Deep Reinforcement Learning for room temperature control: a black-box pipeline from data to policies

Deep Reinforcement Learning for room temperature control: a black-box pipeline from data to policies

L Di Natale[1,2], B Svetozarevic[1], P Heer[1] and C N Jones[2]

[1] Urban Energy Systems Laboratory, Swiss Federal Laboratories for Material Science and Technology (Empa), Dübendorf, Switzerland

[2] Laboratoire d'Automatique, Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland

E-mail: loris.dinatale@empa.ch

Abstract. Deep Reinforcement Learning (DRL) recently emerged as a possibility to control complex systems without the need to model them. However, since weeks long experiments are needed to assess the performance of a building controller, people still have to rely on accurate simulation environments to train and tune DRL agents in tractable amounts of time before deploying them, shifting the burden back to the original issue of designing complex models. In this work, we show that it is possible to learn control policies on simple black-box linear room temperature models, thereby alleviating the heavy engineering usually required to build accurate surrogates. We develop a black-box pipeline, where historical data is taken as input to produce room temperature control policies. The trained DRL agents are capable of beating industrial rule-based controllers both in terms of energy consumption and comfort satisfaction, using novel penalties to introduce expert knowledge, i.e. to incentivize agents to follow expected behaviors, in the reward function. Moreover, one of the best agents was deployed on a real building for one week and was able to save energy while maintaining adequate comfort levels, indicating that low-complexity models might be enough to learn control policies that perform well on real buildings.

1. Introduction

Today, most buildings are still controlled using heuristic rules, which are known to be suboptimal in terms of energy savings and occupant comfort satisfaction. As a counter to the reactive nature of such rule-based approaches, predictive methods, such as Model Predictive Control (MPC), arose to offer better performance [1]. However, MPC relies on accurate models to find the optimal control input at each time step. Such models are hard to derive for buildings due to their complex and highly nonlinear dynamics, which leads to high development costs.

Leveraging the growing connectivity of buildings, several data-driven control algorithms were recently proposed to alleviate some of the issues linked to the design of accurate models. For example, researchers proposed adaptive and robust MPC schemes to deal with model errors, such as in [2], but this only increases the required engineering further. On the other hand, researchers also took advantage of available data to construct black-box models to use in MPC, thus avoiding the complex physics-based modelling of building dynamics, like in [3]. However, these models might not follow the laws of physics and induce complex optimization routines for MPC. In all cases, accurate building models are needed to develop high performance predictive controllers, but they require significant expertise during the design phase.

1.1. Reinforcement Learning for building control

Deep Reinforcement Learning (DRL) recently arose as another interesting control paradigm due to its ability to learn control policies through direct interaction with a system, hence bypassing the need for models. However, the length and complexity of real building control experiments remains a major obstacle in the field and most DRL controllers are never deployed on physical systems [4, 5]. Indeed, people still need to rely on simulations to train DRL agents, as it is not feasible to wait weeks to get results for each experiment. This inevitably shifts the burden back to finding accurate building models, either from first principles [6], with a black-box approach [7], or using tools like EnergyPlus [8], but such models are not trivial to calibrate [9, 10].

1.2. Main contributions

To alleviate the required model engineering, we develop a black-box pipeline from historical data to room temperature control policies[1], similar to [7], with two key contributions: (i) Using linear models to mitigate extrapolation errors of black-box models and a novel reward function, we learn control policies that beat rule-based controllers in simulation both in terms of energy consumption and comfort satisfaction. (ii) One of the best agents was deployed on the real building during a week and it performed better than a rule-based counterpart, saving energy and maintaining adequate comfort levels, thus demonstrating that the learned policy is not only effective in simulation. Remarkably, this pipeline indicates that linear room temperature models might be sufficient to learn meaningful control policies, confirming an intuition from [10].

2. Framework

2.1. Case study

In this work, we control the temperature of one of the two bedrooms in the UMAR unit at the NEST demonstrator at Empa [11]. Since the bedrooms present similar architectures, we design DRL agents to control the heating/cooling system in one of them. At deployment time, we then regulate the temperature of the other bedroom with a rule-based algorithm as a benchmark. Each room has water-based heating/cooling panels on the ceiling and valves that control the water flowing through them. Consequently, DRL agents decide how much to open the valves - hence effectively choosing how much to turn the heating/cooling system on - each 15 minutes, which is deemed enough to capture the slow thermal dynamics of the room.

2.2. Linear room temperature model

To avoid heavy engineering, we use black-box linear autoregressive models with exogenous inputs (ARX) of the rooms in UMAR, taken from [12], to train our agents. Since the sun does not shine at night, the solar irradiation profile is binned into 9 intervals of 2h, creating the one-hot-encoding variables $S^1, \ldots, S^9$ to capture different impacts of the sun on the room temperature depending on its orientation in the sky. The bedroom temperature model then has the form:

$$T_{t+1}^{room} = \left[\beta_1,\ \beta_2,\ \ldots,\ \beta_{13}\right] \times \left[T_t^{room},\ U_t,\ T_t^{neigh},\ T_t^{out},\ S_t^1,\ \ldots,\ S_t^9\right]^T, \tag{1}$$

where $\beta_1, \ldots, \beta_{13}$ are the learned model coefficients, T^{room} the room temperature, U the control input, i.e. how much the valves are open, T^{neigh} the temperature in the neighboring room, T^{out} the ambient temperature, and the subscripts indicate the time step.

Since all the rooms in UMAR are connected to the same thermal energy meter, we cannot access the individual consumption of the modeled bedroom and thus use the valves opening U_t as a proxy for energy consumption throughout this work. U_t is defined as a percentage with values in $[0, 1]$, representing how long the valves are open for each 15 minute time interval.

[1] The code can be found here: `https://gitlab.nccr-automation.ch/loris.dinatale/cisbat21`

As comfort measure for the occupants, we predefine dynamic bounds for the room temperature of [22°C; 23°C] at night, from 8pm to 8am. During the day, when the bedroom is unoccupied, they are relaxed to [20°C; 23°C] in the heating and [22°C; 25°C] in the cooling season. Comfort violations over a given period of time are then expressed in Kelvin Hours, summing the difference between the temperature and the bounds at each time step.

3. Deep Reinforcement Learning agents

3.1. Definition of the DRL agents

In Reinforcement Learning (RL), agents observe the current state of the system and decide which action to take, for which they receive a reward from the environment. Their objective is then to maximize the expected discounted sum of rewards [13]. In our case, the state-space is similar to the inputs of the building models from Section 2.2, with agents additionally knowing the current lower and upper bounds on the room temperature, as well as the *case*[2] they are in, to know if opening the valves will heat or cool the room. Since agents are monitoring one room temperature, there is only one control variable: U_t. We parametrize our agents with recurrent neural networks and solve the RL problem with the Proximal Policy Optimization (PPO) algorithm [13]. We choose a discount factor of $\gamma = 0.95$ and train the agents in episodes of 24h, i.e. 96 steps, with a novel reward function described in the following Section.

3.2. Reward function

We want to find control policies simultaneously minimizing the thermal energy consumption and maintaining satisfactory comfort levels, which is classically achieved with the following reward:

$$R_t^{base} = -\max\{0, B_t^{low} - T_t^{room}\} - \max\{0, T_t^{room} - B_t^{high}\} - \lambda E_t, \tag{2}$$

where B_t^{low} and B_t^{high} are respectively the lower and upper temperature bounds, E_t the energy consumption and λ the balancing factor between the comfort violations and the energy consumption, which is fixed at 10 in our experiments.

In this work, to facilitate the learning process, we add supplementary penalties against unwanted behaviors. Mathematically, we have the following reward in the heating case:

$$\begin{aligned} R_t^{heating} &= R_t^{base} && (3)\\ &\quad - (1 - U_t) \times (\max\{0, B_t^{low} - T_t^{room}\})^2 && (4)\\ &\quad - \quad U_t \quad \times (\max\{0, T_t^{room} - B_t^{high}\})^2 && (5) \end{aligned}$$

The intuition behind it is that we would like to transfer *expert* knowledge to agents, as people typically have straightforward expectations about the behavior of a controller maintaining the temperature of a room between given bounds. For example, occupants want controllers to heat a room if the temperature is below the lower bound. This is reflected in Equation (4), as agents get penalized if $U_t < 1$, i.e. if the valves are not fully opened. Similarly, when the temperature is too high, we want agents to turn the heating off, and we hence penalized them if $U_t > 0$ in Equation (5). Scaling these additional penalties proportionally to how much heating power is used by the agents allows us to penalize them proportionally to their error. Furthermore, we employ quadratic penalties, which are small when the temperature is near the bound, so that agents retain enough expressiveness and might let the temperature go out of bounds on purpose, for example in anticipation of high heat gains.

In the cooling case, for R_t^{cool}, the factors $(1-U_t)$ and U_t are exchanged in Equations (4)-(5) to reflect that agents should cool a room when it is too hot and stop cooling when the temperature drops below the lower bound. These rewards R_t^{heat} and R_t^{cool} turned out to drastically improve the learning of agents, allowing for faster convergence to better performing control policies.

[2] We differentiate between two cases: the heating and the cooling season.

4. Results

To assess the performance of our agents, we compare it to two rule-based controllers: a bang-bang controller with a one degree hysteresis (*Baseline 1*) and another bang-bang controller without hysteresis and tracking a reference defined 0.5 degrees off the bound (*Baseline 2*).

Since we randomly sample initial conditions for each episode, it might start with a room temperature out of the comfort bounds, which leads to *unavoidable* penalties for any controller. To keep track of these unavoidable penalties, we implement an additional algorithm fully opening or closing the valves until the room temperature reaches the bounds for the first time.

4.1. Performance in simulation

A comparison of the performance of one of the best agents[3] and the baselines - on the same room - can be found in Figure 1. All the numbers were obtained after the subtraction of the unavoidable penalties mentioned above, since no controller could have done anything about it. This gives a clearer picture of how much the DRL agent is able to improve upon the performance of the two baselines. Note that the comfort violations discussed in this Section are given by the sum of the first two terms of Equation (2), i.e. without the additional quadratic penalties used in the reward function, as we just want to analyze how far from the bounds each controller was.

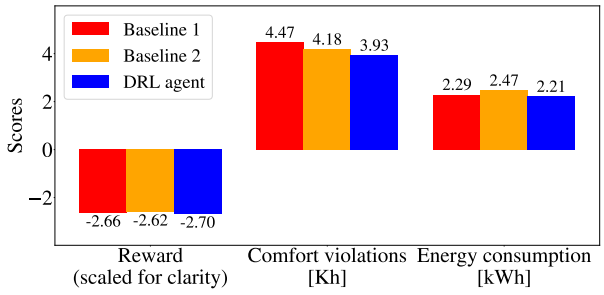

Figure 1. Performance of the three controllers in simulation, computed from their mean performance over more than 5'700 episodes, where the rewards were divided by 10 and the energy consumption multiplied by 10 for clarity.

On average, the agent obtained 1.7% and 3.2% less rewards than *Baseline 1* and *Baseline 2*, respectively, which is a consequence of these rule-based controllers never receiving the additional quadratic penalties from Equations (4)-(5) by definition. Nonetheless, when we look at both the comfort violations and energy consumption, the agent is found to strike a better compromise than both baselines. Indeed, the agent reduced the amount of violation by 12% while saving 3.4% energy compared to *Baseline 1*. Contrasted with *Baseline 2*, on the other hand, the comfort is improved by 6% while using 10.4% less energy. The DRL agent is thus able to simultaneously improve the two main objectives of our room temperature control framework compared to the two industrial rule-based controllers.

4.2. Performance on the real building

To assess the actual performance of the agent analyzed in Section 4.1, we deployed it on the real building at NEST in December 2020. We took advantage of the similarities between the two bedrooms in UMAR to simultaneously deploy *Baseline 2* in the other one, allowing us to compare the behavior of both controllers under similar external conditions in Figure 2. Over the course of five days, both controllers achieved similar comfort levels, with the agent performing 2.1% worse than the rule-based controller. However, it managed to use 13.5% less energy.

Looking at the time series in Figure 2, we can observe desired preheating behaviors from the DRL agent in green shaded areas, when it started to open the valves earlier in the afternoon

[3] Due to a different implementation, the agent was trained with time steps of 2 h instead of 15 min. However, it was then successfully deployed on the real building to provide new control inputs every 15 min (Section 4.2).

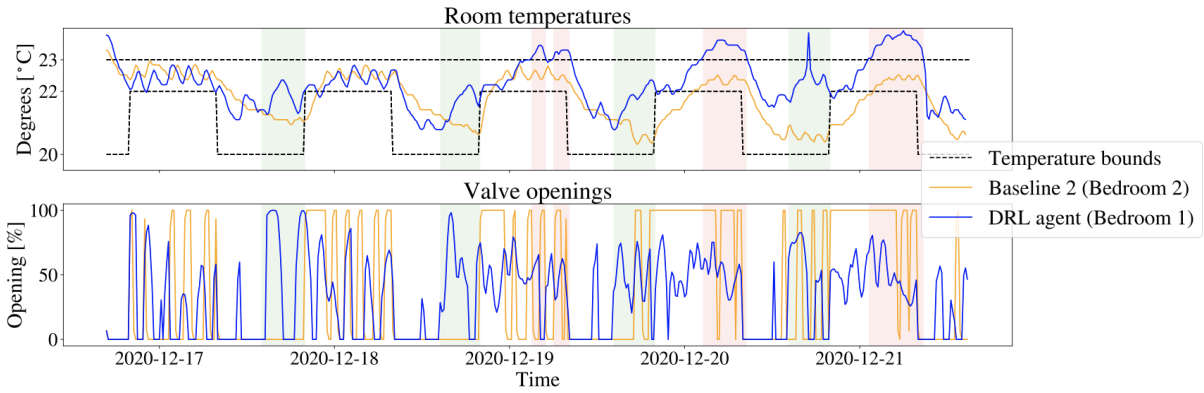

Figure 2. Results of the real experiment in UMAR. For the sake of clarity, the valves pattern of the agent are smoothed with a Gaussian filter. Green shaded areas emphasize expected preheating behaviors of the agent, while red ones expose unwanted overheating situations.

to meet the tightening of the comfort bounds at 8pm. On the other hand however, we can also notice unwanted red shaded overheating situations, when the agent kept heating the room even though the room temperature was already too high. This was unexpected; agents should indeed never use energy leading to comfort violations, as it goes against both their primary objectives and leads to low rewards. This issue is discussed in the following Section.

5. Discussion

5.1. The reward function

The fact that our agent was able to strike a better balance between comfort satisfaction and energy consumption while receiving less rewards than the baselines (Figure 1) indicates that the reward function is not optimal yet. While the novel quadratic penalties in Equations (4)-(5) allow agents to converge faster and to better performing policies, additional considerations are still needed to better shape the reward, and we leave it for future work.

5.2. Real experiments

Looking at Figure 2, one can see the performance of the agent deteriorate along the experiment, with more and more overheating issues each night. We suspect it to be partly due to the episodic training framework and the parametrization of DRL agents with Long Short Term Memory (LSTM) Networks. Indeed, while it was trained on 24 h long episodes, the agent was then deployed in the real building for five days straight, and the LSTMs might thus have built up erroneous memory over these longer sequences of data. To counter that in future experiments, we aim to reset the agent's memory each day to mimic the training framework. One could also increase the length of training episodes, but that would require better and more accurate models, shifting the burden back to the modelling part, which we want to avoid or at least keep at a minimum.

Finally, one has to keep in mind that experiments on real buildings can never be compared in a straightforward manner. Even though we took advantage of the similarities between the two bedrooms in UMAR, they have differences, like their number of doors and external walls, or their occupancy pattern. This leads to slightly different room dynamics, as the temperature in *bedroom* 1 is for example decreasing faster than in *bedroom* 2 when both controllers are off (Figure 2). Remarkably, the DRL agent used less heating energy than the baseline despite the controlled room temperature having a tendency to drop faster.

6. Conclusion

In this work, we designed a black-box pipeline from historical data to room temperature control policies. To avoid the usual heavy engineering required to build accurate models of the system to control, we developed linear temperature models to train Deep Reinforcement Learning agents in simulation. Using an augmented reward function, DRL agents were able to simultaneously maintain adequate comfort levels and save energy compared to industrial rule-based controllers, both in simulation and on the real building. These results suggest that low-complexity black-box models might suffice to train agents to control a room temperature.

In future works, we plan to improve this black-box pipeline, extending the current framework to more complex building control problems, designing more informative black-box building models that still require as little engineering as possible, and developing better performing agents with other reward functions and control policy parametrizations. We already improved the reward function further and obtained agents capable to beat rule-based controllers also in terms of rewards in simulation.

Acknowledgements

This research was supported by the Swiss National Science Foundation under NCCR Automation, grant agreement 51NF40_180545. We would additionally like to thank Felix Bünning, who kindly accepted to share the data and linear models of UMAR with us, and the rest of the team at Empa for their help during the deployment of the controllers.

References

[1] Gianluca Serale, Massimo Fiorentini, Alfonso Capozzoli, Daniele Bernardini, and Alberto Bemporad. Model predictive control (MPC) for enhancing building and HVAC system energy efficiency: Problem formulation, applications and opportunities. *Energies*, 11(3):631, 2018.

[2] Lukas Hewing, Melanie N Zeilinger, et al. Data-Driven Distributed Stochastic Model Predictive Control with Closed-Loop Chance Constraint Satisfaction. *arXiv preprint arXiv:2004.02907*, 2020.

[3] Francesco Smarra, Achin Jain, Tullio De Rubeis, Dario Ambrosini, Alessandro D'Innocenzo, and Rahul Mangharam. Data-driven model predictive control using random forests for building energy optimization and climate control. *Applied energy*, 226:1252–1272, 2018.

[4] José R Vázquez-Canteli and Zoltán Nagy. Reinforcement learning for demand response: A review of algorithms and modeling techniques. *Applied energy*, 235:1072–1089, 2019.

[5] Zhe Wang and Tianzhen Hong. Reinforcement learning for building controls: The opportunities and challenges. *Applied Energy*, 269:115036, 2020.

[6] Hepeng Li, Zhiqiang Wan, and Haibo He. Real-Time Residential Demand Response. *IEEE Transactions on Smart Grid*, 2020.

[7] Bratislav Svetozarevic, Christian Baumann, Simon Muntwiler, Loris Di Natale, Philipp Heer, and Melanie Zeilinger. Data-driven control of room temperature and bidirectional EV charging using deep reinforcement learning: simulations and experiments. *arXiv preprint arXiv:2103.01886*, 2021.

[8] William Valladares, Marco Galindo, Jorge Gutierrez, Wu-Chieh Wu, Kuo-Kai Liao, Jen-Chung Liao, Kuang-Chin Lu, and Chi-Chuan Wang. Energy optimization associated with thermal comfort and indoor air control via a deep reinforcement learning algorithm. *Building and Environment*, 155:105–117, 2019.

[9] Xianzhong Ding, Wan Du, and Alberto Cerpa. OCTOPUS: Deep reinforcement learning for holistic smart building control. In *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 326–335, 2019.

[10] Zhiang Zhang, Adrian Chong, Yuqi Pan, Chenlu Zhang, and Khee Poh Lam. Whole building energy model for HVAC optimal control: A practical framework based on deep reinforcement learning. *Energy and Buildings*, 199:472–490, 2019.

[11] Empa. NEST. `https://www.empa.ch/web/nest/overview`. Accessed: 24.12.2020.

[12] Felix Bünning, Ahmed Aboudonia, Benjamin Huber, Philipp Heer, Roy Smith, and John Lygeros. Linear regression is a competitive approach compared to machine learning methods in building MPC. *Manuscript in preparation*, 2021.

[13] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.