Prior-information Enhanced Reinforcement Learning for Energy Management Systems

Théo Zangato[1], Aomar Osmani[1], and Pegah Alizadeh[1]

LIPN, CNRS-UMR-7030, Université Paris Sorbonne Nord, France

Abstract. Amidst increasing energy demands and growing environmental concerns, the promotion of sustainable and energy-efficient practices has become imperative. This paper introduces a reinforcement learning-based technique for optimizing energy consumption and its associated costs, with a focus on energy management systems. A three-step approach for the efficient management of charging cycles in energy storage units within buildings is presented combining RL with prior knowledge. A unique strategy is adopted: clustering building load curves to discern typical energy consumption patterns, embedding domain knowledge into the learning algorithm to refine the agent's action space and predicting of future observations to make real-time decisions. We showcase the effectiveness of our method using real-world data. It enables controlled exploration and efficient training of Energy Management System (EMS) agents. When compared to the benchmark, our model reduces energy costs by up to 15%, cutting down consumption during peak periods, and demonstrating adaptability across various building consumption profiles.

Keywords: Reinforcement Learning, Energy Management Systems, Time-Series, Clustering

1 Introduction

Global energy consumption has surged from 5,268 TWh in 1974 to 22,315 TWh recently, with buildings accounting for 30% of this usage and their associated greenhouse gas emissions. This is especially significant in urban areas, where 54% of the global population resided in 2018, which puts the share of energy in the budget of European households at 25% in 2021[1]. Buildings, including individual homes, play a critical role in transitioning the energy production and consumption model to combat climate change. The integration of renewable energies into electricity networks requires adaptability to handle shifting flow patterns. Advances in renewable technology, like solar panels and batteries, have become more accessible, emphasizing the need for energy management tools. Leveraging home storage and self-generated renewable energy allows buildings to reduce their carbon footprint, lower energy costs, and alleviate grid strain during peak hours. Grid operators face challenges in maintaining and controlling energy within an increasingly interconnected network with limited new grid infrastructure and growing demand. Ensuring a reliable supply is essential, as disruptions can have severe economic and social consequences. Grid operators employ strategies, including pricing mechanisms to

[1] source: `http://data.europa.eu/88u/dataset/e3td1ejcprfbhotlntxwa`

encourage off-peak load shifting and limiting network energy use when renewable energy is scarce. Self-generated electricity, such as solar energy, empowers buildings to reduce reliance on carbon-emitting sources, reducing carbon footprints. However, the intermittent nature of renewables requires their combination with home batteries for surplus energy storage [1].

Over the past decade, Energy Management Systems (EMS) have gained prominence as reported by [2]. Researchers like [3] or [4], have proposed approaches to integrate renewables into existing grid architectures and enhance energy efficiency. EMS solutions enable monitoring, control, and optimization of building energy consumption, reducing energy usage and costs through domain knowledge, historical data, and statistical analysis. Efficiently managing charging and discharging cycles of Energy Storage Units (ESU) is a key EMS challenge, given its role in reducing operational building costs.

This paper's primary objective is to propose an efficient method for managing charging cycles in electrical or thermal storage systems. We enhance the learning process for EMS agents using reinforcement learning (RL), addressing sample inefficiency concerns denoted by [5–7]. Our solution involves incorporating prior knowledge into the learning model, allowing agents to focus on task optimization rather than learning environmental constraints from scratch. This approach promotes specific subgroup generalization, reducing the need to adapt to a broad range of subgroups based on varying energy needs. Directly integrating prior knowledge streamlines learning and ensures controlled and safe exploration during training, boosting agent effectiveness in real-world scenarios while minimizing extensive modeling and associated approximations [8]. Our method includes analyzing historical energy consumption data to create typical profiles by clustering load curves from buildings. Domain knowledge informs decision-making probabilities, and during inference, a classification module assesses load profile similarities, facilitating policy generalization to similar buildings. This empowers consumers to cut energy expenses and aids network operators in peak period consumption reduction. The approach is adaptable to various building types with specific consumption profiles, further enhancing decision-making with a prediction module for real-time environmental observations.

This paper is structured as follows: in Section 2, we introduce the EMS problem. Section 3 outlines our modeling approach of EMS as an RL problem, while Section 4 details the incorporation of prior knowledge and the training process. The results of our experiments are presented and discussed in Section 5. Finally, in Section 6, we conclude the paper and discuss potential research directions. For reproducibility, the code is available on a GitHub repository[2].

[2] https://github.com/TheoZan/CL

2 Problem description

EMS play a pivotal role in overseeing and regulating energy consumption within buildings. Typically, EMS primarily aim to achieve cost savings in energy usage while also considering other objectives like reducing carbon emissions or enhancing occupants' well-being [9]. It operates within a designated area, which can be a set of buildings forming a micro-network or a single building. These buildings house various energy-consuming systems, and the energy is supplied by an energy provider, typically via the electricity grid. This network offers continuous energy access, with associated costs that can be financial and/or environmental and are subject to hourly rates. Among the energy demands of a building, there are two distinct categories: shiftable loads, which can be delayed without disrupting operations (e.g., private home oven use), and non-shiftable loads, which cannot be rescheduled.
To achieve its objectives, an EMS effectively manages shiftable and controllable loads that can be optimized based on energy tariffs or emissions, environmental conditions (e.g., temperature, humidity, sunlight exposure), and time of day. This involves a controller making decisions about activating specific devices based on available information. Additionally, buildings may incorporate renewable energy generation devices like solar panels and ESUs that provide storage capacities for previously generated electrical or thermal energy.

Various EMS approaches have been proposed, including Model Predictive Control (MPC) [10], Rule-Based Control (RBC), and Fuzzy Logic Control (FLC) [11]. Recently, RL has gained attention, particularly for its success in the gaming field [12], resulting in the emergence of RL-based EMS as a vibrant research area [13]. RL is a machine learning approach where an agent interacts with the environment to learn a sequential decision-making policy, utilizing historical data alongside environment interactions to achieve long-term goals, without relying on a complete model of the environment or a large labeled dataset [14]. These approaches extend to HVAC systems in various settings, such as personal housing and office buildings. For instance, in [15], authors combined expert functions with reinforcement learning, while in [16], Dueling-double deep Q-learning network with a Generalized corr-entropy assisted LSTM prediction module was employed for personal housing HVAC systems.

3 Problem Formulation

In our study, we examine a set of N buildings denoted as $B = \{b_1, b_2, \ldots, b_N\}$. Each building is equipped with a controller responsible for regulating the flow of energy into and out of each Energy Storage Units (ESU). This control mechanism dynamically influences the energy drawn from the grid, increasing the intake when storing energy in the units and decreasing it when releasing previously stored energy. Within the context of a specific building, let's denote the following variables:

– L_t: Non-shiftable loads (NSL) in kWh for the given building at time t.
– E_t^{th}: Electrical consumption in kWh associated with thermal needs (e.g., deep hot water, cooling, or heating) at time t.
– E_t^{ESU}: Energy transfers in kWh occurring in all ESUs at time t. If an storage unit is storing energy, then $E_t^{ESU} > 0$, and if it is releasing energy, then $E_t^{ESU} < 0$.
– E_t^{pv}: Energy produced in kWh by renewable sources at time t, in our case through solar panels.

Additionally, we denote H_t the SoC of any ESU, as the normalized proportion of energy stored in the storage device relative to the device's capacity v. The building's total energy consumption is defined by:

$$E_t = L_t + E_t^{th} + E_t^{ESU} + E_t^{pv} \tag{1}$$

At all times, the building's consumption must be satisfied, achieved through a combination of utilizing self-generated renewable energy, releasing stored energy from storage units, and acquiring energy from the grid. Furthermore, the renewable energy is always prioritised, thus:

$$E_t \geq L_t + E_t^{th} - E_t^{pv} \tag{2}$$

Depending on the action chosen by the controller, the remaining energy $E^r > 0$ is acquired from the grid.

3.1 Objective function

Our case study proposes an EMS for a single building with multiple objectives. We focus on managing the charge/discharge cycles of the building's battery in order to limit energy expenditure while at the same time limiting greenhouse gas emissions. To achieve this, we optimize two key objectives, denoted as C^1 (3) and C^2 (4) in dollars, representing the financial cost and the financial equivalent of CO2 emissions associated with the energy procured from the network. Both objectives vary over a specified time sequence T and depend on the real-time financial and environmental costs of a unit of grid energy, denoted as c_t^1 and c_t^2 at time t.

$$C^1(t) = \sum_{t=0}^{T} E_t^r \times c_t^1 \tag{3}$$

$$C^2(t) = \sum_{t=0}^{T} E_t^r \times c_t^2 \tag{4}$$

3.2 Reinforcement Learning modeling

By defining the control of power flows hourly, one can consider the EMS as a sequential decision problem. In order to solve this problem, every building b_i and its dynamical environment can be modeled as a Markov Decision Process (MDP): $M =< S, A, P, R, \gamma >$ in which S, A, P, R are set of states, set of actions, probability of choosing an action in a given state and ending in another state, and reward function respectively, $\gamma \in [0, 1]$ is the discount factor. In an MDP, the main objective is to find an optimal policy $\pi : S \longrightarrow A$ that maximizes (minimizes) the expected discounted sum of rewards (penalties):

$$J = \mathbb{E}_\tau \left[\sum_{t=0}^{T-1} \gamma^t r_t \right], \tag{5}$$

where τ is a trajectory generated by policy π and T the maximum episode length.

State space Each $s_t \in S$ is defined for a given building as:

$$s_t = \left\{ C_t^{\text{grid}}, \{H_t\}, E_t^{\text{pv}}, L_t, \{K_t\} \right\}. \tag{6}$$

It includes the cost of a grid energy unit C_t^{grid} being the aggregation of $c^1 t$ and c_t^2, a set of ESUs' SoC, the renewable energy production, the building's non-shiftable load and a set of temporal features $\{K_t\}$ representing the month, day and hour.

Action space The agent controls the ESU of the building by taking a single continuous action $a_t \in [-1, 1]$, representing the amount of energy in kWh to store or release as a proportion of total capacity of the device. We call this the theoretical bounds of our action space, representing the entire action space without any constraints other than the capacity.

We can infer the actual valid action space at each timestep based on the nature of the modeling of the different components, as well as the general physics constraint that exists. For example if $H_t = 0$, then $a_t \geq 0$. The valid action space at each time step for a given ESU d is defined by:

$$a_t = \begin{cases} [0, \frac{v_d - H_t}{v_d}] & \text{if } E^{pv} \geq E_t \\ [-\frac{\max(E_t, H_t)}{v_d}, \frac{v_d - H_t}{v_d}] & \text{else} \end{cases} \tag{7}$$

Reward function The reward function provides an interface between the agent's policy and goal. The main objective is to learn an optimal policy that decides the optimal storage device usage in each state to minimize the total cost. Our reward shaping strategy splits into two components: one emphasizes the maximization of self-generated renewable energy utilization, while the other concentrates on managing storage cycles during periods when renewables are not at their peak.

PV specialized We designed the reward function as an adversarial reward signal for situations where the renewable production exceeds the total consumption of the building to incentivize the agent to charge the storage unit when free energy is available. As a baseline situation we choose the building's consumption cost without storage nor PV capacities. The final reward is the difference between the baseline cost C^b and the cost derived from the action of our agent equipped with PV modules and a storage capacity C^a.

$$R(t) = C_t^b - C_t^a \quad = C_t^g \left(E_t^{r,b} - E_t^{r,a} \right), \tag{8}$$

with $E_t^{r,b}$ and $E_t^{r,a}$ the remaining energy coming from the grid in respectively the baseline case and the agent case. In the baseline case, all of the energy is coming from the grid.

Energy storage system management The second part of the reward focuses on managing cycles to effectively reduce operating costs. The reward function depicts the cost of each unit of energy consumed by the building and is defined based on two costs: 1) R^1, associated with energy usage from storage units, and 2) R^2, derived from grid energy usage for residual consumption.

The R^1 computation requires the cost of the energy unit C^d for the storage device. This cost is maintained by tracking the quantity and associated cost of energy transferred to the device E^d.

$$C_t^d = \frac{C_{t-1}^d \times H_{t-1} + E^d \times C_t^{\text{grid}}}{H_t}. \tag{9}$$

Only addressing cost R^1 at discharge fails to convey optimal charging time, which might wrongfully induce the agent to maintain a complete charge rather than optimizing charging phases. To address this issue, we introduce hyper-parameter ζ to the final cost function. It adjusts the cost of using stored energy between charging and discharging periods. A ζ value of 0 or 1 signifies only discharge or charge time cost consideration, respectively.

$$\begin{aligned} R(t) &= R_t^1 + R_t^2 \\ &= (1-\zeta) C_t^d \left(\max(0, H_{t-1} - H_t)(1-\alpha) \right) \\ &+ \zeta C_t^g \sum_{e \in E^{th}} E_t^r + E^{ESU} \end{aligned} \tag{10}$$

The second part of the equation relies on the cost of responding to remaining demands with by using the grid.

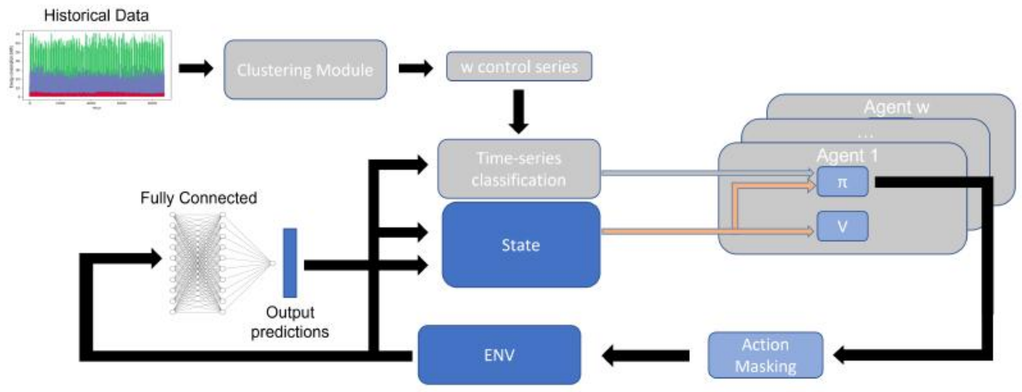


Fig. 1: The overall framework of the proposed approach. The grey part represents the policy mapping based on pre-identified consumption patterns.

4 Proposed Algorithm

Analysis of the train dataset shows different consumption profiles, suggesting that different types of building are present in the dataset, such as residences and stores. The challenge is to find a solution that can be applied to each of these different buildings groups, each of which have their own unique requirements.

4.1 Behaviour identification through time series analysis

To help identify similar behaviours among buildings, we clustered the consumption load of each building. We used the non shiftable loads as it is representative of the building's overall consumption and available in the historical data of the buildings, and thus enables us to distinguish between different building types, in preparation for the application of a given agent to a new building type. This corresponds to the clustering phase.

Time series clustering To group consumption into clusters, we leverage a clustering method defined in [17]. Let $\mathbf{X} \in \mathbb{R}^{n\times m}$ be the the matrix of time series, where n is the number of series and m is the length of each series. For each time series $\mathbf{x}_i$ in $\mathbf{X}$, we compute the derivative $\mathbf{x}'_i$ as follows, which reflects the trend of the series on all points:

$$\mathbf{x}'_i(t) = \frac{d}{dt}\mathbf{x}_i(t) \tag{11}$$

We then perform the Fourier transformation [18] using an FFT (Fast Fourier Transformation) on each derivative series $\mathbf{x}'_i$ to obtain the frequency domain representation, which we denote $\hat{\mathbf{x}}_i$ given by:

$$\hat{\mathbf{x}}_i(\omega) = \mathcal{F}[\mathbf{x}'_i(t)] = \int_{-\infty}^{\infty} \mathbf{x}'_i(t)\, e^{-j\omega t}\, dt \tag{12}$$

We employ the FFT as a feature extractor to accentuate significant frequency-related patterns in the data. By utilizing FFT, the resulting series become less complex, enhancing the efficiency of the Dynamic Time Warping (DTW) algorithm, particularly advantageous when dealing with long time series. This approach enables a targeted focus on patterns within the time series that exhibit pronounced frequency dependencies. It is particularly pertinent in the context of electric consumption in buildings, known for showcasing distinct time-related patterns.

We transform our matrix one last time using Dynamic Time Warping (DTW) algorithm [19]. This technique enables the comparison of similarity between time series that vary in length, speed or frequency, while limiting the effects of shifting and distortion. With its one-to-many approach, the algorithm matches the indexes of each series with at least one index of the other series, enabling a more elastic matching that can detect common patterns and trends even if they are out of phase. To achieve this, the algorithm seeks to arrange the point sequences by minimizing their distance in order to align the series.
There are three main steps. The first is to construct the distance matrix representing the set of pair-wise distances of two series X and Y in some feature space Φ.

$$C_l \in R^{N\times M} : c_{i,j} = \|x_i - y_i\|, i \in [1:N], j \in [1:M] \tag{13}$$

The second step consists of going through the resulting matrix in order to select the alignment. The algorithm exploits the matrix to minimize the cost between points in the sequences while satisfying boundary, monotonicity and step size conditions. The associated cost function is denoted:

$$c_p(X,Y) = \sum_{l}^{L} = c(x_{n_l}, y_{m_l}) \tag{14}$$

The final solution is then generated by finding the path with the lowest cost using the Dynamic Programming algorithm. The resulting DTW distance function is noted:

$$DTW(X,Y) = c_{p*}(X,Y) = \min\left\{c_p(X,Y), p \in P^{N\times M}\right\} \tag{15}$$

The algorithm logic is depicted in Algorithm 1.

Algorithm 1 Transform and Compare Time Series

```
Require: $\mathbf{M} \in \mathbb{R}^{n \times m}$, a matrix of $n$ time series each of length $m$
Ensure: $\mathbf{D} \in \mathbb{R}^{n \times n}$, a matrix of distances between transformed time series
 1: for $i = 1$ to $n$ do
 2:     Compute the derivative of the $i$-th time series $\mathbf{x}_i$:
 3:     $\mathbf{x'}_i[j] = \mathbf{x}_i[j+1] - \mathbf{x}_i[j], \forall j \in [1, m-1]$
 4:     Compute the Fourier transform of the derivative time series $\mathbf{x'}_i$:
 5:     $\hat{\mathbf{x}}_i = \mathcal{F}(\mathbf{x'}_i)$
 6: end for
 7: for $i = 1$ to $n$ do
 8:     for $j = i + 1$ to $n$ do
 9:         Compute the Dynamic Time Warping (DTW) distance between $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}_j$:
10:         $\mathbf{D}_{ij} = \mathrm{DTW}(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j)$
11:         $\mathbf{D}_{ji} = \mathbf{D}_{ij}$
12:     end for
13: end for
```

Clusters are built from the resulting matrix using hierarchical clustering. Hierarchical clustering is used to build nested clusters by recursively merging or splitting each of them. Here, we use the so-called agglomerative form, which consists of considering each observation as a single cluster at the start of the process and then merging pairs of clusters at each generation. To build our clusters hierarchically, we need to provide a distance measure between each observation, as well as a linkage criterion that indicates the dissimilarity of the clusters in terms of the point-to-point distances of the observations in the clusters under study. The distance measure chosen is the Euclidean distance calculated on the basis of our matrix defined between two points a and b by: $d(a,b) = \sqrt{(a-b)^2}$.

The variance between the merged clusters is minimized as a linkage criterion known as linkage ward defined by:

$$\begin{aligned} \frac{|A|.|B|}{|A| \cup |B|} \|\mu_A - \mu_B\|^2 = & \sum_{x \in A \cup B} ||x - \mu_{A \cup B}\|^2 \\ & - \sum_{x \in A} ||x - \mu_A\|^2 - \sum_{x \in B} ||x - \mu_B\|^2 \end{aligned} \tag{16}$$

Time series classification Once the clusters have been defined, we obtain three distinct data models for which we decide to learn an optimal policy for managing energy charge and discharge cycles. This then raises the question of managing a new, unknown building. To apply the right policy to the new building, we need to determine which cluster it relates to most, and as quickly as possible, in order to limit its total operating cost. To classify each building to a known cluster type, and thus map the right policy, we proceed to a classification of the building using the method previously described.

The first step is to recover 1 control time series for each of our known clusters. We then add the new series we want to classify to our time series matrix $\mathbf{M}$ of size $(w+1, m)$, where m is the length of each series and w the number of clusters.
Using Algorithm 1 we compute the dissimilarity vector $V = [V_1, V_2, \ldots, V_k]$ representing the dissimilarity between the new time series and the w control sequences. We then find the index j such that V_j is the minimum value among the elements of V such as:

$$j = \underset{j}{\operatorname{argmin}}(V) \tag{17}$$

The control sequence most analogous to the new time series corresponds to the index j. Consequently, we can map the policy of V_j to the new building, leveraging the insights and strategies derived from the identified similar control sequence.

Time series forecasting Initially, the agent can access values related to energy consumption and generation only after a delay. It can perceive the scenario at a specific moment only after that moment has elapsed. Such a setup is suboptimal for learning the desired policy as the agent is constrained to depend solely on data from the previous hour to infer its actions. The agent's reliance solely on data from the preceding hour to guide its actions becomes a bottleneck. Such a constraint hampers its decision-making, especially when the current situation diverges significantly from the previous one. Factors like volatile energy prices, changing user consumption habits, or variations in impending solar energy production can further complicate the scenario. To address this challenge, we've designed a predictive module. Armed with this predictive insight, the agent is better positioned to select the most beneficial course of action, ensuring optimal energy utilization and cost optimization.

We chose MLPs as prediction models, with 2 hidden layers of 50 neurons followed by reLu activation functions. The model takes as input the temporal features given by the environment and the last 5 observations of the value to be predicted and is trained using Adam optimizer with a decaying learning rate.

4.2 Learning algorithm

This problem is complex because, depending on each state and situation, there is a constraint on the action space. For example if the building's electric demand is already met by solar production, we cannot discharge the battery. In order to learn a decision-making policy that takes optimal actions, the constraints on the action space should be integrated as a priori knowledge into the MDP. We use the maskable version [20] of the Proximal Policy Optimization (PPO) algorithm [21] to address this problem.

PPO The Proximal Policy Optimization (PPO) algorithm acts as a policy gradient method, where an estimator of the policy gradient $\nabla_\theta J(\theta)$ is computed and updated using a stochastic gradient ascent algorithm. It is built on the work of [22] who proposed the Trust Region Policy Optimization (TRPO) algorithm which already implemented the idea of a trust region constraint by indicating that the KL-divergence of the old and new policy should not be greater than a given value δ.

The PPO gradient is of the following form: $\hat{g} = \mathbb{E}_t\left[\nabla_\theta \log \pi_\theta(a_t|s_t)\hat{A}_t\right]$, with π_θ a stochastic policy parameterized by θ and $\hat{A}_t$ an estimator of the advantage function at timestep t defined by:$A(s_t, a_t) = Q(s_t, a_t) - V(s_t)$. We can thus construct an associated objective function whose gradient corresponds to the policy gradient estimator,

$$L^{PG}(\theta) = \mathbb{E}_t\left[\log \pi_\theta(a_t|s_t)\hat{A}_t\right]. \tag{18}$$

The algorithm uses a singular objective function known as the clipped surrogate objective function that constrains the policy change using a clipping function to avoid destructive large weights updates that are too far from the current policy. In that way, it ensure stable updates:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right], \tag{19}$$

with $r_t(\theta)$ being the probability ratio between old and new policies defined as: $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{old}}(a_t|s_t)$

Action space reduction When the agent takes an action that is valid with respect to the theoretical boundaries of the action space but that violates the valid action space, the environment automatically transforms the chosen action by replacing it with the nearest valid action. This operation does not allow to learn efficiently the dynamic bounds of our space and the design of the reward function prevents the agent from knowing its invalidity.

We determine the limits β^1 and β^2 of our valid space using equations 20 and 21, which is later used in action sampling and prediction. Boundaries can be computed using ϕ, the maximum output power of the source charging the specific Energy Storage Units (ESU) as well as Γ and η respectively the maximum input/output power and the efficiency of the ESU .

$$\beta^1 = -\frac{1}{v_d}\max\left(\Gamma_t, \eta_t H_t, E_t\right) \tag{20}$$

$$\beta^2 = \frac{1}{v_d} \min\left(\Gamma_t, \frac{v_d - H_t}{\eta_t}, \phi_t - E_t\right) \tag{21}$$

We apply a discretized, original continuous space $[-1, 1]$—partitioned into two equal sections denoting charging and discharging phases—to incorporate β^1 and β^2 in the action space.

To filter out invalid actions, we employ an action mask, ensuring: 1) trajectories T consist of valid actions and 2) only valid actions are used for gradient computation. The policy network, represented by a neural network, employs a softmax function on unnormalized logits to form an action probability distribution. Invalid actions are masked by assigning significant negative logit values, making their sampling probability null.

5 Experiment

The following section details the environment used for our experiments, the settings and the results of our approach.

5.1 Citylearn envirornment

Our approach was developed within the CityLearn environment, an open-source Gym environment tailored for the creation of agents specializing in building energy coordination and demand response within urban settings [23]. Known for its scalability, CityLearn spans over 60 buildings situated in four distinct climatic zones across the USA. The building model in CityLearn adopts a hybrid approach, combining first-principles-based and data-driven models to simulate both thermal and electrical behaviors, accounting for both occupants' preferences and the physical attributes of buildings.

This versatile environment encompasses diverse systems, including energy storage units (ESUs) and renewable energy devices, each building being uniquely equipped. The central objective within the CityLearn framework is to optimize energy consumption by tactically controlling ESUs in adherence to specific constraints and objectives.

5.2 Experimental settings

In our experiment, the discretized action space contains two segments each with a length of $l = 10$, representing ten evenly spaced capacity fractions for either charging or discharging. An additional observation represents the idle state. The buildings are split between training and test sets, maintaining equal building group distributions.

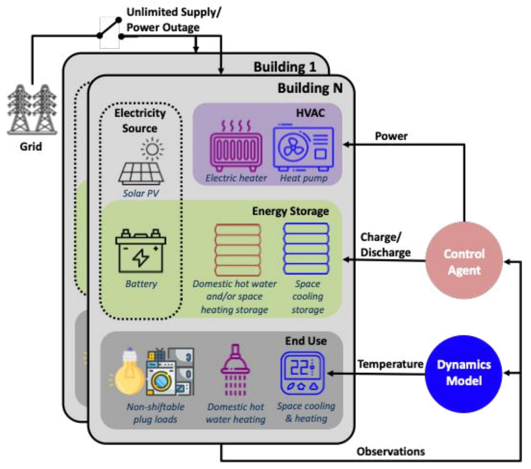
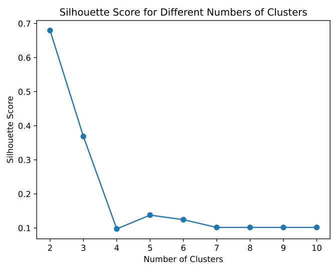


Fig. 2: Citylearn environment overview.
[23]

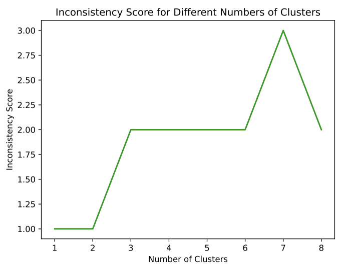
To improve learning performances, we normalized the observations fed to the agents. For normalization, we used $8760 * 5$ observations collected via random agents to center and reduce each new observation during training. Agents took random actions, dictated by a uniform law $U(a, b)$, where a and b are the valid action space bounds. We used a sine-cosine transformation for the temporal features K_t, preserving their cyclical character. Each temporal observation k was replaced by two observations, $k^1 = \cos(2\pi k \times k_{max})$ and $k^2 = \sin(2\pi k \times k_{max})$, where k_{max} is k's maximum value during an episode.

5.3 Results

Clustering and classification We applied the clustering method detailed in Section 4.1 on the annual NSL consumption time-series from the training set buildings, intending to group different consumption patterns. The NSL is an accurate representation of overall consumption as it aligns with occupancy levels that influence thermal needs. NSL equipment is also omnipresent while other devices like cooling devices may not, making NSL time-series a reliable metric for consumption patterns. We used $w = 3$ clusters, selecting w by a balanced consideration of both silhouette scores and inconsistency scores for different numbers of clusters (Figure 3). While the silhouette score slightly favours 2 clusters, the low inconsistency scores for 3 clusters and the observation of a plateau in the inconsistency plot until 6 clusters suggest that 3 clusters provide a reasonable compromise between cluster quality and consistency in the hierarchical structure. This aligns with the domain knowledge of the dataset that suggest more than two types of patterns in the data and that is further showcased in the dendogram's construction 4.



(a) Silhouette Score Analysis for Optimal Number of Clusters in Hierarchical Clustering.



(b) Inconsistency Analysis for Determining Optimal Number of Clusters in Hierarchical Clustering using depth of 2.

Fig. 3: Cluster Evaluation: Silhouette Scores vs. Inconsistency Scores for Different Numbers of Clusters

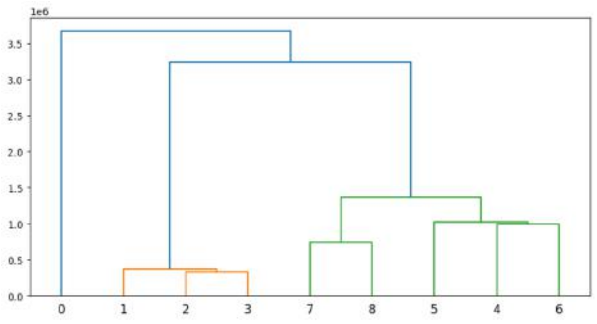
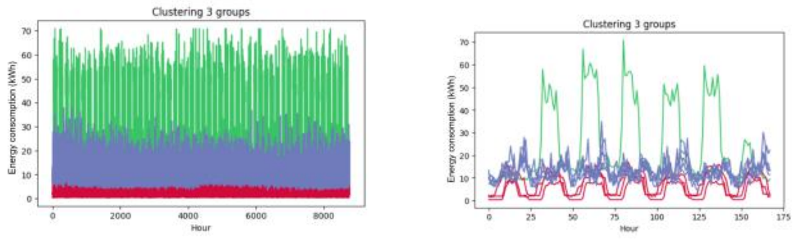


Fig. 4: Hierarchical cluster construction dendogram. Each cluster are sequentially merged during the process maximizing the distance between the clusters by maximizing the length of vertical lines connecting nodes.

To control an ESU in a building with an unknown consumption profile over a year using agents trained on three predefined subgroups, rapid association of the unknown profile with a policy is crucial. We thus employ the classification procedure on the same time-series, retaining only $m = 7 \times 24$ hours (one week or 1.9% of a year).
Both clustering instances identified identical clusters, thus validating one week's worth of data against the annual data for consistency. After one week's simulation, we can precisely assign the building to a group and optimize the agent's response. Figure 5 shows the clustering results for representative buildings for a year and a week. Based on these results, we classify each test set building using $m = 7 \times 24$.



(a) 1 year (b) 1 week

Fig. 5: w=3 group clustering of a subset of buildings using the clustering method applied on a) 1 year (m=8760) and b) 1 week (m=168) time series. Our 3 policy subtasks are defined based on those 3 different consumption patterns of non shiftable loads in kWh.

Forecasting The prediction module focuses on forecasting solar energy production which helps derive the environmental cost, quantified in W/kW. This forecast can be easily adjusted to the building's installed power capacity, ensuring its versatility across different installations. Furthermore, the module predicts financial cost, along with the building's non shiftable loads. Notably, this consumption is unaffected by the controller's actions and reflects the inherent energy usage of the building without any intervention. This approach establishes a baseline against which the controller's effectiveness can be measured. Table 1 presents the outcomes of our approach for each predicted value, benchmarked against the base case, represented by a 1-hour lag observation and Figure 6 presents the training and validation curves of the prediction module.

Policy We trained 3 agents based on the clusters identified previously, each learning a specific policy based on the group's consumption profile. When training the

Table 1: Evaluation metrics comparison of MLP predictor vs. 1-hour lag observations (Base).

Observation predicted	**Evaluation results**	
	MLP	***Base***
Financial cost	**0.007**[a], **0.995**[b]	0.702, 0.5
Solar Generation	**0.128**, **0.952**	0.56, 0.798
Non Shiftable Load	**0.117**, **0.964**	0.220, 0.871

[a]RMSE.[b] R^2.

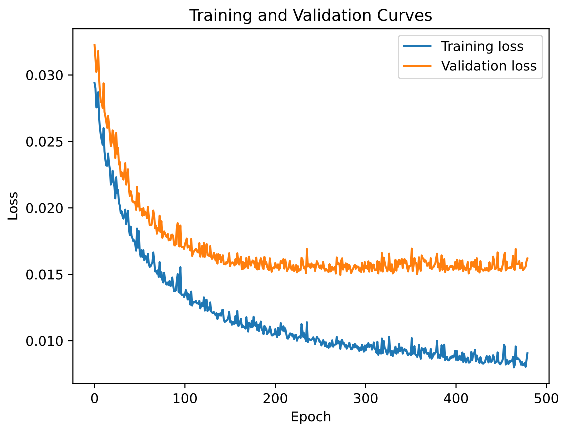


Fig. 6: Training and Validation Loss Evolution of the Predictor Model.

Table 2: Algorithm evaluation according to objective functions on test set buildings.

cluster	**Evaluation results with compared algorithms**			
group	***Ours***	***SAC***	***Random***	***RBC***
1	**1.0**[a], **0.937**[b]	1.018, 0.951	1.159, 1.163	1.004,0.998
2	**1.013**, **0.858**	1.029, 0.876	1.186, 1.171	1.017, 0.998
3	1.02, **0.887**	1.018, 0.935	1.071, 1.062	**1.004**, 0.999

[a]yearly carbon emissions.[b]Yearly price cost.

policy of a given group, a building is randomly selected from the available pool for each episode.

On the test set (T=8760, corresponding to a year), our approach reduces the building's operational cost by approximately 5% compared to scenarios without energy storage. We compare our method to a continuous RL algorithm [24] where their convergence is shown in Figure 7 and a rule-based heuristic based on objective functions defined in 3.1. The impact of hasty storage capacity management on operating costs is demonstrated using a random policy. Results in Table 2 show metrics for carbon emissions and building operating costs over duration T, normalized relative to scenarios without storage. A value of 1.1 indicates a 10% consumption increase.

Our approach reduces financial operating costs by up to 15%, depending on the building type, while maintaining stable environmental costs. This stability is due to daily fluctuations in the price of CO2 per tonne, in contrast to hourly fluctuations in kilowatt-hour prices ($\sigma = 0$ when $T = 24$). The CO2 price experiences minimal yearly variation ($c_v = 0.8892$) throughout the year, and overall electricity consumption remains steady with our solution's implementation.
Our solution demonstrates consistent policy learning. The agent charges the battery regularly before energy costs increase, using it at optimal times. Charging occurs through multiple small charges over time, reducing efficiency losses from excessive energy transfers, indicating the agent's integration of the battery's operating model.

6 Conclusion

This study addresses the complex task of sequential energy storage management within a diverse variety of building types, taking into account their unique consumption characteristics. By studying these features in detail, we constructed a concise yet exhaustive set of policies that optimizes energy behavior across all building types, strengthening the interest of RL methods for EMS. MDP based methods enables the modeling of important constraints directly into the learning framework allowing for safe exploration, risk-free training, and practical deployment. Our research leverages building-specific data and consumption patterns to significantly improve energy management efficiency without modifying existing algorithms only

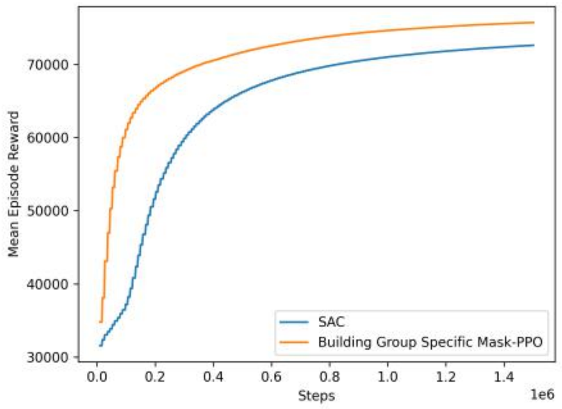


Fig. 7: Convergence of the two algorithms compared on a given group of buildings.

through modeling improvements. Furthermore, we have developed a robust classification module, designed to efficiently associate new, unseen data with existing policies using just a small number of data points. This allows us to utilize trained policies effectively without needing to learn new behaviors for every new building encountered.

Looking towards the future, an intriguing challenge rests in addressing data access issues. The objective would be to devise a solution that requires minimal data to adapt efficiently to previously unknown consumption profiles by leveraging Meta Learning.

Acknowledgments

The authors would like to extend their appreciation to the reviewers for their valuable feedback and insightful comments, which significantly contributed to the improvement of this work.

References

1. F. J. de Sisternes, J. D. Jenkins, and A. Botterud, "The value of energy storage in decarbonizing the electricity sector," *Applied Energy*, 2016.
2. L. Gelazanskas and K. A. Gamage, "Demand side management in smart grid: A review and proposals for future direction," *Sustainable Cities and Society*, 2014.
3. Y. Huang, H. Tian, and L. Wang, "Demand response for home energy management system," *International Journal of Electrical Power & Energy Systems*, 2015.
4. D. Lee and C.-C. Cheng, "Energy savings by energy management systems: A review," *Renewable and Sustainable Energy Reviews*, 2016.

5. J. Buckman, D. Hafner, G. Tucker, E. Brevdo, and H. Lee, “Sample-efficient reinforcement learning with stochastic ensemble value expansion,” in *NeurIPS 2018*, 2018.
6. J. Zhang, J. Kim, B. O’Donoghue, and S. P. Boyd, “Sample efficient reinforcement learning with REINFORCE.” AAAI Press, 2021.
7. Y. Yu, “Towards sample efficient reinforcement learning,” in *IJCAI*, J. Lang, Ed., 2018.
8. M. Beaudin and H. Zareipour, “Home energy management systems: A review of modelling and complexity,” *Renewable and Sustainable Energy Reviews*, 2015.
9. K. Kurte, K. Amasyali, J. Munk, and H. Zandi, “Deep reinforcement learning based hvac control for reducing carbon footprint of buildings,” in *2023 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, 2023.
10. D. Mariano-Hernández, L. Hernández-Callejo, A. Zorita-Lamadrid, O. Duque-Pérez, and F. S. 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*, 2021.
11. M. Motevasel and A. R. Seifi, “Expert energy management of a micro-grid considering wind energy uncertainty,” *Energy Conversion and Management*, vol. 83, pp. 58–72, 2014.
12. D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel *et al.*, “A general reinforcement learning algorithm that masters chess, shogi, and go through self-play,” *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.
13. L. Yu, S. Qin, M. Zhang, C. Shen, T. Jiang, and X. Guan, “A review of deep reinforcement learning for smart building energy management,” *IEEE Internet Things J.*, vol. 8, no. 15, pp. 12 046–12 063, 2021.
14. R. S. Sutton and A. G. Barto, *Reinforcement learning - an introduction*, ser. Adaptive computation and machine learning. MIT Press, 1998.
15. S. Xu, Y. Fu, Y. Wang, Z. Yang, Z. O’Neill, Z. Wang, and Q. Zhu, “Accelerate online reinforcement learning for building HVAC control with heterogeneous expert guidances,” in *International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation, 2022*, 2022, pp. 89–98.
16. M. Ren, X. Liu, Z. Yang, J. Zhang, Y. Guo, and Y. Jia, “A novel forecasting based scheduling method for household energy management system based on deep reinforcement learning,” *Sustainable Cities and Society*, vol. 76, p. 103207, 2022.
17. M. Łuczak, “Hierarchical clustering of time series data with parametric derivative dynamic time warping,” *Expert Systems with Applications*, vol. 62, pp. 116–130, 2016.
18. I. N. Sneddon, *Fourier transforms.* Courier Corporation, 1995.
19. M. Müller, “Dynamic time warping,” *Information retrieval for music and motion*, pp. 69–84, 2007.
20. S. Huang and S. Ontañón, “A closer look at invalid action masking in policy gradient algorithms,” *CoRR*, vol. abs/2006.14171, 2020.
21. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *CoRR*, vol. abs/1707.06347, 2017.
22. J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, “Trust region policy optimization,” in *International conference on machine learning.* PMLR, 2015, pp. 1889–1897.
23. 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*, 2019, pp. 356–357.
24. T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor,” in *International conference on machine learning.* PMLR, 2018, pp. 1861–1870.