limiting influence of the randomly initialized input air temperature, the algorithm significantly improves in exploiting the rewards from the training environment from the beginning. The training is stopped at this point to avoid over-adaption to the simulation and to keep a sufficient degree of adaptability in the online interaction phase. It is also visible how the optimal selection of the DQN hyper-parameters not only accelerates the algorithm's search for the optimal policy, but also stabilizes the training significantly. While the DQN with optimized hyper-parameters follows a stable policy after 2,000 training episodes, the default hyper-parameters lead to a significant performance drop after 2,000 episodes and the maximum performance is also reduced. Generally, the positive trend also indicates that the MDP, consisting of a reward signal, a state-space and an action-space, generally provides the appropriate information for the algorithm to improve its policy over the course of the training episodes.

4.3.3 IT infrastructure

The IT infrastructure is shown in figure 4.26, which illustrates the communication between the optimization script and the AHU. The central tool for the interaction is a cloud-based data

infrastructure [Bode et al., 2019], whose main components are a time-series database, a REST API, and a MQTT broker. The time-series database contains all actual measurements and setpoint values of the AHU automation. For communication with the database via HTTPS, the REST API provides standardized interfaces. The MQTT broker is used for the secure and high-performance communication between the AHU automation and the platform via the MQTT protocol. The MDP and the DQN are implemented as a script on an ordinary PC. For the interaction via MQTT, an industrial PC (data logger) is integrated into the local automation system which exchanges actual measurements and setpoint values with the AHU automation and acts as a translator between the local BACnet protocol and the MQTT protocol. With this setup, it is possible to control the system remotely via the internet and to perform the experiments without being physically close to the system.

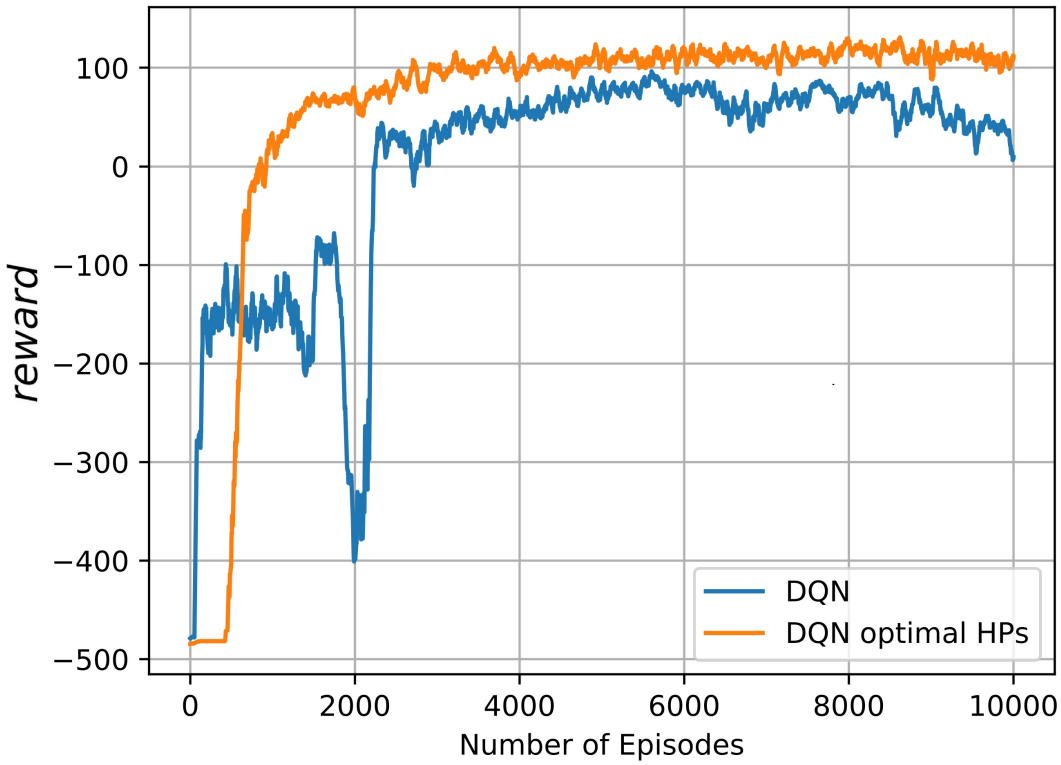

Figure 4.25: Visualization of the DQN learning curve with optimized hyper-parameters. The influence of hyper-parameter optimization on the training of the algorithm is clearly visible. While the algorithm with default parameters shows several partly heavy performance drops, the DQN with optimized hyper-parameters shows a rapid performance increase and stabilizes at a higher level after 3000 episodes.

4.3.4 Results of case study three

Online valve control

Figure 4.27 shows the development of the average reward of the DQN in interaction with the real cooler valve (top subplot) and of the associated exploration (bottom subplot). A positive trend in the average rewards is observed. Due to the selected minimum replay memory size, the DQN controls the valve based on the initial policy from the simulation pre-training in the first eight hours. During this period, the standard deviation (shaded area) is relatively high compared to a later interaction where the policy is updated using data from the online interaction.

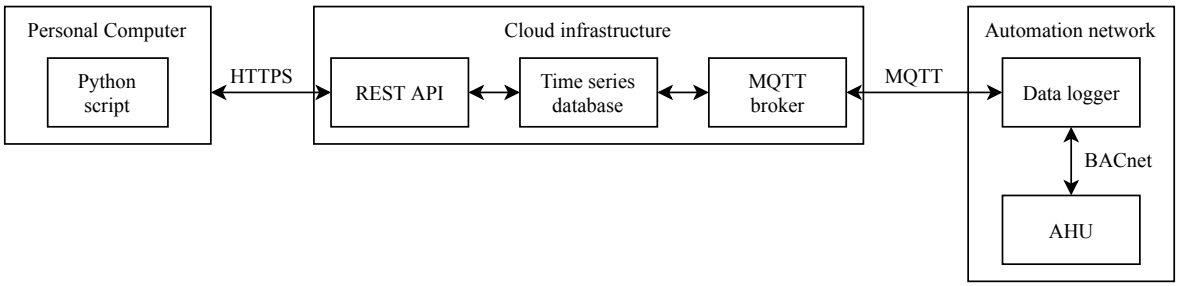

Figure 4.26: The required IT infrastructure. The state variables from the AHU are translated from the BACnet protocol into the MQTT protocol via an industrial PC (data logger) in the automation network and sent to a MQTT broker. This broker writes the data to a time-series database where it is then available to a personal computer running a Python script and accessing it via HTTPS, using a REST API. The setpoints from the script are written back to the AHU automation network via the same path.

After ten more hours of interaction, the rewards drop, increase again in the following 30 hours, and after 60 hours, the rewards rise to 5 with a significant lower standard deviation. A high standard deviation indicates a high fluctuation of the outgoing air temperature, a decreasing standard deviation can therefore be interpreted as a reduced oscillation around the setpoint. The real heat exchanger reacts more sensitively to valve opening changes than the simulated heat exchanger. Therefore, $a_{cooler} = [+3, \pm 0, -3]$ is chosen. Further, using the DQN for the cooler requires to negate the deviations from the setpoint in the cooler's DQN state-vector, because unlike with the simulated heating water/air heat exchanger, opening leads to a temperature reduction and closing to a temperature increase. The DQN shows the expected behavior: if the temperature is too high, the algorithm increases the valve opening, whereas the valve opening is decreased when the temperature is too low. On the other hand, the DQN starts with strong oscillations around the setpoint.

Figure 4.28 shows the learned Q-values with respect to the current deviation from the setpoint. The current deviation from the setpoint is displayed on the x-axis while the y-axis shows the

available actions. The surface illustrates the Q-value of each state-action pair. The red line shows the greedy action. In spite of the initial policy, where the algorithm does not change the valve position if the deviations are smaller than 1 °C, the control behavior improved. There is no deviation from the setpoint where the DQN does not change the valve position. Naturally, the Q-value and thus the action in a given state depends on more factors than the current deviation (namely the history of deviations and actions), but the visualization in the form of such a 3D Q-value plot is a good option to show the relationship between values in the state-vector and their influence on the Q-values. The red line shows the highest Q-values in each state, and thus the action that would be selected by a greedy policy.

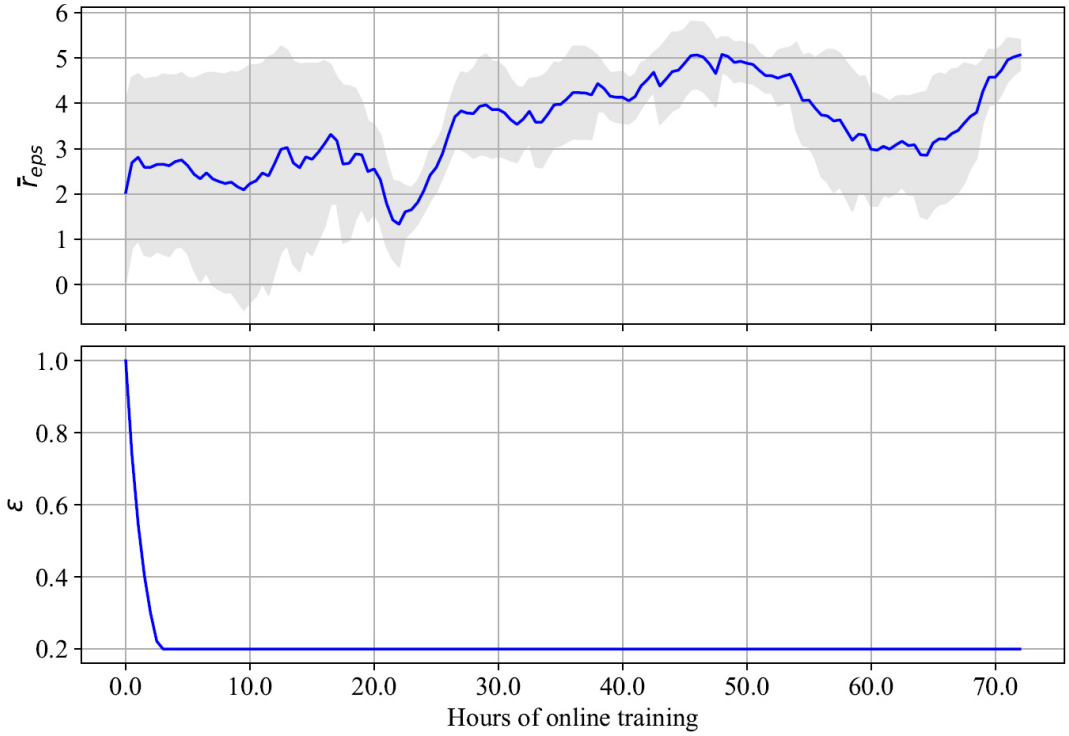

Figure 4.27: Rewards and exploration during online interaction. The upper plot shows the progression of the reward, averaged over one hour of interaction. The lower plot shows the exploration value, decreasing to 0.2 during the first 5 hours.

Additionally, figure 4.29 shows the plot of the temperatures and valve positions on the last day of online training. The outgoing air temperature no longer oscillates around the setpoint. At the beginning (07:00 - 09:00), the temperature is slightly below the setpoint. However, the DQN correctly recalibrates the temperature firstly by gradually reducing the valve position (09:00 - 10:00), then gradually increasing the valve position again as the water temperature rises from 14 °C to 15.5 °C (10:00 - 11:00). From that point on, the DQN selects to alternately increase and decrease the valve position. After 70 hours, the algorithm has already adapted to the real valve and learned to keep the temperature within a good range around the setpoint temperature. It is worth mentioning that, in contrast to its initial policy, the DQN now

only very rarely chooses the action of not changing the valve position, which leads to good temperature maintenance on the one hand, but also to a high number of valve position changes on the other, which is not optimal in terms of maintenance-free operation.

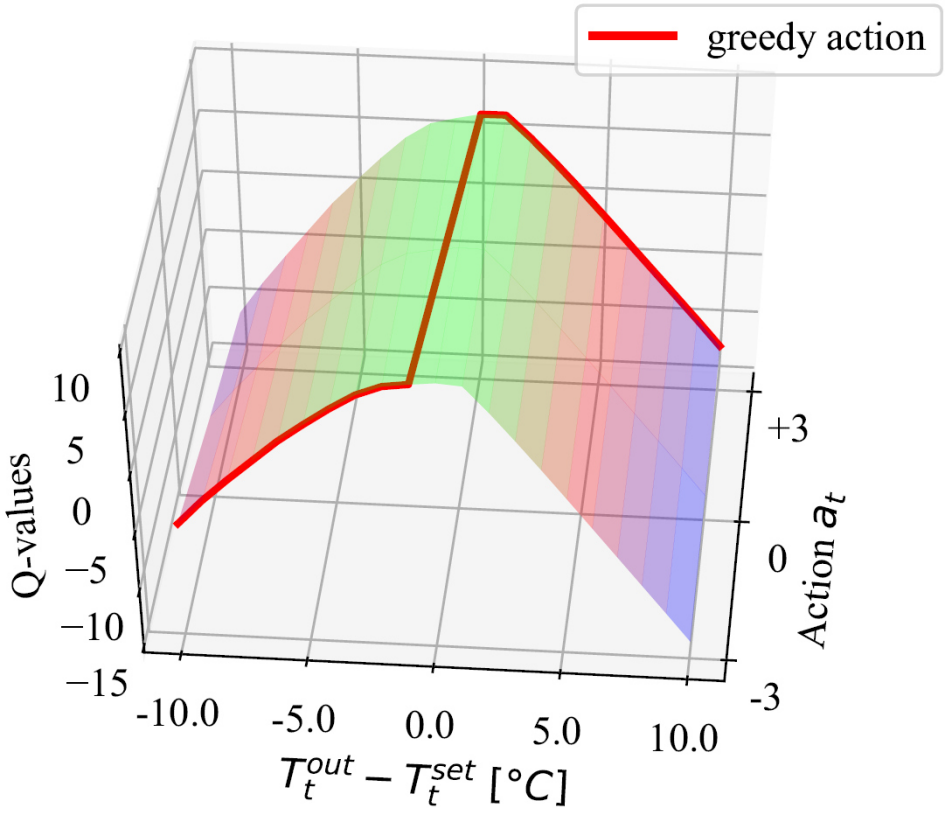

Figure 4.28: Three dimensional representation of the Q-values of the three possible actions for different deviations from the set temperature. The greedy action (the action that is selected without exploration based on the highest Q-values) is plotted in red. It can be seen that the algorithm has learned to open or close the valve according to the deviation.

Setpoint jump

In order to compare the control behavior of the DQN after 70 hours of online interaction against an established control using a PID, a setpoint jump from 18 °C to 16 °C is performed with both variants under similar boundary conditions. The results of this setpoint jump experiment are shown in figure 4.30. The PID parameters are set manually, achieving a significantly better control performance than the setting according to the established method of Ziegler and Nichols [Abel, 2018]. Nevertheless, it can be observed that the PID leads to an oscillation around the setpoint. A similar situation, but with higher frequency and amplitude, can be observed with the DQN: Although the algorithm that performed well after 70 hours is used here, the set temperature is now barely maintained. It should be noted that the supply

temperature is now cooler than in the period shown in figure 4.29. This indicates that the behavior of the real valve is not yet sufficiently well-learned in all areas of the state-space and further training would be necessary to obtain a better control performance even under more varying boundary conditions.

It is noticeable that both the PID and the DQN stop oscillating after the setpoint jump and strive towards a good maintenance of the setpoint temperature. Since the difference between the cooling circuit and the setpoint temperature is smaller after the setpoint jump, changes in the valve position show a smaller change in the output temperature, so the system is less sensitive in this state, easier to control, and the differences in the state-vector are closer to those from the period in figure 4.29. For the DQN and the PID, the summed deviation from the setpoint, and thus the control performance for this setpoint jump, is calculated ($J_{abs} = \int |T_{set} - T_4| \; dt$):

- $J_{abs}^{DQN} = 126.17$
- $J_{abs}^{PID} = 81.28$.

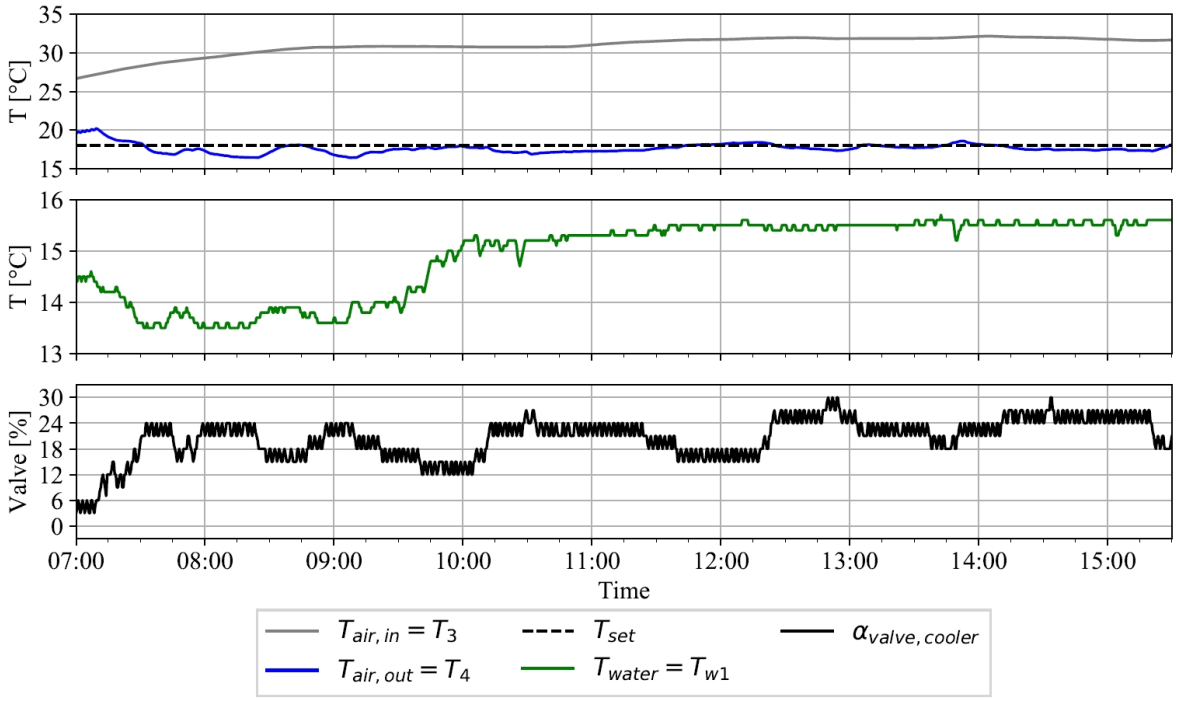

Figure 4.29: Undisturbed policy after 70 hours of online training. Plotted are the outgoing air temperature of the AHU and the incoming air temperature before the cooler, as well as the set temperature (dashed). In the middle plot, the cooling water temperature is shown as a significant influencing variable. The lowest plot shows the valve position set by the algorithm. It is noticeable that the algorithm almost never maintains the valve position, but achieves the stable temperature by alternately opening and closing the valve.

The results show that the DQN does not yet reach the control performance of a manually calibrated PID after 70 hours of online interaction. In particular, the strong oscillation before the setpoint jump and a larger (although generally small) constant deviation from the setpoint after the jump degrades the result.

Therefore, it has been shown that by pre-training with a simulation, a DQN significantly improves its control behavior after 70 hours of operation on a real valve. However, the real behavior with its thermal inertia under varying boundary conditions is not yet sufficiently well-adapted in the policy of the DQN to outperform a manually calibrated PID in its control performance.

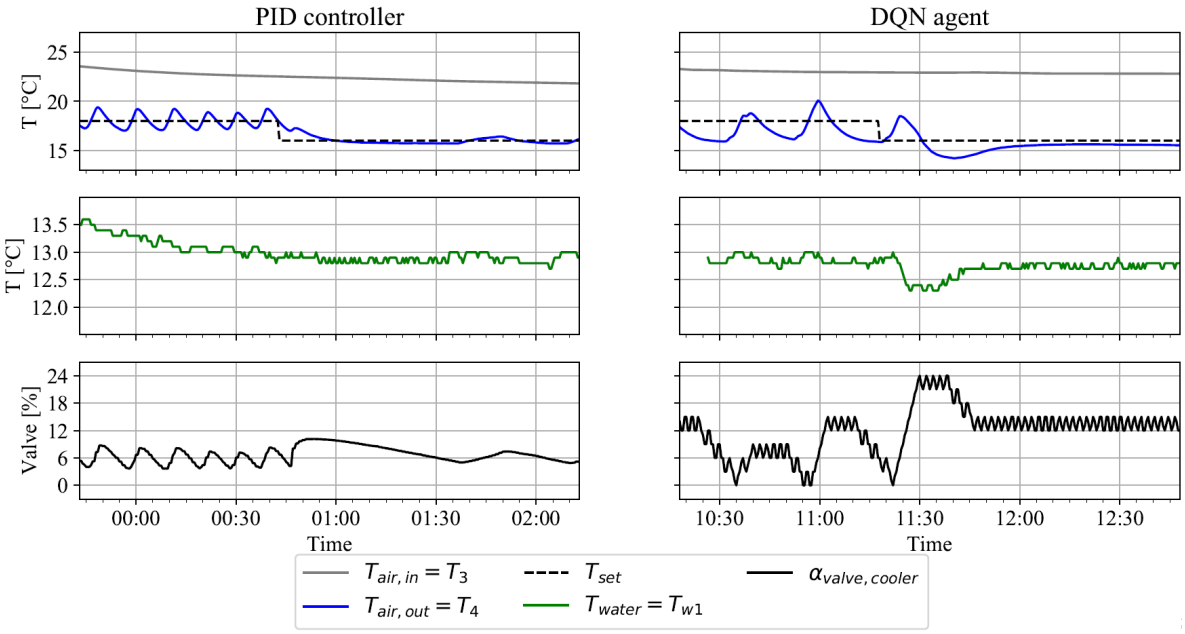

Figure 4.30: Setpoint jump comparison between DQN and PID control. In both cases, the system starts oscillating around the setpoint. Both the PID controller and the DQN algorithm can control the system to reach the setpoint after the setpoint jump. However, the alternating valve position is visible again in the valve position plot of the DQN.

4.3.5 Discussion and lessons learned from case study three

The presented results show that with the introduced MDP formulation, a pre-trained DQN can significantly improve the learned policy within 70 hours of online interaction. In contrast to the beginning, there is no oscillation around the setpoint on the last day of online training. Against this observation, changes in the boundary conditions led to significantly lower control performance, therefore it is expected that there is still a high potential for improvement. During the pre-training, the DQN was trained with 75,000 interactions, while 70 hours of online training are equivalent to 4,200 interactions. The fact that the learned

DQN valve control policy improved visibly in such a short time highlights the benefits of the compact formulation of state- and action-space. Interestingly, without its own past actions, the algorithm is not able to control the valve correctly. It is concluded that this is important for the DQN to distinguish between its own influence and the influence of changing boundary conditions. In order to realize a more precise control, it could also be of interest to extend the action-space with more control options ($A = [+Y, +X, \pm 0, -X, -Y]$) or directly use a continuously controlling algorithm, like the SAC algorithm (with the risk of extreme action during online training).

This case study contributes to the development of self-calibrating building automation systems. It has been demonstrated that with a generic MDP formulation, a fast adaption of a pre-trained DQN algorithm to a real hydraulic system is possible. The results show that the time required for online training can be significantly reduced by pre-training. In the case of the real cooler valve, the simulated heat exchanger has both an opposite dynamic and a different hydraulic system. After oscillating around the setpoint at the beginning of the online training, the outgoing air temperature was maintained at the setpoint at the end, without oscillating. At the same time, the setpoint jump experiment showed, that there is still a high potential for improvements. The application of Long Short-Term Memory (LSTM) function approximators could be investigated to move historical data from the state-space to the internal memory of the neural networks. Additionally, the pre-training process could be further refined by running specific scenarios such as setpoint jumps.

Based on the study, the following lessons learned can be derived for future work:

- Through a generic formulation of recurring control tasks and pre-training with simulations, it is possible to train RL algorithms that adapt to the behavior of real systems within feasible training times.
- Besides the history of state-values from the environment, it is important to include the history of the selected actions in the state-vector. This is important for the algorithm to learn the relationship between its own actions and the state change in the environment.
- With a suitable automation infrastructure, extended by a cloud platform, it is possible to establish a direct communication between state-of-the-art RL Python implementations and a BES control. Besides implementing the trained algorithms directly on the BES automation hardware, this is a way to detach the operation logic from the BES location and make use of higher programming languages during implementation.

5 Discussion

In this chapter, the application of novel RL algorithms for different BES application areas is discussed. Based on the results of the conducted case studies and the derivation of the potential based on the literature, different aspects are addressed. These include the performance of the methods and typical RL design principles, as well as aspects of practical applicability, future challenges in science and practice, and possible future products.

5.1 Reinforcement Learning control performance for energy management tasks

In general, the investigations carried out in case study one and case study two confirm the promising results that have also been published for other application examples. In both cases, state-of-the-art algorithms were used to exploit flexible elements in the BES for energy management with respect to dynamic boundary conditions. Using the design principles presented in chapter 3, in both case studies RL algorithms were selected, parameterized, and trained to solve the given optimal control problem. The two case studies, which were published in different scientific journals and conference articles, addressed some of the more pressing research questions currently being discussed in the scientific community.

For case study one, the simulation model, modeled based on a real cooling network, was used as a training and evaluation environment for state-of-the-art RL algorithms. While the pipe network itself was physically modeled, real monitoring data was used for consumer load data, electricity prices, and weather data. The state-space can be considered comparatively large, since all data points (environmental values, historical values, and forecast data) were initially passed to the algorithms unprocessed in each state of the environment. The two compared algorithms, widely used in the literature (DQN and DDPG), are typical representatives for discrete and continuous action-spaces. Only a slight superiority of the DDPG was observed. This suggests that dicretized action-spaces can also obtain good results if the control problem is well-formulated. The learned policies are able to exploit the dynamics of the thermal masses in the considered cooling network under time-variable boundary conditions. Depending on the capacity, flexibility can be exploited, saving 14 % operation costs on average. The learned policies are based on generalized relationships and lead to meaningful control actions under unknown test boundary conditions. However, only the adaptation of the exploration in the course of a hyper-parameter optimization in the discrete case (DQN) and the use of an adjusted algorithm in the continuous case (SAC) led to training times that would have

been practicable in interaction with the real system. Thus, state-of-the-art RL algorithms are indeed suitable to map high-dimensional state-vectors to optimal actions, but in order to keep the training times within practical time frames, it is necessary to use the latest algorithms with well-chosen hyper-parameters. Although the results of the simulation study in case study one are very promising, further investigations are needed regarding the interpretability and transferability of RL algorithms in BES. For interpretability, correlations between the features in the state-space and the Q-values that determine the control action should be investigated in future studies. Thus, it can be visualized what kind of correlation is encoded in the state-action-value function. Regarding the transferability of RL algorithms to similar control tasks, two promising approaches are proposed: Firstly, state-space definitions with relative and thus generic features can be chosen, which favors to adapt the pre-trained state-action-value function to similar problems. The second alternative was elaborated in a comprehensive literature review [Wang and Hong, 2020]. The authors have elaborated that on-policy algorithms are particularly suitable for transfer learning, which deals with the transferability of trained neural networks to other application areas. This feature should be used in future work to apply trained algorithms to similar problems. When it comes to interactions with real systems, one promising approach is also the targeted manipulation of hyper-parameters during the interaction. For example, exploration and learning rate can be increased in a systematic way after the reward signal drops under known conditions. The adaptivity of the algorithm can thus be purposefully controlled during operation.

In case study two, the creation of the training and evaluation environment itself was data-driven. By modeling the energy supply system, consisting of two compression chillers and an ice storage, a complete workflow from monitoring data to the trained RL algorithm was demonstrated. The task for the RL algorithm was to use the ice storage optimally under dynamic electricity prices and outdoor temperatures. To get from the monitoring data to the trained algorithm, five different pre-training strategies were compared and evaluated. The training procedures include offline and online training as well as variations with and without guiding RBC. The case study contributed to the objective of enhancing the real-world applicability of RL by demonstrating a fast learning approach on the one hand and an approach that outperforms a guiding RBC from the first moments of online interaction on the other hand. The latter represents a safe option for cases where failing to meet a baseline is intolerable. Against the initial assumptions, with pure try and error online training, the highest cost savings and the fastest convergence was achieved with the used DQN with well selected hyper-parameters. By a compact formulation of the forecast data in the form of statistical features of the time-series, the dimension of the state-vector was kept comparatively small. Therefore, a try and error online learning on the real system would lead after a few months to a policy that outperforms a well-designed RBC policy. The goal of adaptive BES management systems, which are both robust on the one hand and able to cope with changing environment conditions on the other hand, is therefore achievable when using compact MDP

formulations. The intuitive approach of using monitoring data for training did not produce promising results. The algorithm quickly adapted to the suboptimal policy in the training data, an effect that had to be compensated by extensive exploration during the online training. Although the use of monitoring data for the training of algorithms seems to be promising, there is a crucial challenge. When value-based algorithms are trained offline with monitoring data, there is a tendency for the algorithms to over-adapt to the observed policy [Hester et al., 2018]. This should then be addressed with an early stopping strategy within the offline training process, which should be based on the measurement of the adaption of the neural networks, internal weights or the tracking of the course of the reward signal. Seventy weeks (case study one) might be too little data in this case, thus it would take more data, especially with different observable policies, to extract a superior policy.

When applying the trained algorithms in real systems, some additional difficulties are expected. The required system data must go through extensive pre-processing to reduce the data to the physical relationships, to reach a similar quality as observed from the simulation. The used sensors have to be monitored carefully to replace faulty sensors as soon as possible. In such situations, the training of the algorithms must be stopped in order to avoid adaptation to physically implausible system behavior. The data for the prediction of the boundary conditions must be available to the algorithm. For this purpose, an interface to the respective database operators must be available. The uncertainty in the forecasts of the boundary conditions is not considered to be very critical, because the control decision is continuously corrected, similar to MPC.

An important limitation to the practical application of RL for energy management applications is the availability of dynamic electricity price tariffs for end customers. Although the algorithms can also be used to improve the efficiency of BES under dynamically changing usage and outdoor temperatures, the full potential is realized (as with MPC) when energy can be purchased at certain times at particularly good conditions. This was assumed in both case studies. Another limitation may arise from the fact that higher programming languages often cannot be executed on state-of-the-art building automation hardware. The automation hardware should ideally be able to execute higher programming languages or (if not) should have an internet connection to calculate the optimized actions independent from the location on a different hardware. The potential of RL algorithms for BES energy management applications has been shown in the literature as well as in the conducted investigations. By identifying recurring tasks, pre-trained algorithms could be provided in the future that can adapt to the individual conditions in specific application scenarios in a manageable time. If monitoring data from different application scenarios with different operation policies are available in large quantities for a BES, offline training procedures can become of interest again, since over-fitting to a single specific, suboptimal policy can then be avoided. In addition to pre-trianing, compact problem formulation has shown to be effective for rapid training. To enable the coordination of multiple BESs in an area in the future (for example, for co-

ordinated power purchase), the exploration of RL algorithms with multiple agents is also proposed. Such a system could reward coordination and thus avoid all RL-controlled BESs drawing power at the same time [Vázquez-Canteli et al., 2020]. Following these proposals, a self-calibrating, self-optimizing energy management system becomes possible, which could reduce manual engineering in the future.

5.2 Reinforcement Learning for feedback control automation tasks

The experiment which was conducted in the course of case study three shows a less clear picture compared to the application for the case studies focusing on energy management tasks. Also in the literature, while RL for BES energy management has been investigated with increasing interest in recent years, the application for automation and feedback control on the actuator level remains behind in the scientific literature. While RL is used for these purposes in the fields of robotics or autonomous driving, applications in the building sector mostly remain on the level of energy management.

Case study three has shown that the workflow of pre-training on a generic simulation model in combination with a generic MDP formulation leads to promising online adaption times, where the trained algorithm can be adapted to a real valve within a few days. On the other hand, the trained DQN algorithm could not yet outperform a manually calibrated PID controller in its control performance in a setpoint jump experiment. Further investigations are necessary to find the optimal combination of algorithm, hyper-parameter set, and MDP formulation, which by training on the simulation, not only improves for the special model, but also leaves potential for fast adaptation to real systems. If this is achieved, there is a high potential for product manufacturers and automation companies, which could provide a self-calibrating, self-optimizing valve controller after a one-time engineering effort.

In contrast to energy management applications, however, the potential is rather ambiguous. While optimal control methods are clearly recommended with dynamic electricity prices or for optimal operation of BES storage capacity under fluctuating outdoor temperatures, good BES automation at the actuator level can also be achieved with well-adapted PID controllers. However, in practice, the controllers are often set under time pressure on construction sites, following standard procedures, therefore there is great potential for the use of automated calibration methods. Here RL can be a promising alternative, but other methods are also feasible, such as automated Bayesian calibration of the PID parameters in the initial phase of the BES. As for energy management applications, modern IT and automation infrastructure is crucial. This can be achieved by a local hardware which can execute higher programming languages or by outsourcing the automation to the cloud in the context of an IoT-based BES automation.

5.3 Importance of Markov Decision Process formulation

In the course of the case studies, it was found that the formulation of the optimal control problem is of major importance in order to be able to use state-of-the-art RL algorithms for BES control applications. The assumption that RL algorithms are simply connected to sensor data and actuators and only improve in the selection of their actions unfortunately does not hold. Crucial is program code, which is executed between the sensors and the algorithm and translates the sensor data from the BES to be controlled into a MDP, which can be processed by RL algorithms, and translates the actions of the algorithm into corresponding actuator signals.

For an efficient and error-free processing of the MDP, it is important that it represents all influencing factors of the targeted control problem. To satisfy the Markov Property [Sutton and Barto, 2018], the future of a MDP, i.e. the further development of its states and its reward signals, should only depend on the current state and the policy of the RL algorithm. However, since BES depend on a large number of influencing factors that are difficult to predict and, in some cases, unobservable, the MDP is usually only partially observable. Another aspect is the inertia in thermal systems, which leads to the property that actions (like increasing the heating power) become observable in the sensor data (i.e. in the state-vector) only after some time. A pragmatic solution that appears repeatedly in the literature, which was also successfully implemented in case study one and three, is the integration of historical state and action values from past time-steps in the current state-vector. This makes not only the current state but also the current dynamics of the MDP observable for the RL algorithm. External influencing factors that are not directly observable in the sensor data are also encoded in the history and can be taken into account as indicators by the algorithm for its actions. Other options to consider the history of the MDP in the current state is to include gradients of the sensor values time-series into the state-vector or in the use of recurrent neural networks, which themselves have an internal memory for past input data.

A simple linear reward function has also shown to be important for guiding the algorithm. Although RL algorithms can also be trained with MDPs in which there are only infrequent rewards, the search of the RL algorithm for the optimal policy is much more difficult and many policies have to be tried until it can be learned which action in which state leads to a later reward. Therefore, the reward function should be guiding, in the sense that small rewards lead to larger ones in optimal operation states. For example, minimizing the deviation from a target temperature value can be rewarded in an inversely-proportional way to the temperature deviation. Long chains of quadratic elements of the reward function seem problematic. While quadratic reward functions are suitable for many MPC approaches, the RL algorithm's lack of observability about the function can lead to divergence when signals fluctuate widely. Since the reward function is part of the MDP black-box, unlike in MPC, large changes in the quadratic function terms here can lead to unstable training and eventually poor policies. It is

recommended, similar to case studies one and two, that simple concatenations of the target variables with simple weights should be used. Forecast data, used in both case studies, also proved to be promising. While RL methods also promise some forecasting capability, all the data needed for an informed forecast (e.g., outdoor temperature) would be too complex to process for the problem, so integrating available forecast data is more appropriate.

In summary, the MDP should be designed such that the information contained in the state data is sufficient to predict well the evolution of the reward signal over the optimization horizon. Then, RL aLgorithms can learn and compute an optimal policy (encoded in a sequence of actions) for the respective MDP.

5.4 Importance of state-of-the-art algorithms, design principles, and hyper-parameter optimization

In all case studies conducted, state-of-the-art RL algorithms from the family of value-based algorithms (Q-Learning) were used for discrete control tasks and hybrid algorithms between value-based and policy-based algorithms for continuous control tasks.

In case study one, the comparison of the continuous controlling DDPG and the discrete controlling DQN showed that a discretized action-space can also lead to comparably promising results. By using the DDPG, which can be seen as an extension of the DQN for continuous control tasks, two similar algorithms were tested for the same problem. While significant savings could be achieved with both algorithms, only the optimization of the exploration led to training times that would be feasible in practice on real systems. This confirms the statement that it is problematic that many recent studies on RL for BES do not use the latest algorithms and design principles [Wang and Hong, 2020]. Subsequent to the direct comparison of the two basic algorithms, significant improvements in training times have been achieved through improved algorithms and optimized hyper-parameters. The used DQN is continuously developed further and the extensions which are available in the form of robust implementations should be used. Currently, this concerns the use of a replay buffer, the use of target-networks, prioritized replay, and dueling networks. In the case of case study one, however, the most significant improvement was achieved by optimizing the decrease of the exploration, which was significantly too slow in the base implementation. For the continuous case, the use of SAC instead of DDPG led to a similar improvement. SAC can be seen as an extension of DDPG with a stochastic policy and an entropy-based exploration mechanism. These extensions resulted in similar fast convergence for the case study one system as it was achieved using the optimized DQN, but without any adjustment of the exploration. Thus, for the specific application, consideration should always be given to the MDP (discrete or continuous action-space), and based on this, the latest algorithms should be used for the case under consideration. No algorithms from the area of model-based RL algorithms or from the

area of direct policy optimization were used in the course of the investigations carried out. Model-based algorithms were intentionally excluded because the transition to data-driven MPC is fuzzy and the algorithms share the same strengths and weaknesses. Nevertheless, future work can carry out valuable investigations at these interfaces and address the question for which applications in the BES area it makes sense to provide the required higher computing power in order to achieve even better optimization results by model-based methods. As discussed in section 3.3, algorithms from the area of direct policy optimization can be useful for certain use cases. The PPO is the most popular algorithm from this family and achieved comparable results to the DQN in many studies. However, the limitations that no offline training is possible and that the policy must always be applied for a complete episode before it is updated are disadvantages compared to the DQN, which in turn has no disadvantageous compared to the PPO in terms of convergence speed and performance. The selection of algorithms in the course of this work thus follows the trends that can be observed in current studies in the field of RL.

In all three case studies, hyper-parameter optimizations were performed and training times over 10 times faster were observed with the same, or better, final performance. This observation underlines the importance of optimized hyper-parameters, which is discussed in the current RL literature. In particular, the hyper-parameters learning rate, exploration factor, and discount factor should be well chosen for the considered control problem, since they significantly influence the stability of the training, the number of random actions, and the weighting of direct versus future rewards. Bayesian hyper-parameter optimization has proven to be a suitable method, which quickly leads to good hyper-parameters via informed experiments and also takes into account the interdependencies between the hyper-parameters. Based on the observations, it is recommended to use this method and to replace the common practice of using the default parameters or performing a sensitivity analysis on it.

5.5 Frameworks and implementations

In the course of this work, existing open source frameworks were built upon. Particularly worth mentioning and recommended here are:

- Stable-Baselines [Hill et al., 2018] with a Tensorflow [Agarwal et al., 2015] backend for RL algorithms, training, and evaluation routines;
- Open AI Gym [Brockman et al., 2016] for standardized interfaces between RL algorithms and the environment;
- the AixLib [Müller et al., 2016] library for physical system models in Modelica;
- FMPy [Sommer et al., 2017] for running Modelica FMUs in Python environments;
- Scikit-Learn [Pedregosa et al., 2011] for data-driven machine-learning models in Python;

- HyperOpt [Bergstra et al., 2015] for Bayesian optimization in Python;
- as well as standard Python libraries for the implementation of wrappers for the respective environments and evaluation and visualization routines.

For engineers working with artificial intelligence, the Python programming language is clearly recommended. This is mainly due to the intuitive and comparatively user-friendly syntax, the wide distribution and thus very active community, and the availability of further supporting libraries. In addition to Python, deep-learning frameworks exist in the programming languages JavaScript and R, as well as C++ (which can be interesting for projects targeting applications on edge devices).

The investigations have shown that a great deal of programming work, and thus time-consuming debugging, takes place in the wrapper implementation, where the signals from the environment and the actions from the algorithm are formulated in a MDP. Therefore, it is important to perform a careful and structured approach in this step. Well-documented Python code is well-suited for this purpose. Well-documented frameworks such as Stable-Baselines, Tensorflow, or Scikit-Learn help with code examples and their active community to make the latest innovations from the fields of machine-learning, deep-learning, and RL usable for engineers.

Engineers should generally avoid programming complex neural networks themselves. Publications in which RL algorithms are described are strongly influenced by the scientific language in the research field and can only be interpreted and implemented by engineers with a high degree of time effort and required specialization. It should be rather developed upon well maintained, documented, and tested libraries. The libraries used in this work do have alternatives. There are numerous comparably good libraries such as Google Dopamine, Keras-RL, or Tensorflow Agents. When making the selection, especially with GitHub projects, attention should be paid to how many developers actively maintain the repositories (contributors), how recent the last commit is (last commit), how many users have favorited the project (stars), how many problems have been solved (closed issues), how actively people contribute (commit activity), and finally whether the syntax and the available documentation are well-suited and fit the individual way of working and programming.

According to these criteria, the used libraries were selected and in the course of this work successfully used to produce the results. Since the formulation of the MDP requires the most engineering effort and the debugging must be performed efficiently, a setup should be established in which the RL BES developer is able to design structured and (if possible) automated experiments.

5.6 Critical discussion of the engineering effort compared to other methods

In this section, a critical discussion of the engineering effort of RL compared to other BES control methods is conducted. A distinction has to be made between the effort for the selection of the algorithm and the appropriate hyper-parameters, the formulation of the MDP, and the training process. The methods with which a critical comparison can be made are the state-of-the-art BES automation, consisting of RBC logics and conventional feedback controllers as the lower benchmark and MPC (physics-driven and data-driven) as the upper benchmark.

In case studies one and two, the algorithms were trained and tested on the same environment. In both cases, modeling was performed (physics-driven in case study one and data-driven in case study two) to represent the behavior of the real systems and to test RL applications on them. In both cases, considerable effort was invested in formulating the MDP and in optimizing the hyper-parameters. In principle, the resulting models would have been suitable for direct use in a MPC application to optimize the real systems. In these cases, the advantage of RL over MPC (physics-driven and data-driven) is reduced to the significantly lower computing power required after training and the inherent adaptivity. However, in both cases, MDP formulations and hyper-parameter combinations have been found, which (assuming a robust data infrastructure) would have allowed training on the real system in feasible times. The technical effort has thus been worthwhile in the sense that results have been produced that support future work addressing the development of RL BES controllers (which can be adapted to real systems in a reasonable time). Thus, assuming a state-of-the-art algorithm, a compact and complete MDP, and a good hyper-parameter selection, the effort for optimization can be significantly reduced compared to MPC. At the same time, the implementation of a monitoring, is even more necessary, which detects security of supply effecting actions of the algorithm and activates a conventional automation in such cases. This increases the effort compared to physics-driven MPC, but not compared to data-driven MPC, where a similar mechanism is required. Furthermore, in the course of case study two, it was demonstrated that a RBC approach for predictive storage management was outperformed by the RL-based approach in operational performance optimization after a few months of operation.

In case study three, the training was also realized by means of a physical simulation model. However, it was shown that with a generic MDP formulation, a transfer to different, real systems with similar dynamics is possible. Although the algorithm trained on the simulation of a mixing valve did not yet perform better than a manually calibrated PID, within the context of a setpoint jump experiment on a throttle circuit of a real heat exchanger, a significant adaptation to the real system already took place after 70 hours of online interaction. The results show that a generic problem formulation makes it possible to provide self-optimizing, self-calibrating BES energy management and automation systems for recurring use cases. The engineering effort for algorithm selection, MDP formulation, and hyper-parameter optimization thus becomes a one-time operation and the actual application can be implemented with

little effort after that. This represents an advantage over MPC in terms of effort and required computing power. It remains an open question for which use cases RL achieves the control performance required to represent a feasible alternative to MPC. The self-calibration potential is a promising feature compared to established methods where plant automation engineers are often engaged for several years with the calibration of complex BES. A critical point is that even with the application of RL at the automation level, a backup control must be provided in order to exclude actions that are harmful to operation.

In summary, RL has the potential to reduce the engineering effort for optimized BES energy management and automation. Compared to physics-driven MPC, it has the potential to avoid manual modeling of systems. Compared to data-driven MPC, the advantage is reduced to the lower required computational power and the inherent system exploration, which has to be implemented manually in data-driven MPC. When using RL, the required know-how shifts slightly from physics to computer science compared to MPC. Since only the necessary inputs, outputs, and rewards for actions to solve a MDP need to be defined, less know-how about physical processes in BES and disturbance modeling is required. At the same time, the application of RL requires a solid basic understanding of the algorithms, the MDP typical design principles, and the influence of hyper-parameters. Therefore, a similar amount of know-how is needed for RL, only from other technology fields. With both RL and data-driven MPC, BES system providers have the opportunity to offer products that are self-calibrating and self-optimizing for individual environmental conditions, and continuously improving in operation after a one-time engineering effort.

5.7 Challenges and opportunities in research

Research in the field of RL for BES faces several challenges and opportunities, some of which are specific to RL and others of which apply to all approaches to optimal control for BES.

One of the most important challenges, which applies to all methods, is the identification of optimal application scenarios for the methods. While there are numerous successful application studies for RL algorithms, MPC algorithms, and other methods, there is no structured analysis of which use cases have which inherent properties (flexibility potential, speed of system dynamics, relevance of user comfort, or available computing power in the automation). Such a structured discussion, e.g. in the form of a combined review and method/problem classification, could identify the optimal application areas for the respective methods and show the need for further comparative studies if no clear statement can be made.

A similar issue is the reproducibility of results. Especially studies performed on real systems are often difficult to reproduce and a high degree of trust in the carefulness of the publishing researchers is necessary. Therefore, publications in the field of BES optimal control should increase their potential for comparability by standardized description of their experimental

setup. This should include not only the textual description of the problem, but also formulas and tables of system properties, models, or hyper-parameters. To develop a standardized template for this would be a great help for the research area. In the field of RL research, there are also efforts to provide standardized training and test environments for BES applications in the standardized Open AI Gym format. This is a promising opportunity, since different methods can be tested in these standardized open source environments and thus be made comparable. Unfortunately, the available environments so far cover only very isolated application scenarios, which leaves a great potential for the development of such environments open.

Another aspect concerns experiments on real systems. As in this work, a large part of the studies is performed on simulated environments. When applied to real systems, a number of challenges arise especially concerning the aspects of safe operation, data quality, and debugging of the optimization algorithm. These challenges are similar for all optimization methods and require further research in the area of robust integration of novel methods into state-of-the-art automation systems and robust BES and sensor monitoring.

In terms of available algorithms, there is also still a high need for research. One of the most promising research paths is the one of transfer learning. Transfer learning approaches address the question of how trained algorithms can be made applicable to similar problems or which components of algorithms can be transferred and adapted to other individual problems in order to speed up the otherwise long online training process. For this purpose, the identification of corresponding recurring automation tasks is needed. When using neural networks in RL algorithms, classical ANNs are usually employed. However, by using CNNs or LSTMs, it is also possible to design algorithms that use image data to determine their actions and ones that have a memory for past system states via an internal data storage. Finally, the research field of online hyper-parameter adaptation after training is still hardly addressed. In principle, the hyper-parameters learning rate and exploration rate could be (triggered by certain events during interaction) adjusted selectively, and thus a faster adaptation to changing system behavior could be achieved.

5.8 Challenges and opportunities in practical application

The challenges in the practical application of RL for BES products and services are partly different from those in research. In order to bring RL algorithms into practical application, several questions need to be addressed. For economic and safety reasons, the complete exploration of the real system environment cannot be carried out in all cases starting from zero. On the other hand, performing expensive, time-consuming, and error-prone model creation should be avoided for applications where the effort does not lead to substantial performance improvements.

All libraries and algorithms used in the course of this work are freely available and can be used for real-world application projects. BES component manufacturers, automation companies, and energy providers only have to make the strategic decision to build up know-how in the area of artificial intelligence and RL. For this, corresponding methods should also be covered in engineering studies, in application-oriented lectures. In the long term, artificial intelligence will be established alongside simulation, experimentation, surveys, and optimization as a method for BES engineers and the necessary know-how should therefore already be covered in the course of the respective studies.

Especially in case study three, some of the challenges that have to be addressed in the practical application of RL have been addressed. In addition to a robust data infrastructure and data pre-processing, which ensures comparable data quality as in a simulation, regular backups of the RL algorithms should be created at different points in the interaction history. Thus, after an adaptation to non-physical behavior, for example, due to the failure of a sensor, an earlier version of the algorithm, before the failure, can be accessed. The technical requirements for the operation optimization thus shift from computational power to data storage. Since (after training) RL algorithms map system states directly to actions, the computational power required is minimal compared to MPC (where the optimization must be performed at each time-step). At the same time, MPC does not necessarily require backups or a training data storage.

Product developers face several challenges when applying RL. In particular, a suitable algorithm has to be selected and a training strategy has to be developed. Training by means of data-driven models from monitoring data can be reasonable for highly cost-intensive, existing BES. For products that are sold in large quantities (such as heat pumps or boilers), where the physical behavior should not differ but which are used in changing environmental conditions, a more complex training procedure may be considered. Here, an algorithm for performance maximization could be pre-trained on a simulation model and set with optimal hyper-parameters. To ensure that the initialization of the ANN parameters does not have an overly large influence on the final policy, the optimization should be tested several times for each setup, or alternatively always with the same initialization of the ANN. After the pre-training phase, the algorithm can be tested on a test bench, further trained, and the hyper-parameters tuned for a quick adaptation to the real system. Care should be taken that the algorithm is not trained too long with a flat performance gradient on the simulation, as there is a risk of over-fitting to the simulation model. For components with sufficient monitoring data from real operation, the data can be used for additional offline training. However, if the components are operated with the same operation policy over the entire dataset, the danger of over-fitting to a suboptimal policy (as observed in case study two) must be taken into account. Using the described workflow, a component manufacturer can offer an adaptive, self-optimizing control for its product, which continuously improves during operation.

A major challenge in the practical application of RL for BES is the guarantee that no actions harmful to the operation are performed. In particular, this is crucial during the exploration of the BES dynamics but also afterwards. Therefore, as with other methods, a comprehensive monitoring of the system behavior is necessary. Fault detection and threshold monitoring are crucial requirements and should be implemented as pre-defined software components for each RL-supported BES. If the RL control is deactivated, a robust state-of-the-art backup control can take over. At the same time, the RL algorithm should be trained by a large negative reward signal not to return to a known failing system state in the future.

An important prerequisite for the implementation of the described methods is a building automation able to process optimization results. As buildings become more complex in the coming years due to greater pressure to decarbonize their operation and the associated increase in electrification, the automation and IT infrastructure must provide a suitable environment for executing modern methods and higher programming languages. This can be done by outsourcing the energy management level, or even the automation level (in the course of IoT automation), to the cloud. A key advantage is also that the operation logic can be tested and maintained independently of the location. However, a decisive obstacle (especially in Germany) is data protection, which places high demands on data collection and processing. It is hoped that in the future, a political decision will be made in the field of energy systems that defines sensitive data and facilitates the collection of data needed to increase the efficiency of energy systems. A second option is the implementation of RL automation on a local industrial PC. Here the lower required computing power is a clear advantage compared to MPC, which makes RL also interesting for legacy BES with recurring structures.

5.9 Possible products in the coming years

It can be assumed that RL algorithms will be incorporated into product development around BES software in the future. The challenges and opportunities were discussed in the last two sections. In particular, the property of RL algorithms to encode optimization results from a large number of system states into state-action relations in a performant way, and at the same time to keep open a high potential for further adaptation to new environmental conditions, makes RL algorithms promising here.

In the field of home energy management systems (HEMS), system providers could offer different RL-based expert systems. For recurring configurations (e.g. the combination E-vehicle, heat pump, PV system, and thermal storage), an algorithm pre-trained on a generic simulation covering the basic dynamics could be used to make optimal energy management decisions based on the individual user behavior. Furthermore, if a variable electricity price is made available by the energy provider, such a system could also be used to operate a cluster of buildings in a grid-stabilizing manner via the RL algorithms implemented in the HEMS.

In the field of building energy management systems (BEMS), complex non-residential buildings could be automated using pre-trained RL algorithms. While currently the building automation is implemented and calibrated under great time pressure during the construction of a building, RL algorithms can support automating this process. Since the builder of a building is usually not the later operator, efficient and well-adjusted automation and energy management layers are often not prioritized enough and building technicians are occupied with setting up the BES correctly for several years. Using specific software components, where RL algorithms as well as training and safety backup rules are included, this process could be automated and a product family of self-calibrating, self-optimizing BEMS could be developed.

Room control also offers great potential for RL applications due to the recurring sensor and actuator structure. Since the target variables air quality (in the form of CO_2 and VOC) and room temperature are often the same and the available actions (valve position cooling, valve position heating, and fresh air flow rate) are also recurring, RL algorithms could be pre-trained here on the basic interrelationships with generic MDPs and adapt to individual rooms (different due to room size, building age, and use). An automation of the room shading would also be worthy of consideration. Due to the property of RL algorithms to map even high-dimensional state-vectors to actions in a performant way, image-processing CNNs could be used, which include radiation intensity and sun position in the optimized room control.

Further application scenarios are possible. For example, research is currently being conducted in the area of optimized de-icing control of outdoor air heat pumps. Also, a heating curve adapted to individual BES and user needs could be realized via RL algorithms. In addition to the products listed here, many other products are possible, in application scenarios where generic pre-training and policy fine tuning in an online training phase is technically and economically feasible. The prerequisites for a successful application are only the available know-how and the potential to invest in the RL specific workflow. Then adaptive, self-optimizing energy management and automation systems can be realized for various use cases.

6 Summary and outlook

In this dissertation, the applicability and potential of novel artificial intelligence algorithms from the field of RL for the optimization of BES was investigated. For this purpose, the necessity of using novel optimization methods for BES was first described and a differentiation of artificial intelligence-based methods from more established methods was presented. Subsequently, the research area of RL, as a subfamily of artificial intelligence research, was introduced with its different algorithm families; advantages and disadvantages were described and the most promising methods for the building sector were identified. The fundamental assessment was concluded by a review of the literature in the area of RL for BES, where selected publications were discussed in detail and open research questions were identified.

Based on the introduced fundamentals, a workflow was defined and presented according to which engineers can develop a RL-supported building automation. For this purpose, all relevant aspects from the problem formulation, the selection of suitable algorithms, and training strategies to the optimization of hyper-parameters were described based on different initial settings. In addition, application-oriented aspects such as the procedure for implementation and interaction strategies between algorithms and real systems were discussed in separate sections.

Based on the literature and the definition of a suitable workflow, three specific case study systems were investigated in the application chapter, thus demonstrating the applicability and potential of RL for BES. All case studies were inspired from real optimization tasks, which are part of current work packages in public research projects in Germany, kindly funded by the Federal Ministry for Economic Affairs and Climate Action. The results were published in international conference proceedings as well as in scientific journals and discussed in the scientific community. The current state of research is thus extended by studies addressing in particular algorithm comparisons, training and problem formulation strategies, and algorithm parameterization. Among the insights gained, Bayesian hyper-parameter optimization has been identified as an efficient tool that should be preferred over manual parameterization of algorithms in future studies. In addition, the application of RL for energy management, load shifting, and control applications has been investigated during the studies. In the course of the case studies, energy management applications and control applications were investigated on the basis of physical simulations, on the basis of data-driven models, and in interaction with a real system. The research has shown that RL is well suited to automate optimization tasks for BES, considering the design principles defined in the workflow. At the same time,

however, the problem formulation, the choice of algorithms, and the use of optimized hyperparameters have a major impact on the final performance and speed of training. However, if these aspects are carefully considered, algorithms can learn superior control policies within a few month, even without pre-training.

Finally, based on the previous chapters, an extensive discussion of the case studies, the lessons learned, and best practices resulting from the work, was conducted. Based on the experience gained in the course of this work, the applicability of RL for energy management and control tasks was discussed. The relevance of the different design aspects as well as concrete suggestions for the implementation were presented. In addition, a critical evaluation of the methods compared to other methods was performed. The chapter was concluded with a discussion of the challenges and opportunities in research as well as in practice, including possible products that could arise in the future from the knowledge gained.

However, a number of research questions remain open. On the one hand, in the area of algorithms, further studies can help to select the best ones for families of application scenarios and to distinguish the optimal use cases from the ones for other methods such as MPC. In the area of energy management tasks, it should be investigated how the property of RL algorithms to process even high-dimensional state-vectors in a performant way can be used to realize coordination tasks of many technical systems or even whole districts. In the area of closed loop control applications, problem classes should be identified that can be described and learned as generically as possible. In this way, algorithms can be pre-trained and transferred to different but similar application scenarios. For both areas, safe interaction and operation strategies have to be developed. Since RL, like all artificial intelligence methods, is a stochastic technique, robust backup and security mechanisms need to be implemented. Studies on the robust use of RL on real systems are still rare and successful applications should be published in a comprehensible way.

With the increasing availability of low-cost sensors and data infrastructures, artificial intelligence is expected to become another permanent group of tools in the toolbox of energy system engineers. Wherever substantial sensor and data infrastructure is envisioned, artificial intelligence algorithms can provide significant benefits by processing data in a performant manner, identifying structures in the data that are invisible to humans, pre-processing data for expert decision-making, or as in the case of RL, even making optimal operation decisions autonomously. As the algorithms continue to improve on corresponding benchmark problems, the challenge in the coming years will be for technicians and engineers to develop a broad understanding of the possibilities and problems in order to recognize the problems that can be addressed by artificial intelligence in their respective fields of work. Then, these algorithms can make a decisive contribution to the efficient achievement of climate policy goals.

In view of the current climate and geopolitical events in the world, it can be assumed that there will be an increasing demand for decarbonization measures in the major energy sectors.

As one of the largest consumers of gas, and thus emitters of climate emissions, the building sector is an important area of work here. With rising commodity prices, energy management and optimal control applications are becoming increasingly important for the building sector as well. At the same time, it cannot be assumed that these challenges can be addressed individually for each building, due to the high engineering efforts. It is therefore expected that RL algorithms together with other optimization techniques will be integrated in many products in the next decades. Wherever the same arrangement of systems and sensors is required and an adaptation to individual user and environmental conditions is desired, the design and pre-training of RL-supported energy management and control systems can be beneficial. In this way, current research in the field of RL can make a decisive contribution to making the vision of a self-calibrating, self-optimizing building control system a reality.

Bibliography

Abel, D. [2018], *Umdruck zur Vorlesung Regelungstechnik und Ergänzungen (Höhere Regelungstechnik)*, Verlaghaus Mainz.

Afram, A. and Janabi-Sharifi, F. [2014], 'Theory and applications of hvac control systems – a review of model predictive control (mpc)', *Building and Environment* **72**, 343–355.
URL: *http://www.sciencedirect.com/science/article/pii/S0360132313003363*

Afram, A., Janabi-Sharifi, F., Fung, A. S. and Raahemifar, K. [2017], 'Artificial neural network (ann) based model predictive control (mpc) and optimization of hvac systems: A state of the art review and case study of a residential hvac system', *Energy and Buildings* **141**, 96–113.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778816310799*

Agarwal, A., Barham, P., Brevdo, E., Chen, Z. and et al [2015], Tensorflow:large-scale machine learning on heterogeneous distributed systems. Software available from tensorflow.org.
URL: *https://www.tensorflow.org/*

Ahmad, M. W., Mourshed, M. and Rezgui, Y. [2017], 'Trees vs neurons: Comparison between random forest and ann for high-resolution prediction of building energy consumption', *Energy and Buildings* **147**, 77–89.
URL: *https://www.sciencedirect.com/science/article/pii/S0378778816313937*

Al-jabery, K., Xu, Z., Yu, W., Wunsch, D. C., Xiong, J. and Shi, Y. [2017], 'Demand-side management of domestic electric water heaters using approximate dynamic programming', *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **36**(5), 775–788.
URL: *https://ieeexplore.ieee.org/document/7536214*

Alfred, R. [2016], The rise of machine learning for big data analytics, *in* '2nd International Conference on Science in Information Technology (ICSITech)', pp. 1–1.
URL: *https://ieeexplore.ieee.org/document/7852593*

Álvarez, J. D., Redondo, J. L., Camponogara, E., Normey-Rico, J., Berenguel, M. and Ortigosa, P. M. [2013], 'Optimizing building comfort temperature regulation via model predictive control', *Energy and Buildings* **57**, 361–372.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778812005610*

Bahrami, S., Wong, V. W. S. and Huang, J. [2018], 'An online learning algorithm for demand response in smart grid', *IEEE Transactions on Smart Grid* **9**(5), 4712–4725.
URL: *https://ieeexplore.ieee.org/document/7849144*

Baranski, M., Fütterer, J. and Müller, D. [2018], 'Distributed exergy-based simulation-assisted control of hvac supply chains', *Energy and Buildings* **175**, 131–140.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778817339804*

Barsce, J. C., Palombarini, J. A. and Martínez, E. C. [2017], Towards autonomous reinforcement learning: Automatic setting of hyper-parameters using bayesian optimization, *in* '2017 XLIII Latin American Computer Conference (CLEI)', pp. 1–9.
URL: *http://arxiv.org/pdf/1805.04748v1*

Bellman, R. [1954], 'Dynamic programming and a new formalism in the calculus of variations', *Proceedings of the National Academy of Sciences of the United States of America* **40**(4), 231–235.
URL: *https://www.ncbi.nlm.nih.gov/pmc/articles/PMC527981/*

Bellman, R. [1956], *Dynamic programming*, Dover Books on Computer Science, Dover Publications, Newburyport.
URL: *http://gbv.eblib.com/patron/FullRecord.aspx?p=1897424*

Bergstra, J. and Bengio, Y. [2012], 'Random search for hyper-parameter optimization', *Journal of Machine Learning Research* **13**, 281–305.
URL: *https://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf*

Bergstra, J., Komer, B., Eliasmith, C., Yamins, D. and Cox, D. D. [2015], 'Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures', *Computational Science & Discovery* (8(1)), 014008.
URL: *https://proceedings.mlr.press/v28/bergstra13.html*

Biemann, M., Scheller, F., Liu, X. and Huang, L. [2021], 'Experimental evaluation of model-free reinforcement learning algorithms for continuous hvac control', *Applied Energy* **298**, 117164.
URL: *https://www.sciencedirect.com/science/article/pii/S0306261921005961*

Bischl, B., Binder, M., Lang, M., Pielok, T., Richter, J., Coors, S., Thomas, J., Ullmann, T., Becker, M., Boulesteix, A.-L., Deng, D. and Lindauer, M. [2023], 'Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges', *WIREs Data Mining and Knowledge Discovery* **13**(2), e1484.
URL: *https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.1484*

Blockwitz, T., Otter, M., Akesson, J., Arnold, M., Clauss and et al. [2012], Functional mockup interface 2.0: The standard for tool independent exchange of simulation models, *in* 'Proceedings of the 9th International MODELICA Conference, September 3-5, 2012, Munich, Germany', pp. 173–184.
URL: *https://ep.liu.se/ecp/076/017/ecp12076017.pdf*

Bode, G., Baranski, M., Schraven, M., Kümpel, A., Storek, T., Nürenberg, M., Müller, D., Rothe, A., Ziegeldorf, J. H., Fütterer, J. and Scheuffele, B. [2019], 'Cloud, wireless technology, internet of things: the next generation of building automation systems?', *Journal of Physics: Conference Series* **1343**(1), 012059.
URL: *https://doi.org/10.1088/1742-6596/1343/1/012059*

Bode, G., Fütterer, J. and Müller, D. [2018], 'Mode and storage load based control of a complex building system with a geothermal field', *Energy and Buildings* **158**, 1337–1345.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778816317832*

Brandi, S., Piscitelli, M. S., Martellacci, M. and Capozzoli, A. [2020], 'Deep reinforcement learning to optimise indoor temperature control and heating energy consumption in buildings', *Energy and Buildings* **224**, 110225.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778820308963*

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J. and Zaremba, W. [2016], 'Openai gym'. Accessed: 2022-06-19.
URL: *https://arxiv.org/pdf/1606.01540*

Bryson, A. E. [1996], 'Optimal control-1950 to 1985', *IEEE Control Systems* **16**(3), 26–33.
URL: *https://ieeexplore.ieee.org/document/506395*

Bschorer, S., Buchholz, R., Hanßke, A., Langemeyer, S., Petermann, C. and Rohde, F. [2019], *Energienetz Berlin Adlershof: Schlussbericht. Förderkennzeichen 03ET1038E.*
URL: *https://www.tib.eu*

Buchanan, B. G. [2005], 'A (very) brief history of artificial intelligence', *AI Magazine* (26).
URL: *https://ojs.aaai.org//index.php/aimagazine/article/view/1848*

Chen, Y., Norford, L. K., Samuelson, H. W. and Malkawi, A. [2018], 'Optimal control of hvac and window systems for natural ventilation through reinforcement learning', *Energy and Buildings* **169**, 195–205.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778818302184*

Chollet, F. [2018], *Deep learning with Python*, Safari Tech Books Online, Manning, Shelter Island, NY.
URL: *http://proquest.safaribooksonline.com/9781617294433*

Chollet, F. and et al. [2015], 'Keras'. Accessed: 2022-06-19.
URL: *keras.io/*

Deng, K., Sun, Y., Li, S., Lu, Y., Brouwer, J., Mehta, P. G., Zhou, M. and Chakraborty, A. [2015], 'Model predictive control of central chiller plant with thermal energy storage via dynamic programming and mixed-integer linear programming', *IEEE Transactions on Automation Science and Engineering* **12**(2), 565–579.
URL: *https://ieeexplore.ieee.org/document/6899700*

Deutsche Energie-Agentur [2018], 'Integrierte energiewende - impulse für die gestaltung des energiesystems bis 2050'.
URL: *https://www.dena.de/integrierte-energiewende*

Di Natale, L., Svetozarevic, B., Heer, P. and Jones, C. N. [2021], 'Deep reinforcement learning for room temperature control: a black-box pipeline from data to policies', *Journal of Physics: Conference Series* **2042**(1), 012004.
URL: *https://iopscience.iop.org/article/10.1088/1742-6596/2042/1/012004*

Dong, Z., Huang, X., Dong, Y. and Zhang, Z. [2020], 'Multilayer perception based reinforcement learning supervisory control of energy systems with application to a nuclear steam supply system', *Applied Energy* **259**, 114193.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S030626191931880X*

Dorokhova, M., Martinson, Y., Ballif, C. and Wyrsch, N. [2021], 'Deep reinforcement learning control of electric vehicle charging in the presence of photovoltaic generation', *Applied Energy* **301**(18), 117504.
URL: *https://www.sciencedirect.com/science/article/pii/S0306261921008874*

Dowling, J. and Haridi, S. [2008], Decentralized reinforcement learning for the online optimization of distributed systems, *in* C. Weber, M. Elshaw and N. M. Mayer, eds, 'Reinforcement Learning', IntechOpen, Rijeka, chapter 8.
URL: *https://doi.org/10.5772/5279*

Drgoňa, J., Arroyo, J., Cupeiro Figueroa, I., Blum, D., Arendt, K., Kim, D., Ollé, E. P., Oravec, J., Wetter, M., Vrabie, D. L. and Helsen, L. [2020], 'All you need to know about model predictive control for buildings', *Annual Reviews in Control* **50**, 190–232.
URL: *https://www.sciencedirect.com/science/article/pii/S1367578820300584*

EnBA-M [2018], 'Energienetz berlin adlershof: Monitoring und optimierung 2018-2021'. Accessed: 2022-06-19.
URL: *http://www.energienetz-berlin-adlershof.de/monitoring-und-optimierung-2018-2021/*

Ernst, D., Glavic, M., Capitanescu, F. and Wehenkel, L. [2009], 'Reinforcement learning versus model predictive control: a comparison on a power system problem', *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society* **39**(2), 517–529.
URL: *https://ieeexplore.ieee.org/document/4717266*

Filipe, J., Bessa, R. J., Reis, M., Alves, R. and Póvoa, P. [2019], 'Data-driven predictive energy optimization in a wastewater pumping station', *Applied Energy* **252**, 113423.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261919310979*

Francois-Lavet, V., Henderson, P., Islam, R., Bellemare, M. G. and Pineau, J. [2018], 'An introduction to deep reinforcement learning', *Foundations and Trends® in Machine Learning* **11**(3-4), 219–354.
URL: *http://arxiv.org/pdf/1811.12560v2*

Frigg, R. and Hartmann, S. [2020], Models in Science, *in* E. N. Zalta, ed., 'The Stanford Encyclopedia of Philosophy', Spring 2020 edn, Metaphysics Research Lab, Stanford University.

Fütterer, J., Schild, T. and Müller, D. [2017], 'Building automation systems in practice'.
URL: *http://publications.rwth-aachen.de/record/691150*

Görges, D. [2017], 'Relations between model predictive control and reinforcement learning', *IFAC-PapersOnLine* **50**(1), 4920–4928.
URL: *https://www.sciencedirect.com/science/article/pii/S2405896317311941*

Gruber, M., Trüschel, A. and Dalenbäck, J.-O. [2014], 'Model-based controllers for indoor climate control in office buildings – complexity and performance evaluation', *Energy and Buildings* **68**, 213–222.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778813005951*

Haarnoja, T., Zhou, A., Abbeel, P. and Levine, S. [2018], 'Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor', *arXiv e-prints* p. arXiv:1801.01290.
URL: *https://arxiv.org/pdf/1801.01290.pdf*

Haji Hosseinloo, A., Ryzhov, A., Bischi, A., Ouerdane, H., Turitsyn, K. and Dahleh, M. A. [2020], 'Data-driven control of micro-climate in buildings: An event-triggered reinforcement learning approach', *Applied Energy* **277**, 115451.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261920309636*

Han, D.-M. and Lim, J.-H. [2010], 'Design and implementation of smart home energy management systems based on zigbee', *IEEE Transactions on Consumer Electronics* **56**(3), 1417–1425.
URL: *https://ieeexplore.ieee.org/document/5606278*

Han, M., May, R., Zhang, X., Wang, X., Pan, S., Yan, D., Jin, Y. and Xu, L. [2019], 'A review of reinforcement learning methodologies for controlling occupant comfort in buildings', *Sustainable Cities and Society* **51**, 101748.
URL: *https://www.sciencedirect.com/science/article/pii/S2210670719307589*

Hassan, M. H., Awada, M., Khoury, H. and Srour, I. [2019], 'A machine learning approach for predicting office energy consumption in a mediterranean region', *Proceedings of ECOS 2019* .
URL: *https://www.proceedings.com/52893.html*

Hasselt, H. v., Guez, A. and Silver, D. [2016], Deep reinforcement learning with double q-learning, *in* 'Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence', AAAI Press, p. 2094–2100.
URL: *https://dl.acm.org/doi/10.5555/3016100.3016191*

Hessel, M., Modayil, J., van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M. and Silver, D. [2017], 'Rainbow: Combining improvements in deep reinforcement learning'.
URL: *https://arxiv.org/pdf/1710.02298*

Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., Horgan, D., Quan, J., Sendonaris, A., Dulac-Arnold, G., Osband, I., Agapiou, J., Leibo, J. Z. and Gruslys, A. [2018], 'Deep q-learning from demonstrations'.
URL: *https://arxiv.org/pdf/1704.03732*

Hill, A., Raffin, A., Ernestus, M., Gleave, A., Kanervisto, A., Traore, R., Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S. and Wu, Y. [2018], 'Stable baselines', *GitHub repository* .
URL: *https://github.com/hill-a/stable-baselines*

Hirth, L., Mühlenpfordt, J. and Bulkeley, M. [2018], 'The entso-e transparency platform – a review of europe's most ambitious electricity data platform', *Applied Energy* **225**, 1054–1067.
URL: *https://www.sciencedirect.com/science/article/pii/S0306261918306068*

Hüttermann, A., Leenders, L., Bahl, B. and Bardow, A. [2019], 'Automated data-driven model generation of energy systems using piecewise linear regression', *Proceedings of ECOS 2019* .
URL: *https://www.proceedings.com/52893.html*

IEA - International energy Agency [2013], *Transition to sustainable buildings: Strategies and opportunities to 2050*, Organisation for Economic Cooperation and Development, Paris.
URL: *https://www.iea.org/reports/transition-to-sustainable-buildings*

IEA - International energy Agency [2018], '2018 global status report: For the global alliance for buildings and construction (globalabc)'.
URL: *https://globalabc.org/resources/publications*

IPCC [2018], 'Global warming of 1.5°c. an ipcc special report on the impacts of global warming of 1.5°c above pre-industrial levels and related global greenhouse gas emission pathways, in the context of strengthening the global response to the threat of climate change, sustainable development, and efforts to eradicate poverty'.
URL: *https://www.ipcc.ch/sr15/download/*

Jain, A., Behl, M. and Mangharam, R. [2017], 'Data predictive control for building energy management', *Proceedings of 2017 American Control Conference (ACC)* pp. 44–49.
URL: *https://ieeexplore.ieee.org/document/7962928*

Jain, A., Mangharam, R. and Behl, M. [2016], 'Data predictive control for peak power reduction', *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments - BuildSys '16* pp. 109–118.
URL: *https://repository.upenn.edu/mlab$_p$apers/92/*

Jain, R. K., Smith, K. M., Culligan, P. J. and Taylor, J. E. [2014], 'Forecasting energy consumption of multi-family residential buildings using support vector regression: Investigating the impact of temporal and spatial monitoring granularity on performance accuracy', *Applied Energy* **123**, 168–178.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261914002013*

Javed, A., Larijani, H., Ahmadinia, A. and Des Gibson [2017], 'Smart random neural network controller for hvac using cloud computing technology', *IEEE Transactions on Industrial Informatics* **13**(1), 351–360.
URL: *https://ieeexplore.ieee.org/document/7529229*

Jia, R., Jin, M., Sun, K., Hong, T. and Spanos, C. [2019], 'Advanced building control via deep reinforcement learning', *Energy Procedia* **158**, 6158–6163.
URL: *https://www.sciencedirect.com/science/article/pii/S187661021930517X*

Kämper, A., Leenders, L., Bahl, B. and Bardow, A. [2021], 'Automog: Automated data-driven model generation of multi-energy systems using piecewise-linear regression', *Computers & Chemical Engineering* **145**(6), 107162.
URL: *https://www.sciencedirect.com/science/article/pii/S0098135420306852*

Kathirgamanathan, A., de Rosa, M., Mangina, E. and Finn, D. P. [2021], 'Data-driven predictive control for unlocking building energy flexibility: A review', *Renewable and Sustainable Energy Reviews* **135**, 110120.
URL: *https://www.sciencedirect.com/science/article/pii/S1364032120304111*

Kazmi, H., Suykens, J., Balint, A. and Driesen, J. [2019], 'Multi-agent reinforcement learning for modeling and control of thermostatically controlled loads', *Applied Energy* **238**, 1022–1035.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261919301564*

Khayatian, F., Nagy, Z. and Bollinger, A. [2021], 'Using generative adversarial networks to evaluate robustness of reinforcement learning agents against uncertainties', *Energy and Buildings* p. 111334.
URL: *https://www.sciencedirect.com/science/article/pii/S0378778821006186*

Kieback&Peter [2020]. Accessed: 2022-06-19.
URL: *https://www.kieback-peter.com/de/*

Kilkis, İ. B. [1999], 'Utilization of wind energy in space heating and cooling with hybrid hvac systems and heat pumps', *Energy and Buildings* **30**(2), 147–153.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778898000826*

Kofinas, P., Dounis, A. I. and Vouros, G. A. [2018], 'Fuzzy q-learning for multi-agent decentralized energy management in microgrids', *Applied Energy* **219**, 53–67.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261918303465*

Kolokotsa, D., Pouliezos, A., Stavrakakis, G. and Lazos, C. [2009], 'Predictive control techniques for energy and indoor environmental quality management in buildings', *Building and Environment* **44**(9), 1850–1863.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0360132308002990*

Kotevska, O., Munk, J., Kurte, K., Du, Y., Amasyali, K., Smith, R. W. and Zandi, H. [2020], Methodology for interpretable reinforcement learning model for hvac energy control, *in* '2020 IEEE International Conference on Big Data (Big Data)', IEEE, pp. 1555–1564.
URL: *https://ieeexplore.ieee.org/document/9377735*

Lee, W.-S., Chen, Y. T. and Wu, T.-H. [2009], 'Optimization for ice-storage air-conditioning system using particle swarm algorithm', *Applied Energy* **86**(9), 1589–1595.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261908003528*

Leibowicz, B. D., Lanham, C. M., Brozynski, M. T., Vázquez-Canteli, J. R., Castejón, N. C. and Nagy, Z. [2018], 'Optimal decarbonization pathways for urban residential building energy services', *Applied Energy* **230**, 1311–1325.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261918313552*

Li, Y. [2017], 'Deep reinforcement learning: An overview'.
URL: *http://arxiv.org/pdf/1701.07274v6*

Liessner, R., Schmitt, J., Dietermann, A. and Bäker, B. [2019], 'Hyperparameter optimization for deep reinforcement learning in vehicle energy management', *Proceedings of the 11th International Conference on Agents and Artificial* pp. 134–144.
URL: *https://www.scitepress.org/Link.aspx?doi=10.5220/0007364701340144*

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D. and Wierstra, D. [2015], 'Continuous control with deep reinforcement learning'.
URL: *http://arxiv.org/pdf/1509.02971v5*

Liu, S. and Henze, G. P. [2006], 'Experimental analysis of simulated reinforcement learning control for active and passive building thermal storage inventory', *Energy and Buildings* **38**(2), 148–161.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S037877880500085X*

Lork, C., Li, W.-T., Qin, Y., Zhou, Y., Yuen, C., Tushar, W. and Saha, T. K. [2020], 'An uncertainty-aware deep reinforcement learning framework for residential air conditioning energy management', *Applied Energy* **276**, 115426.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261920309387*

Lu, Y., Wang, S., Sun, Y. and Yan, C. [2015], 'Optimal scheduling of buildings with energy generation and thermal energy storage under dynamic electricity pricing using mixed-integer nonlinear programming', *Applied Energy* **147**, 49–58.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261915002378*

Maddalena, E. T., Lian, Y. and Jones, C. N. [2020], 'Data-driven methods for building control — a review and promising future directions', *Control Engineering Practice* **95**, 104211.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0967066119301832*

Marantos, C., Siozios, K. and Soudris, D. [2020], 'Rapid prototyping of low-complexity orchestrator targeting cyberphysical systems: The smart-thermostat usecase', *IEEE Transactions on Control Systems Technology* **28**(5), 1831–1845.
URL: *https://ieeexplore.ieee.org/document/8770296*

Mason, K. and Grijalva, S. [2019], 'A review of reinforcement learning for autonomous building energy management', *Computers Electrical Engineering* **78**, 300–312.
URL: *https://www.sciencedirect.com/science/article/pii/S0045790618333421*

McCartney, M., Haeringer, M. and Polifke, W. [2020], 'Comparison of machine learning algorithms in the interpolation and extrapolation of flame describing functions', *Journal of Engineering for Gas Turbines and Power* **142**(6).
URL: *https://asmedigitalcollection.asme.org*

MeteoViva GmbH [2020]. Accessed: 2022-06-19.
URL: *https://meteoviva.com/en/*

Minoli, D., Sohraby, K. and Occhiogrosso, B. [2017], 'Iot considerations, requirements, and architectures for smart buildings—energy optimization and next-generation building management systems', *IEEE Internet of Things Journal* **4**(1), 269–283.
URL: *https://ieeexplore.ieee.org/document/7805265*

Mirinejad, H., Sadati, H., Maryam, G. and Hamid, T. [2008], 'Control techniques in heating, ventilating and air conditioning (hvac) systems 1', *Journal of Computer Science* **4**.
URL: *https://thescipub.com/pdf/jcssp.2008.777.783.pdf*

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D. and Riedmiller, M. [2013], 'Playing atari with deep reinforcement learning'.
URL: *https://www.deepmind.com/publications*

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S. and Hassabis, D. [2015], 'Human-level control through deep reinforcement learning', *Nature* **518**(7540), 529–533.
URL: *https://www.nature.com/articles/nature14236*

Mocanu, E., Nguyen, P. H. and Gibescu, M. [2018], Chapter 7 - deep learning for power system data analysis, *in* R. Arghandeh and Y. Zhou, eds, 'Big Data Application in Power Systems', Elsevier, pp. 125–158.
URL: *https://www.sciencedirect.com/science/article/pii/B9780128119686000073*

Müller, D., Lauster, M., Constantin, A. and Remmen, P. [2016], Aixlib - an open-source modelica library within the iea-ebc annex 60 framework, Proceedings of BauSim 2016, pp. 3–9.
URL: *https://www.iea-annex60.org/downloads/2016-bausim-aixlib.pdf*

Nagy, A., Kazmi, H., Cheaib, F. and Driesen, J. [2018], 'Deep reinforcement learning for optimal control of space heating', *Proceedings of Building Simulation and Optimization 2018* .
URL: *http://www.ibpsa.org/proceedings/BSO2018/1C-4.pdf*

Nagy, Z., Vázquez-Canteli, J. R., Dey, S. and Henze, G. [2021], The citylearn challenge 2021, *in* 'Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation', BuildSys '21, Association for Computing Machinery, New York, NY, USA, p. 218–219.
URL: *https://doi.org/10.1145/3486611.3492226*

Oldewurtel, F., Parisio, A., Jones, C. N., Gyalistras, D., Gwerder, M., Stauch, V., Lehmann, B. and Morari, M. [2012], 'Use of model predictive control and weather forecasts for energy

efficient building climate control', *Energy and Buildings* **45**, 15–27.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778811004105*

Ooka, R. and Ikeda, S. [2015], 'A review on optimization techniques for active thermal energy storage control', *Energy and Buildings* **106**, 225–233.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778815301420*

OpenAI Spinning Up [2020]. Accessed: 2022-06-19.
URL: *https://spinningup.openai.com/en/latest/*

Palensky, P. and Dietrich, D. [2011], 'Demand side management: Demand response, intelligent energy systems, and smart loads', *IEEE transactions on industrial informatics* pp. 381–388.
URL: *https://ieeexplore.ieee.org/document/5930335*

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J. and Chintala, S. [2019], Pytorch: An imperative style, high-performance deep learning library, *in* H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox and R. Garnett, eds, 'Advances in Neural Information Processing Systems 32', Curran Associates, Inc., pp. 8024–8035.
URL: *http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf*

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E. [2011], 'Scikit-learn: Machine learning in python'.
URL: *http://www.jmlr.org/papers/volume12/pedregosa11a/ pedregosa11a.pdf*

Perera, A. and Kamalaruban, P. [2021], 'Applications of reinforcement learning in energy systems', *Renewable and Sustainable Energy Reviews* **137**(4), 110618.
URL: *https://www.sciencedirect.com/science/article/pii/S1364032120309023*

Pinto, G., Deltetto, D. and Capozzoli, A. [2021], 'Data-driven district energy management with surrogate models and deep reinforcement learning', *Applied Energy* **304**, 117642.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261921010096*

Pinto, G., Piscitelli, M. S., Vázquez-Canteli, J. R., Nagy, Z. and Capozzoli, A. [2021], 'Coordinated energy management for a cluster of buildings through deep reinforcement learning', *Energy* **229**, 120725.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0360544221009737*

Piotr Żymełka, M. S. [2019], 'Short-term scheduling of gas-fired chp plant with thermal storage using optimiz ation algorithm', *Proceedings of ECOS 2019* .
URL: *https://www.proceedings.com/52893.html*

Putatunda, S. and Rama, K. [2018], A comparative analysis of hyperopt as against other approaches for hyper-parameter optimization of xgboost, *in* 'Proceedings of the 2018 International Conference on Signal Processing and Machine Learning', Association for Computing Machinery, New York, NY, USA, p. 6–10.
URL: *https://doi.org/10.1145/3297067.3297080*

Rätz, M., Javadi, A. P., Baranski, M., Finkbeiner, K. and Müller, D. [2019], 'Automated data-driven modeling of building energy systems via machine learning algorithms', *Energy and Buildings* **202**, 109384.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778819316585*

Rayati, M., Sheikhi, A. and Ranjbar, A. M. [2015], Applying reinforcement learning method to optimize an energy hub operation in the smart grid, *in* '2015 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)', IEEE, pp. 1–5.
URL: *https://ieeexplore.ieee.org/document/7131906*

Recknagel [2017], *Taschenbuch für Heizung + Klimatechnik 2017/2018*, Recknagel - Sprenger - Albers.

Recogizer Group GmbH [2020]. Accessed: 2022-06-19.
URL: *https://recogizer.com/*

Rosenblatt, F. [1958], 'The perceptron: A probabilistic model for information storage and organization in the brain', *Psychological Review* **65**(6), 386–408.
URL: *https://doi.org/10.1037/h0042519*

Ross, S. and Bagnell, D. [2010], 'Efficient reductions for imitation learning', *Journal of Machine Learning Research - Proceedings* (9), 661–668.
URL: *https://proceedings.mlr.press/v9/ross10a.html*

Ruelens, F. [2016], *Residential Demand Response Using Reinforcement Learning: From Theory to Practice.*
URL: *https://limo.libis.be/primo-explore/search?vid=KADOCfromLogin=true*

Ruelens, F., Claessens, B. J., Quaiyum, S., de Schutter, B., Babuska, R. and Belmans, R. [2018], 'Reinforcement learning applied to an electric water heater: From theory to practice', *IEEE Transactions on Smart Grid* **9**(4), 3792–3800.
URL: *https://ieeexplore.ieee.org/document/7792709*

Ryu, S., Noh, J. and Kim, H. [2017], 'Deep neural network based demand side short term load forecasting', *Energies* **10**(1), 3.
URL: *https://www.mdpi.com/1996-1073/10/1/3*

Sammut, C. and Webb, G. I., eds [2017], *Encyclopedia of machine learning and data mining*, Springer Reference, 2nd ed. edn, Springer, New York.
URL: *https://link.springer.com/referencework/10.1007/978-1-4899-7687-1*

Sanaye, S. and Shirazi, A. [2013], 'Thermo-economic optimization of an ice thermal energy storage system for air-conditioning applications', *Energy and Buildings* **60**, 100–109.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778813000133*

Santos, S. F., Fitiwi, D. Z., Cruz, M. R., Cabrita, C. M. and Catalão, J. P. [2017], 'Impacts of optimal energy storage deployment and network reconfiguration on renewable integration level in distribution systems', *Applied Energy* **185**, 44–55.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261916314970*

Schaul, T., Quan, J., Antonoglou, I. and Silver, D. [2015], 'Prioritized experience replay', *arXiv preprint (2015)* .
URL: *https://www.deepmind.com/publications/prioritized-experience-replay*

Schild, T. P., Rademacher, M., Baranski, M. A. and Müller, D. [2019], 'Building automation systems in practice - advanced control methods'.
URL: *http://publications.rwth-aachen.de/record/753892*

Schubnel, B., Carillo, R., Hutter, A. and Alet, P.-J. [2018], 'Data-driven reinforcement learning for smart controllers in large building facilities', *CSEM Scientific and Technical Report* .
URL: *https://www.csem.ch/Doc.aspx?id=123853*

Shah, A., Nasir, H., Fayaz, M., Lajis, A. and Shah, A. [2019], 'A review on energy consumption optimization techniques in iot based smart building environments', *Information* **10**(3), 108.
URL: *https://www.mdpi.com/2078-2489/10/3/108*

Shang, Y., Wu, W., Guo, J., Ma, Z., Sheng, W., Lv, Z. and Fu, C. [2020], 'Stochastic dispatch of energy storage in microgrids: An augmented reinforcement learning approach', *Applied Energy* **261**, 114423.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261919321105*

Shekhar, S., Bansode, A. and Salim, A. [2021], A comparative study of hyper-parameter optimization tools, *in* '2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)', pp. 1–6.
URL: *https://doi.org/10.1145/3297067.3297080*

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D. and Riedmiller, M. [2014], 'Deterministic policy gradient algorithms', *Proceedings of Machine Learning Research* **2014**.
URL: *http://proceedings.mlr.press/v32/silver14.pdf*

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I. and et al. [2017], 'Mastering the game of go without human knowledge', *Nature* **550**(7676), 354–359.
URL: *https://www.nature.com/articles/nature24270*

Smarra, F., Jain, A., de Rubeis, T., Ambrosini, D., D'Innocenzo, A. and Mangharam, R. [2018], 'Data-driven model predictive control using random forests for building energy optimization and climate control', *Applied Energy* **226**, 1252–1272.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261918302575*

Sommer, T., Kessler, M. and Thorade, M. [2017], 'Fmpy: Free python library to simulate functional mock-up units (fmus)'. Accessed: 2022-05-31.
URL: *https://github.com/CATIA-Systems/FMPy*

Stepancic, M., Grancharova, A. and Kocijan, J. [2015], 'Adaptive mpc based on probabilistic black-box input-output model', *Comptes rendus de l'Académie bulgare des Sciences* **68**(6).
URL: *https://inis.iaea.org/search/*

Stoffel, P., Maier, L., Kümpel, A., Schreiber, T. and Müller, D. [2023], 'Evaluation of advanced control strategies for building energy systems', *Energy and Buildings* **280**, 112709.
URL: *https://www.sciencedirect.com/science/article/pii/S0378778822008805*

Sun, Y., Haghighat, F. and Fung, B. C. [2020], 'A review of the-state-of-the-art in data-driven approaches for building energy prediction', *Energy and Buildings* **221**(4), 110022.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778819339313*

Sutton, R. S. and Barto, A. [2018], *Reinforcement learning: An introduction*, Adaptive computation and machine learning, second edition edn, The MIT Press, Cambridge, MA and London.
URL: *https://mitpress.mit.edu/books/reinforcement-learning-second-edition*

Sutton, R. S., McAllester, D. A. and Singh, S. P. [2000], 'Policy gradient methods for reinforcement learning with function approximation', *Advances in Neural Information Processing Systems* pp. 1057–1063.
URL: *https://dl.acm.org/doi/10.5555/3009657.3009806*

Touzani, S., Prakash, A. K., Wang, Z., Agarwal, S., Pritoni, M., Kiran, M., Brown, R. and Granderson, J. [2021], 'Controlling distributed energy resources via deep reinforcement learning for load flexibility and energy efficiency', *Applied Energy* **304**(5), 117733.
URL: *https://www.sciencedirect.com/science/article/pii/S0306261921010801*

Vandael, S., Claessens, B., Ernst, D., Holvoet, T. and Deconinck, G. [2015], 'Reinforcement learning of heuristic ev fleet charging in a day-ahead electricity market', *IEEE Transactions on Smart Grid* **6**(4), 1795–1805.
URL: *https://ieeexplore.ieee.org/document/7056534*

Vázquez-Canteli, J., Kämpf, J. and Nagy, Z. [2017], 'Balancing comfort and energy consumption of a heat pump using batch reinforcement learning with fitted q-iteration', *Energy Procedia* **122**, 415–420.
URL: *https://www.sciencedirect.com/science/article/pii/S1876610217332629*

Vázquez-Canteli, J. R. e., Detjeen, T., Henze, G., Kämpf, J. and Nagy, Z. [2019], 'Multi-agent reinforcement learning for adaptive demand response in smart cities', *Journal of Physics: Conference Series* **1343**, 012058.
URL: *https://iopscience.iop.org/article/10.1088/1742-6596/1343/1/012058*

Vázquez-Canteli, J. R., Henze, G. and Nagy, Z. [2020], Marlisa: Multi-agent reinforcement learning with iterative sequential action selection for load shaping of grid-interactive connected buildings, *in* 'Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation', BuildSys '20, Association for Computing Machinery, New York, NY, USA, p. 170–179.
URL: *https://doi.org/10.1145/3408308.3427604*

Vázquez-Canteli, J. R. and Nagy, Z. [2019], 'Reinforcement learning for demand response: A review of algorithms and modeling techniques', *Applied Energy* **235**, 1072–1089.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261918317082*

Vázquez-Canteli, J. R., Ulyanin, S., Kämpf, J. and Nagy, Z. [2019], 'Fusing tensorflow with building energy simulation for intelligent energy management in smart cities', *Sustainable Cities and Society* **45**, 243–257.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S2210670718314380*

Wan, Z., Li, H. and He, H. [2018], Residential energy management with deep reinforcement learning, *in* '2018 International Joint Conference on Neural Networks (IJCNN): 2018 proceedings', IEEE, Piscataway, NJ, pp. 1–7.
URL: *https://ieeexplore.ieee.org/document/8489210*

Wang, S. and Ma, Z. [2008], 'Supervisory and optimal control of building hvac systems: A review', *HVAC&R Research* **14**(1), 3–32.
URL: *https://www.tandfonline.com/doi/abs/10.1080/10789669.2008.10390991*

Wang, Y., Velswamy, K. and Biao, H. [2017], 'A long-short term memory recurrent neural network based reinforcement learning controller for office heating ventilation and air conditioning systems', *Processes* **5**(4), 46.
URL: *https://www.mdpi.com/2227-9717/5/3/46*

Wang, Z. and Hong, T. [2020], 'Reinforcement learning for building controls: The opportunities and challenges', *Applied Energy* **269**, 115036.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261920305481*

Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M. and De Freitas, N. [2016], Dueling network architectures for deep reinforcement learning, *in* 'Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48', ICML'16, JMLR.org, p. 1995–2003.
URL: *https://dl.acm.org/doi/10.5555/3045390.3045601*

Wang, Z. and Srinivasan, R. S. [2015], A review of artificial intelligence based building energy prediction with a focus on ensemble prediction models, *in* '2015 Winter Simulation Conference (WSC)', pp. 3438–3448.
URL: *https://ieeexplore.ieee.org/document/7408504*

Wemhoff, A. [2012], 'Calibration of hvac equipment pid coefficients for energy conservation', *Energy and Buildings* **45**, 60–66.
URL: *https://www.sciencedirect.com/science/article/pii/S0378778811004658*

Wietschel, M., Plötz, P., Pfluger, B., Klobasa, M., Eßer, A., Haendel, M., Müller-Kirchenbauer, J., Kochems, J., Hermann, L., Grosse, B., Nacken, L., Küster, M., Pacem, J., Naumann, D., Kost, C., Kohrs, R., Fahl, U., Schäfer-Stradowsky, S., Timmermann, D. and Albert, D. [2018], 'Sektorkopplung: Definition, chancen und herausforderungen'.
URL: *https://www.econstor.eu/handle/10419/175374*

Wooldridge, M. [2001], Intelligent agents: The key concepts, Vol. 2322, pp. 3–43.
URL: *https://link.springer.com/chapter/10.1007/3-540-45982-0*$_1$

Yang, L., Nagy, Z., Goffin, P. and Schlueter, A. [2015], 'Reinforcement learning for optimal control of low exergy buildings', *Applied Energy* **156**, 577–586.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S030626191500879X*

Yang, S., Wan, M. P., Chen, W., Ng, B. F. and Dubey, S. [2020], 'Model predictive control with adaptive machine-learning-based model for building energy efficiency and comfort optimization', *Applied Energy* **271**, 115147.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261920306590*

Yuan, X., Pan, Y., Yang, J., Wang, W. and Huang, Z. [2019], 'Study on the application of reinforcement learning in the operation optimization of hvac system', *Building Simulation* **14**.
URL: *https://link.springer.com/article/10.1007/s12273-020-0602-9*

Zhang, F., Deb, C., Lee, S. E., Yang, J. and Shah, K. W. [2016], 'Time series forecasting for building energy consumption using weighted support vector regression with differential

evolution optimization technique', *Energy and Buildings* **126**, 94–103.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0378778816303899*

Zhang, X., Lu, R., Jiang, J., Hong, S. H. and Song, W. S. [2021], 'Testbed implementation of reinforcement learning-based demand response energy management system', *Applied Energy* **297**(1), 117131.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261921005705*

Zhang, Z., Chong, A., Pan, Y., Zhang, C., Lu, S. and Lam, K. [2018], 'A deep reinforcement learning approach to using whole building energy model for hvac optimal control', *ASHRAE/IBPSA-USA Building Performance Analysis Conference and SimBuild* .
URL: *https://cpb-us-w2.wpmucdn.com/blog.nus.edu.sg/dist/8/9027/files/2018/01/1308-revision-26hr9b4.pdf*

Zhao, Y., Li, T., Zhang, X. and Zhang, C. [2019], 'Artificial intelligence-based fault detection and diagnosis methods for building energy systems: Advantages, challenges and the future', *Renewable and Sustainable Energy Reviews* **109**, 85–101.
URL: *https://ideas.repec.org/a/eee/rensus/v109y2019icp85-101.html*

Zhong, S., Wang, X., Zhao, J., Li, W., Li, H., Wang, Y., Deng, S. and Zhu, J. [2021], 'Deep reinforcement learning framework for dynamic pricing demand response of regenerative electric heating', *Applied Energy* **288**, 116623.
URL: *https://www.sciencedirect.com/science/article/abs/pii/S0306261921001586*

Appendix

A List of publications in the course of this dissertation

A.1 Journal articles integrated into this dissertation

Schreiber, Thomas and Eschweiler, Sören and Baranski, Marc and Müller, Dirk. Application of two promising Reinforcement Learning algorithms for load shifting in a cooling supply system. Energy and Buildings. 2020. 229 (110490).
URL: https://www.sciencedirect.com/science/article/pii/S0378778820320922

Schreiber, Thomas and Netsch, Christoph and Baranski, Marc and Müller, Dirk. Monitoring data-driven Reinforcement Learning controller training: A comparative study of different training strategies for a real-world energy system. Energy and Buildings. 2021. 239 (110856).
URL: https://www.sciencedirect.com/science/article/pii/S0378778821001407

Schreiber, Thomas and Netsch, Christoph and Sören, Eschweiler and Wang, Tianyuan and Storek, Thomas and Baranski, Marc and Müller, Dirk. Application of data-driven methods for energy system modelling demonstrated on an adaptive cooling supply system. Energy. 2021. 230 (120894).
URL: https://www.sciencedirect.com/science/article/pii/S0360544221011427

Stoffel, Phillip and Maier, Laura and Kümpel, Alexander and Schreiber, Thomas and Müller, Dirk. Evaluation of advanced control strategies for building energy systems. Energy and Buildings. 2023. 280 (112709).
URL: https://www.sciencedirect.com/science/article/pii/S0378778822008805

A.2 Conference articles integrated into this dissertation

Schreiber, Thomas and Schwartz, Aron and Müller, Dirk. Towards an intelligent HVAC system automation using Reinforcement Learning. Journal of Physics: Conference Series. 2021. 2042 (Proceedings of CISBAT 2021).
URL: https://iopscience.iop.org/article/10.1088/1742-6596/2042/1/012028

Schreiber, Thomas and Evenschor, Vincent and Müller, Dirk. Improving the application of Reinforcement Learning for load shifting in a cooling system through state-of-the-art algorithms and hyper-parameter optimization. Proceedings of ECOS 2022.
URL: https://orbit.dtu.dk/en/publications/proceedings-of-ecos-2022-the-35th-international-conference-on-eff

A.3 Journal articles that were contributed to in the course of this dissertation

Bode, Gerrit and Schreiber, Thomas and Baranski, Marc and Müller, Dirk. A time series clustering approach for Building Automation and Control Systems. Applied Energy. 2019. 238 (1337-1345).
URL: https://www.sciencedirect.com/science/article/abs/pii/S0306261919302089

Wirtz, Marco and Hahn, Maria and Schreiber, Thomas and Müller, Dirk. Design optimization of multi-energy systems using mixed-integer linear programming: Which model complexity and level of detail is sufficient?. Energy Conversion and Management. 2021. 240 (114249).
URL: https://www.sciencedirect.com/science/article/pii/S0196890421004258

Mans, Michael and Blacha, Tobias and Schreiber, Thomas and Müller, Dirk. Development and Application of an Open-Source Framework for Automated Thermal Network Generation and Simulations in Modelica. Energies. 2022. 15(12). 4372.
URL: https://doi.org/10.3390/en15124372

Wirtz, Marco and Schreiber, Thomas and Müller, Dirk. Survey of 53 Fifth-Generation District Heating and Cooling (5GDHC) Networks in Germany. Energy Technology. 2022. 10(9). 2200749.
URL: https://doi.org/10.1002/ente.202200749

Wirtz, Marco and Heleno, Miguel and Moreira, Alexandre and Schreiber, Thomas and Müller, Dirk. 5th generation district heating and cooling network planning: A Dantzig–Wolfe decomposition approach. Energy Conversion and Management. 2023. 276 (116593).
URL: https://www.sciencedirect.com/science/article/abs/pii/S0196890422013711

Wirtz, Marco and Heleno, Miguel and Romberg, Hannah and Schreiber, Thomas and Müller, Dirk. Multi-period design optimization for a 5th generation district heating and cooling network. Energy and Buildings. 2023. 284 (112858).
URL: https://www.sciencedirect.com/science/article/abs/pii/S0378778823000889

A.4 Conference articles that were contributed to in the course of this dissertation

Bode, Gerrit and Schreiber, Thomas and Baranski, Marc and Müller, Dirk. Comparing unsupervised and supervised machine learning techniques to improve time-series classification in building and energy data. Proceedings of ECOS 2018.
URL: https://www.researchgate.net/publication/327281617

Bode, Gerrit and Stinner, Florian and Baranski, Marc and Brümmendorf, Erik and Cai, Xiaoye and Kümpel, Alexander and Schraven, Markus and Schreiber, Thomas and Stoffel, Phillip and Storek, Thomas and Müller, Dirk. From plans to programs: A holistic toolchain

for building data applications Journal of Physics: Conference Series. 2019. 1343 (Proceedings of CISBAT 2019).
URL: https://iopscience.iop.org/article/10.1088/1742-6596/1343/1/012117

Stoffel, Phillip and Baranski, Marc and Schreiber, Thomas and Müller, Dirk. Monitoring, Analyse der Energieeffizienz und nachhaltige Nutzung eines Geothermiefeldes zur Gebäudeklimatisierung. Konferenzband: Der Geothermie Kongress 2019.
URL: https://publications.rwth-aachen.de/record/780999

Schreiber, Thomas and Bode, Gerrit and Baranski, Marc and Müller, Dirk. An automated feature selection for time-series classification in building automation and control systems. Proceedings of ECOS 2019.
URL: https://www.proceedings.com/52893.html

Stinner, Florian and Yang, Yingying and Schreiber, Thomas and Bode, Gerrit and Baranski, Marc and Müller, Dirk. Generating Generic Data Sets for Machine Learning Applications in Building Services Using Standardized Time Series Data. Proceedings of ISARC 2019.
URL: https://www.iaarc.org/publications/

Cai, Xiaoye and Schild, Thomas and Schreiber, Thomas and Müller, Dirk. Modeling of Petri-Net-based control algorithms for the simulation-based improvement of the planning process of building energy systems. Journal of Physics: Conference Series. 2019. 1343 (Proceedings of CISBAT 2019).
URL: https://iopscience.iop.org/article/10.1088/1742-6596/1343/1/012123

Oberkirsch, Laurin and Kriwet, Jonathan and Baranski, Marc and Schreiber, Thomas and Storek, Thomas and Müller, Dirk. Auto-generation of hybrid automata for real-time operation optimization of building energy systems. Proceedings of ECOS 2020.
URL: https://www.proceedings.com/55242.html

Maier, Laura and Schreiber, Thomas and Kümpel, Alexander and Mehrfeld, Philipp and Müller, Dirk. Integration of Advanced Control Methods into Mode-Based Control Logics of Building Energy Systems. Proceedings of ECOS 2021.
URL: https://doi.org/10.52202/062738-0058

Beckhölter, Tobias and Matthes, Julian and Schreiber, Thomas and Müller, Dirk. Comparison of optimization-based operation strategies for a hydrogen-based district energy system. Proceedings of ECOS 2022.
URL: https://orbit.dtu.dk/en/publications/proceedings-of-ecos-2022-the-35th-international-conference-on-eff

Zhang, Yizhuo and Wiederhöft, Tori and Schreiber, Thomas and Müller, Dirk. Optimization of a grid-interactive building energy system considering user satisfaction. Proceedings of BauSim Conference 2022: 9th Conference of IBPSA-Germany and Austria.
URL: https://publications.ibpsa.org/conference/?id=bausim2022

Schreiber, Thomas and Beckhölter, Tobias and Derzsi, Kai and Droste, Kai and Karuvingal, Rahul and Nie, Yi and Wackerbauer, David and Wirtz, Marco and Welter, Sarah and Zhang, Yizhuo and Müller, Dirk. Planning the Design and Operation of Urban Energy Systems with Limited Data Availability: A Holistic Open-Source Tool Chain. Journal of Physics: Conference Series. 2023. 2600 (Proceedings of CISBAT 2023).
URL: https://https://iopscience.iop.org/article/10.1088/1742-6596/2600/8/082023