Low Emission Building Control with Zero-Shot Reinforcement Learning

Scott Jeen[13], **Alessandro Abate**[23], **Jonathan M. Cullen**[1]
[1] University of Cambridge
[2] University of Oxford
[3] Alan Turing Institute
{srj38, jmc99}@cam.ac.uk & alessandro.abate@cs.ox.ac.uk

Abstract

Heating and cooling systems in buildings account for 31% of global energy use, much of which are regulated by Rule Based Controllers (RBCs) that neither maximise energy efficiency nor minimise emissions by interacting optimally with the grid. Control via Reinforcement Learning (RL) has been shown to significantly improve building energy efficiency, but existing solutions require access to building-specific simulators or data that cannot be expected for every building in the world. In response, we show it is possible to obtain emission-reducing policies without such knowledge *a priori* – a paradigm we call zero-shot building control. We combine ideas from system identification and model-based RL to create PEARL (**P**robabilistic **E**mission-**A**bating **R**einforcement **L**earning) and show that a short period of active exploration is all that is required to build a performant model. In experiments across three varied building energy simulations, we show PEARL outperforms an existing RBC once, and popular RL baselines in all cases, reducing building emissions by as much as 31% whilst maintaining thermal comfort. Our source code is available online via `https://enjeeneer.io/projects/pearl/`.

1 Introduction

Heating and cooling systems in buildings account for 31% of global energy use, primarily in managing occupant thermal comfort and hygiene (Cullen and Allwood 2010). Such systems are usually regulated by rule-based controllers (RBCs) that take system temperature as input, use a temperature setpoint as an objective, and actuate equipment to minimise the error between objective and current state. Whilst usefully simple, RBCs do not maximise energy efficiency, nor can they perform *demand response*: the manipulation of power consumption to better match demand with supply. New techniques that demonstrate this capability across generalised settings would prove valuable climate change mitigation tools.

An option for obtaining control polices across complex, unknown settings is Reinforcement Learning (RL) (Sutton and Barto 2018). Where existing advanced control techniques, like the receding-horizon architecture Model Predictive Control (MPC), require the specification of a dynamical model that can be expensive to obtain (Borrelli, Bemporad, and Morari 2017), RL's strength is in obtaining polices *tabula rasa*, and updating their parameters *online* as the environment evolves. Recent successes in complex physics tasks (Lillicrap et al. 2016), gaming (Silver et al. 2017), and robotics (Gu et al. 2017) have highlighted these characteristics. With this generality, one can imagine RL agents being placed in *any* energy-intensive setting, and feasibly learning to control them more efficiently, minimising emissions.

Recent applications of RL to building control have indeed shown marked energy efficiency improvements over conventional controllers. Chen et al. (2018) used Q-learning to control HVAC and window actuation in a residential building with 23% less energy than the existing RBC. Similarly, Zhang et al. (2019b) used Asynchornous Advantage Actor Critic (A3C) to reduce heating demand in a real office building by 16.7%. However, these contributions are limited by their deployment paradigm, which is:

1. **Simulate** the real-world environment's transition dynamics $p(s_{t+1}|s_t, a_t)$ using a physics-based/generative model;
2. **Pre-train** the agent by sampling the simulator sequentially and updating parameters until convergence; and
3. **Deploy** the pre-trained agent in the real-world environment.

We contend that the need to *simulate* the environment *a priori* limits real-world scalability. In the context of building control, creating an accurate simulator in *EnergyPlus* (Crawley et al. 2001), the favoured software, can take an expert months, and is impossible without knowledge of the building topology and thermal parameters. An alternative, is to deploy agents in the real environment *without* pre-training, granting them a short commissioning period of 3 hours to collect data and model the state-action space (Lazic et al. 2018). We call this **zero-shot** building control after Socher et al. (2013)'s use for out-of-distribution image classification. Zero-shot control could scale to any building given sufficient sensor installation, but Lazic et al. (2018)'s agent only improved cooling costs by 9%, poorer performance than the agents trained in simulation. To maximise the emission-abating potential of RL building control, we need new systems that can elicit the performance of pre-trained agents whilst being deployed zero-shot.

In this paper, our primary contribution is showing that deep reinforcement learning algorithms can find performant

building control policies *online*, without pre-training. We achieve this with a new approach called PEARL, showing it can reduce annual emissions by up to 31.46% compared with an RBC whilst maintaining thermal comfort. PEARL is simple to commission, requiring no historical data or simulator access *a priori*, and capable of generalising across varied building archetypes. The scaled deployment of such systems could prove a cost-effective method for tackling climate change.

2 Related Work

Previous research on RL for building control has focused mainly on model-free algorithms (Yu et al. 2021). In this setting, the agent uses data collected from the environment to learn a policy π that maps states s_t from a state-space $\mathcal{S}$ to a probability distribution over the action-space $\mathcal{A}$ i.e. $\pi : \mathcal{S} \rightarrow \mathcal{P}(\mathcal{A})$. To obtain an optimal policy, the agent must necessarily visit many states $s \in \mathcal{S}$ and trial many actions $a \in \mathcal{A}$. Doing so is data inefficient, with Deep-Q Learning (Mnih et al. 2013), Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al. 2016), and Proximal Policy Optimisation (PPO) (Schulman et al. 2017) each taking in the order 10^7 samples in complex, simulated environments to obtain optimal polices. Such data inefficiency has been corroborated in the building control literature. Wei, Wang, and Zhu (2017) train a Deep-Q agent to control the HVAC equipment of a 5-zone building with 35% reduction in energy cost, but pre-train for 8 years in a simulator. Similarly, Valladares et al. (2019) require 10 years of simulated data to pre-train a Double-Q agent to control the HVAC in a university classroom with 5% energy savings. Such data-intensive pre-training can only be achieved in bespoke building simulators that are time-consuming to create, or impossible to specify, in most cases.

In contrast, model-based RL algorithms have demonstrated better sample efficiency. Here, the agent learns the system dynamics mapping from current state-action pair (s_t, a_t) to next state s_{t+1}, often called the agent's *model* i.e. $f_\theta : (s_t, a_t) \rightarrow s_{t+1}$. Between interactions with the environment, the agent samples its model to create additional data for updating the policy, or to predict the expected reward of a range of candidate action sequences $a_{t:t+H-1}$ to time horizon H. Either procedure reduces the agent's reliance on samples from the environment, improving data efficiency.

A popular choice of function approximator are Gaussian Processes (GP), which offer uncertainty quantification and work well with small datasets (Williams and Rasmussen 2006). PILCO uses GPs, showing state-of-the-art data efficiency on robotic tasks (Deisenroth and Rasmussen 2011), and Jain et al. (2018) used a similar approach to curtail hotel energy-use during a simulated demand response event. However, inference using a data set of size n has complexity $\mathcal{O}(n^3)$ which becomes intractable with more than a few thousand samples (Hensman, Fusi, and Lawrence 2013), limiting their applicability for modelling building transition functions with arbitrarily large training sets.

An alternative is to model the transition function using Deep Neural Networks (DNNs). Nagabandi et al. (2018) combined DNNs with MPC to solve the *Swimmer* task on the MuJoCo benchmark using 20x fewer datapoints than a model-free approach. In the building control setting, Zhang et al. (2019a) built a similar agent that reduced energy consumption in a data centre by 21.8% with 10x fewer training steps than a model-free algorithm. Jain et al. (2020) then tested the algorithm in-situ, finding an 8% energy reduction. Despite encouraging data efficiency, the performance of such agents is hampered by overfitting, or *model bias*, in the low-data regime, causing poor generalisation to unobserved transitions. Ding, Du, and Cerpa (2020) attempt to mitigate model bias by deploying an ensemble of DNNs to model the transition function of a large multi-zone building showing they can achieve 8.2% energy savings in 10.5x fewer timesteps than model-free approaches. Although the ensemble allows for the quantification of *epistemic* uncertainty, *aleatoric* uncertainty is not captured, potentially limiting performance.

To the best of our knowledge, Lazic et al. (2018) is the only work that attempts to learn a zero-shot building control policy by interacting with the real environment. Their agent fits a linear model of a datacentre's thermal dynamics using data obtained in a three-hour commissioning period, and selects actions by optimising planned trajectories using MPC. During commissioning, the agent explores the state-action space by performing a uniform random walk in each control variable, bounded to a safe operating range informed by historical data. Their choice of model expedites learning, but limits agent performance as the building's non-linear dynamics are erroneously linearised. In this study we aim to preserve the data efficiency of this approach whilst improving performance with expressive deep networks.

3 PEARL: Probabilistic Emission-Abating Reinforcement Learning

Problem formulation. We consider the standard reinforcement learning setup in which an agent takes actions in an environment at discrete timesteps to maximise the cumulative sum of future rewards. We model the environment as an infinite-horizon Markov Decision Process (MDP) characterised by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \gamma)$, where $\mathcal{S} \in \mathbb{R}^{d_s}$ and $\mathcal{A} \in \mathbb{R}^{d_a}$ are continuous state and action spaces, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is a reward function, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ is a transition function, and $\gamma \in [0, 1]$ is a discount parameter. In our case we do not discount and set $\gamma = 1$. Despite this formulation, the state s_t is not fully observed, and is instead partially observed via an observation space $\mathcal{O} \in \mathbb{R}^{d_o}$. To allow us to apply standard RL techniques for MDPs, the agent is passed a recent trajectory of observations as a state representation i.e. $s_t = (o_t, o_{t-1}, \ldots, o_{t-h})$ where h is the history length (Kaelbling, Littman, and Cassandra 1998). The agent selects actions using its policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$, which it manipulates to maximise expected return from the current state $G_t = \mathbb{E}[\sum_{t=0}^{H} \gamma^t \mathcal{R}(s_t, a_t)]$, where H is a finite time horizon.

We use the following sub-sections to detail the components of our proposed approach PEARL, and summarise our system in Figure 1 and Algorithm 1.

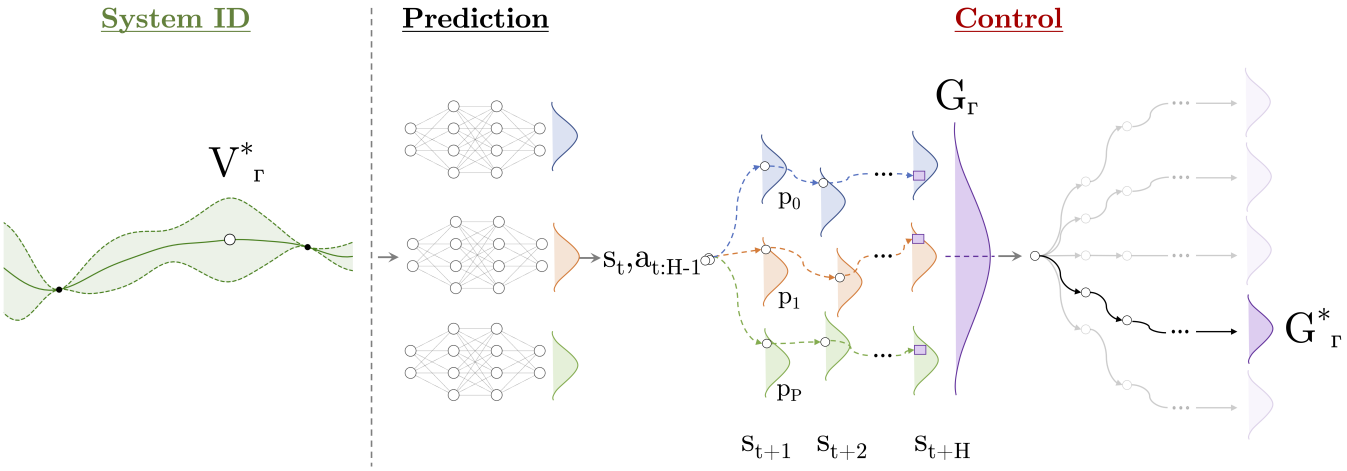

Figure 1: PEARL: our end-to-end deep reinforcement learning approach. System ID: the agent takes actions to explore parts of the state-space with highest predictive variance $\circ = V^*_\Gamma$ to maximise information gain. Prediction: system dynamics modelled as an ensemble of probabilistic deep neural networks. Control: trajectory sampling used to predict future rewards G_Γ of one action sequence $a_{t:H-1}$, which is compared with many others to find the trajectory with optimal return G^*_Γ.

3.1 Prediction

We follow the schema of model-based reinforcement learning where our task is to fit a function $\tilde{f}_\theta$ that approximates the true forward dynamics of the system $f(s_{t+1}, (s_t, a_t))$ given a dataset of experience collected from the environment $\mathcal{D} = [s_{n+1}, (s_n, a_n)]_{n=1}^N$. We employ probabilistic DNNs to learn this mapping, which provide data-efficient approximations of complex system dynamics and allow agents to incorporate prediction uncertainty into action selection (Gal, McAllister, and Rasmussen 2016; Higuera, Meger, and Dudek 2018). Where traditional, *deterministic* DNNs output point predictions given an input, here our probabilistic DNNs output distributions over the output nodes parameterised by a multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and diagonal covariance matrix Σ; i.e: $\tilde{f}_\theta(s_t, a_t) = \mathcal{N}(\mu_\theta(s_t, a_t), \Sigma_\theta(s_t, a_t))$. The agent maximises the likelihood of a target variable being drawn from the predicted distribution i.e. it performs Maximum Likelihood Estimation (MLE):

$$Loss(\theta) = \sum_{n=1}^{N} -\log(P(s_n; \mu_\theta, \Sigma_\theta)) \ . \tag{1}$$

By outputting a distribution over the next state our network can quantify *aleotoric* uncertainty. Ensembling multiple probabilistic DNNs, training each on different subsets of the data, and averaging over their predictions can quantify *epistemic* uncertainty (Lakshminarayanan, Pritzel, and Blundell 2017). Here, we employ K-many models, averaging the predictions to ensure both types of uncertainty are captured

$$\tilde{f}_\theta(s_t, a_t) = \frac{1}{K} \sum_{k=1}^{K} \tilde{f}_{\theta_k}(s_t, a_t) \ . \tag{2}$$

3.2 Control

Between interactions with the environment, our agent plans by combining Model Predictive Path Integral (MPPI) (Williams, Aldrich, and Theodorou 2015) with Trajectory Sampling (Chua et al. 2018). Here, we sample action sequences $a^u_{t:t+H-1} \ \forall \ u \in U$ from our policy π, an initially arbitrary multivariate Gaussian distribution with diagonal covariance and parameters $(\mu^0, \sigma^0)_{t:t+H} \in \mathbb{R}^a$. Then, we duplicate P-many state-action pairs at the first planning timestep $(s^p_t, a_t) \ \forall \ p \in P$ called *particles*. Each particle is assigned one bootstrap from the dynamics function ensemble and iteratively passed through it to create next-state distributions which can be sampled: $s^p_{t+1} \sim \mathcal{N}(\mu^p_{t+1}, \Sigma^p_{t+1}; \theta)$. We estimate the return G_Γ from each trajectory Γ by taking an expectation across particles

$$G_\Gamma = \mathbb{E}_P \left[\sum_{t=0}^{H} \mathcal{R}(s_t, a_t) \right] \ . \tag{3}$$

We select the top-e returns G^*_Γ, sometimes called the *elite* sequences, and update the parameters of π at iteration j using a G^*_Γ-normalised estimate:

$$\mu^j = \frac{\sum_{i=1}^{e} \Pi_i \Gamma^*_i}{\sum_{i=1}^{e} \Pi_i}, \ \sigma^j = \sqrt{\frac{\sum_{i=1}^{e} \Pi_i (\Gamma^*_i - \mu^j)^2}{\sum_{i=1}^{e} \Pi_i}} \ , \tag{4}$$

where $\Pi_i = e^{\tau(G^*_\Gamma, i)}$, τ is a temperature parameter that mediates weight given to the optimal trajectory, and Γ^*_i is the ith top-e trajectory corresponding to expected return G^*_Γ (Hansen, Wang, and Su 2022). After n optimisation iterations, μ^n represents the optimal action sequence, and the first action is taken in the real environment i.e. we perform receding horizon control.

3.3 System Identification

We grant the agent a window, or *commissioning period* C, to explore the state-action space and fit the one-step dynamics function $\tilde{f}_\theta(s_t, a_t)$. Our task is to sample the state-action space sequentially, and update the parameters θ of our model,

Algorithm 1: PEARL

Require: $\mathcal{D}$, $\tilde{f}(s_t, a_t)_\theta$: memory, dynamics model
μ^0, σ^0, N: π_{MPPI} params
$\mathbf{s}_t, \mathbf{a}_0$: current state, random init action
C, U, H: comm. steps, act seqs., horizon

for step $t = 0...T$ **do**
  for iteration $n = 1...N$ **do**
    for act. seq. $u = 1...U$ **do**
      $a^u_{t:t+H-1} \sim \pi$ ◁ *Sample actions*
      $s^p_{t+H} = \tilde{f}(s^p_t, a^i_{t:t+H-1})_\theta \; \forall \; p \in P$ ◁ *Plan*
      if $t \leq C$ **then**
        $V_\Gamma = Var_P \left[\sum_{t=0}^{H} \mathcal{R}(s_t, a_t)\right]$
        $\mu^{n+1}, \sigma^{n+1} \leftarrow \mu^n(V_\Gamma), \sigma^n(V_\Gamma)$ ◁ *Eq. 4*
        update $\tilde{f}(s_t, a_t)_\theta$ given $\mathcal{D}$ ◁ *Eq. 1*
      else
        $G_\Gamma = \mathbb{E}_P \left[\sum_{t=0}^{H} \mathcal{R}(s_t, a_t)\right]$
        $\mu^{n+1}, \sigma^{n+1} \leftarrow \mu^n(G_\Gamma), \sigma^n(G_\Gamma)$ ◁ *Eq. 4*
    $a_t \leftarrow \mu^N_t$
    $\mathcal{D} \leftarrow (s_t, a_t, r_t, s_{t+1})$
  if end of day **then**
    update $\tilde{f}(s_t, a_t)_\theta$ given $\mathcal{D}$ ◁ *Eq. 1*

such that we minimise the error in model predictions by the end of commissioning period C. Inspired by Bayesian Optimisation (Snoek, Larochelle, and Adams 2012), we leverage the predictive variance in our models to select trajectories that transition the agent to parts of the state-space where it is most uncertain, sometimes called Maximum Variance (MV) exploration (Jain et al. 2018). We adapt the routine from the previous section to evaluate the variance V_Γ in the predicted rewards across particle trajectories

$$V_\Gamma = Var_P \left[\sum_{t=0}^{H} \mathcal{R}(s_t, a_t)\right] \; . \tag{5}$$

Now the elite trajectories Γ^* in Equation (4) correspond to the action sequences and that maximise predictive variance V^*_Γ. We update the parameters of our model θ after each sample, thereafter we update θ at the end of each day.

3.4 Reward Function

The reward $\mathcal{R}(s_t, a_t)$ is a linear combination of an *emissions term* $r_E[t]$ and a *temperature term* $r_T[t]$ i.e. $\mathcal{R}(s_t, a_t) = r_E[t] + r_T[t]$. Our goal is to motivate the agent to minimise emissions whilst satisfying thermal comfort in the building. If $r_E[t]$ is an emissions-term reward at timestep t, $E[t]$ is the total energy consumption in the environment at time t, and $C[t]$ the grid carbon intensity at time t, then the emissions-term reward is

$$r_E[t] = -\phi \left(E[t]C[t]\right) \; , \tag{6}$$

where ϕ is a tunable parameter that sets the relative emphasis of emission-minimisation over thermal comfort (cf. Technical Appendix). The reward is negative because our goal is to minimise emissions, or maximise the negative of emissions produced. If $r^i_T[t]$ is a temperature-term reward at timestep t for thermal zone i, T^i_{obs} is the observed temperature in thermal zone i, and T_{low} and T_{high} are the lower and upper temperature bounds on thermal comfort respectively. The temperature reward is then given by

$$r^i_T[t] = \begin{cases} 0 : & T_{low} \leq T^i_{obs} \leq T_{high} \\ -\min[(T_{low} - T^i_{obs}[t])^2, \\ (T_{high} - T^i_{obs})^2] : & \text{otherwise} \; , \end{cases} \tag{7}$$

where the second term can be thought of as a *penalty* that punishes the agent in proportion to deviations from the thermal comfort zone. The total temperature reward $r_T[t]$ is obtained by summing the rewards across thermal zones i.e. $\sum_{i=1}^{N} r^i_T[t]$.

4 Experimental Setup

4.1 Environments

We evaluate the performance of our proposed approach using *Energym*, an open-source building simulation library for benchmarking smart-grid control algorithms (Scharnhorst et al. 2021) that offers more candidate buildings that any other open-source package. *Energym* provides a *Python* interface for ground-truth building simulations designed in *EnergyPlus* (Crawley et al. 2001), and presents buildings with varied equipment, geographies, and structural properties. We perform experiments in the following three buildings:

- ***Mixed-Use*** facility in Athens, Greece. 566.38m^2 surface area; 13 thermal zones; $\mathcal{A} \in \mathbb{R}^{12}$ and $\mathcal{S} \in \mathbb{R}^{37}$. Temperature setpoints and air handling unit (AHU) flowrates are controllable.
- ***Office*** block in Athens, Greece. 643.73m^2 surface area; 25 thermal zones; $\mathcal{A} \in \mathbb{R}^{14}$ and $\mathcal{S} \in \mathbb{R}^{56}$. Only temperature setpoints are controllable.
- ***Seminar Centre*** in Billund, Denmark. 1278.94m^2 surface area; 27 thermal zones; $\mathcal{A} \in \mathbb{R}^{18}$ and $\mathcal{S} \in \mathbb{R}^{59}$. Only temperature setpoints are controllable.

In all cases, environment states are represented by a combination of temperature, humidity and pressure sensors (among others). Full state and action spaces for each building are reported in the Technical Appendix for brevity. Experiments were run in each environment for one year starting on 01/01/2017, advancing in k-minute timestep increments, with k being environment dependent. Weather and grid carbon intensity match the true data in each geography for this period.

4.2 Baselines

We compare the performance of our agent against several strong RL baselines, and an RBC:

- **Soft Actor Critic** (SAC; (Haarnoja et al. 2018)), a state-of-the-art algorithm, known for lower variance performance than other popular model-free algorithms like PPO and DDPG.
- **Proximal Policy Optimisation** (PPO; (Schulman et al. 2017)), a popular model-free algorithm in production and

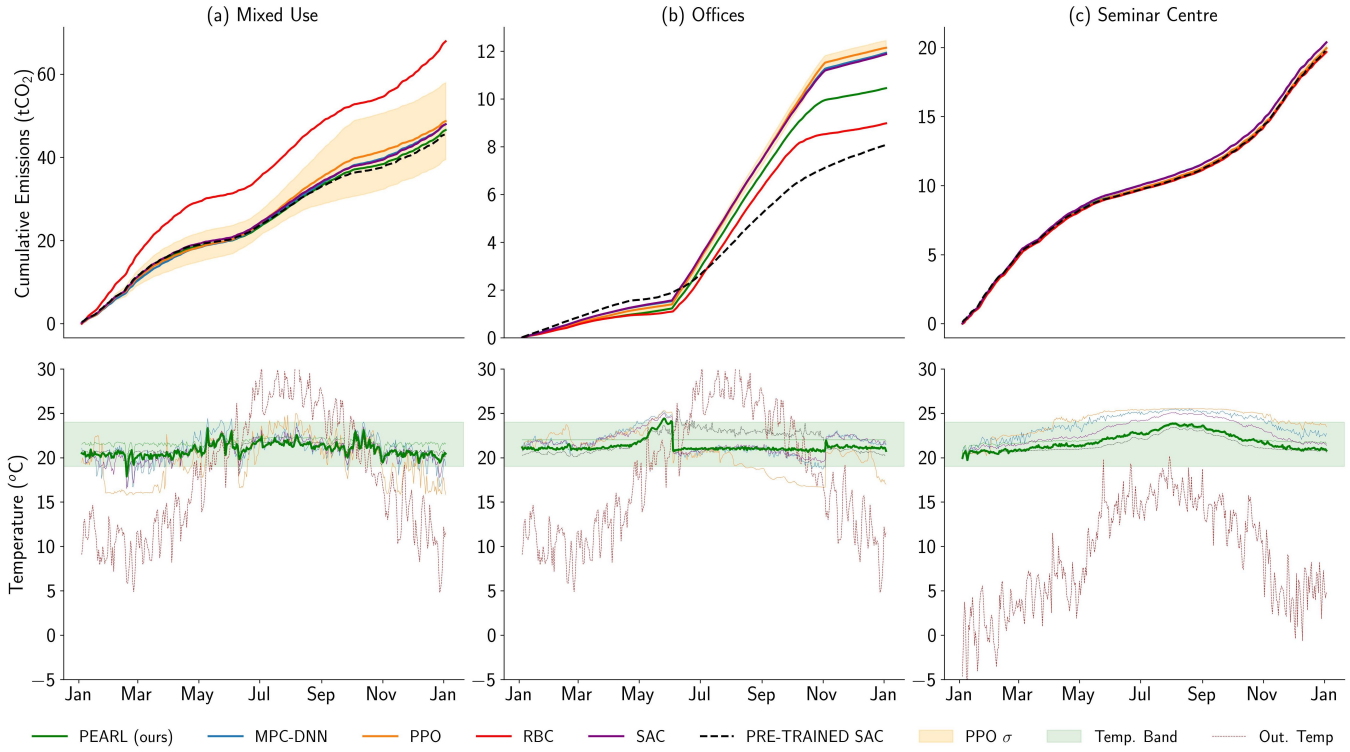

Figure 2: Energym Performance. Top: Cumulative emissions produced by all agents across the (a) *Mixed Use*, (b) *Offices*, and (c) *Seminar Centre* environments. Curves represent the mean of 3 runs of the experiment, shaded areas are one standard deviation (too small to see in all cases except PPO). Bottom: Mean daily building temperature produced by all agents, the green shaded area illustrates the target temperature range [19, 24].

	Agent	Emissions (tCO_2)	Temp. Infractions	Latency	Mean Reward
Mixed-Use	RBC	68.09 ± 0.00	$2.47\% \pm 0\%$	–	–
	PPO	48.80 ± 10.35	$48.49\% \pm 6.08\%$	$\mathbf{0.021} \pm 0.01$	$-1.47e7 \pm 2.84e6$
	SAC	48.80 ± 0.14	$\mathbf{0}\% \pm 0\%$	0.028 ± 0.02	$-6.02e5 \pm 3.85e4$
	MPC-DNN	48.03 ± 0.46	$13.42\% \pm 0.59\%$	0.030 ± 0.01	$-1.12e6 \pm 3.47e4$
	PEARL	$\mathbf{46.67 \pm 0.09}$	$0.55\% \pm 0.08\%$	0.870 ± 0.15	$\mathbf{-5.76e10^5} \pm 2.12e10^3$
	Oracle	45.49	0%	0.027	$-4.77e5$
Office	RBC	$\mathbf{9.61} \pm 0.00$	$1.64\% \pm 0\%$	–	–
	PPO	12.14 ± 0.31	$31.51\% \pm 7.19\%$	$\mathbf{0.018} \pm 0.006$	$-2.26e6 \pm 1.07e6$
	SAC	11.87 ± 0.01	$6.58\% \pm 1.12\%$	0.025 ± 0.01	$-2.75e5 \pm 2.03e4$
	MPC-DNN	11.93 ± 0.022	$9.86\% \pm 0.84\%$	0.029 ± 0.001	$-5.50e5 \pm 2.89e4$
	PEARL	10.45 ± 0.07	$\mathbf{1.52}\% \pm 0\%$	0.845 ± 0.14	$\mathbf{-5.51e10^4} \pm 1.82e10^3$
	Oracle	8.08	2.47%	0.023	$-2.63e4$
Sem. Centre	RBC	$\mathbf{19.74} \pm 0.00$	$\mathbf{0}\% \pm 0\%$	–	–
	PPO	20.01 ± 0.27	$51.23\% \pm 14.99\%$	$\mathbf{0.028} \pm 0.01$	$-4.15e6 \pm 9.67e5$
	SAC	20.37 ± 0.08	$29.32\% \pm 1.87\%$	0.033 ± 0.02	$-1.95e6 \pm 7.11e4$
	MPC-DNN	20.44 ± 0.02	$49.13\% \pm 0.56\%$	0.035 ± 0.001	$-2.45e6 \pm 3.26e4$
	PEARL	20.02 ± 0.10	$\mathbf{0}\% \pm 0\%$	0.911 ± 0.17	$\mathbf{-1.18e10^6} \pm 1.50e10^3$
	Oracle	19.75	0%	0.031	$-1.14e6$

Table 1: Energym Performance. Results for all agents across our three *Energym* environments. We define the temperature infraction metric as the percentage of days where mean building temperature falls outside the target range [19, 24], and latency as the mean compute time each agent requires to select an action given its policy measured in seconds per action. Results are averaged across 3 runs and presented as mean $\pm$ standard deviation, except for the Oracle which has converged on a policy prior to deployment with multiple runs showing the same performance.

used in previous works by Ding, Du, and Cerpa (2020) and Zhang et al. (2019a) as a baseline.

MPC with Deterministic Neural Networks (MPC-DNN; Nagabandi et al. (2018)), a simple, high-performing model-based architecture. Varying implementations have been used by previous authors, notably Ding, Du, and Cerpa (2020) and Zhang et al. (2019a). We use the original implementation by Nagabandi et al. (2018).

RBC, a generic, bang-bang controller found in most heating/cooling equipment that follows the heuristics outlined in the Technical Appendix.

Oracle, an SAC agent with hyperparameters fit to each environment using Bayesian Optimsation in Weights and Biases (Biewald 2020), and **pre-trained** in each building simulation for 10 years prior to test time.

We ensure both model-based agents plan with the same number of candidate actions over the same time horizon H to ensure performance variation is a consequence only of their differing design. For each agent we adopt the implementations from their original papers, except for the number of network layers and their dimensions which are set to 5 and 200 respectively to attempt to capture the full complexity of the system dynamics. Full hyperparameter specifications are provided in the Technical Appendix–PEARL's trajectory sampling and MPPI hyperparameters follow implementations by Chua et al. (2018) and Hansen, Wang, and Su (2022) respectively.

5 Results and Discussion

5.1 Performance

Table 1 reports key metrics for our six controllers across our *Energym* environments. Results are reported as the mean $\pm$ standard deviation for 3 runs of each experiment. The RBC performs identically across all experiments because both its policy and the environment are deterministic; varying RL agent performance is a consequence of policy and initialisation stochasticity.

Emissions. We find PEARL produces minimum emissions in the *Mixed-Use* environment, with cumulative emissions 31.46% lower than the RBC. In the *Office* block, the RBC exhibits lowest cumulative emissions, and PEARL outperforms all RL baselines. In the *Seminar Centre*, the RBC minimises emissions, and PPO marginally outperforms PEARL, but does so at the cost of erroneous temperature control. Low external temperatures in the *Seminar Centre* make experiments there less informative as the optimal policy is to heat most of the year, providing little room for improved control. Stepwise emission totals for each environment are illustrated in the top half of Figure 2. We provide an illustrative example of PEARL showing an ability to load shift in Figure 3.

Temperature. We find that PEARL produces minimum daily mean temperature infractions in the *Office* and *Seminar Centre* environments, and is slightly outperformed by SAC in the *Mixed-Use* environment. The RBC is comparably performant across all environments, as would be expected. The remaining RL baselines miss the thermal bounds regularly, with PPO and MPC-DNN exhibiting temperature infraction

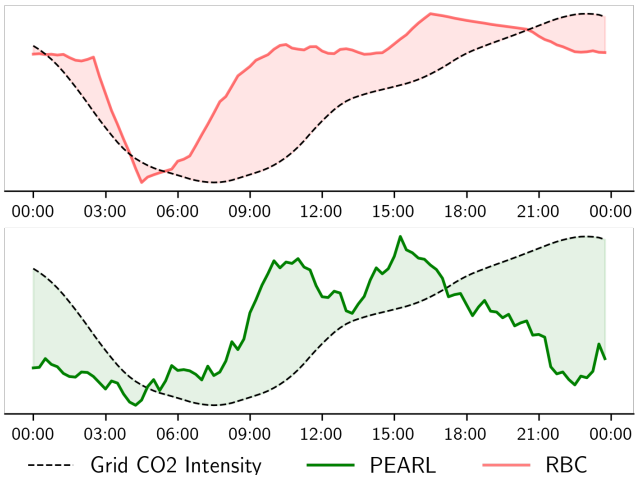

Figure 3: Load Shifting. Power consumption w.r.t. grid carbon intensity for the RBC (top) and PEARL (bottom) on an exemplar day in the *Office* environment. We wish to maximise the shaded area to minimise emissions. PEARL minimises power draw in the early morning and late evening when grid carbon intensity is highest.

rates as high as 51.23% and 49.13% respectively. In some cases the strong emissions performance of these baselines is a direct consequence of shutting off HVAC equipment and forcing uncomfortable internal temperatures. Mean daily building temperatures for each agent across the environments are plotted in the lower half of Figure 2.

Latency. PPO is the lowest-latency (mean compute time per action) controller, selecting actions in two thirds of the time required by MPC-DNN, and 41 times faster than PEARL on average, but we note the latency of all agents is far smaller than the sampling period of each environment, meaning all implementations would prove adequate for real-world deployment. Were they deployed in situ, the model-based agents could utilise the time between environment interactions fully to plan with greater numbers of action sequences which we expect would improve performance.

Reward. Mean annual reward captures an agent's ability to minimise emissions *and* maintain thermal comfort; this is the primary measure of agent performance. PEARL exhibits maximum mean reward across all environments suggesting it strikes this balance better than the other controllers. The reward curves for each agent are reported in the Technical Appendix for brevity.

Oracle comparison. The pre-trained oracle outperforms the baselines and PEARL in all cases as expected. However, its performance is surprisingly close to PEARL's, showing only 2.5% and 1.3% lower emissions in the *Seminar Centre* and *Mixed-Use* environments, and exhibiting similar thermal performance. From these results one could conclude that PEARL has produced a near-optimal policy, but one cannot be certain the oracle has reached optimality given the unusual, non-convergent reward curves this problem setting creates (cf. Technical Appendix). Indeed, the shape of the reward curve associated with an optimal policy is unclear, unlike

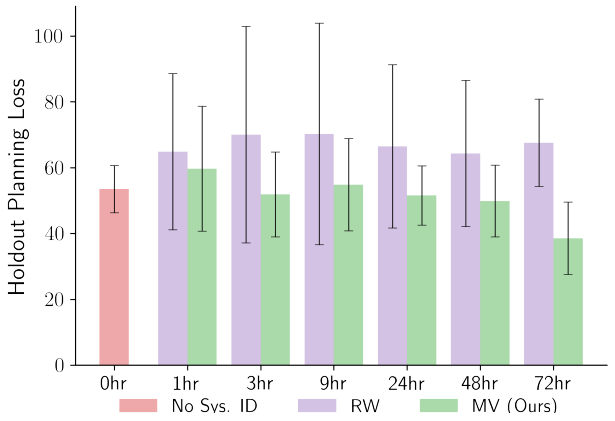

Figure 4: System ID. Planning MSE post-commissioning on a holdout set of 100 randomly sampled state-action trajectories, given varying system ID duration. Black bars represent one standard deviation across three runs.

the canonical episodic RL tasks (cartpole, mountain car etc.) where optimal solutions can be quantified by a fixed episodic return. Identifying optimality in this context is challenging and has been hitherto unexplored by the community.

Why is *Mixed-Use* performance an outlier? An important observation from Table 1 is that PEARL only outperforms the RBC in the *Mixed-Use* facility. Why would this be the case? The outcome of the *Seminar Centre* results has been discussed above, but one may expect PEARL to perform as well in the Office environment as it does in the *Mixed-Use* environment. Unlike the Office environment, in the *Mixed-Use* facility the agent has access to thermostat setpoints *and* continuous AHU flowrate control. This greatly increases action-space complexity and moves the control problem away from a setting where simple heuristics can be readily applied. This would suggest that RL building controllers should only be deployed when the action space is sufficiently complex, likely owing to some access to continuous control parameters.

5.2 System Identification

We test the sensitivity of PEARL's performance to system ID duration. We vary the commissioning period at seven intervals between 0 (no system ID) and 72 hours, and test predictive accuracy on a holdout set of 100 randomly sampled state-action trajectories produced by another controller in the same environment. We compare our method (MV) with Lazic et al. (2018)'s Random Walk (RW) and plot the results in Figure 4–see the Technical Appendix for Lazic et al. (2018)'s exploration policy.

We find that the accuracy of models fitted via MV system ID correlates with commissioning period length, as would be expected. We observe that, in expectation, models fit using data collected via Lazic et al. (2018)'s RW perform no better than randomly initialised networks. Using MV exploration, model accuracy is noticeably better than random only after 72 hours of system ID. However, we note from Figure 2, that an agent with 3 hours of system ID time remains capable of reducing emissions and maintaining thermal comfort despite poorer predictive accuracy.

	Deterministic	Probabilistic
Random Shooting (RS)	$-7.11e5$	$-7.30e5$
MPPI	$\mathbf{-5.28e5}$	$-5.65e5$
Oracle	$-4.77e5$	

Table 2: Agent Decomposition. Mean daily reward for four instantiations of PEARL varying the choice of network and planning algorithm. The *Oracle* is reported as a baseline. Experiments conducted in the Mixed-Use environment for one year.

5.3 Agent Decomposition

What components of PEARL enable performant control? We note two differences between PEARL and our model-based RL baseline MPC-DNN: 1) The use of probabilistic networks, rather than deterministic networks, and 2) The use of MPPI planning, rather than random shooting (RS). To understand their relative importance, we vary the design of PEARL to either include or exclude these components and compare performance–Table 2.

We find the performance of PEARL to be sensitive to the choice of planner and, to our surprise, insensitive to the choice of network, with agents composed of deterministic and probabilistic networks performing similarly when optimiser choice is controlled for. This is in contradiction to many works suggesting probabilistic modelling of dynamics function improves performance over deterministic models, particularly in complex, partially-observed state-action spaces like those exhibited in this study (Deisenroth and Rasmussen 2011; Chua et al. 2018; Levine 2018). Given changes to training time between networks are insignificant, we continue to endorse probabilistic dynamics functions despite this result, as they may improve performance in settings beyond those used in this study.

6 Conclusion

In this work we consider the task of learning policies *tabula rasa* that minimise emissions in buildings whilst ensuring thermal comfort, a considerably harder task than pre-training models in simulation before deployment. We have proposed PEARL (**P**robabilistic **E**mission-**A**bating **R**einforcement **L**earning), and shown it can reduce emissions from buildings by up to 31.46% when compared with an RBC, by fitting polices *online* without pre-training in simulation. When compared with existing RL baselines, our algorithm performs favourably, showing reduced emissions in all cases bar one, whilst maintaining thermal comfort more effectively. Our approach is simple to commission, requiring no historical data or simulator access *a priori*, and capable of generalising across varied building archetypes. The scaled deployment of such systems could prove effective climate change mitigation tools.

References

Biewald, L. 2020. Experiment Tracking with Weights and Biases. Software available from wandb.com.

Borrelli, F.; Bemporad, A.; and Morari, M. 2017. *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press.

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.

Chua, K.; Calandra, R.; McAllister, R.; and Levine, S. 2018. Deep Reinforcement Learning in a Handful of Trials Using Probabilistic Dynamics Models. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, 4759–4770.

Crawley, D. B.; Lawrie, L. K.; Winkelmann, F. C.; Buhl, W. F.; Huang, Y. J.; Pedersen, C. O.; Strand, R. K.; Liesen, R. J.; Fisher, D. E.; Witte, M. J.; et al. 2001. EnergyPlus: creating a new-generation building energy simulation program. *Energy and buildings*, 33(4): 319–331.

Cullen, J.; and Allwood, J. 2010. The efficient use of energy: Tracing the global flow of energy from fuel to service. *Energy Policy*, 38(1): 75–81.

Deisenroth, M.; and Rasmussen, C. E. 2011. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, 465–472.

Ding, X.; Du, W.; and Cerpa, A. E. 2020. MB2C: Model-Based Deep Reinforcement Learning for Multi-zone Building Control. In *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, 50–59.

Gal, Y.; McAllister, R.; and Rasmussen, C. E. 2016. Improving PILCO with Bayesian neural network dynamics models. In *Data-Efficient Machine Learning workshop, ICML*, volume 4, 25.

Gu, S.; Holly, E.; Lillicrap, T.; and Levine, S. 2017. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE international conference on robotics and automation (ICRA)*, 3389–3396. IEEE.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, 1861–1870.

Hansen, N.; Wang, X.; and Su, H. 2022. Temporal Difference Learning for Model Predictive Control. *arXiv 2203.04955*.

Hensman, J.; Fusi, N.; and Lawrence, N. D. 2013. Gaussian Processes for Big Data. In *Uncertainty in Artificial Intelligence*, 282. Citeseer.

Higuera, J. C. G.; Meger, D.; and Dudek, G. 2018. Synthesizing Neural Network Controllers with Probabilistic Model based Reinforcement Learning. *CoRR*, abs/1803.02291.

Jain, A.; Nghiem, T.; Morari, M.; and Mangharam, R. 2018. Learning and Control Using Gaussian Processes. In *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, 140–149.

Jain, A.; Smarra, F.; Reticcioli, E.; D'Innocenzo, A.; and Morari, M. 2020. NeurOpt: Neural network based optimization for building energy management and climate control. In *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, volume 120 of *Proceedings of Machine Learning Research*, 445–454. The Cloud: PMLR.

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2): 99–134.

Lakshminarayanan, B.; Pritzel, A.; and Blundell, C. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30.

Lazic, N.; Lu, T.; Boutilier, C.; Ryu, M.; Wong, E. J.; Roy, B.; and Imwalle, G. 2018. Data Center Cooling using Model-predictive Control. In *Proceedings of the Thirty-second Conference on Neural Information Processing Systems (NeurIPS-18)*, 3818–3827. Montreal, QC.

Levine, S. 2018. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2016. Continuous control with deep reinforcement learning. In *ICLR (Poster)*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. A. 2013. Playing Atari with Deep Reinforcement Learning. *CoRR*, abs/1312.5602.

Nagabandi, A.; Kahn, G.; Fearing, R. S.; and Levine, S. 2018. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 7559–7566. IEEE.

Scharnhorst, P.; Schubnel, B.; Fernández Bandera, C.; Salom, J.; Taddeo, P.; Boegli, M.; Gorecki, T.; Stauffer, Y.; Peppas, A.; and Politi, C. 2021. Energym: A Building Model Library for Controller Benchmarking. *Applied Sciences*, 11(8): 3518.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Silver, D.; Schrittwieser, J.; Simonyan, K.; Antonoglou, I.; Huang, A.; Guez, A.; Hubert, T.; Baker, L.; Lai, M.; Bolton, A.; et al. 2017. Mastering the game of go without human knowledge. *nature*, 550(7676): 354–359.

Snoek, J.; Larochelle, H.; and Adams, R. P. 2012. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25.

Socher, R.; Ganjoo, M.; Manning, C. D.; and Ng, A. 2013. Zero-shot learning through cross-modal transfer. *Advances in neural information processing systems*, 26.

Sutton, R.; and Barto, A. 2018. *Reinforcement Learning: An Introduction*. The MIT Press, second edition.

Valladares, W.; Galindo, M.; Gutiérrez, J.; Wu, W.-C.; Liao, K.-K.; Liao, J.-C.; Lu, K.-C.; and Wang, C.-C. 2019. Energy

optimization associated with thermal comfort and indoor air control via a deep reinforcement learning algorithm. *Building and Environment*, 155: 105 – 117.

Wei, T.; Wang, Y.; and Zhu, Q. 2017. Deep Reinforcement Learning for Building HVAC Control. In *Proceedings of the 54th Annual Design Automation Conference 2017*, DAC '17. New York, NY, USA: Association for Computing Machinery. ISBN 9781450349277.

Williams, C. K.; and Rasmussen, C. E. 2006. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA.

Williams, G.; Aldrich, A.; and Theodorou, E. 2015. Model predictive path integral control using covariance variable importance sampling. *arXiv preprint arXiv:1509.01149*.

Yu, L.; Qin, S.; Zhang, M.; Shen, C.; Jiang, T.; and Guan, X. 2021. A Review of Deep Reinforcement Learning for Smart Building Energy Management. *IEEE Internet of Things Journal*, 8(15): 12046–12063.

Zhang, C.; Kuppannagari, S. R.; Kannan, R.; and Prasanna, V. K. 2019a. Building HVAC scheduling using reinforcement learning via neural network based model approximation. In *Proceedings of the 6th ACM international conference on systems for energy-efficient buildings, cities, and transportation*, 287–296.

Zhang, Z.; Chong, A.; Pan, Y.; Zhang, C.; and Lam, K. P. 2019b. Whole building energy model for HVAC optimal control: A practical framework based on deep reinforcement learning. *Energy and Buildings*, 199: 472–490.