Reward Shaping for Battery Control in CityLearn: Penalizing Peak Increases and Export Waste

Jingzeng Xie (`jx2668@columbia.edu`)
Linghao Tian (`lt3035@columbia.edu`)

Department of Electrical Engineering, Columbia University

Abstract

We study reinforcement learning (RL) control for a small district of battery-equipped buildings using CityLearn. Starting from the official CityLearn tutorial baselines—a rule-based controller (RBC), tabular Q-learning (TQL), and Soft Actor-Critic (SAC)—we introduce a shaped reward that explicitly penalizes (i) increases in the *daily* district peak import, (ii) unnecessary grid import while storage is nearly full, and (iii) grid export that discards renewable generation. On the CityLearn `citylearn_challenge_2022_phase_all` dataset with a 4-building, 7-day evaluation window, our reward-shaping variant (SAC-Our) improves district-level KPIs relative to the SAC-Net-Zero-Energy reward: **cost** 0.806, **emissions** 0.873, **consumption** 0.893, and **average daily peak** 0.823, all normalized to the baseline KPI of 1.0.

1. Introduction

Buildings with behind-the-meter solar and battery storage can provide flexibility to the grid by shifting demand away from high-price or high-emission periods and by reducing peak import. CityLearn is a widely used benchmark environment that standardizes such grid-interactive building control tasks for RL research (Vázquez-Canteli et al., 2019; 2020). However, the reward used to train RL agents can strongly influence which objectives are optimized and how stable learning becomes. In this project, we keep the SAC algorithm fixed (Haarnoja et al., 2018) and focus on *reward shaping*: we design a denser and more goal-aligned reward that encourages peak shaving and avoids counterproductive behaviors such as exporting excess solar while simultaneously importing power.

Our goal is to improve CityLearn evaluation KPIs at both building and district levels, with particular focus on cost, emissions, and peak-related metrics.

2. Background and Related Work

CityLearn provides building-level dynamics (loads, PV, batteries) and district-level evaluation KPIs, enabling reproducible comparisons across controllers (Vázquez-Canteli et al., 2019; 2020). The tutorial baselines used in this project include RBC, TQL (a discrete tabular method) (Watkins & Dayan, 1992), and SAC (a continuous-control deep RL method) (Haarnoja et al., 2018).

Reward shaping is a classic technique to speed up RL and improve alignment with downstream objectives. Potential-based shaping provides policy invariance guarantees under certain transformations (Ng et al., 1999). While our shaping is not strictly potential-based, we adopt the same philosophy: provide denser feedback that encodes domain knowledge (e.g., peak shaving and avoiding wasteful export) without changing the underlying environment dynamics.

We propose and implement a shaped reward function ("SAC-Our") tailored to CityLearn battery control that: (1) penalizes *increases* in daily peak import (not just instantaneous import), (2) discourages importing when state-of-charge (SoC) is already high, and (3) discourages exporting energy (often indicating curtailment or inefficiency). We evaluate our shaping against the tutorial controllers and show improved district KPIs.

3. Approach

3.1. Experimental setup

We modify `project.ipynb` to add our custom reward. Unless otherwise stated:

- Dataset: `citylearn_challenge_2022_phase_all`.
- Buildings: 4 buildings sampled with seed 0 (Buildings 2, 7, 9, 10 in our run).

- Simulation period: 7 consecutive days (168 hourly timesteps) sampled from the dataset with seed 0.
- Agent: centralized control with SAC using Stable-Baselines3 (Raffin et al., 2021).

3.2. Baselines

We compare against: **RBC** (rule-based battery charging/discharging), **TQL** (tabular Q-learning with discretized actions), and two tutorial SAC variants: **SAC-Original** and **SAC-Net-Zero-Energy** (a provided custom reward in the tutorial).

3.3. Our shaped reward

Let I_t be the district net import (positive for import, negative for export) and p_t the electricity price. We compute:

$$\text{imp}_t = \max(I_t, 0), \quad \text{exp}_t = \max(-I_t, 0).$$

We additionally track the daily maximum import $\max_{\tau\in\text{day}} \text{imp}_\tau$ and penalize *increases* in this quantity.

Our reward is a weighted combination of penalties and a small bonus term:

$$\bar{s}_t \;=\; \overline{\max(0, \text{SoC}_t - 0.85)}. \tag{1}$$

$$\begin{aligned} r_t = -\Big(& w_c \tanh(\text{imp}_t p_t) + w_e \tanh(\text{exp}_t) \\ &+ w_s \bar{s}_t \tanh(\text{imp}_t)\Big) \\ - \Big(& w_p (\Delta\text{Peak}_t)^2 + w_r \tanh(\text{Ramp}_t) \\ &+ w_b \tanh(\Delta\text{SoC}_t)\Big) + w_a \cdot \text{Arb}_t. \end{aligned} \tag{2}$$

Here ΔPeak_t is the (normalized) increase in the running daily peak import, Ramp_t is the normalized change in district import relative to the previous step, and ΔSoC_t is the aggregate SoC change normalized by total storage capacity. In our implementation, we use:

$$\begin{aligned} (w_c, w_e, w_s, w_p) &= (0.6, 0.2, 0.5, 0.8), \\ (w_r, w_b, w_a) &= (0.25, 0.1, 0.5). \end{aligned} \tag{3}$$

Algorithm 1 summarizes the main computations.

3.4. Training details

We train SAC for 15 episodes on the selected 7-day window. We use an MLP policy with two 256-unit hidden layers and standard SAC hyperparameters: discount 0.99, $\tau = 0.005$, learning rate 3×10^{-4}, batch size 256, replay buffer size 10^5, and entropy coefficient `auto_0.2`.

Algorithm 1 SAC-Our reward shaping (per timestep)

```
1: Compute district import/export $(\text{imp}_t, \text{exp}_t)$ from net load.
2: Update daily max import; compute normalized peak increase $\Delta\text{Peak}_t$.
3: Compute ramping penalty from change in district import.
4: Compute SoC-based penalties (import when SoC high, cycling penalty).
5: Return reward $r_t$ using Eq. (2).
```

Table 1: District-level KPIs (lower is better, baseline = 1.0).

KPI	RBC	TQL	SAC-Orig	SAC-NZE	SAC-Our
Consumption	1.423	2.321	1.000	0.927	**0.893**
Cost	1.524	2.135	1.000	0.860	**0.806**
Emissions	1.418	2.382	1.000	0.915	**0.873**
Avg. daily peak	1.446	1.580	1.000	0.886	**0.823**
Ramping	1.730	3.531	1.000	**0.941**	0.965
1 - load factor	0.968	0.954	1.000	1.000	0.995

The active observation set for SAC-Our is: `[hour, day_type, net_electricity_consumption, electricity_pricing, solar_generation, electrical_storage_soc]`.

All experiments can be reproduced by running `project.ipynb` from top to bottom. The notebook prints the selected buildings and simulation window, and generates all figures reported here.

4. Experimental Results

CityLearn evaluates controllers using normalized KPIs where 1.0 corresponds to the baseline (no control) and lower is better. Figure 1 shows district-level KPIs. SAC-Our improves cost, emissions, consumption, and average daily peak relative to SAC-Original and also improves upon the tutorial SAC-Net-Zero-Energy on most KPIs (except ramping, where SAC-Net-Zero-Energy is slightly better).

Figure 2 shows per-building cost, emissions, and consumption. SAC-Our consistently improves over SAC-Original and yields the strongest improvements on Buildings 9 and 10.

Figure 3 provides time-series evidence: SAC-Our smooths district net load and produces coherent battery actions that reduce peaks. Figure 4 shows the learning curve over episodes.

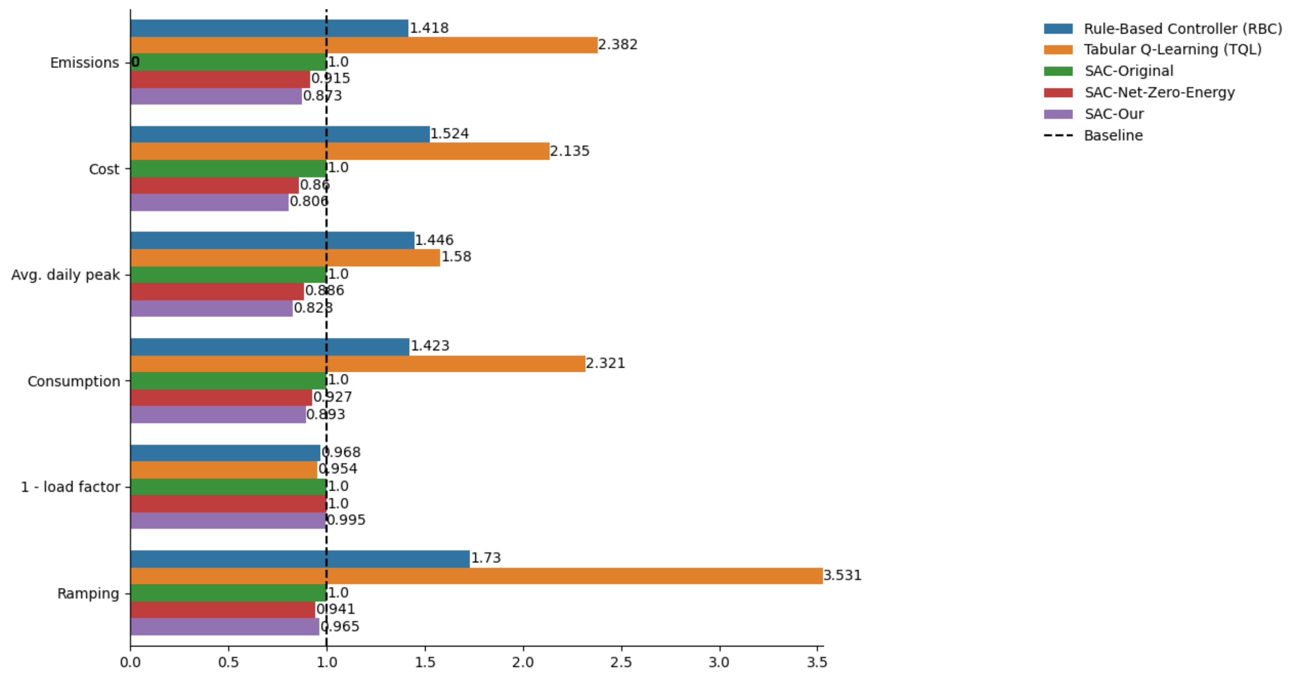


Figure 1: District-level KPI comparison across controllers (baseline = 1.0 shown as dashed line).

5. Conclusion

We implemented and evaluated a reward-shaping strategy for CityLearn battery control. By explicitly penalizing daily peak increases, wasteful export, and importing while nearly fully charged, our SAC agent improves key district KPIs, particularly cost and peak-related metrics. Future work includes training for more episodes, evaluating multiple random seeds and longer horizons, and refining the arbitrage term to account for simultaneous import/export at the *building* level.

References

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018. URL `https://arxiv.org/abs/1801.01290`.

Ng, A. Y., Harada, D., and Russell, S. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning (ICML)*, 1999. URL `https://people.eecs.berkeley.edu/~russell/papers/icml99-shaping.pdf`.

Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., and Dormann, N. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL `https://jmlr.org/papers/v22/20-1364.html`.

Vázquez-Canteli, J. R., Kämpf, J., Henze, G., and Nagy, Z. 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)*, pp. 356–357. Association for Computing Machinery, 2019. doi: 10.1145/3360322.3360998.

Vázquez-Canteli, J. R., Dey, S., Henze, G., and Nagy, Z. Citylearn: Standardizing research in multi-agent reinforcement learning for demand response and urban energy management. *arXiv preprint arXiv:2012.10504*, 2020.

Watkins, C. J. and Dayan, P. Q-learning. *Machine Learning*, 8(3):279–292, 1992. doi: 10.1007/BF00992698.

Team Contributions

Jingzeng Xie: implemented reward shaping (SAC-Our) in `project.ipynb`, ran all experiments, generated plots, and wrote the report.

Linghao Tian: implemented reward shaping (SAC-Our) in `project.ipynb`, ran all experiments, generated plots, and wrote the report.

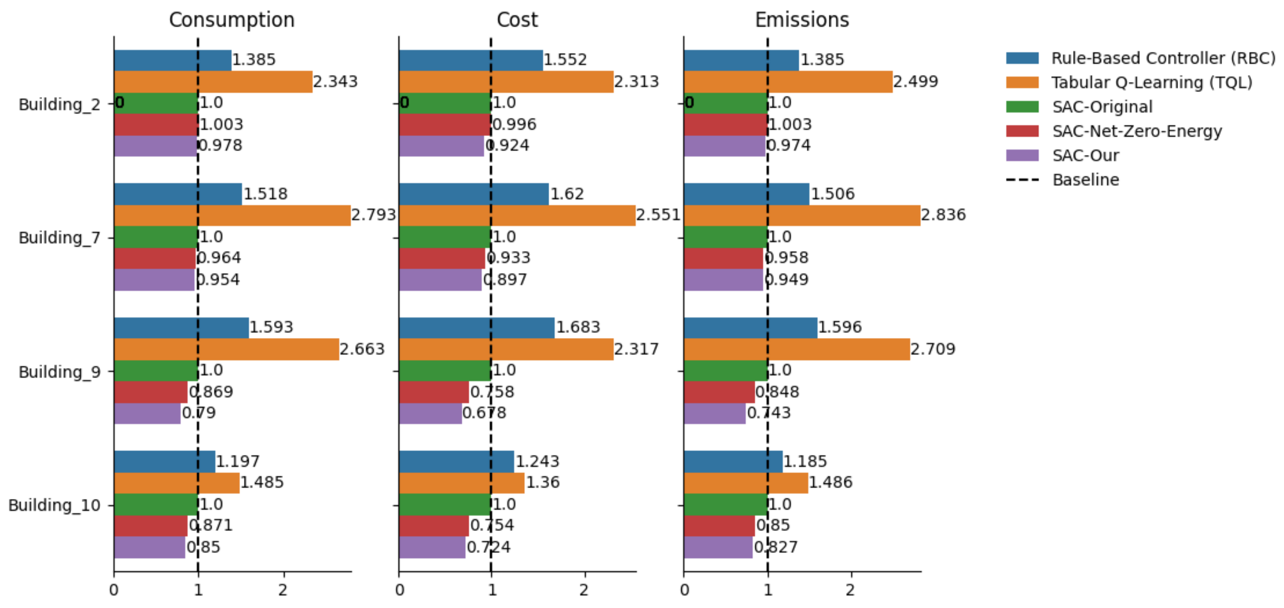
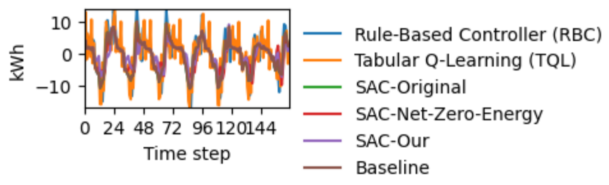
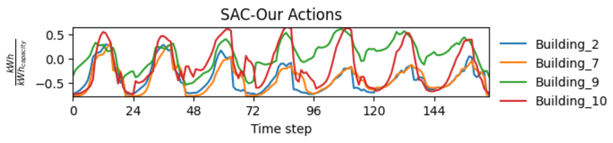


Figure 2: Building-level KPI comparison (lower is better).



(a) District net load.



(b) SAC-Our actions (one per building).

Figure 3: Time-series behavior for SAC-Our.

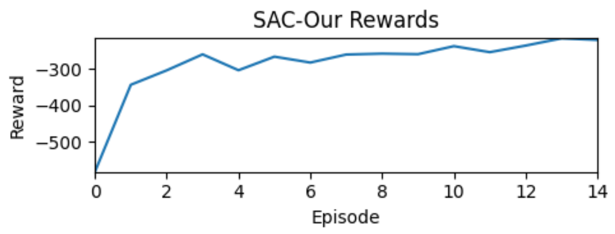


Figure 4: Episode rewards during SAC-Our training.