ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240757

Winning the 2023 CityLearn Challenge: A Community-Based Hierarchical Energy Systems Coordination Algorithm

Andoni I. Garmendia[a,*], Francesco Morri[b,], Quentin Cappart[c] and Hélène Le Cadre[b]**

[a]University of the Basque Country UPV/EHU, Donostia, Spain
[b]Inria, Univ. Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL, F59000, Lille, France
[c]Polytechnique Montréal, Montreal, Canada

Abstract. The effective management and control of building energy systems are crucial for reducing the energy consumption peak loads, CO_2 emissions, and ensuring the stability of the power grid, while maintaining optimal comfort levels within buildings. The difficulty to accommodate this trade-off is amplified by dynamic environmental conditions and the need for scalable solutions that can adapt across various building types and geographic locations. Acknowledging the importance of this problem, NeurIPS conference hosted since 2020 the *CityLearn control challenge* to foster the design of innovative solutions in building energy management. Participants were tasked with developing strategies that not only enhance energy efficiency but also prioritize sustainability and occupant comfort. This paper introduces the *Community-based Hierarchical Energy Systems Coordination Algorithm* (CHESCA), the winning approach of the 2023 edition. We rely on a hierarchical approach adaptable to an arbitrary number of buildings, first optimizing building-level metrics individually, and later refining these through a central community-level controller to improve grid-related metrics. Compared to the other high-ranked competitors, our approach demonstrated fast inference capabilities like learning-based methods, while offering a better interpretability and a superior generalization capabilities with minimal data requirements. This paper details our approach, supported by comprehensive experimental results and ablation studies.

1 Introduction

Urban energy systems are becoming increasingly complex, necessitating innovative strategies for the efficient management of their energy consumption. Modern buildings, key components of the urban system, significantly influence the energy consumption patterns and grid stability, due to the integration of renewable energy sources, storage systems, electric vehicles, etc. These require solving dynamically complex problems, which are essential for reducing grid strain and improving sustainability.

As shown in [9, 18], the emergence of smart grids allowed for a flexible control of the energy demand in buildings. A review of the hardwares that have been used during the last 40 years to develop more efficient controllers is provided in [18], thus paving the way towards more sophisticated control algorithms. In particular, [9] highlights that the traditional methods consisting of heuristics and heavy numerical calculations, are starting to be replaced by Machine Learning techniques.

The *NeurIPS CityLearn Challenge*, launched in 2020 [22], addresses these multi-objective challenges by encouraging the development of algorithms leveraging modern Machine Learning techniques. These algorithms aim to optimize the control of building-based technologies, which in turn helps to lower operational costs, reduce CO_2 emissions, and decrease the frequency of blackouts. The 2023 edition of this competition [2] attracted over 600 participants and received more than 2500 submissions across two main tracks: i) the *forecast track*, focused on developing models to predict various variables; and ii) the *control track*, aimed at coordinating multiple buildings to optimize a set of criteria simultaneously.

One of the main challenges in the competition was the lack of abundant historical data, which required competitors to design algorithms using a small training dataset. As a direct consequence, most of the submissions were prone to overfitting, resulting in suboptimal performance when applied to unseen scenarios. We participated in the control track of the competition and emerged as winners by employing an effective strategy that directly addressed this data scarcity.

This paper presents CHESCA (*Community-based Hierarchical Energy Systems Coordination Algorithm*), the algorithm which won the control track of the competition. Designed to offer a broadly applicable solution, CHESCA stands out from many competitors' efforts that encountered issues with model overfitting. Our approach integrates heuristics, traditional control systems, and Machine Learning models to improve its generalization ability across various scenarios. More precisely, the contributions of the paper are as follows:

1. We introduce CHESCA, a generic and *feedback-loop* framework adaptable to multi-building operational scenarios and requiring few data to generalize properly;
2. We instantiate CHESCA relying on the CityLearn environment, allowing our team to win the NeurIPS 2023 edition of the competition (*control track*);
3. We propose a comprehensive evaluation against the 2 highest-ranked competitors and highlight the superior generalization capabilities of CHESCA;
4. We discuss the influence of different components within our framework through an ablation study.

* Corresponding Author. Email: andoni.irazusta@ehu.eus
** Corresponding Author. Email: francesco.morri@inria.fr

The rest of this paper is structured as follows: in Section 2, we present the relevant literature related to this problem; in Section 3, we introduce the CityLearn challenge environment. In Section 4, we describe our proposed algorithm, while Section 5 details the results of the challenge and the subsequent tests we ran with the algorithm. Finally, Section 6 provides a discussion about the approach and future works. Section 7 concludes the paper.

2 Related Literature

The existing literature on energy control within buildings can be categorized into three primary groups: (1) *rule-based control* (RBC) methods, (2) *model predictive control* (MPC) methods, and (3) *learning-based approaches*. For an extensive survey on classical control methods (i.e., MPC and RBC) we refer to the survey of Mariano-Hernández et al. [11]. For learning-based approaches, we refer to the survey of Vamvakas et al. [20].

2.1 Classical Control Techniques

RBC methods typically utilize predefined rules derived from best practices or heuristic data to manage technologies within buildings, like *heating, ventilation, and air conditioning* (HVAC). These rules are generally straightforward to implement, making RBC a simple, yet popular benchmark against more complex techniques.

On the other hand, methods based on a MPC involve the use of a dynamic model of the system to predict and optimize future behavior. These methods often require detailed models of the building's thermal dynamics to effectively schedule tasks like temperature set point adjustments in HVAC systems.

Salakij et al. [16] describe a physical model to simulate heat and moisture transfer in buildings. A simplified version of this model was then used to enhance HVAC scheduling, comparing it against traditional RBC methods. Rocha et al. [14] propose a two-level optimization framework where temperature control is the subordinate problem and energy optimization is the primary concern. This approach not only aims to meet demand but also explores the potential of selling energy surplus to the grid. The work by Ruusu et al. [15] targets the development of a computationally efficient strategy for energy management through the use of linear programming to handle nonlinear constraints. Finally, Biyik and Kahraman [4] seek to leverage interdependencies among different zones within the building to minimize energy use by means of a constrained optimization problem. An interesting common thread between these works is that most of them consider the control of a single building at a time, possibly because a grid-aware model allowing energy exchanges among buildings, is computationally expensive and would require to deal with the nonconvexities of the power-flow equations.

2.2 Learning-based Approaches

Learning-based approaches can model complex and dynamic environments like building energy management, and may provide faster responses compared to MPC [9]. Most applications of learning-based methods to the control of energy systems rely on Reinforcement Learning (RL) algorithms, due to its capability to operate without labeled data and the potential for training agents in simulated environments. We report below a non-exhaustive list of some RL algorithms used to tackle similar problems.

One of them is *Q-Learning* [23], a method tailored for environments with discrete states and actions. Chen et al. [6] use *Q*-Learning to manage the operation of air conditioning and window openings in buildings. This model simplifies the control to binary decisions, which, while restrictive, allows for direct comparisons with heuristics in simulation. Wei et al. [24] propose an extension relying on *Deep Q-Learning* (DQN) [12], which accommodates continuous state spaces but still relies on discrete actions.

Further developments of RL are demonstrated with more sophisticated algorithms such as *Deep Deterministic Policy Gradient* (DDPG) [10] and *Proximal Policy Optimization* (PPO) [17]. Du et al. [7] conducted a comparative analysis of DDPG, DQN, and heuristics-based controls for temperature regulation in a dual-zone residential building, obtaining a reduction in operational costs and improved comfort and an algorithm able to generalize effectively to new, unseen building environments. Azuatalam et al. [3] expanded the scope of reinforcement learning to include demand-response scenarios within an energy market, managing HVAC operations along with energy transactions between buildings.

However, RL encounters several practical difficulties: these methods require long training on simulated environments built with large amount of real-world data, which may be hard to obtain; furthermore, they usually lack interpretability and generalization capacities, struggling to adapt to unseen data.

2.3 Distinctive Aspects of our Contribution

Distinct from these works, our approach `CHESCA` melds the best attributes of both classical and learning-based methodologies. It achieves a good level of interpretability and the capacity to generalize effectively even with minimal data, overcoming the inherent limitations associated with RL methods. Importantly, our method does not depend on intricate mathematical models, which facilitates swift deployment in online control settings. Additionally, `CHESCA` is scalable allowing to consider a wide number of buildings.

3 CityLearn Challenge Environment

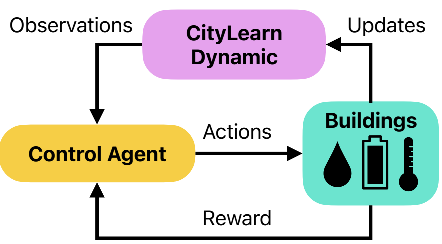

Figure 1. Overview of the CityLearn control environment.

The *CityLearn environment* [21] is an open-source environment created by *Farama Foundation* to develop classical and learning-based control algorithms in the context of building energy management. This environment allows the end-user to simulate an arbitrary number of buildings interconnected through the same power grid. We use the term *community* to refer to a group of buildings located in a common geographic area, producing their own electricity to meet their own demand and sharing the surplus among the community.

Figure 1 presents a schematic representation of the CityLearn environment for the control track. Briefly, the goal is to design control algorithms that optimize the energy utilization across a set $\mathcal{B}$ of buildings during a period of T time-steps. A time-step is typically 1 hour long. The agents' decision-making process relies on such algorithms. The environment provides observations $\boldsymbol{o} \in \mathcal{O}$ at each time step, that each agent uses to update its decisions $\boldsymbol{a} \in \mathcal{A}$ to control different aspects of the building. Executing an action results in an *update* of the building environment. At each time step, the agent receives a feedback by means of a *reward* interpreted as a signal from the buildings. We also note the potential for *power outages*, which must be managed appropriately.

3.1 Observations and Actions

The observations in $\mathcal{O}$ encompass all relevant and realistic information necessary for controlling the buildings. These include *external parameters* such as ambient temperature and solar irradiance, *internal parameters* like each building's internal temperature and the state of battery charge, and *various demands* specific to each building, for example, hot-water demand. A comprehensive list of all observations used is provided in the supplementary material[1].

At each time step, three actions must be carried out by the agent for each building: (1) controlling the HVAC system, (2) the (dis)charge of the battery and, (3) the (dis)charge of hot water storage. Let $\boldsymbol{a} := \langle \boldsymbol{a}^\theta, \boldsymbol{a}^e, \boldsymbol{a}^w \rangle \in \mathcal{A}^{|\mathcal{B}|}$ be a vector, where $\boldsymbol{a}^\theta := (a^\theta[b])_b$, $\boldsymbol{a}^e := (a^e[b])_b$, $\boldsymbol{a}^w := (a^w[b])_b$, and $a^\theta[b] \in [0,1], a^e[b] \in [-1,1]$ and $a^w[b] \in [-1,1]$ represent the normalized power given to the HVAC system, the charge of the electric battery and the charge of the water storage resp. in each building $b \in \mathcal{B}$. Besides, the specific (dis)charge amount of energy (resp. hot water) allowed is constrained by the state-of-charge of the battery (resp. water storage). Such a restriction is embeddeded into the dynamics of the environment.

3.2 Agents' Objective Functions

Consider an agent responsible for coordinating a set of $\mathcal{B}$ buildings, all connected to the same micro-grid. The agent's primary objective is to minimize a predefined cost function, $C : \mathcal{O} \times \mathcal{A} \to \mathbb{R}$, over the course of the simulation. This cost function is designed to comprehensively reflect various aspects of the agent's performance through four distinct components: (1) a *comfort cost*, (2) an *emission cost*, (3) a *grid cost* and, (4) a *resilience cost*. The cost function to minimize is formalized in Equation (1), where $f_c(\cdot)$, $f_e(\cdot)$, $f_g(\cdot)$, $f_r(\cdot)$ are real-valued functions capturing criteria which compose the cost function with weights w_c, w_e, w_g, w_r:

$$\begin{aligned} C(\boldsymbol{o}, \boldsymbol{a}) = & w_c f_c(\boldsymbol{o}, \boldsymbol{a}) + w_e f_e(\boldsymbol{o}, \boldsymbol{a}) + \\ & + w_g f_g(\boldsymbol{o}, \boldsymbol{a}) + w_r f_r(\boldsymbol{o}, \boldsymbol{a}). \end{aligned} \tag{1}$$

$C(\cdot)$ is a weighted linear combination of 4 criteria described below:

Comfort Cost $f_c(\cdot)$. This component quantifies the proportion of time-steps during which the indoor temperature strays from the established comfort range while the building is occupied.

Emission Cost $f_e(\cdot)$. This metric calculates the total CO_2 emissions produced by the buildings.

Grid Cost $f_g(\cdot)$. Differing from other metrics that are calculated individually for each building, this criterion evaluates the performance of the power grid as a whole, encompassing all connected buildings. It is composed of four equally weighted sub-components.

1. *Ramping r.* This component evaluates the smoothness of the community energy consumption. Formally, let E_t be the total energy consumption for the community at time-step t and T the length of the simulation, then we have $r := \sum_{t=1}^{T} |E_t - E_{t-1}|$.
2. *Load Factor l.* This component calculates the ratio of average daily consumption to the daily peak load. Formally, we have

$$l := \frac{1}{D} \left(\sum_{d=1}^{D} 1 - \frac{\frac{1}{h}\bar{E}_d}{\widehat{E}_d} \right),$$

where D is the number of days in the simulation, h is the number of hours in a day, $\bar{E}_d$ is the average consumption for day d, and $\widehat{E}_d$ is the maximum consumption for day d. Finally, l is bounded between 0 (perfect efficiency) and 1 (worst case inefficiency).
3. *Daily Peak.* This value is the highest daily electricity usage.
4. *All-time Peak.* This information captures the maximum power usage at any point during the entire simulation.

Resilience Cost $f_r(\cdot)$. This metric evaluates the system's ability to sustain operational effectiveness during power interruptions. It includes the following two sub-components:

1. *Resilience.* This metric measures the temperature comfort levels maintained during power outages.
2. *Unserved Energy.* This metric quantifies the unmet energy demand, including non-shiftable loads and essential services such as hot water and cooling, during power outages.

Finally, each term in Equation (1) is normalized based on baseline values derived from control-free simulation runs. The total cost $C(\cdot)$ is calculated at the end of the simulations and serves as the basis for the final rankings in the contest. Comprehensive details on the formulation of each function can be found on the official challenge page [1].

3.3 Dataset Description

The challenge utilizes the open-source *end-use load profiles for the U.S. building stock* dataset [25] to model a community of six single-family buildings in an undisclosed location within the U.S. This dataset provides hourly data for a four-months period, covering all the observations previously described. To accurately simulate environmental dynamics, each building is initialized with specific data, including weather conditions, building characteristics, and demand profiles. Observations are continuously updated throughout the simulation, reflecting both the dataset inputs (e.g., demand levels) and the outcomes of actions taken by the controller. Three distinct datasets are employed.

Training Set. This dataset encompasses one month of data for three buildings, and has been specifically prepared for participants to develop and refine their algorithms. It includes hourly recordings, yielding a total of $30 \times 24 \times 3 = 2160$ data points.

Public Evaluation Set. This dataset covers three months of data for the same three buildings and is used to calculate the online public leaderboard. This leaderboard can be used by participants to asses their performance, while not having direct access to the data. The dataset includes $3 \times 30 \times 24 \times 3 = 6480$ data points.

Private Evaluation Set. This dataset incorporates data from the same three buildings featured in the public dataset, along with three additional buildings, covering the last three months of the simulation. It is employed to set the final ranking and to determine the winner of

[1] The supplementary material and source code is available at https://github.com/TheLeprechaun25/CHESCA [19]

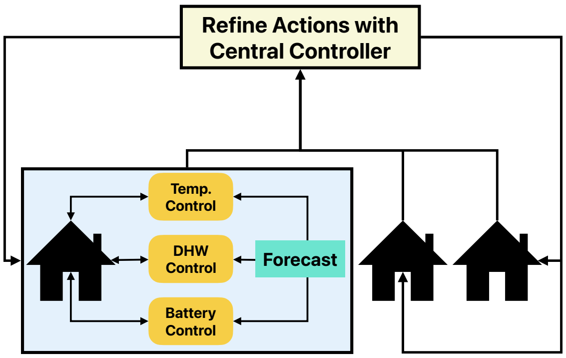

Figure 2. Overview of the structure of the algorithm: in yellow we depict the sub-controllers for *temperature*, *DHW* and *battery* control. In green we represent the forecast module. The upper module represents the central controller, highlighting with the arrows the feedback process.

the competition. The access to both the data and the results were restricted until the winner announcement. It encompasses $3 \times 30 \times 24 \times 6 = 12960$ data points, a quantity significantly larger than that available in the training dataset.

4 Optimizing Energy Management with CHESCA

This section describes the operational framework of CHESCA, designed to optimize energy management in a multi-building environment across discrete time steps. The algorithm operates by processing real-time observations along with forecast data to dynamically adjust control actions for temperature (i.e., via cooling or heating systems), domestic hot water (DHW), and battery usage. Control actions are devised at two distinct levels: *locally* and *globally*.

First, the *building level* operates locally. Within this scope, each building is managed by an individual controller tasked with optimizing building-specific costs. Besides, each building-level controller is provided with a forecast $\langle \widehat{\boldsymbol{y}}_t, \ldots, \widehat{\boldsymbol{y}}_{t+\tau} \rangle$, giving a prediction of future values of some variables for the next τ time-steps, starting from step t. The motivation is to help the controller to take decisions suitable for a longer term. The goal of each building-level controller is to take the actions $\boldsymbol{a}_t$ minimizing the cost $\widehat{C}(\cdot)$ at each time step t, from the current observations $\boldsymbol{o}_t$, the executed action, and the prediction $\langle \widehat{\boldsymbol{y}}_t, \ldots, \widehat{\boldsymbol{y}}_{t+\tau} \rangle$. The cost function $\widehat{C}(\cdot)$ is an approximation of cost $C(\cdot)$ described in Equation (1), obtained by using the forecast variables. The problem is formally expressed in Equation (2):

$$\boldsymbol{a}_t[b] := \operatorname*{argmin}_{\boldsymbol{a} \in \mathcal{A}} \widehat{C}(\boldsymbol{o}_t, \boldsymbol{a}, \langle \widehat{\boldsymbol{y}}_t, \ldots, \widehat{\boldsymbol{y}}_{t+\tau} \rangle), \quad \forall\, b \in \mathcal{B}. \tag{2}$$

Second, the *community level* operates globally. Here, a central controller evaluates the actions proposed by the individual controllers and perform modifications to improve the grid efficiency and its stability. Its goal is mainly to minimize the grid cost $\widehat{f}_g(.)$ derived by the function presented in Section 3.2. However, this cost directly depends on the actions carried out by the building-level controller and also on the predictions. Formally, the cost to be minimized by the central controller is given in Equation (3):

$$\boldsymbol{a}_t := \operatorname*{argmin}_{\boldsymbol{a} \in \mathcal{A}} \widehat{f}_g(\boldsymbol{o}_t, \boldsymbol{a}, \langle \widehat{\boldsymbol{y}}_t, \ldots, \widehat{\boldsymbol{y}}_{t+\tau} \rangle). \tag{3}$$

To take into account this hierarchical aspect, our method employs an *iterative feedback loop* where the community-level controller provides feedback (the proposed actions) to the building-level controllers for further refinement. This iterative process ensures a balance between the individual needs of buildings and the overall grid efficiency of the community, and provides better predictions that consider the future actions. This process continues until a predefined stopping criterion is satisfied, such as a fixed number of iterations or when the modifications have values falling below a specific threshold. A high-level representation of CHESCA architecture is provided in Figure 2 for a community of three buildings.

4.1 Iterative Feedback Loop Algorithm

CHESCA pseudo code is formalized in Algorithm 1. Let T be the length of the simulation. At each time step t, CHESCA receives a set of observations $\boldsymbol{o}_t$ from the environment and initializes action proposals $\boldsymbol{a}_t$. The initial step involves forecasting the future values of certain variables ($\widehat{\boldsymbol{y}}_t$) τ steps ahead. Control at the building level is conducted through three specialized sub-controllers, each dedicated to manage specific action variables. We define each subcomponent in the following subsections, but at a high-level: the ControlTemperature(.) function is a *proportional-integral-derivative* (PID) controller used for temperature regulation, the ControlDHW(.) function is an hourly-based heuristic used for hot-water control, and the OptimizeBatteryUsage(.) function uses the predictions to optimize the battery control by means of a tree search.

Algorithm 1 Pseudo-code of CHESCA.

for each time step $t \in \{1, \ldots, T\}$ **do**
  input: Observation vector $\boldsymbol{o}_t$, Set of action initializations $\boldsymbol{a}_t$
  while Stopping Criteria Not Met **do**
    predict: Update forecast variables $\widehat{\boldsymbol{y}}_t$
    for each building b in the set of buildings $\mathcal{B}$ **do**
      $a_t^{\theta}[b] \leftarrow \textsc{ControlTemperature}(\boldsymbol{o}_t, \widehat{\boldsymbol{y}}_t, \boldsymbol{a}_t)$
      $a_t^{w}[b] \leftarrow \textsc{ControlDHW}(\boldsymbol{o}_t, \widehat{\boldsymbol{y}}_t, \boldsymbol{a}_t)$
      $a_t^{e}[b] \leftarrow \textsc{OptimizeBatteryUsage}(\boldsymbol{o}_t, \widehat{\boldsymbol{y}}_t, \boldsymbol{a}_t)$
    end for
    $\boldsymbol{a}_t \leftarrow \textsc{Aggregate}(a_t^{\theta}[b], a_t^{w}[b], a_t^{e}[b]\ \forall\, b \in \mathcal{B})$
    $\boldsymbol{a}_t \leftarrow \textsc{CentralController}(\boldsymbol{o}_t, \widehat{\boldsymbol{y}}_t, \boldsymbol{a}_t)$
  end while
  output: $\boldsymbol{a}_t$
end for

As previously mentioned, the decisions of individual sub-controllers for each variable and each building are aggregated (Aggregate(.)), and shared with a central controller (CentralController(.)), that aims to refine the individual decisions to optimize the metrics for the entire community. The refined actions are used again to update those predicted variables $\widehat{\boldsymbol{y}}_t$ that depend on future consumption estimates.

The final output of the algorithm is the vector of refined actions $\boldsymbol{a}_t$ once the iterative refinement process is finished. The output actions are passed to the environment to proceed with the updates of the buildings. In scenarios where a power outage is detected, the algorithm adapts by conserving more energy and prioritizing essential demands. In that case, we separately define an alternative control pipeline. The numerical values for all the parameters used to configure the controllers are described in the supplementary material.

4.2 Forecasting

The variables we forecast include: (1) the outdoor temperature, $T_{out} \in [0, 50]$ [$°C$], (2) the solar generation, $S \geq 0$, [*W/kW*], (3) the hot water demand, $H \geq 0$, [*kWh*], and (4) the non-shiftable load, $N \geq 0$, [*kWh*]. Forecasting is performed by an ensemble approach that combines the outputs of three distinct models:

(1) Pre-trained XGBoost $\widehat{\boldsymbol{y}}_t^{\mathtt{PX}}$. An XGBoost regression model [5] is trained offline with the training-set data and remains fixed during the competition evaluation.

(2) Online XGBoost $\widehat{\boldsymbol{y}}_t^{\mathtt{OX}}$. An XGBoost regression model [5] is dynamically trained online using data acquired during the evaluation period, with retraining occurring every $T_{retrain}$ discrete steps.

(3) Online Historical Aggregation $\widehat{\boldsymbol{y}}_t^{\mathtt{H}}$. This is an aggregation (average value) of the predicted variables in previous days at the corresponding hour. The previous-hour value is taken when there is not any historical data.

The performance of each model $i \in \{\mathtt{PX}, \mathtt{OX}, \mathtt{H}\}$, measured as the absolute error ($e_t^i := \|\widehat{\boldsymbol{y}}_t^i - \boldsymbol{y}_t^i\|$), varies depending on the stage of the process. For instance, online XGBoost is expected to under-perform initially due to the limited availability of training data. To mitigate this, a dynamically adjusted weighted average of the three models' outputs is employed:

$$\widehat{\boldsymbol{y}}_t = w_t^{\mathtt{PX}} \widehat{\boldsymbol{y}}_t^{\mathtt{PX}} + w_t^{\mathtt{OX}} \widehat{\boldsymbol{y}}_t^{\mathtt{OX}} + w_t^{\mathtt{H}} \widehat{\boldsymbol{y}}_t^{\mathtt{H}}, \tag{4}$$

where w_t^i denotes the dynamically adapted weight for model i at time t, calculated based on the model's recent performance as follows:

$$w_t^i = \frac{\sum_{j=t-T}^{t-1} (e_j^i)^{-1}}{\sum_{k=1}^{3} \sum_{j=t-T}^{t-1} (e_j^k)^{-1}}. \tag{5}$$

In this formulation, e_t^i denotes the absolute error of model i at time step t, where a lower error contributes to a higher weighting. This adaptive weighting mechanism ensures that each model's influence on the overall forecast is proportional to its recent accuracy. Initially, weights can be set uniformly or based on prior knowledge to influence early predictions until adequate real-time data is available.

4.3 Temperature Control

In order to maintain the indoor temperature inside the comfort zone (i.e., the temperature setpoint decided by the occupants), we make use of a classical PID controller. The PID operates by constantly determining the difference between the target or set-point temperature ($T_t^{sp}[b]$) and the current temperature ($T_t[b]$), using this error value to fine-tune the cooling output. Specifically, the output $a_t^\theta[b]$ of the PID controller obeys the discrete Equation 6, where $a_t^\theta[b]$ is the power given to the cooler/heater at instant t, $e_t[b] = T_t[b] - T_t^{sp}[b]$ is the error signal between the current indoor temperature and the set-point, and k_p, k_i, k_d are the proportional, integral and derivative gains respectively, which need to be fine-tuned.

$$a_t^\theta[b] = k_p e_t[b] + k_i \sum_{i=0}^{t} e_i[b] + k_d \frac{e_t[b] - e_{t-1}[b]}{dt}. \tag{6}$$

In order to fine-tune k_p, k_i and k_d, we employed *Bayesian optimization*, an approach for optimizing complex functions that are expensive to evaluate [8]. Bayesian optimization works by building a probabilistic model of the function and then uses that model to make guesses about the best possible values for the parameters.

In our application, we aim to achieve a balance between maintaining a comfortable environment and ensuring energy efficiency. Therefore, Bayesian optimization was used in a multi-objective setting, where the objectives were to minimize discomfort and reduce cooler's electricity consumption. Evaluation was conducted on three buildings from the training dataset, where the objectives of minimizing discomfort and energy use were somewhat conflicting.

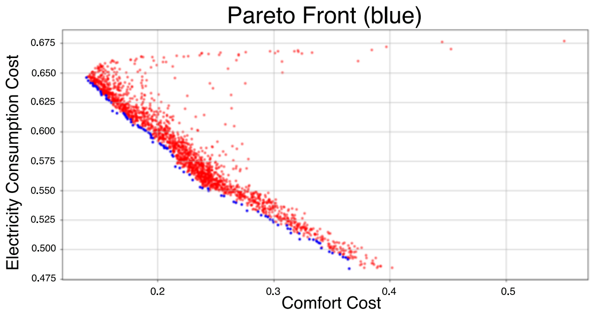

Figure 3. The Pareto front from multi-objective Bayesian optimization, showing evaluated PID parameters (k_p, k_i, k_d) using three buildings.

After completing the optimization process, we focused on the parameter values situated on the Pareto front (see Figure 3). These values were assessed according to the competition's scoring system, from which we selected the set of parameters that achieved the lowest overall cost. This revealed that the competition's criteria placed a higher emphasis on comfort over other metrics, guiding our final selection of parameters, which resulted in: $k_p = -0.290$, $k_i = -2.490$, and $k_d = 0.009$.

4.4 Hot Water Storage Control

For hot water control, we employ a simple RBC heuristic designed to leverage periods of low energy consumption and high solar energy production. We prioritize heating water during nighttime (1AM-5AM), when the overall energy consumption is low. Additionally, we heat water when the predicted solar energy generation (P_{sg}) exceeds the predicted consumption (P_{pred}). This occasionally happens during midday (12PM-2PM). We set the power p_{hw} given to the heater in each of the allowed hours as a hyperparameter. Outside the designated charging windows, our system maintains a focus on supplying the available hot water as needed. The heating heuristic is described as follows:

$$\forall\, b \in \mathcal{B} : a_t^w[b] = \begin{cases} p_{hw} & \text{if } 1\,\text{AM} \leq t \leq 5\,\text{AM}, \\ p_{hw} & \text{if } P_{sg} > P_{pred}, \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

Note that, even though this is a simple RBC method, the output action a_t^w, together with the actions proposed by the rest of the individual sub-modules, can be later refined by the central controller. Therefore, the output of this sub-module serves as a baseline action that can be later modified to optimize the community-based metrics, i.e., heat more water if there is a surplus of energy or reduce it if the consumption is larger than average.

4.5 Electricity Storage Control

The battery storage control is mainly guided by forecast values detailed in Section 4.2, aiming to optimally manage the battery's charge and discharge actions, denoted as $a_t^e[b]\ \forall\ b \in \mathcal{B}$, across a defined horizon. The objective is to optimize key metrics such as ramping, load factor, and consumption peaks, represented by the objective function $f^b(\cdot)$. To achieve this, we forecast the system's needs over the next τ hours and adjust each action to ensure these metrics are optimized for this period.

For practicality, we discretize each action within a range $[-1.0, 1.0]$, with a granularity g. Thus, we aim at selecting the optimal sequence of discrete actions, $\mathbf{a}_\tau^e$. We constraint the process to ensure system reliability by maintaining the battery's state-of-charge (SOC_t, at time t) between a lower bound ($\text{LB}(h)$) and an upper bound ($\text{UB}(h)$) which depend on the hour h of the day. The cost function to minimize is described as:

$$
\begin{aligned}
\min_{(a_j^e[b])_j} \quad & f^b(a_{t+1}^e[b], a_{t+2}^e[b], \ldots, a_{t+\tau}^e[b]), \quad \forall\, b \in \mathcal{B}, \\
\text{s.t.} \quad & a_j^e[b] \in \{-1, (-1+g), \ldots, (1-g), 1\}, \\
& \text{LB}(h) \le SOC_j(a_j^e[b]) \le \text{UB}(h), \quad \forall\, j \in [t+1, t+\tau].
\end{aligned}
$$

This optimization problem is solved using a standard A* algorithm using the grid cost $f_g(\cdot|\mathbf{a}_\tau^e)$ as the heuristic.

4.6 Outage mode

To effectively manage energy distribution during grid outages, we define a control pipeline that is operational immediately upon the detection of an outage event. During outages, the primary energy sources are the solar generation and energy stored within battery systems. In the following, we detail the management of each controlled variable during outages.

Hot Water Storage. Considering the limitations on energy availability during outages, we have chosen to suspend water heating operations. As a result, the availability of hot water is confined to the amount stored prior to the outage. This measure ensures that energy is conserved for more critical functions, thereby optimizing resource usage under constrained conditions.

Energy Storage. We implement an energy storage and distribution strategy that utilizes predictive analytics. Specifically, we forecast both the energy demand and expected solar generation using the forecast ensemble. In scenarios where the solar generation exceeds the forecasted demand, we allocate the surplus energy towards recharging the battery storage systems. Conversely, when the anticipated demand surpasses the solar generation, the system draws on the stored energy to meet the shortfall.

Temperature control. Managing temperature control presents several challenges, the primary issue being the potential for PID controller saturation. This saturation typically occurs when the control signal reaches its maximum limit due to restricted energy supply during outages. Prolonged saturation can lead to a phenomenon known as integral wind-up, where the accumulated temperature control errors in the integral term may result in excessive corrective actions once the power is restored. To quantify the extent of saturation ($\mathtt{S}$), we define it as the ratio of the discrepancy between the energy output dictated by the controller in the last step (E_{t-1}^{out}) and the actual energy delivered by the cooling system (E_{t-1}^{real}) to the controller's output energy:

$$
\mathtt{S} := \frac{E_{out} - E_{real}}{E_{out}}. \tag{8}
$$

To mitigate the effects of integral wind-up and avoid aggressive corrective actions after outages, we employ a saturation ratio to adjust the integral term ($\mathtt{I}$). This adjustment is detailed in Equation (9). The adjustment is inversely proportional to the degree of saturation.

$$
\widehat{\mathtt{I}} := \mathtt{I}\,(1 - \mathtt{S}). \tag{9}
$$

4.7 Central controller

Of the four cost functions detailed in Section 3.2, three are focused at the building level: comfort (f_c), emissions (f_e), and resilience (f_r). These functions require optimization individually for each building involved in the problem. Conversely, the grid metric (f_g) aims to manage energy consumption across the entire community, minimizing energy usage fluctuations (ramping) and reducing peak demand, which includes the load factor, daily peaks, and historical peaks. In order to optimize the grid usage for the whole community, the central controller gathers the decisions taken by individual sub-controllers from each building and modifies previously decided actions. The central controller primarily functions in two key areas: *smoothing overall consumption* and *reducing peak demand*.

Consumption Smoothing. The primary aim here is to align the community's total energy consumption closer to a mean consumption trajectory. This involves adjusting the predicted energy usage for the upcoming hour towards the cumulative average consumption rate. The average consumption $\bar{E}_t$ at any given simulation step t is calculated by dividing the total energy usage by all buildings up to that point by the number of elapsed steps. To guide these adjustments, upper (B_{up}) and lower (B_{low}) bounds are established based on the standard deviation of consumption rates. Adjustments to the system are made based on forecasted consumption levels. If consumption is anticipated to exceed the upper limit, the system initially reduces non-essential water heating. If further reductions are necessary, it then decreases cooling capacity, but only up to a predefined maximum limit, Θ_{max}. Conversely, when consumption is predicted to fall below the lower threshold, it presents an opportune moment to enhance water heating and battery charging.

Peak Demand Reduction. After efforts to smooth consumption, consumption spikes may still occur. We identify a spike when the consumption exceeds two standard deviations above the mean $\bar{E}$. In such scenarios, the cooling action is either halted entirely or reduced until the total energy consumption returns to within the predefined acceptable range established by the bounds B_{up} and B_{low}.

5 Experiments

5.1 Experimental Protocol

In the 2023 CityLearn challenge, the weights in the cost function were determined as follows: $w_c = 0.3$ for comfort, $w_e = 0.1$ for emissions, $w_g = 0.3$ for grid stability, and $w_r = 0.3$ for resilience. Within the resilience and grid stability categories, all subcomponents were weighted equally. The implementation of our algorithm is based on Python with standard libraries (*Numpy*, *Pandas*), the 2.1b12 version of the *CityLearn* library for the environment, and the *xgboost* library for the forecasting. The details about the values used for the different hyperparameters are left in the supplementary material.

We mentioned in Section 3 that the terms in the cost function are normalized over a baseline cost. This value is obtained by not controlling any device in the house, so the simulation is ran without taking any actions. Furthermore, the environment provides some default baseline models for rule based control and reinforcement learning agents. We report in Table 1 the results of the most simple rule based controller with the name `RBC`.

Table 1. Final leaderboard of the challenge. We highlighted the best cost for the private and public dataset. Lower is better.

Team	**Private Cost**	**Public Cost**	Comfort	Emissions	Ramp.	Load	Peak	All-Time Peak	Resilience	Unserved En.
RBC (baseline)	1.124	1.085	2.190	0.994	1.045	0.673	1.432	1.436	0.803	0.750
CHESCA	**0.565**	0.562	0.132	0.944	0.892	0.951	0.875	0.811	0.715	0.448
Team 2	0.575	**0.464**	0.304	0.883	0.783	0.869	0.844	0.789	0.398	0.278
Team 3	0.582	0.508	0.203	0.932	0.798	0.911	0.887	0.792	0.445	0.508
CHESCA*	0.548	0.522	0.129	0.930	0.845	0.954	0.873	0.958	0.642	0.316

Table 2. Ablation study of the prediction method, evaluated in the grid cost of the training dataset. Lower is better.

Method	Ramping	Load factor	Daily peak	All-time peak	Grid Cost
Oracle (Real Values)	0.791	0.929	0.840	0.829	0.848
Our Ensemble Model	0.812	0.933	0.853	0.897	0.874
XGBoost	0.851	0.951	0.871	0.928	0.900
Online XGBoost	0.856	0.955	0.874	0.899	0.896
Hist. Aggregation	0.867	0.942	0.865	0.913	0.896

5.2 Results: Performances on CityLearn 2023 Contest

The evaluation of the challenge was divided following the division of the data highlighted in Section 3.3. The leaderboard for the public dataset was available during the months of the competition, without access to the data itself. The private cost was instead computed over a different dataset, and the leaderboard obtained with these costs was not visible until the announcement of the winner. The final cost in each case is computed following what is described in Section 3.2.

We compare our algorithm with the second and third teams in the challenge [2], which employed diverse strategies. The second team used MPC for battery and temperature control, and RBC for DHW. The third team used a RL-based approach with curriculum learning and hand-crafted penalties for consumption smoothness.

In Table 1 we report the final results of the challenge, detailing the average cost along with individual sub-terms for all the approaches. Notably, compared to competitors and baselines, our algorithm exhibited consistent performance, with minimal variance in cost from the public to the private dataset. Furthermore, it achieved a significantly lower cost in terms of comfort, which was crucial in our victory over competing teams, which perform equally or slightly better in the emission, grid and outage terms. For the outage terms, we identified a minor enhancement that improved outage-related costs shortly after the challenge's final submission deadline. The cost obtained with this improvement is reported as CHESCA* in Table 1.

5.3 Analysis: Ablation study

This experiment was carried out post-competition, once the full dataset was released. We investigated the influence of the forecasting component on our algorithm's performance. To assess this, we tested the algorithm using actual future observations instead of forecasts. We used grid-related costs as metrics, since these were the terms affected by the forecasts. The comparisons between the *ground truth* (i.e., the real data) and our ensemble model, as well as the individual models constituting the ensemble, are detailed in Table 2.

This evaluation serves as an ablation study, revealing that each individual model within the ensemble yields a worse overall cost compared to the ensemble. Interestingly, the results with perfect information (actual future observations) are comparable to those of the ensemble model, suggesting that the forecasting component is highly effective. This similarity also implies that there is limited potential for further improvements in forecasting.

6 Discussion and Future Works

As illustrated in the previous sections, the methods based on RL lacked the ability to generalize, likely because of the small amount of data they were trained on. We avoided the generalization problem developing an algorithm that is inherently general: nothing in CHESCA is tuned specifically on the dataset, all the rules and techniques we applied aim to control a general building. This problem is very common in RL applications, especially when the data is not trivially obtained or generated and is well-studied in the literature [27, 26, 13]. Building a deep RL algorithm that is sample efficient is a challenging task, but it is a fundamental step towards larger real world applications.

Going forward, one possible approach is to merge the problem solving capabilities of RL with the interpretability and general nature of heuristic-based optimization. As we mentioned in the previous section, our algorithm is already performing close to its best, while more data would improve the performance of RL algorithms, especially on new, unseen data. Developing an algorithm exploiting the best of the two techniques is the natural next step for tackling building energy management problems.

7 Conclusion

In this work, we presented an algorithm to coordinate energy management in buildings belonging to the same community, i.e., a common geographic area. Our solution is interpretable and generalizable without the need of large amount of training data, combining heuristics and classical optimization techniques. Nonetheless, we validated our method by winning the CityLearn Challenge 2023, beating solutions employing more complex RL-based algorithms. We explored the structure of the proposed algorithm, and examined the factors contributing to its performance on both the training dataset and the previously unknown evaluation sets. Furthermore, we highlighted the limitations of our approach regarding the quality of the forecast variables. As future research direction, we aim at building algorithms for energy exchange management combining heuristics and deep RL.

References

[1] AIcrowd | Control Track: CityLearn Challenge | Challenges. URL https://www.aicrowd.com/challenges/neurips-2023-citylearn-challenge/problems/control-track-citylearn-challenge.

[2] The CityLearn Challenge 2023. URL https://neurips.cc/virtual/2023/competition/66590.

[3] D. Azuatalam, W.-L. Lee, F. de Nijs, and A. Liebman. Reinforcement learning for whole-building HVAC control and demand response. *Energy and AI*, 2:100020, Nov. 2020. ISSN 2666-5468.

[4] E. Biyik and A. Kahraman. A predictive control strategy for optimal management of peak load, thermal comfort, energy storage and renewables in multi-zone buildings. *Journal of Building Engineering*, 25: 100826, Sept. 2019. ISSN 2352-7102.

[5] T. Chen and C. Guestrin. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, Aug. 2016. Association for Computing Machinery. ISBN 978-1-4503-4232-2.

[6] Y. Chen, L. K. Norford, H. W. Samuelson, and A. Malkawi. Optimal control of HVAC and window systems for natural ventilation through reinforcement learning. *Energy and Buildings*, 169:195–205, June 2018. ISSN 0378-7788.

[7] Y. Du, H. Zandi, O. Kotevska, K. Kurte, J. Munk, K. Amasyali, E. Mckee, and F. Li. Intelligent multi-zone residential HVAC control strategy based on deep reinforcement learning. *Applied Energy*, 281:116117, Jan. 2021. ISSN 0306-2619.

[8] R. Garnett. *Bayesian optimization*. Cambridge University Press, 2023.

[9] J. Leitão, P. Gil, B. Ribeiro, and A. Cardoso. A Survey on Home Energy Management. *IEEE Access*, 8:5699–5722, 2020. ISSN 2169-3536. Conference Name: IEEE Access.

[10] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[11] D. Mariano-Hernández, L. Hernández-Callejo, A. Zorita-Lamadrid, O. Duque-Pérez, and F. Santos García. A review of strategies for building energy management system: Model predictive control, demand side management, optimization, and fault detect & diagnosis. *Journal of Building Engineering*, 33:101692, Jan. 2021. ISSN 2352-7102.

[12] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518 (7540):529–533, 2015.

[13] O. Nachum, S. S. Gu, H. Lee, and S. Levine. Data-Efficient Hierarchical Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/hash/e6384711491713d29bc63fc5eeb5ba4f-Abstract.html.

[14] P. Rocha, A. Siddiqui, and M. Stadler. Improving energy efficiency via smart building energy management systems: A comparison with policy measures. *Energy and Buildings*, 88:203–213, Feb. 2015. ISSN 0378-7788.

[15] R. Ruusu, S. Cao, B. Manrique Delgado, and A. Hasan. Direct quantification of multiple-source energy flexibility in a residential building using a new model predictive high-level controller. *Energy Conversion and Management*, 180:1109–1128, Jan. 2019. ISSN 0196-8904.

[16] S. Salakij, N. Yu, S. Paolucci, and P. Antsaklis. Model-Based Predictive Control for building energy management. I: Energy modeling and optimal control. *Energy and Buildings*, 133:345–358, Dec. 2016. ISSN 0378-7788.

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

[18] H. Shareef, M. S. Ahmed, A. Mohamed, and E. Al Hassan. Review on Home Energy Management System Considering Demand Responses, Smart Technologies, and Intelligent Controllers. *IEEE Access*, 6:24498–24509, 2018. ISSN 2169-3536. Conference Name: IEEE Access.

[19] TheLeprechaun25. Chesca, 2024. URL https://github.com/TheLeprechaun25/CHESCA. Accessed: 2024-08-24.

[20] D. Vamvakas, P. Michailidis, C. Korkas, and E. Kosmatopoulos. Review and Evaluation of Reinforcement Learning Frameworks on Smart Grid Applications. *Energies*, 16(14):5326, Jan. 2023. ISSN 1996-1073. Number: 14 Publisher: Multidisciplinary Digital Publishing Institute.

[21] J. R. Vázquez-Canteli, J. Kämpf, G. Henze, and Z. Nagy. CityLearn v1.0: An OpenAI gym environment for demand response with deep reinforcement learning. In *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, BuildSys '19, page 356–357, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450370059.

[22] J. R. Vázquez-Canteli, S. Dey, G. Henze, and Z. Nagy. The CityLearn challenge 2020. In *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 320–321, 2020.

[23] C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

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

[25] E. Wilson, A. Parker, A. Fontanini, E. Present, J. Reyna, R. Adhikari, C. Bianchi, C. CaraDonna, M. Dahlhausen, J. Kim, et al. End-use load profiles for the us building stock. Technical report, DOE Open Energy Data Initiative (OEDI); National Renewable Energy Laboratory ..., 2021.

[26] D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, and R. Fergus. Improving Sample Efficiency in Model-Free Reinforcement Learning from Images. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):10674–10681, May 2021. ISSN 2374-3468. Number: 12.

[27] J. Zhang, J. Kim, B. O'Donoghue, and S. Boyd. Sample Efficient Reinforcement Learning with REINFORCE. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):10887–10895, May 2021. ISSN 2374-3468. Number: 12.