On the use of conditional TimeGAN to enhance the robustness of a reinforcement learning agent in the building domain

Marta Fochesato*
Automatic Control Laboratory,
ETH Zürich
Zürich, Switzerland
mfochesato@control.ee.ethz.ch

Fazel Khayatian*
Urban Energy System Laboratory, EMPA
Dübendorf, Switzerland
fazel.khayatian@empa.ch

Doris Fonseca Lima
Automatic Control Laboratory,
ETH Zürich
Zürich, Switzerland
dolima@student.ethz.ch

Zoltan Nagy
Intelligent Environments Laboratory,
University of Texas at Austin
Austin, Texas, USA
nagy@utexas.edu

ABSTRACT

This paper develops an end-to-end data-driven pipeline to improve the out-of-sample performance of a Reinforcement Learning (RL) agent operating in the domain of building energy management. The approach can benefit researchers and practitioners that are confronted with the challenge of training robust control architectures when only few historical data are available to them. Under these circumstances, in fact, the RL agent is generally unable to respond robustly to unseen (possible, rare) events. To tackle this issue, we propose a data-driven procedure composed of two steps: *(i)* we develop a novel Generative Adversarial Network (GAN) architecture to create synthetic time series profiles of building performance; *(ii)* we infuse these artificial profiles into the original training dataset. The procedure is found to increase the robustness of the RL agent to rare events, without compromising the performance during "standard" operations. Extended simulations conducted on the CityLearn OpenAI Gym environement show that the GAN-enhanced RL agent's response displays better performance metrics with respect to a rule-based controller, with results generally improving with the data-enhancement process.

CCS CONCEPTS

• **Applied computing** → *Engineering*.

KEYWORDS

Generative Adversarial Network, Time-series forecasting, Reinforcement Learning, Building energy management.

ACM Reference Format:
Marta Fochesato, Fazel Khayatian, Doris Fonseca Lima, and Zoltan Nagy. 2022. On the use of conditional TimeGAN to enhance the robustness of a reinforcement learning agent in the building domain. In *The 9th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys '22), November 9–10, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3563357.3564080

1 INTRODUCTION

The increase in the availability of building performance data has fuelled the interest in the development of data-driven models capable of improving buildings energy management [14]. As a consequence, data-driven models are gaining a central position in buildings automation research with successful applications ranging from electricity cost minimization [10], to peak shaving [16], to emissions reduction, often displaying increased performances with respect to conventional model-based or ruled-based controllers (see, for instance, [5]). However, the majority of data-driven models proposed in the literature are very data-intensive, requiring large training datasets to achieve usable performances . Additionally, it is unclear how these models respond to unseen uncertainties beyond the training dataset [26]. These issues represent a bottleneck in the widespread adoption of such methods in the building domain, where data collection still remains a time-consuming process.

This data inadequacy can potentially be addressed via synthetic building performance profiles generation. The topic has indeed caught the literature attention with recent works exploring the possibility of *(i)* infusing random noise into established physics-based models [15] [25], and *(ii)* build generative black-box models directly from measurements [30]. Despite the attractiveness of the first category due to the large availability of archetypes describing different buildings typologies, approaches based on physics-based models have the drawback of being susceptible to modelling assumptions. The second category bypasses this issue, but it faces the inherent complexity of generating high-dimensional time-series profiles by only relying on historical data.

In this sense, Generative Adversarial Networks (GANs) [11] are a promising direction. They aim at producing synthetic samples that adequately represent the underlying probability distribution of the original input data. A GAN consist of two "adversarial" blocks:

[*] Both authors contributed equally to this research. Names are stated in alphabetical order.

BuildSys '22, November 9–10, 2022, Boston, MA, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9890-9/22/11…$15.00
https://doi.org/10.1145/3563357.3564080

a generative model G that captures the data distribution, and a discriminator model D that estimates the probability of a sample coming from the training data rather than being generated by G. More formally, given a vector space of features $\mathcal{X}$, we denote as $\mathbf{X} \in \mathcal{X}$ a random vectorial instance of $\mathcal{X}$ and as $\mathbf{x}$ a specific realization of $\mathbf{X}$. Given a set of data $\mathbf{x}$ originated from the (true) distribution $p(\mathbf{X})$, we are interested in learning a distribution $\hat{p}(\mathbf{X})$ that best resembles $p(X)$. To this end, the generator builds a (nonlinear) map from a prior noise distribution $p_z(\mathbf{Z})$, with $\mathbf{Z} \in \mathcal{Z}$ in line with the introduced notation, to a data space $G(\mathbf{z})$. Then, the discriminator $D(\mathbf{x})$ outputs a scalar representing the probability that $\mathbf{x}$ stems from $p(\mathbf{X})$ or from $\hat{p}(\mathbf{X})$. The two "players" are trained simultaneously in a min-max game with loss function $V(D,G)$ modeled

$$\begin{aligned}\min_G \max_D V(D,G) =& \mathbb{E}_{\mathbf{x}\sim p(\mathbf{X})}[\log D(\mathbf{x})] \\ &+ \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{Z})}[\log(1-D(G(\mathbf{z})))],\end{aligned} \tag{1}$$

where D aims at maximizing the probability of correct labeling of samples $[\log D(\mathbf{x})]$, while G aims at minimizing the probability that D correctly classifies samples $[\log(1-D(G(\mathbf{z})))]$. Note that, while explicit density models utilize true data to estimate a distribution, implicit ones as the GANs do not directly estimate or fit the data distribution (see [13]). Aside computational advantages over explicit density models, the use of GANs can avoid overfitting, since the generator does not directly come into contact with the real samples. These appealing features justify the widespread adoption of such learning methods [13].

Despite the success of GANs in static feature generation (i.e. image reconstruction [18]), the "traditional" GAN architecture is not directly applicable in the context of buildings performance simulation. This is mainly due to two reasons: *(i)* the inability of the traditional GAN to learn temporal dynamics which play an important role in heavily seasonal-based environment; and *(ii)* the inability of the traditional GAN to exploit the availability of contextual (or side) information (in our case, weather conditions or operating modes) to learn better distributions.

Related works. There have been a number of works in the literature exploiting GANs for projecting buildings occupancy patterns [6], generating electricity load patterns [30] [12] [23] [12], and creating cooling load profiles [9]. However, these works only consider one-dimensional energy profile data. Authors in [17] develop a conditional GAN architecture for projecting multi-dimensional building data, conditioned on weather and operating conditions, and discuss in depth the plausibility of the synthetic profiles generated. The proposed GAN, however, fails to specifically learn temporal transitions. In fact, by construction, it can only capture the distributions of features within each time point, but it is unable to learn the potentially complex dynamics of those variables across time. On a different direction, a smaller body of literature departs from GANs and proposes the usage of autoregressive recurrent networks trained via the maximum likelihood principle as a method to learn temporal correlation within (possibly multi-dimensional) time series [31]. However, these approaches suffer of two drawbacks: first, as they account for stepwise transition dynamics they are prone to potentially large prediction errors when performing multi-step sampling; second, they are essentially deterministic, thus failing to explicitly accommodate for the synthetic generation task. As a (partial) solution to this issue, recent works invoke to cast GANs into a temporal setting, by making use of recurrent neural networks (RNN) or long-short term memory (LSTM) networks for generator and discriminator due to their ability to process sequences of data [22]. The drawback is the large datasets required for training the resulting architecture, making it unsuitable for building application due to the aforementioned scarcity of historical data.

Main contributions. The contributions of our work can be summarized as follows.

(1) We propose a novel GAN framework, which we call conditional TimeGAN (cTimeGAN), that is able to generate multi-dimensional synthetic time series that preserve the temporal consistency and the correlations of the original profiles. Our approach intends to combine the flexibility of the unsupervised GAN framework with the control afforded by supervised training in autoregressive models and the expressiveness of conditioning (i.e., labelling) both the generator and the discriminator on a set of exogenous features.
(2) We propose qualitative and quantitative methods to determine how well the generated distributions resemble the original ones and how accurately the generated data preserves the predictive characteristics of the original profile.
(3) We show the potential of synthetic profiles in improving the robustness of a reinforcement learning agent's policy to unseen (high-probably low-impact and low-probably high-impact) events. While the usage of synthetic data to enhance the performance of data-driven methods in not new in the machine learning community (see, for instance, [19]), to the best of our knowledge this is the first work to show the benefits of this method in the building domain.

It is worth mentioning that the authors came across to a recently published paper [3] proposing a similar GAN architecture. However, we point out some key differences between the two works, supporting the novelty of our contributions. On one side, the focus of [3] is mainly devoted to the analysis of the GAN architecture and its generalization properties over different building types (i.e., residential vs commercial). On the contrary, our work primarily focuses on the usage of the newly-developed cTimeGAN to enhance the robustness of the RL agent to unseen disturbances via the establishment of an end-to-end training pipeline.

2 THE CONDITIONAL TIMEGAN

Before describing the proposed cTimeGAN, we briefly recall the main features of the two state-of-the-art frameworks that inspired our architecture, namely the conditional GAN and the TimeGAN. We then outline in details the proposed architecture and we propose an extensive comparative analysis.

2.1 Conditional GAN

The conditional GAN is an extension of the original framework proposed by [21]. Both the generator and discriminator benefit from additional information y, which can include class labels or other data types that help the GAN to produce better synthetic samples. A schematic representation of its architecture is reported in Figure 1 panel (a). The original two-player min-max game (see (1)) is enhanced as:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim p(\mathrm{X}|\mathrm{C})}[\log D(\mathbf{x}|\mathbf{c})] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathrm{Z}|\mathrm{C})}[\log(1 - D(G(\mathbf{z}|\mathbf{c})))]. \tag{2}$$

This architecture has been applied in the context of building energy management [17] using a convolutional neural network.

2.2 TimeGAN

Proposed by [32], the TimeGAN aims to capture intricacies of the temporal dynamics of time-series. Given a vector space of static features $\mathcal{S}$ and one of temporal features $\mathcal{X}$, we denote as $\mathrm{S} \in \mathcal{S}$ and $\mathbf{X} \in \mathcal{X}$ two random vectorial instances of them that can be instantiated with specific values denoted $\mathbf{s}$ and $\mathbf{x}$. We consider tuples of the form $(\mathrm{S}, \mathbf{X}_{1:T})$ with some joint distribution $p(\mathrm{S}, \mathbf{X}_{1:T})$. Given a training dataset composed of $n = 1, \dots, N$ of such tuples, $\mathcal{D} = \{(\mathbf{s}^n, \mathbf{x}^n_{1:T})^N_{n=1}\}$, we are interested in learning a density $\hat{p}(\mathrm{S}, \mathbf{X}_{1:T})$ that best resembles the true one, $p(\mathrm{S}, \mathbf{X}_{1:T})$. Exploiting the autoregressive decomposition of the joint distribution $p(\mathrm{S}, \mathbf{X}_{1:T}) = p(\mathrm{S}) \prod_t p(\mathbf{X}_t \mid \mathrm{S}, \mathbf{X}_{1:t-1})$ yields to the simpler objective of learning a density $\hat{p}(\mathbf{X}_t|\mathrm{S}, \mathbf{X}_{1:t-1})$ approximating the true one at any time t. This makes it possible to divide the learning objective into a global objective and a local objective. The global objective,

$$\min_{\hat{p}} d(p(\mathrm{S}, \mathbf{X}_{1:T})||\hat{p}(\mathrm{S}, \mathbf{X}_{1:T})), \tag{3}$$

intends to minimize an appropriately selected notion of distance between the real and the learned probability distributions. The local objective,

$$\min_{\hat{p}} d(p(\mathbf{X}_t|\mathrm{S}, \mathbf{X}_{1:t-1})||\hat{p}(\mathbf{X}_t|\mathrm{S}, \mathbf{X}_{1:t-1})), \tag{4}$$

uses the original data to probe via maximum-likelihood training whether the learned density approximates the original density one in an accurate manner.

As can be seen in Figure 1 panel (b), the architecture consists in four elements. Namely, to the standard architecture composed of a generator and a discriminator, two autoencoding elements - an embedding and a recovery function, respectively - are added. This choice allows to learn the underlying temporal dynamics of the data via lower-dimensional representations. The embedder E consists on a embedding function $e : \mathcal{S} \times \prod_t \mathcal{X} \to \mathcal{H}_\mathcal{S} \times \prod_t \mathcal{H}_\mathcal{X}$ mapping the static and temporal features $(\mathbf{s}, \mathbf{x}_{1:T})$ to their latent codes $\mathbf{h}_s, \mathbf{h}_{1:T} = e(\mathbf{s}, \mathbf{x}_{1:T})$ living in the lower-dimensional spaces $\mathcal{H}_\mathcal{S}$ and $\prod_t \mathcal{H}_\mathcal{X}$, respectively. On the other direction, the recovery component R performs the inverse operation via a function $r : \mathcal{H}_\mathcal{S} \times \prod_t \mathcal{H}_\mathcal{X} \to \mathcal{S} \times \prod_t \mathcal{X}$ mapping the latent codes back to their approximate representations $\tilde{\mathbf{s}}, \tilde{\mathbf{x}}_{1:T} = r(\mathbf{h}_s, \mathbf{h}_{1:T})$. The embedding and recovery functions can be parameterized by any architecture of choice, with the only requirement of obeying causal ordering. Recurrent neural networks are a natural choice.

The generator function $g : \mathcal{Z}_\mathcal{S} \times \prod_t \mathcal{Z}_\mathcal{X} \to \mathcal{H}_\mathcal{S} \times \prod_t \mathcal{H}_\mathcal{X}$ takes a tuple of static and temporal realizations $(\mathbf{z}_s, \mathbf{z}_{1:T})$ drawn from the vector spaces $(\mathcal{Z}_\mathcal{S}, \prod_t \mathcal{Z}_\mathcal{X})$ over which known distributions are defined and produces synthetic latent codes $\hat{\mathbf{h}}_s, \hat{\mathbf{h}}_{1:T} = g(\mathbf{z}_s, \mathbf{z}_{1:T})$ in the embedding space. Finally, the discriminator function $d : \mathcal{H}_\mathcal{S} \times \prod_t \mathcal{H}_\mathcal{X} \to [0,1] \times \prod_t [0,1]$ receives the static and temporal codes in the embedded space and returns classifications $\tilde{\mathbf{y}}_s, \tilde{\mathbf{y}}_{1:T} = d(\tilde{\mathbf{h}}_s, \tilde{\mathbf{h}}_{1:T})$ Again, there are no restrictions on architecture beyond the requirement for the generator to be autoregressive.

2.3 Proposed architecture

By proposing the conditional TimeGAN architecture, we aim at combining the positive features of both the cGAN and the TimeGAN, as summarized in Table 1.

While the architecture itself does not change much with respect to the TimeGAN (see Figure 1 panel (c)), the proposed conditional TimeGAN possesses specific features that make it more desirable in the building energy domain. In particular, the cTimeGAN aims to enable both conditioning and continuity from one sequence to another by keeping track of the mixing of data before the training and by implementing conditioning of sequences on static and temporal features. As a result, it is able to generate synthetic time profiles of the desired length starting at any point in time, while maintaining the inherent seasonality and ensuring that the synthetic profiles co-vary with the weather condition and the building's occupancy pattern.

2.3.1 *Training procedure.*

The training procedure of the cTimeGAN is structured into three parts. The key feature, borrowed from TimeGAN training, is that the autoencoding components are trained jointly with the adversarial components. As a result, the cTimeGAN simultaneously learns to encode features, generate representations, and iterate across time.

In the first part of the training (called "embedding training phase"), the autoencoder is trained for dimensionality reduction in the latent space. The reconstruction loss is

$$\mathcal{L}_R = \mathbb{E}_{\mathbf{x}_{1:T}\sim p}\Big[\sum_t \|\mathbf{x}_t - \tilde{\mathbf{x}}_t\|_2 + \sum_t \|\mathbf{c}_t - \tilde{\mathbf{c}}_t\|_2 + \frac{1}{T}\sum_t \|\mathbf{c}_{s,t} - \tilde{\mathbf{c}}_{s,t}\|_2\Big], \tag{5}$$

where $\mathbf{c}_t$ denotes the dynamic conditional features and $\mathbf{c}_s$ the static conditional ones. The reconstruction loss allows for an accurate reconstruction of all input features (contextual information included) from the latent code to the original. Here, we drop the static features learning as not relevant for the considered building application. Furthermore, we drop the argument of the density p to ease the notation. Sequential conditions (such as irradiation or humidity in the case of building application) are included into the cost in the same form of the sequences to be learnt. The static conditions are divided by the length T of the sequence to balance out their weight on the total reconstruction cost.

In the second part of training the generator is trained with the supervised loss capturing the discrepancy between distributions $p(\mathbf{H}_t \mid \mathbf{H}_\mathcal{S}, \mathbf{H}_{1:t-1}, \mathbf{H}_C)$ and $\hat{p}(\mathbf{H}_t \mid \mathbf{H}_\mathcal{S}, \mathbf{H}_{1:t-1}, \mathbf{H}_C)$ (note that we do not explicitly differentiate between static conditional features and dynamical ones for ease of notation; nonetheless it is possible to encode both types). The supervised loss reads as

$$\mathcal{L}_S = \mathbb{E}_{\mathbf{x}_{1:T}\sim p}\Big[\sum_t ||\mathbf{h}_t - g_\mathcal{X}(\mathbf{h}_\mathcal{S}, \mathbf{h}_{t-1}, \mathbf{h}_C, \mathbf{z}_t)||_2 + \frac{1}{T}\sum_t ||\mathbf{h}_{s,t} - g_\mathcal{X}(\mathbf{h}_\mathcal{S}, \mathbf{h}_{t-1}, \mathbf{h}_C, \mathbf{z}_t)||_2\Big], \tag{6}$$

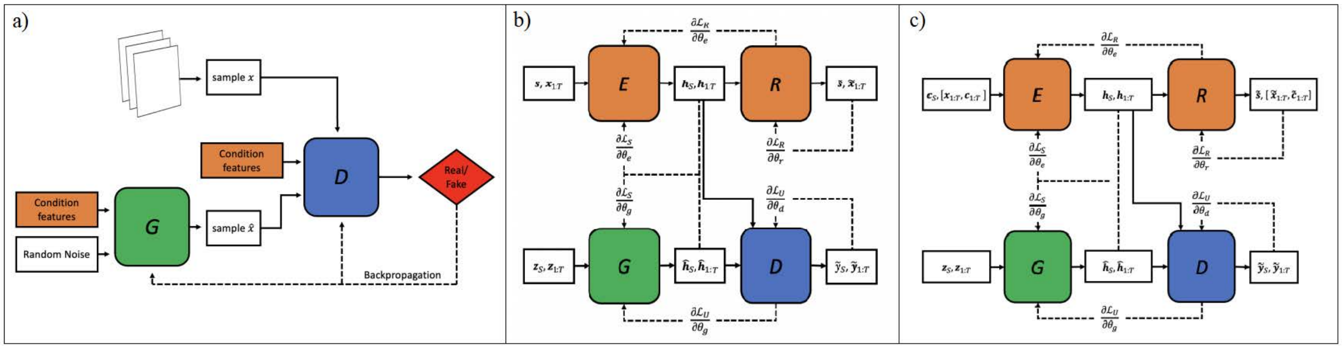
a)
sample x
Condition features
D
Real/ Fake
Condition features
G
sample $\hat{x}$
Random Noise
Backpropagation
b)
$\frac{\partial \mathcal{L}_R}{\partial \theta_e}$
$s, x_{1:T}$
E
$h_S, h_{1:T}$
R
$\tilde{s}, \tilde{x}_{1:T}$
$\frac{\partial \mathcal{L}_S}{\partial \theta_e}$
$\frac{\partial \mathcal{L}_R}{\partial \theta_r}$
$\frac{\partial \mathcal{L}_S}{\partial \theta_g}$
$\frac{\partial \mathcal{L}_U}{\partial \theta_d}$
$z_S, z_{1:T}$
G
$\hat{h}_S, \hat{h}_{1:T}$
D
$\tilde{y}_S, \tilde{y}_{1:T}$
$\frac{\partial \mathcal{L}_U}{\partial \theta_g}$
c)
$\frac{\partial \mathcal{L}_R}{\partial \theta_e}$
$c_S, [x_{1:T}, c_{1:T}]$
E
$h_S, h_{1:T}$
R
$\tilde{s}, [\tilde{x}_{1:T}, \tilde{c}_{1:T}]$
$\frac{\partial \mathcal{L}_S}{\partial \theta_e}$
$\frac{\partial \mathcal{L}_R}{\partial \theta_r}$
$\frac{\partial \mathcal{L}_S}{\partial \theta_g}$
$\frac{\partial \mathcal{L}_U}{\partial \theta_d}$
$z_S, z_{1:T}$
G
$\hat{h}_S, \hat{h}_{1:T}$
D
$\tilde{y}_S, \tilde{y}_{1:T}$
$\frac{\partial \mathcal{L}_U}{\partial \theta_g}$

Figure 1: Representation of the GAN architectures: a) Conditional GAN, b) TimeGAN, c) Conditional TimeGAN.

Feature	cGAN	TimeGAN	cTimeGAN	Non-GAN deep network
Conditions possible	✓	✗	✓	✓
Creates continuous sequences	✗	✗	✓	✓
Specifically learning temporal transitions	✗	✓	✓	✓
Type of deep architecture	Convolutional	Recurrent in latent space	Convolutional + Recurrent in latent space	Recurrent
Data intensity	fair	fair	fair	high
Generative power	✓	✓	✓	✗

Table 1: Comparison between different frameworks.

where the first part refers to the temporal sequences (and conditions), whereas the second part refers to the static features (again weighted by the sequence length T). Here, $g_{\mathcal{X}}(\mathbf{h}_{\mathcal{S}}, \mathbf{h}_{t-1}, \mathbf{h}_C, \mathbf{z}_t)$ approximates $\mathbb{E}_{\mathbf{z}_t \sim \mathcal{N}}[\hat{p}(\mathbf{H}_t \mid \mathbf{H}_{\mathcal{S}}, \mathbf{H}_{1:t-1}, \mathbf{H}_C)]$ with respect to the sample $\mathbf{z}_t$, as typically done in stochastic gradient descent.

In the third part of the training, generator, discriminator and autoencoder are jointly trained. The embedder is trained on the supervised loss (6) and reconstruction loss (5) as $\min_{\theta_e, \theta_r}(\mu \mathcal{L}_S + \mathcal{L}_R)$, with θ_e, θ_r being the weights of embedder and recover networks, respectively, and $\mu \geq 0$ being a weighting parameter. Generator and discriminator are trained adversarially on the supervised loss (6) and on the unsupervised loss $\mathcal{L}_U$ as $\min_{\theta_g}(\lambda \mathcal{L}_S + \max_{\theta_d} \mathcal{L}_U)$, where $\lambda \geq 0$ is a weighting parameter and

$$\begin{aligned}\mathcal{L}_U =& \mathbb{E}_{\mathbf{s}, \mathbf{x}_{1:T} \sim p}[\log y_{\mathcal{S}} + \sum_t \log y_t] \\ &+ \mathbb{E}_{\mathbf{s}, x_{1:T} \sim \hat{p}}[\log(1 - \hat{y}_{\mathcal{S}}) + \sum_t \log(1 - \hat{y}_t)]\end{aligned} \tag{7}$$

forces the generator to create realistic sequences evaluated by an imperfect adversary. Here, θ_g, θ_d denotes the weights of the generator and discriminator network, respectively.

The optimal training parameters of the networks are explored with a grid search using the discriminative and predictive scores proposed by [32] as metrics to monitor the convergence of the cTimeGAN. In practice, we found beneficial to train the generator and the embedder is k times as often as the discriminator. During our experiments, we observed the best performance with $k = 2$, $\lambda = 1$ and $\mu = 1$.

3 GAN VALIDATION

We demonstrate the potential of the cTimeGAN for generating synthetic data by using the CityLearn OpenAI Gym environment [29] and benchmarking its performance against the state-of-the-art cGAN architecture from [17].

The CityLearn environment hosts a virtual community of nine heterogeneous buildings including residential buildings, commercial buildings and offices, with dissimilar occupancy profiles, geometry and construction characteristics.

The model contains pre-computed building demand profiles, as well as indoor air temperature and relative humidity. For the GAN validation we focus on Climate Zone 5 among CityLearn datasets and train both the cTimeGAN and the cGAN from [17] on the first year out of the available four. Both GANs produce synthetic samples of six building performance profiles, i.e. indoor air temperature, average unmet cooling set-point difference, indoor relative humidity, electric power used by equipment, domestic hot water, and cooling loads. Both GAN architectures are fed with the same input features, as reported in Table 2.

3.1 Evaluation metrics

In this study we evaluate the plausibility of synthetic projections by resorting to both qualitative and quantitative measures. To qualitatively evaluate the plausibility of the synthetic data, we contrast GAN's projections against the original dataset. Visual comparisons of annual and daily building performance profiles in Figure 2 show that the cTimeGAN is able to capture both daily and annual seasonality. Furthermore, the proposed GAN architecture is capable of associating projections with a class of occupancy that is dependant

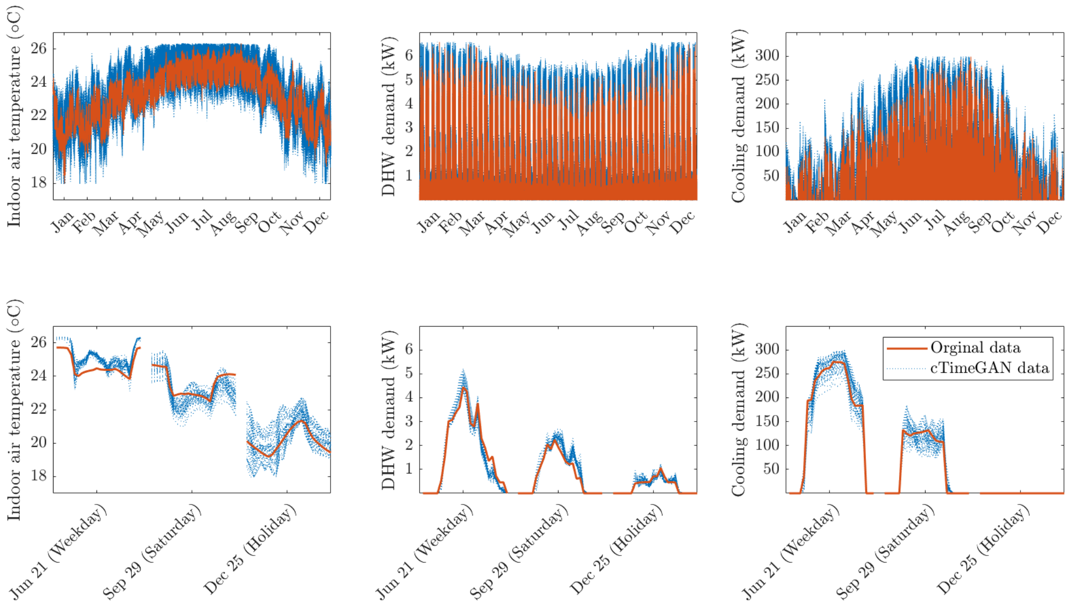

Figure 2: Visual comparison between the original profiles and the synthetic profiles generated by the conditional TimeGAN.

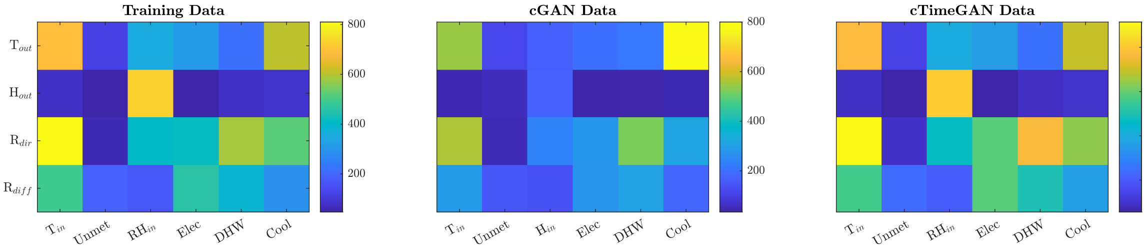

Figure 3: Quantitative assessment of the GAN architectures in terms of Granger metric. Further details on the variables are provided in Table 2.

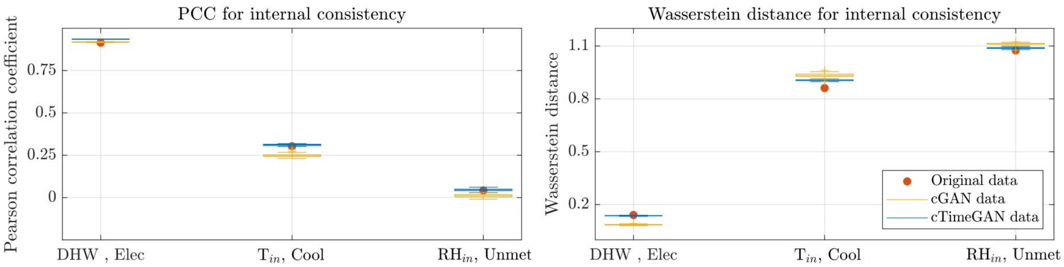

Figure 4: Quantitative assessment of the GAN architectures in terms of internal correlations. Further details on the variables are provided in Table 2.

on the type of the day. While the daily plots in Figure 2 (bottom row) may indicate certain bias in the generator's projections, quantitative assessment of the mean and variance of projections confirm that any systematic bias in the synthetic data is extremely small, and thus, may be deemed negligible.

For the quantitative assessment, we evaluate two sets of characteristics in the GAN's outputs: *(i)* external causality and *(ii)* internal consistency. The former metric reveals cause and effect relations between weather conditions and building performance (external causality). The latter metric shows the covariations within the six building performance profiles (internal consistency). External causality is evaluated by studying whether removing a "cause" variable from the inputs diminishes the accuracy of predicting outputs, i.e., "effects". Internal consistency is evaluated through element-wise and statistical comparisons between the original dataset and the synthetic projections. We use selected pairs of climate and building performance features with the objective to cover both strong (e.g. DHW, Elec) and weak correlations (e.g. RH_{in}, Unmet).

Figure 3 and Figure 4 display the goodness of the cTimeGAN projections in comparison with the cGAN projections and the original training data, using terms of the Granger causality score, the Pearson Correlation Coefficient, and the Wasserstein distance. The Granger causality test reveals whether past and present values of "cause" variables affect the predictive distribution of "effect" variables [8]. When any of the climate variables is assumed to be the "cause" variable, the other three climate variables are set to conditioning endogenous variables, and all six building performance features shape the "effect" variables.

Figure 3 shows the chi-squared-based Wald statistic score that is calculated based on the Granger causality test. Higher chi-squared values corresponds to greater statistical significance, i.e. stronger Granger-causality. The figure shows that relative to the original data (Figure 3, left), the cTimeGAN (Figure 3, right) outperforms the cGAN (Figure 3, middle) in capturing causalities between weather conditions and building performance features. On the other side, the cTimeGAN projections slightly over-represent some causalities. For instance, the cTimeGAN slightly exaggerates how direct solar radiation (R_{dir}) Granger-causes domestic hot water consumption (DHW).

Figure 4 shows that the cTimeGAN outperforms the cGAN in learning co-variations within the building performance profiles. This is evident from the Pearson Correlation Coefficient (PCC) that reveals the co-linearity (Figure 4, left), as well as the Wasserstein distance that measures the similarity between the distribution of variables (Figure 4, right). The Wasserstein distance shows the minimum cost of converting one distribution to the other. Smaller Wasserstein values correspond to greater similarity between the probability distributions, while a value of zero indicates that the two distribution are identical [27]. Both the cGAN and the cTimeGAN successfully capture the internal consistency within building performance features. However, the cTimeGAN once again surpasses the cGAN.

Feature name (shorthand)	Size and unit
Indoor Temperature (T_{in})	24 h/day in [°C]
Avg. Unmet Cool. Setpoint Diff. (Unmet)	24 h/day in [°C]
Indoor Relative Humidity (RH_{in})	24 h/day in [%]
Equipment Electric Power (Elec)	24 h/day in [kWh]
DHW Heating (DHW)	24 h/day in [kWh]
Cooling Load (Cool)	24 h/day in [kWh]
Outdoor Drybulb Temperature (T_{out})	24 h/day in [°C]
Outdoor Relative Humidity (RH_{out})	24 h/day in [%]
Diffuse Solar Radiation (R_{diff})	24 h/day in [W/m^2]
Direct Solar Radiation (R_{dir})	24 h/day in [W/m^2]
Day Type (n.a.)	1 label per day: [Weekday, Holiday]
Daylight Savings Status (n.a.)	1 label per day: [on, off]

Table 2: Summary of energy performance metrics, climate and operation constraints.

NDB score	K=25	K=50	K=100
cGAN	0.29	0.24	0.16
cTimeGAN	0.24	0.20	0.15

Table 3: Mode collapse test express in terms of NDB score for different K values.

Finally, we report mode-collapse, that is a sub-optimal convergence during the training process of GANs in which the generator only produces a small variety of outputs. Mode-collapse can be measured by various metrics, the most popular of which is the inception score [4]. However, calculating the inception score requires pre-trained models that are unavailable for building performance prediction. As an alternative, the Number of statistically-Different Bins (NDB) has proven to return a good correlation with the inception score, and therefore, is a reliable metric for quantifying mode-collapse [24].

Table 3 shows the NDB score, i.e., the fraction of the total number of bins (K) that are statistically dissimilar between the training data and the projected data. A smaller NDB corresponds to a better match between the training data and the generated data w.r.t the diversity of samples. While both the cGAN and the cTimeGAN return NDB values that are smaller than 0.3 (which corresponds to acceptable diversity), cTimeGAN marginally surpasses the cGAN in the diversity of the projections.

4 GAN-ENHANCED REINFORCEMENT LEARNING

In this section, we propose a procedure to enhance the out-of-sample performance of a RL agent by infusing synthetic data into the original training dataset. Indeed, this corresponds to adding (disciplined) randomness in our training dataset to prevent the RL agent from overfitting. For the following analyses, we use the predefined RL model available within CityLearn OpenAI Gym environment. The chosen model is a single soft-actor critic agent that controls all nine buildings in the community in a centralized manner [28].

Data enhancement is conducted by randomly selecting portions of the original training dataset and replacing them with projections from the conditional TimeGAN. We replace the original dataset in 5% increments from 0 to 100, returning a total of 21 combinations. To contrast the results of this study against the literature, we assume that the RL is trained for 15 epochs or 131400 (15*8760) hours. Thus, we concatenate 15 copies of the original 1-year dataset, and randomly replace the entries depending on the portion of data enhancement. The RL is only trained for one epoch on a single 15-year building performance dataset. Given the importance of

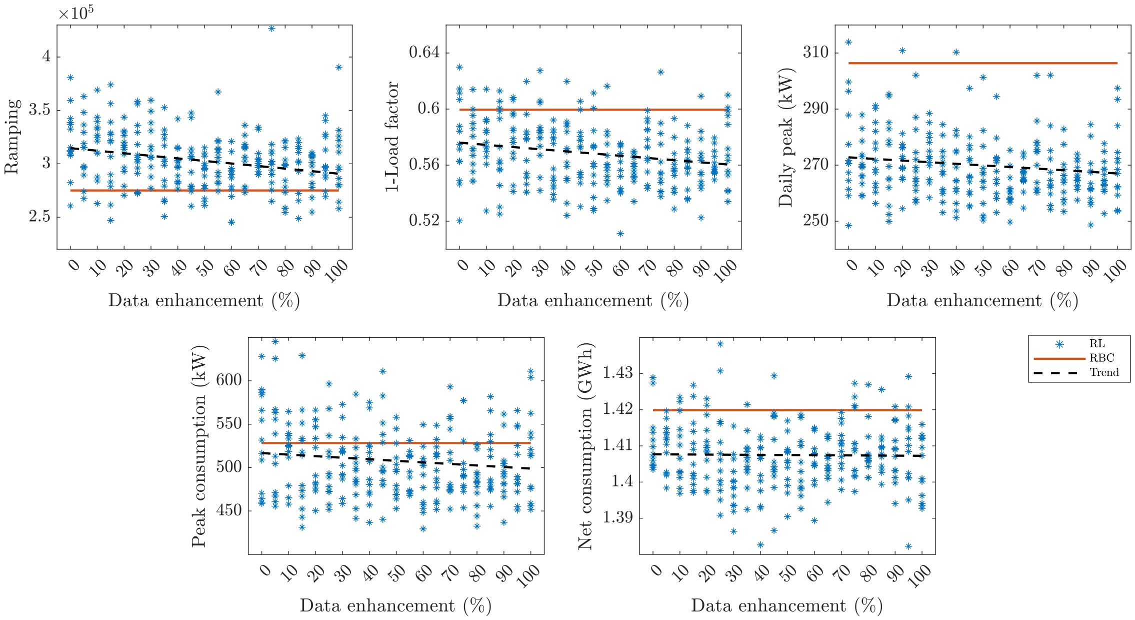

Figure 5: The performance of data-enhanced RL agents on high-probable low-impact disturbances

random initialization on the RL's performance, the training process with enhanced data is repeated 20 times.

4.1 Simulation results

We evaluate the characteristics of the RL agent that is trained on synthetic data from two perspectives. First, we assess whether the enhancement of the training data with synthetic examples affects the overall performance of the trained agents, particularly with respect to events with high-probable low-impact disturbances. For this purpose, the performance of the trained agent is assessed on a new dataset which has three years of unseen building performance data. These three years of building performance data are available as part of the CityLearn OpenAI Gym environment. Second, we evaluate whether enhancing the original training dataset with synthetic data improves the robustness of the RL agent against unseen events with low-probable high-impact disturbances. One example of such cases is a weekday in which the occupants are on holidays or vice-versa, i.e., a weekday label assigned to holiday-like occupant behavior. The test dataset available at [1] contains adversarial examples that are hallucinated by a cGAN. The adversarial examples in this dataset are projected through poisoning the labels of the generator when training the model [2]. The plausibility of the disturbances infused in the test dataset and their suitability for challenging RL agents that are trained on scarce data are discussed in [17].

Figure 5 contrasts the performance of data-enhanced RL agents against that of a conventional RL agent with no data enhancement, as well as a rule-based controller (RBC). While the general trend of data enhancement from 0 to 100% has a positive impact on the agent's performance, the magnitude of improvements vary by the metric considered. "Ramping" and "1-Load factor" metrics show the largest improvements due to enhancement of the training data with synthetic samples. Conversely, the "Net energy consumption" is hardly affected by the data enhancement. It is important to note that the insensibility of this metric to our approach comes with no surprise: by "design", the CityLearn OpenAI Gym Environment only allows shifting the loads in time, but not to reduce them; therefore, the annual energy consumption remains approximately the same. Furthermore, the trend of improvements in the performance metrics is non-monotonic, and therefore 100% replacement of data does not always result in the best performance. Given that this study is a first attempt to evaluate the feasibility of improving RL performance with GAN projections, a deep dive into the response of the trained RL agents is beyond the scope and limits of this study. However, we reckon that the non-monotonic performance of the GAN-enhanced RL models is a result of random data replacement, i.e., (1) the hours of the day, as well as (2) the days of the year which are replaced with synthetic data. When compared to the original dataset (0% enhancement) the proposed data enhancement strategy always returns comparable to better performance on high-probable low-impact disturbances.

Figure 6 shows the performance of data-enhanced RL agents in the presence of low-probable high-impact disturbances. The left y-axes (blue dashed line) shows how data enhancement affects the mean of the performance and the right y-axis (orange dash-dotted line) displays the variance of the performance for each metric. Among all five performance metrics, the trend of "Ramping" clearly shows improvements with data enhancement. Other metrics such as the "1-Load factor", "Daily peak", and "Net consumption" also display subtle improvements in the performance mean. Yet, this trend is not monotonic and could be affected by the random initialization of the RL agent, as well as the placement of the replaced synthetic

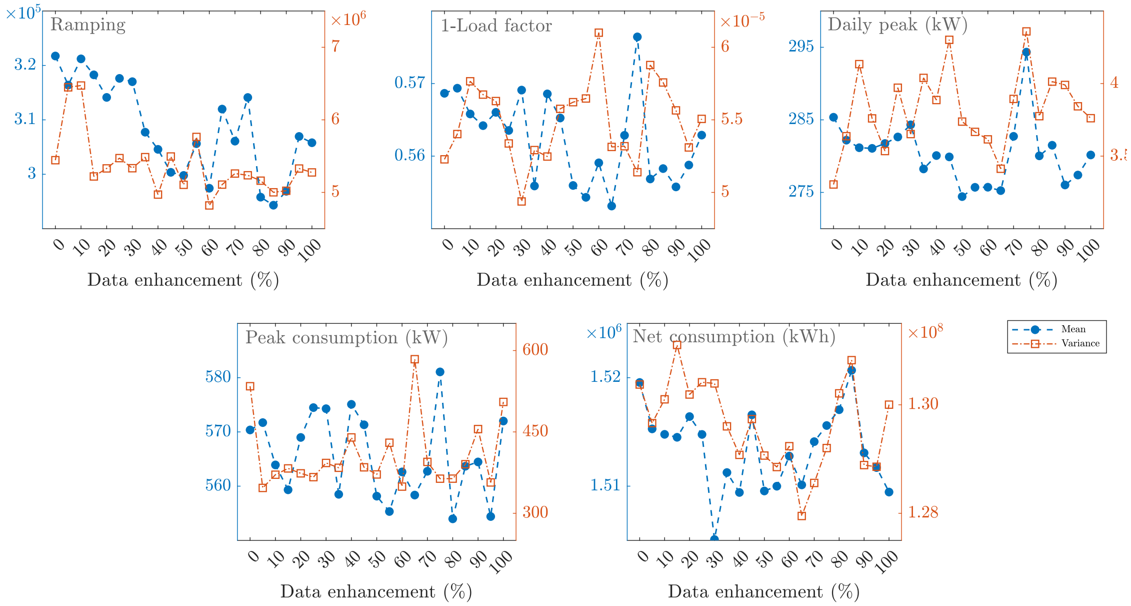

Figure 6: The performance of data-enhanced RL agents on low-probable high-impact disturbances

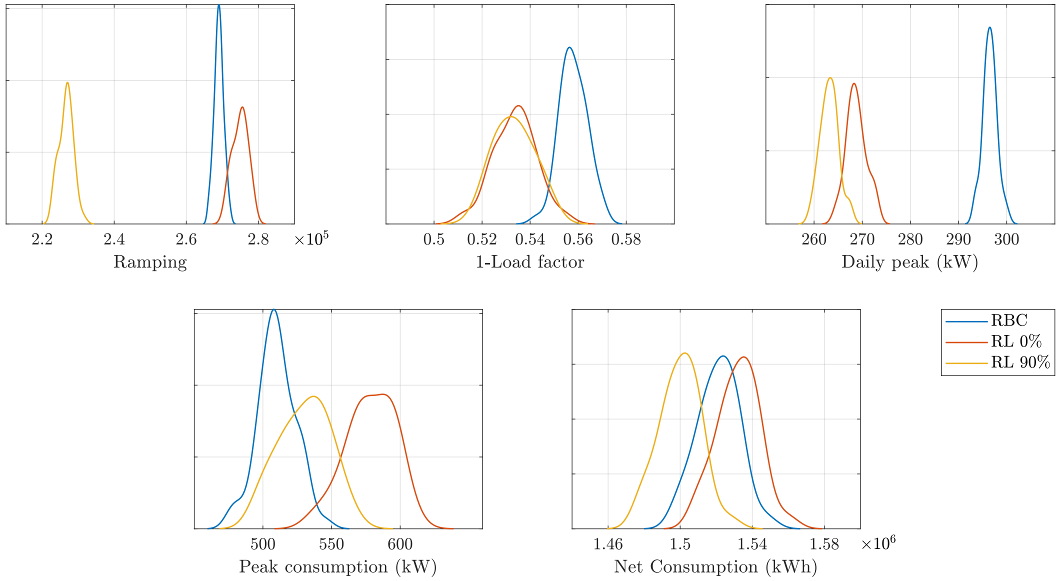

Figure 7: Contrasting the performance of 90% data-enhanced RL agent against a conventionally trained RL agent with no data enhancement and a rule-based controller

data. Among the five metrics, only the "Peak consumption" displays no signs of improvement and occasionally shows a slight degradation when compared to no data enhancement. While the "Ramping" metric also shows a general pattern of smaller variance with higher data enhancement, the trend is again non-monotonic, especially when the data enhancement is very small (5% - 10%). The effect of data enhancement on the variance of "1-Load factor", "Daily peak", and "Net consumption" is extremely small and may be considered negligible. While it is observed that data enhancement can occasionally reduce the variance of "Peak consumption", no specific correlations are observed between the portion of data enhancement and the variance in the RL agent's response to disturbances. Despite

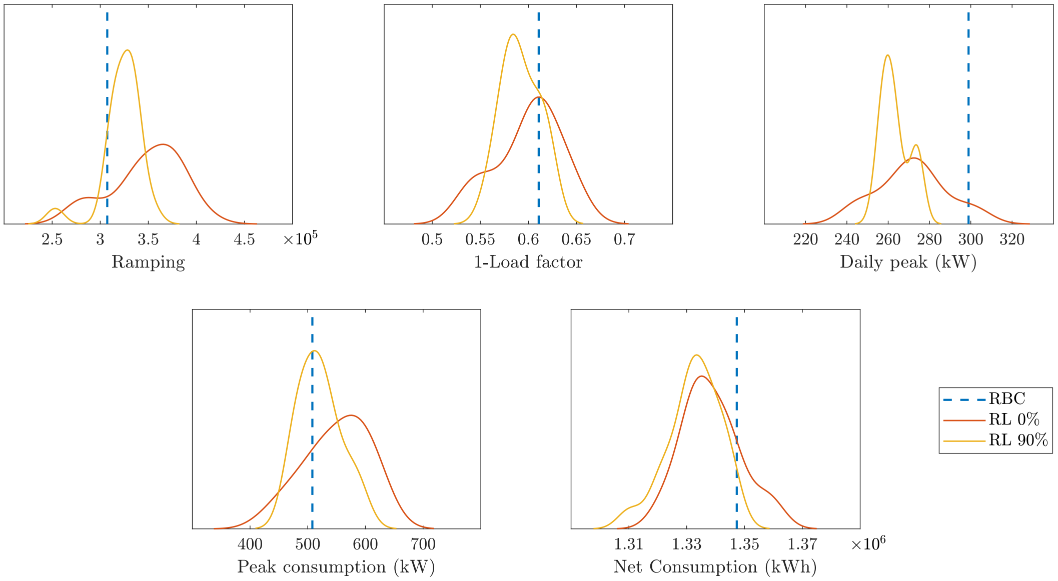

Figure 8: Contrasting the performance of 90% data-enhanced RL agent against a conventionally trained RL agent with no data enhancement and a rule-based controller on a faulty dataset.

the non-monotonic performance of data-enhanced RL agents, the overall pattern points to improvements in the studied metrics, be it at dissimilar rates. We believe that a systematic study on the data replacement techniques would shed additional light on the contribution of the proposed GAN-based data-enhancement for improving the performance and robustness of RL agents.

To highlight the effectiveness of the proposed method in a clearer manner, we select one data-enhanced RL agent and contrast its performance against that of the RBC, as well as a conventional RL with no data enhancement (Figure 7). Specifically, we opt for an RL agent that is trained on 90% of enhanced data, as it shows the best overall performance in all five metrics. The data-enhanced RL surpasses the conventional RL with no data enhancement in all performance metrics. The data-enhanced RL also surpasses the rule-based controller in most metrics, with the exception of the "Peak consumption", in which the difference between the performance of the two models is relatively small. On the other hand, the variance of the distributions underline that resorting to a data-enhanced RL would not necessarily reduce the variance of responses to low-probable high-impact disturbances, when compared to a conventional RL with no data enhancement. One exception is the "Ramping" performance metric, in which the data-enhance RL returns a smaller variance when compared to a conventional training of a RL agent.

Given that the test data set are perturbed projections from another generative model, we resort to a secondary test set (Figure 8), in which the original training data is infilled with unseen disturbances. This time we mimic encountering missing data as a result of faulty sensors. The probability of encountering missing sensors is detailed in [20]. We replace the original data in sequences from a single missing entry to a sequence of 24 missing values (one day). Regardless of the time and date of the faulty sensor events, missing data are replaced with a nominal value according to EnergyPlus protocols [7]. The rule-base controller, conventionally trained RL with no data-enhancement, and a trained RL with 90% of data enhancement are deployed on the new test set. The data-enhanced RL outperforms the conventional RL with no data enhancement in all performance metrics. The data-enhanced RL also returns performances with a narrow distribution than the conventional RL, which highlights the robustness of the model to unseen disturbances. The data-enhanced RL also returns comparable or better performance than the RBC in all metrics, with the exception of "Ramping."

4.2 Limitations

Despite the promising results of this study, several questions remain open. We briefly comment them here in the hope to fuel novel research directions.

(1) Stronger conclusions could be drawn in the presence of a more variegate validation dataset affected by stronger disturbance. It is therefore advisable to corroborate the preliminary result we got on a simulated faulty dataset with a real-world experimental campaign.
(2) Given that both the GAN and the RL model require extensive training data for proper generalization, an investigation on the minimum length of the training dataset that yields a given level of robustness is an interesting future direction.
(3) The current data-enhancement procedure is susceptible to the random replacement of the original data with synthetic ones. This randomness could potentially explain the non-monotonic behaviour of performance metrics as a function

of the percentage of enhanced data, making a univocal interpretation of the results a complicated task which requires further investigations.

5 CONCLUSIONS

This paper proposes a novel GAN architecture, here dubbed conditional TimeGAN (cTimeGAN), and evaluate its usage to enhance the performance of a RL agent in the context of building energy management. By design, the conditional TimeGAN is able to learn temporal dynamics, while associating projections with conditional features. When compared with the state-of-the-art cGAN on a highly-seasonal dataset such as building energy performance, the conditional TimeGAN is found to produce more coherent synthetic profiles. Enhancing the training dataset of a RL agent with synthetic projections is found to generally improve the agent's performance when compared to the non-enhanced counterpart and to produce more satisfactory results of a rule-based controller when applied in out-of-sample tests. Nonetheless, further investigation is required to corroborate such findings. Overall, the inclusion of synthetic profiles in the training dataset influences the learnt policies in several ways: on one side, they include reliance on operation- related and climate-related profiles (thus, paving the way to the transfer of pre-trained RL models to buildings with different operational and weather characteristics); on the other side, they are less susceptible to overfitting and more robust to unseen events due to the infusion of disciplined disturbance.

REFERENCES

[1] 2021. *Synthetic building performance data.* https://doi.org/10.5281/zenodo.4696060

[2] 2021. Synthetic Energy and Environment Replicator. https://github.com/Khayatian/seer.

[3] Gaby Baasch, Guillaume Rousseau, and Ralph Evins. 2021. A Conditional Generative adversarial Network for energy use in multiple buildings using scarce data. *Energy and AI* 5 (2021), 100087.

[4] Shane Barratt and Rishi Sharma. 2018. A Note on the Inception Score. https://doi.org/10.48550/ARXIV.1801.01973

[5] Felix Bünning, Benjamin Huber, Philipp Heer, Ahmed Aboudonia, and John Lygeros. 2020. Experimental demonstration of data predictive control for energy optimization and thermal comfort in buildings. *Energy and Buildings* 211 (2020), 109792.

[6] Zhenghua Chen and Chaoyang Jiang. 2018. Building occupancy modeling using generative adversarial network. *Energy and Buildings* 174 (2018), 372–379. https://doi.org/10.1016/j.enbuild.2018.06.029

[7] Drury B. Crawley, Linda K. Lawrie, Frederick C. Winkelmann, W.F. Buhl, Y.Joe Huang, Curtis O. Pedersen, Richard K. Strand, Richard J. Liesen, Daniel E. Fisher, Michael J. Witte, and Jason Glazer. 2001. EnergyPlus: creating a new-generation building energy simulation program. *Energy and Buildings* 33, 4 (2001), 319–331. https://doi.org/10.1016/S0378-7788(00)00114-6 Special Issue: BUILDING SIMULATION'99.

[8] Cees Diks and Valentyn Panchenko. 2006. A new statistic and practical guidelines for nonparametric Granger causality testing. *Journal of Economic Dynamics and Control* 30, 9 (2006), 1647–1669. https://doi.org/10.1016/j.jedc.2005.08.008 Computing in economics and finance.

[9] Cheng Fan, Yongjun Sun, Yang Zhao, Mengjie Song, and Jiayuan Wang. 2019. Deep learning-based feature engineering methods for improved building energy prediction. *Applied Energy* 240 (2019), 35–45. https://doi.org/10.1016/j.apenergy.2019.02.052

[10] P.M. Ferreira, A.E. Ruano, S. Silva, and E.Z.E. Conceição. 2012. Neural networks based predictive control for thermal comfort and energy savings in public buildings. *Energy and Buildings* 55 (2012), 238–251.

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger (Eds.), Vol. 27. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf

[12] Yuxuan Gu, Qixin Chen, Kai Liu, Le Xie, and Chongqing Kang. 2019. GAN-based Model for Residential Load Generation Considering Typical Consumption Patterns. In *2019 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*. 1–5.

[13] Jie Gui, Zhenan Sun, Yonggang Wen, Dacheng Tao, and Jieping Ye. 2020. A Review on Generative Adversarial Networks: Algorithms, Theory, and Applications. *CoRR* abs/2001.06937 (2020). https://arxiv.org/abs/2001.06937

[14] Tianzhen Hong, Zhe Wang, Xuan Luo, and Wanni Zhang. 2020. State-of-the-art on research and applications of machine learning in the building life cycle. *Energy and Buildings* 212 (2020), 109831.

[15] Tianzhen Hong1, Daniel Macumber, Han Li, Katherine Fleming, and Zhe Wang. 2020. Generation and representation of synthetic smart meter data. In *proceedings of Building Simulation Conference*, Vol. 13. https://doi.org/10.1007/s12273-020-0661-y

[16] Achin Jain, Francesco Smarra, Madhur Behl, and Rahul Mangharam. 2018. Data-Driven Model Predictive Control with Regression Trees—An Application to Building Energy Management. *ACM Trans. Cyber-Phys. Syst.* 2, 1, Article 4 (2018), 21 pages.

[17] Fazel Khayatian, Zoltán Nagy, and Andrew Bollinger. 2021. Using generative adversarial networks to evaluate robustness of reinforcement learning agents against uncertainties. *Energy and Buildings* 251 (2021), 111334. https://doi.org/10.1016/j.enbuild.2021.111334

[18] Yann LeCun and Yoshua Bengio. 1998. *Convolutional Networks for Images, Speech, and Time Series.* MIT Press, Cambridge, MA, USA, 255–258.

[19] Ping Lu, Matt Morris, Seth Brazell, Cody Comiskey, and Yuan Xiao. 2018. Using generative adversarial networks to improve deep-learning fault interpretation networks. *The Leading Edge* 37, 8 (2018), 578–583.

[20] Maryam MeshkinKiya and Riccardo Paolini. 2020. Preparing Weather Data for Real-Time Building Energy Simulation. https://doi.org/10.48550/ARXIV.2011.09733

[21] Mehdi Mirza and Simon Osindero. 2014. Conditional Generative Adversarial Nets. https://arxiv.org/abs/1411.1784

[22] Olof Mogren. 2016. C-RNN-GAN: Continuous recurrent neural networks with adversarial training. *CoRR* abs/1611.09904 (2016). http://arxiv.org/abs/1611.09904

[23] Yue Pang, Xiangdong Zhou, Donghui Xu, Zijing Tan, Mingxi Zhang, Naiwang Guo, and Yingjie Tian. 2019. Generative Adversarial Learning Based Commercial Building Electricity Time Series Prediction. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. 1800–1804. https://doi.org/10.1109/ICTAI.2019.00271

[24] Eitan Richardson and Yair Weiss. 2018. On GANs and GMMs. *CoRR* abs/1805.12462 (2018). arXiv:1805.12462 http://arxiv.org/abs/1805.12462

[25] Jonathan Roth, Amory Martin, Clayton Miller, and Rishee K. Jain. 2020. SynCity: Using open data to create a synthetic city of hourly building energy estimates by integrating data-driven and physics-based methods. *Applied Energy* 280 (2020), 115981.

[26] Adarsh Subbaswamy, Roy Adams, and Suchi Saria. 2021. Evaluating Model Robustness and Stability to Dataset Shift. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 130)*, Arindam Banerjee and Kenji Fukumizu (Eds.). PMLR, 2611–2619.

[27] S. S. Vallender. 1974. Calculation of the Wasserstein Distance Between Probability Distributions on the Line. *Theory of Probability & Its Applications* 18, 4 (1974), 784–786. https://doi.org/10.1137/1118101 arXiv:https://doi.org/10.1137/1118101

[28] Jose R. Vazquez-Canteli, Gregor Henze, and Zoltan Nagy. 2020. MARLISA: Multi-Agent Reinforcement Learning with Iterative Sequential Action Selection for Load Shaping of Grid-Interactive Connected Buildings. In *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation* (Virtual Event, Japan) *(BuildSys '20)*. Association for Computing Machinery, New York, NY, USA, 170–179.

[29] José R. Vázquez-Canteli, Jérôme Kämpf, Gregor Henze, and Zoltan Nagy. 2019. 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* (New York, NY, USA) *(BuildSys '19)*. Association for Computing Machinery, New York, NY, USA, 356–357.

[30] Zhe Wang and Tianzhen Hong. 2020. Generating realistic building electrical load profiles through the Generative Adversarial Network (GAN). *Energy and Buildings* 224 (2020), 110299. https://doi.org/10.1016/j.enbuild.2020.110299

[31] Ronald J. Williams and David Zipser. 1989. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. 1, 2 (1989). https://doi.org/10.1162/neco.1989.1.2.270

[32] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. 2019. Time-series Generative Adversarial Networks. In *Advances in Neural Information Processing Systems*, Vol. 32. Curran Associates, Inc.