Energy and AI 20 (2025) 100509

Contents lists available at ScienceDirect

Energy and AI

journal homepage: www.elsevier.com/locate/egyai

Advanced day-ahead scheduling of HVAC demand response control using novel strategy of Q-learning, model predictive control, and input convex neural networks☆

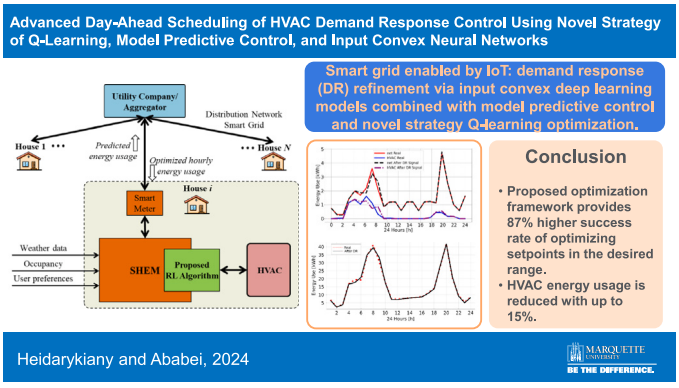
Rahman Heidarykiany *, Cristinel Ababei

MESS Laboratory, ECE Department, Marquette University, Milwaukee, WI, USA

HIGHLIGHTS

- ICLSTM models HVAC and indoortemperature dynamics; embedded in convex, tractable closed-loop MPC for multi-shot, energy-optimal scheduling.
- ICLSTM-MPC is wrapped by epsilon-greedy Q-learning in SHEM; a dynamic Q-table refines DR signals, balancing exploration and exploitation for optimal rewards.
- Simulations show the framework boosts success by 87 %+, cuts HVAC energy up to 15 %, with zero indoor-comfort loss.

GRAPHICAL ABSTRACT



ARTICLE INFO

Keywords:
Model-free reinforcement learning (MF-RL)
Q-learning
Input convex long short-term memory networks (ICLSTM)
Model predictive control (MPC)
Control of nonlinear physical systems
Thermal comfort
HVAC energy usage control
Thermostatically controlled loads
Smart home energy management (SHEM)
Load shifting
Internet of things (IoT) applications
Smartgrid
Virtual power plant (VPP)
Microgrid
Deep learning (DL)
Demand-side management (DSM)
Demand response (DR)

ABSTRACT

In this paper, we present a Q-Learning optimization algorithm for smart home HVAC systems. The proposed algorithm combines new convex deep neural network models with model predictive control (MPC) techniques. More specifically, new input convex long short-term memory (ICLSTM) models are employed to predict dynamic states in an MPC optimal control technique integrated within a Q-Learning reinforcement learning (RL) algorithm to further improve the learned temporal behaviors of nonlinear HVAC systems. As a novel RL approach, the proposed algorithm generates day-ahead HVAC demand response (DR) signals in smart homes that optimally reduce and/or shift peak energy usage, reduce electricity costs, minimize user discomfort, and honor in a best-effort way the recommendations from utility/aggregator, which in turn has impact on the overall well being of the distribution network controlled by the aggregator. The proposed Q-Learning optimization algorithm, based on epsilon-model predictive control (ϵ-MPC), can be implemented as a control agent that is executed by the smart house energy management (SHEM) system that we assume exists in the smart home, which can interact with the energy provider of the distribution network, i.e., utility/aggregator, via the smart meter. The output generated by the proposed control agent represents day-ahead local DR signals in the form of temperature setpoints for the HVAC system that are found by the optimization process to lead to desired trade-offs between electricity cost and user discomfort. The proposed algorithm can be used in smart homes with passive HVAC controllers, which solely react to end-user setpoints, to transform them into smart homes with active HVAC controllers. Such systems not only respond to the preferences of the end-user but also incorporate an external control signal provided by the utility or aggregator. Simulation experiments

☆ This research was financially supported by an award from the National Science Foundation, USA grant ECCF 1936494..
* Corresponding author.
E-mail addresses: rahman.heidarykiany@marquette.edu, rahman.heidary.kiany@gmail.com (R. Heidarykiany).

https://doi.org/10.1016/j.egyai.2025.100509
Received 9 July 2024; Received in revised form 25 February 2025; Accepted 26 March 2025
Available online 15 April 2025

conducted with a custom simulation tool demonstrate that the proposed optimization framework can offer significant benefits. It achieves 87% higher success rate in optimizing setpoints in the desired range, thereby resulting in up to 15% energy savings and zero temperature discomfort.

1. Introduction

Buildings account for 40% of total electricity energy usage [1]. Therefore, it is important for both building owners and energy providers to optimize energy usage in a way that benefits both parties. That is, building owners can benefit from reduced costs achieved through a reduction and/or shifting of energy usage without degradation of their thermal comfort. Energy providers benefit from improving distribution network operation through better energy efficiency, grid stability, environmental sustainability, etc. One of the most promising approaches to achieve combined distribution network plus buildings optimization is through the smart grid approach. More specifically, demand-side management (DSM) can be implemented by energy providers (i.e., utility/aggregator) that can solve distribution network level day-ahead optimization problems, whose solutions translate into communication with individual smart home energy management (SHEM) systems. SHEMs of all buildings receive DR signals from utility/aggregator that specify for example offered energy usage schedules, peak reduction and/or shifting, etc. Locally, at the level of each building, SHEMs further fine-tune such DR signals in a way that takes into consideration the home user's preferences that can be specified via a smart phone app or in-house graphics interface. Finally, the fine-tuned DR signals translate into actual controls of the HVAC system, thereby achieving a trade-off between the energy usage scheduling offered by the utility/aggregator and user's preferences.

Many control methods have been proposed to improve the operational efficiency of HVAC systems and to address the need for efficient and effective techniques in home energy management. However, these methods often suffer from two limitations: the requirement for detailed physics-based models, which can take a long time to develop, and the use of linear control algorithms, which may not perform well in highly nonlinear building dynamics [2]. To address these challenges, data-driven machine deep learning techniques emerged and seem promising [3], enabled by the increasing availability of smart meters and internet of things (IoT) technologies that allow the collection of vast amounts of fine-grained data on smart homes. These techniques can construct models with minimal assumptions, making them effective in handling uncertainties in data and models. The work presented in this paper is in this category of approaches.

2. Related work

In this section, we review previous literature related to different models and methods that correspond to different elements that we combine in the proposed approach. We split the review into several sections specific to those elements for a better readability.

2.1. Methods for system identification of nonlinear systems

System identification methods develop mathematical models of dynamical systems from measured input and output data. One of the objectives for these models is to minimize the prediction error of nonlinear dynamic behaviors. Such models are then used to design controllers that regulate system outputs based on the inputs. Most efficient model-based control algorithms use simple function estimators for system dynamics identification, such as linear models [4] and Gaussian processes [5]. Despite being efficient, such models may suffer from insufficient representation capacity to model large-scale or high-dimension nonlinear systems. This challenge can be addressed by employing machine learning models for system identification, such as neural networks (NNs) [6], which can be good at capturing or learning nonlinear behavior. Such models have been motivated also by the large amounts of operational data that have become available with the increasing deployment of sensors in physical systems, particularly in smart buildings [7]. Thus, deep neural networks (DNNs) have emerged as a popular method for parameterizing complex physical system dynamics and capturing complex relationships [8,9]

Several neural network architectures for system identification were investigated in [10] for developing improved MPC techniques. A feed-forward network was used for system identification in [11]. As another example, recurrent neural networks (RNNs) were employed to develop an implicit model structure, which enabled convex parameterization [12]. The study in [13] combined RNNs with traditional Kalman filters to develop a nonlinear system identification method. As yet another ML model, the study in [14] used a cluster of convex-based long short term memory (LSTM) models for identification of a nonlinear series-parallel system. The study in [15] also used LSTM models for time-varying and parameterized time-varying inputs. Deep autoencoders for identification of nonlinear state-space models for dimensionality reduction and neural networks for learning direct acyclic computational graphs for dimensionality reduction were presented in [16]. In the study from [17], deep learning was used to identify black-box nonlinear systems. Constrained block nonlinear neural state space models are used to identify nonlinear dynamical systems in [18]. An approach to identifying nonlinear systems based on state-space deep neural networks (SS-NNs) was proposed in [19]. Generally, such machine learning models pose their own challenges, particularly related to their non-convex nature [20,21]. Quadratic constraints were used to represent nonlinear specifications and activation functions in [22], in order to arrive at convex inner approximations to all acceptable parameters and their non-convex sets.

2.2. Controllers for nonlinear systems

Despite the emergence of many machine learning (ML) models for system modeling and identification, few studies explored how to integrate such deep learning models into closed-loop control strategies of physical systems. That may be in part due to the significant challenge posed by the non-convex nature of such models. On the other hand, linear models can be computationally tractable, but, they often have poor fitting performance. To address this tradeoff between modeling accuracy and control tractability, the study in [23] proposed a solution that leverages input convex neural networks (ICNN) to learn system dynamics and to determine optimal control policies. They demonstrated that ICNN can achieve classification results comparable to traditional neural networks for inference and prediction problems. More specifically, the study introduced fully input convex neural network (FICNN) and partially input convex neural network (PICNN) architectures. By ensuring convexity from input to output, optimization problems remain tractable computationally while achieving high prediction accuracies.

The work in [23] was extended by [24] to input convex recurrent neural networks (ICRNNs), which can perform one-shot multi-step ahead predictions. Therefore, this approach enables prediction of a sequence of outputs using a sequence of inputs in a single step. The authors employed this approach to develop optimal control methods for energy systems. Their method utilizes an updated input convex neural network model to learn the system dynamics and then computes the best control decisions via solving a convex MPC problem, which is tractable and has global optimality guarantees. While the approach from [23] only provides convexity for one-step predictions, the study in [25] proposed a further extension for multi-shot multi-step predictions.

2.3. ML models and optimal model predictive control (MPC)

Model predictive control (MPC) techniques have been used to control temperature and energy consumption in buildings [26,27]. Also, machine learning models started to be increasingly used in MPC techniques. For example, to achieve precise trajectory tracking of piezoelectric actuators, [28] presented a linear MPC technique that employed LSTM neural networks. ResNet was used for non-linear dynamical systems identification and improved MPC in [29]. A convex BiLSTM-based controller was implemented by the study in [30] to control the dynamics of a robotic system. LSTM networks were utilized to model dynamic processes in [31]. Three different models – physics-informed ARMAX, RF, and ICNN – were used in MPC for building climate control in the study from [32], which found the ARMAX model to be the best. A neural network based MPC of HVAC systems in sports facilities was presented in [33] and reduction of energy consumption of up to 46% was reported improving thermal comfort and indoor air quality.

2.4. Reinforcement learning and MPC

Reinforcement learning may reduce training time, parameter sharing, and pretraining time. For example, a multi-agent RL framework to minimize energy consumption by optimizing HVAC control and incorporating feedback from users regarding uncomfortable room temperature was presented in [34]. The importance of domain knowledge and data preparation as the means of integrating machine learning with operations research (OR) for SHEMS control was highlighted in [35]. The study transformed SHEMS MPC implementation into an RL environment using deep deterministic policy gradients (DDPGs). In the study from [36], the Gnu-RL method was described as a practical and scalable RL method for HVAC control. Differentiable MPC was used to bootstrap the agent with domain knowledge, which allowed efficient learning from limited samples. The study in [37], proposed MBRL-MPC as an integrated control strategy that combined model-based deep reinforcement learning (DRL) and MPC of HVAC systems. The strategy involved learning a thermal dynamic model through supervised learning and employing a neural network planning framework based on RL and MPC principles. A two-stage RL policy search method with expedited convergence was proposed in [38]. More examples and reviews of recent advances in data-driven MPC and RL-based control algorithms for building energy management systems (BEMS) were presented in [39,40].

3. Contributions of this work

We propose a novel combination of data-driven deep learning models with model predictive control (MPC) to improve the learning process and control of nonlinear physical systems such as HVAC systems. More specifically, input convex long short-term memory (ICLSTM) models are employed to predict dynamic states in an MPC optimal control technique integrated within a Q-Learning model-free deep reinforcement learning (MF-DRL) algorithm. The output of the proposed algorithm is in the form of demand response (DR) signals as heating and cooling setpoints for the optimal control of HVAC systems. The proposed approach addresses limitations of previous RL approaches that have been ineffective at controlling nonlinear physical systems with large inertia such as HVAC systems.

A simplified system level diagram to describe the context of the proposed framework is presented in Fig. 1. We assume that optimization is done in a day-ahead manner. The utility/aggregator in Fig. 1 generates optimized hourly energy usage schedules for the houses in the distribution network. Such schedules are generated using a game theory approach, which we described in our recent study [41]. The SHEM of each house in the distribution network receives its schedule. The proposed control algorithm is run as an agent inside the SHEM. It takes as input the hourly energy usage schedule as well as weather data (via WiFi) and home user preferences through an app or GUI interface. It then produces refined control signals for the HVAC system. Our optimization approach is different from existing approaches that rely on model-free or model-based end-to-end controllers. We present a novel model-free Q-Learning optimization approach that builds upon and improves the already enhanced model-based MPC technique. As a result, our approach exhibits much better computational tractability and convexity, which facilitates the derivation of an optimal control sequence based on the energy usage offered by the energy provider. The innovation of this article lies in the formulation and integration of new Input Convex Long Short-Term Memory (ICLSTM) models, which are employed to predict dynamic states in a model predictive control technique. This is integrated within a novel strategy of Q-Learning reinforcement learning framework that incorporates the concept of epsilon-model predictive control (ϵ-MPC). To this end, our main contributions are as follows:

1. We develop novel input convex long short-term memory (ICLSTM) networks to learn the nonlinear dynamics model of a building's HVAC system and of indoor temperature. These models are integrated with a closed-loop MPC to optimize HVAC energy usage scheduling as a multi-shot multi-step process. The ICLSTM-based MPC is specially-designed to be convex and computationally tractable.
2. The ICLSTM-based MPC control is further integrated within a Q-learning algorithm (which is a model-free learning algorithm for solving RL problems), which further improves the output from the MPC control to generate in this way refined DR signals for the local HVAC system. The proposed Q-learning algorithm relies at its core on a Q-table that is dynamically learned, inside the custom simulation tool that we developed for this purpose. The implementation of the proposed Q-Learning algorithm employs the concept of ϵ-greedy policy to construct an effective ϵ-MPC policy, where the agent (again, in our case it resides inside the SHEM) can look-up the table to: (1) either randomly select an action with probability ϵ to explore the environment for learning, (2) or select the greedy action (in our case MPC) to optimize reward. This policy effectively balances exploration of new actions and exploitation of the best-known or optimal actions captured into the current Q-values.
3. We present simulation experiments that demonstrate that the proposed optimization framework achieve more than 87% higher success rate, resulting in up to 15% energy savings and zero temperature discomfort (see Table 1).

4. How proposed approach is different from existing methods

The main difference between the proposed ICLSTM-based MPC Q-Learning method and existing research lies in its unique integration of several key components that collectively enhance HVAC control performance. We highlight below several key differences:

1. Handling Nonlinear HVAC Dynamics: ICLSTM networks are specifically designed to model the nonlinear dynamics of HVAC systems with large inertia. Existing methods often rely on linear models or basic Neural Networks (NNs), which struggle to accurately capture these complex dynamics. The ICLSTM models facilitate system identification using data-driven approaches, reducing reliance on time-consuming physics-based model development required by many existing HVAC control systems.
2. Convex Neural Network Integration Control: The proposed ICLSTM-based MPC method employs Input Convex Neural Networks (ICNNs), ensuring convexity in the mapping from inputs to outputs. ICNN-based MPC guarantees global optimality and

Table 1
Comparative table that contrasts proposed optimization algorithm it with existing models.

Attribute	ICLSTM- model-base RL-model-free RL model (Proposed)	LSTM- model-free RL models	Model-base RL models	ICRNN- model based RL models
Model structure (Control strategy)	Input convex LSTM (ICLSTM) based MPC with Q-learning	LSTM with free reinforcement learning	Physics-based Model Predictive Control (MPC)	Input Convex Recurrent Neural Network (ICRNN) based MPC
Convexity of model	Convexity through ICLSTM	Non-convex, leading to local minima	Convex under linear assumptions	Limited convexity
Adaptability to Nonlinear systems	High, specifically designed for nonlinear (e.g., HVAC) dynamics	Moderate adaptability	Low adaptability for complex, nonlinear systems	Moderate, depending on complexity
Computational Efficiency	Improved due to convex structure of ICLSTM	Slower/local convergence, higher computational cost	High for linear systems, low for nonlinear systems	Moderate, depending on complexity
Dynamic Adjustment	High, Epsilon-greedy (Epsilon-MPC) Q-learning for high adjustment	Slow adaptation due to very large number of interactions nature	Typically static or predefined	Moderate adjustment capability
User comfort optimization	Balances user comfort and cost	Basic trade-offs with limited flexibility	User comfort constraints may be fixed	Limited focus on user comfort
Handling system inertia	Optimized for large inertia systems like HVAC	Limited effectiveness in high-inertia environments	Not effective in high-inertia settings	Moderate effective
Demand Response (DR) signal integration	Integrated within MPC-Q learning feasible sets	Not directly integrated, relies on Model free exploration	Requires predefined DR settings	Limited DR adaptability
Energy savings & Success rate	Up to 15% energy savings with 87% success rate	Lower energy savings due to convergence issues	Typically lower due to limited DR settings adaptability	Moderate savings depending on complexity and conditions

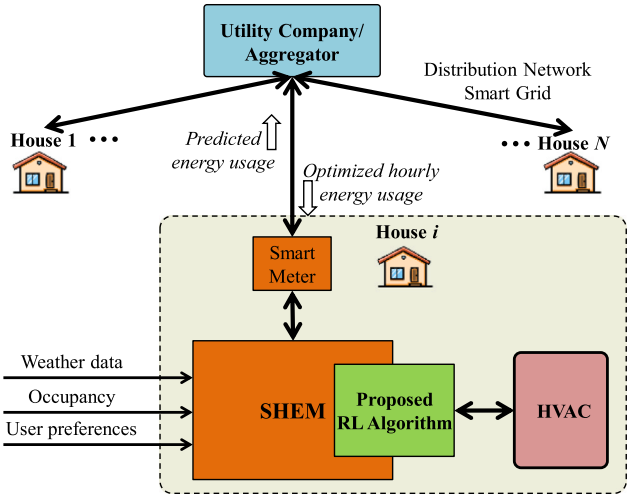


Fig. 1. System level diagram that illustrates that the proposed algorithm for optimization of day-ahead energy usage scheduling for HVAC resides inside the house SHEM.

computational efficiency (computational tractability) in the optimization process, unlike traditional non-convex Neural Networks (NNs) RL models that may lead to suboptimal solutions and increased computational complexity.

3. Integration with Model Predictive Control (MPC): By combining new formulation of ICLSTM with MPC, the convex optimization problem enables multi-step, multi-shot predictions and control over extended horizons. This holistic approach contrasts with existing methods that typically perform simpler, one-step predictions, limiting their ability to optimize energy use effectively over time.
4. Incorporation of model-free Q-Learning and model-based MPC for Reinforcement Learning: The proposed method integrates a model-free reinforcement learning algorithm—Q-learning with an epsilon-MPC (epsilon-greedy) policy to refine/update optimized decisions (MPC decisions). This integration balances exploration and exploitation, enhancing adaptability to dynamic and uncertain environments. Existing methods often lack this hybrid mechanism, reducing their effectiveness in adaptation.
5. Advanced Demand Response (DR) Integration: This paper approach incorporates DR signals coming from higher level (i.e., offers form aggregator in the distribution system) into the local control framework, optimizing energy costs while maintaining user comfort in each house. To the best of our knowledge, we are not aware of any previous approach that focuses on models integrating DR signals from higher level (i.e., aggregator) within a local reinforcement learning strategy to this extent. Implemented as a local agent within each smart home's energy management system, the method allows for decentralized optimization while coordinating with global objectives (offer form aggregator distribution system). This is a departure from prior research that focuses on centralized control without efficient local–global coordination.
6. Demonstrated Superior Performance: Simulation results demonstrate that the proposed method significantly improves the optimization of HVAC setpoints while ensuring zero user discomfort, outperforming prior approaches. This highlights its practical effectiveness over existing methods.

These key differences set method proposed in this paper apart from existing research. In addition to these distinctions, we also note the following points drawn from comparisons with recently published works. While [42] employs a deterministic actor-critic RL coupled with MPC to manage Home Energy Management Systems (HEMS), it primarily focuses on handling real-time energy consumption with limited day-ahead coordination. By contrast, our method integrates Q-Learning with ICLSTM-based MPC for day-ahead scheduling, providing dynamic DR signals from higher-level aggregators while ensuring zero discomfort. [43] highlights the power of convex neural networks (ICNNs) in modifying cost functions for improved MPC robustness and stability. Our approach similarly leverages convex properties through ICLSTM networks yet extends the concept by embedding a model-free Q-Learning algorithm. This combination addresses not only local optimization needs but also enhances adaptability under uncertain DR signals and large HVAC inertia an aspect not fully covered by

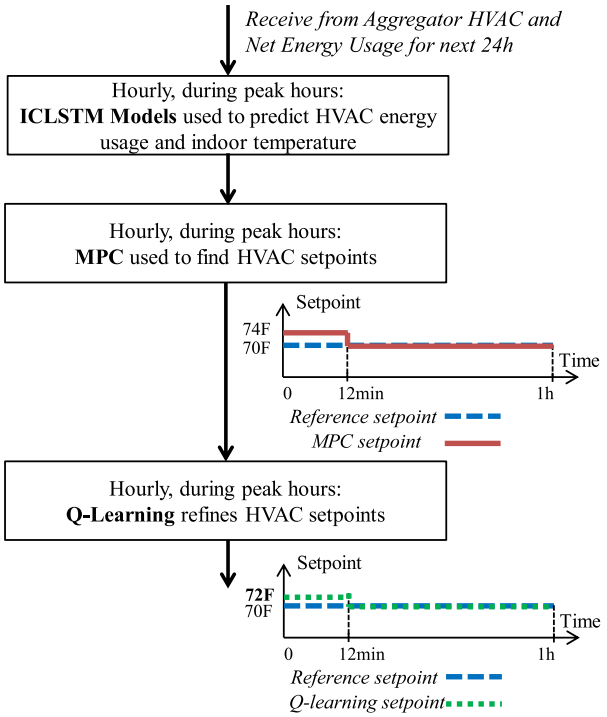


Fig. 2. Block diagram that shows the main steps of the proposed optimization algorithm.

purely economic MPC cost modifications. [44] introduces a data-driven Economic NMPC formulation using RL to optimize cost objectives in broader domains. Our method is narrower in scope HVAC demand response but benefits from a hybrid design (model-based MPC plus model-free Q-Learning) that yields effective handling of building inertia. The result is an approach suited to local comfort constraints while achieving superior energy efficiency over purely data-driven economic NMPC. [45] extends MPC-based RL to manage multi-household microgrids, using Shapley value for cooperative cost sharing. In contrast, our framework emphasizes a single smart home's HVAC control while allowing DR signals to be integrated in a decentralized manner. This provides a balance: local user comfort is secured with global (aggregator) DR requirements still respected crucial in aligning with aggregator offers for residential demand response programs. Where [46] employs RNN-based system identification for general, possibly high-dimensional, nonlinear control, our approach maintains computational tractability by using Input Convex LSTM. This ensures a convex optimization structure within MPC, leading to guaranteed global solutions specific to HVAC dynamics with large inertia. [47] surveys reinforcement learning approaches for MPC, discussing theoretical concerns such as stability, constraint satisfaction, and exploration strategies. Our work, however, demonstrates an *applied* solution Q-Learning with MPC and ICLSTM showing real-world viability for HVAC demand response, bridging the gap between theoretical RL-MPC discussions and practical implementations. Although [48] proposes robust MPC to enforce strict safety constraints in reinforcement learning, our method focuses primarily on ensuring occupant comfort and energy efficiency in an uncertain DR environment rather than guaranteeing robustness against all forms of disturbances. Nevertheless, we address partial robustness by employing convex learning models, reducing the risk of infeasible control actions.

5. Proposed optimization algorithm

In this section, we describe the proposed optimization framework, which relies on a new deep machine learning model that we introduce

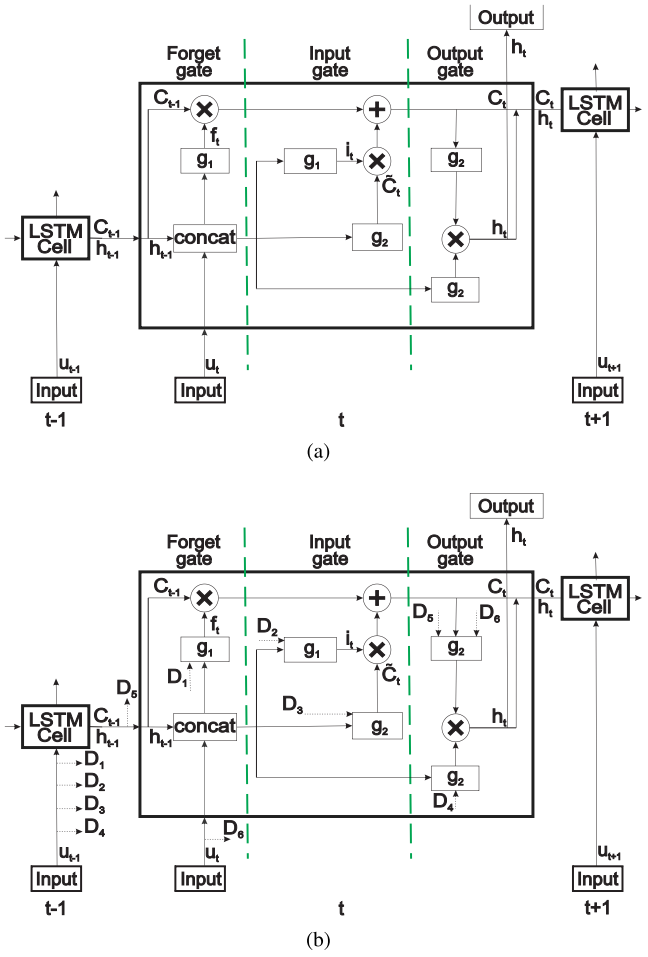


Fig. 3. (a) Diagram of traditional LSTM cell. (b) Diagram of proposed input convex LSTM cell.

(to learn indoor temperature and HVAC energy usage dynamics in the house), which then is used to formulate the convex model predictive control problem that then will be framed in Q-Learning reinforcement learning algorithm. The objective of the proposed algorithm is to find the best control decisions (setpoints) for the HVAC system, starting from the day-ahead energy use schedule offered by the utility/aggregator and under user set preferences and actual outdoor temperature hourly. A block diagram illustrating the main steps involved in the proposed algorithm is shown in Fig. 2. In the next sections, we discuss each of the three main components shown in this figure.

5.1. Novel input convex LSTM

We introduce here the proposed input convex LSTM model. Our contribution is to extend the ICRNN ideas presented in [24] and the FICNN ideas from [25] to the case of LSTM models, to arrive at input convex LSTM models. These models will be used for predicting the house's indoor temperature and HVAC energy usage. To better explain the proposed model, we show in Fig. 3 side by side the block diagrams of the traditional LSTM model and the proposed input convex LSTM model cells.

The equations that govern the operation of the traditional LSTM cell are as follows [49]:

$$\mathbf{f}_t = g_1(\mathbf{W}_f \hat{\mathbf{u}}_t + \mathbf{U}_f \mathbf{h}_{t-1}) \tag{1}$$

$$\mathbf{i}_t = g_1(\mathbf{W}_i \hat{\mathbf{u}}_t + \mathbf{U}_i \mathbf{h}_{t-1}) \tag{2}$$

$$\mathbf{o}_t = g_1(\mathbf{W}_o\hat{\mathbf{u}}_t + \mathbf{U}_o\mathbf{h}_{t-1}) \tag{3}$$

$$\widetilde{\mathbf{c}}_t = g_2(\mathbf{W}_c\hat{\mathbf{u}}_t + \mathbf{U}_c\mathbf{h}_{t-1}) \tag{4}$$

$$\mathbf{c}_t = \mathbf{f}_t \otimes \mathbf{c}_{t-1} + \mathbf{i}_t \otimes \widetilde{\mathbf{c}}_t \tag{5}$$

$$y_t = g_2(\mathbf{c}_t) \otimes \mathbf{o}_t \tag{6}$$

where $\hat{\mathbf{u}}_t$ is the input vector to the memory cell at time t. $\mathbf{W}_f, \mathbf{W}_i, \mathbf{W}_o, \mathbf{W}_c, \mathbf{U}_f, \mathbf{U}_i, \mathbf{U}_o, \mathbf{U}_c$ are weight matrices. h_{t-1} represents the hidden state at time $t-1$ of the LSTM cell. c_{t-1} is the cell state at time $t-1$. g_1 and g_2 are the activation and recurrent activation functions. f_t, i_t and o_t are values of the input gate, the forget gate, and the output gate at time t. $\widetilde{\mathbf{c}}_t$ based on the input, a new candidate cell state is computed. $\mathbf{c}_t$ represents the updated cell state, obtained by the combination of the old state of the cell (scaled by the forget gate) with the new state of the cell (scaled by the input gate).

Following the same procedure as in [24,25], we modify the traditional LSTM cell to include both $\mathbf{u}$ and $-\mathbf{u}$. Also, we constrain the weights $\theta = \{\mathbf{W}_f, \mathbf{W}_i, \mathbf{W}_o, \mathbf{W}_c, \mathbf{U}_f, \mathbf{U}_i, \mathbf{U}_o, \mathbf{U}_c, \mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{D}_4, \mathbf{D}_5, \mathbf{D}_6\}$ to be non-negative and activation functions g_1 and g_2 to be convex and nondecreasing (e.g., ReLU). Following the same rationale as that in Proposition 1 from [24], which exploits the composition rule of convex functions [50], the newly modified LSTM cell shown in Fig. 3.b) and characterized by the following equations is an input convex LSTM cell:

$$\mathbf{f}_t = g_1(\mathbf{W}_f\hat{\mathbf{u}}_t + \mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{D}_1\hat{\mathbf{u}}_{t-1}) \tag{7}$$

$$\mathbf{i}_t = g_1(\mathbf{W}_i\hat{\mathbf{u}}_t + \mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{D}_2\hat{\mathbf{u}}_{t-1}) \tag{8}$$

$$\mathbf{o}_t = g_1(\mathbf{W}_o\hat{\mathbf{u}}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{D}_4\hat{\mathbf{u}}_{t-1}) \tag{9}$$

$$\widetilde{\mathbf{c}}_t = g_2(\mathbf{W}_c\hat{\mathbf{u}}_t + \mathbf{U}_c\mathbf{h}_{t-1} + \mathbf{D}_3\hat{\mathbf{u}}_{t-1}) \tag{10}$$

$$\mathbf{c}_t = \mathbf{f}_t \otimes \mathbf{c}_{t-1} + \mathbf{i}_t \otimes \widetilde{\mathbf{c}}_t \tag{11}$$

$$y_t = g_2(\mathbf{c}_t + \mathbf{D}_5\mathbf{h}_{t-1} + \mathbf{D}_6\hat{\mathbf{u}}_t) \otimes \mathbf{o}_t \tag{12}$$

where:

$$\theta = \{\mathbf{W}_f, \mathbf{W}_i, \mathbf{W}_o, \mathbf{W}_c, \mathbf{U}_f, \mathbf{U}_i, \mathbf{U}_o, \mathbf{U}_c, \mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{D}_4, \mathbf{D}_5, \mathbf{D}_6\} \tag{13}$$

where $\mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{D}_4, \mathbf{D}_5, \mathbf{D}_6$ represent non-negative weights of the passthrough layers and $\hat{\mathbf{u}}_{t-1}$ is the input vector to the memory cell at time $t-1$.

Multiple such cells will be chained together and the output of the last cell will be convex with respect to the input of the first cell. The model developed in this way can then be used to handle time series and make multi-shot multi-step decisions in a close-loop control. More specifically, such ICLSTM models will be used to predict the house HVAC energy usage and indoor temperature given several input features as illustrated in Fig. 4. The inputs into these models include several values from previous time steps and the predicted outputs are for several time steps into the futures. In our experiments we work with a time step granularity of 3 min, which we found to be a good compromise between model development time and performance of the proposed optimization. These models are used inside the MPC as model-based reinforcement learning (MB-RL) discussed in the next section. Details on model training datasets generation are provided later in this paper.

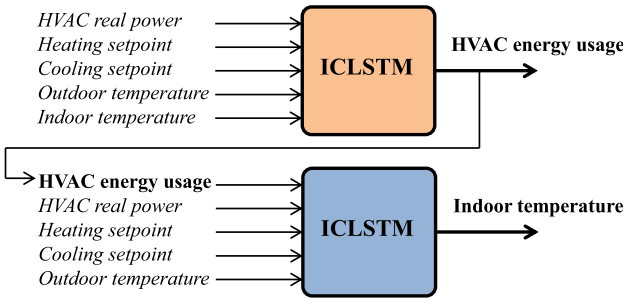


Fig. 4. System level description of the ICLSTM models to illustrate the relation between inputs and outputs. The models will perform multi-step multivariate input multi-step univariate output predictions.

5.2. Closed-loop MPC control using data-driven ICLSTM models

Generally, traditional model predictive control (MPC) entails several steps as follows. First, the current states are measured and future disturbances are obtained. Second, a constrained optimization problem based on a predictive model, time-dependent boundaries, and a predefined cost function is solved for an entire prediction horizon. Third, the optimal control input is applied to the system. These steps are then repeated periodically with a period given by the selected time-step, which is also called an epoch.

In this paper, rather than using traditional physics-based analytical models (which are difficult to develop, take time, and do not perform well), we propose to employ the input convex LSTM models from the previous section. This aligns well with the current trends of increasingly relying on data-driven ML models such as neural models, that have been proven to learn and capture rich and complex system dynamics better than physics-based models. Another advantage of these ML models is that they can also be updated frequently as new data is collected with a vast amount of IoT sensors that smart homes are equipped with. The only requirement to these models to be readily used in an MPC is that they must be convex to guarantee global optimal solutions. And that is where the proposed input convex LTSM models presented in the previous section come in. We use them within a MPC formulation.

The system equations characterizing an HVAC system are follows [24]:

$$y_t = f(\mathbf{s}_t, \mathbf{u}_t) \tag{14}$$

$$\mathbf{s}_{t+1} = G(\mathbf{s}_t, \mathbf{u}_t) \tag{15}$$

where two nonlinear dynamic behaviors are included: one is G that defines the system dynamics described as the coupling between the current inputs to the future system states and the other is f, which defines the system dynamics as the coupling between the current inputs to the future system output. At time t the HVAC system state is $\mathbf{s}_t$, which includes the inside house temperature, outside temperature, etc. $\mathbf{u}_t$ denotes the house temperature setpoints and y_t is the energy usage by the HVAC system. $\mathbf{s}_{t+1}$ represents the system state at time step $t+1$.

Starting from the above system equations, we formulate the problem of optimal HVAC MPC control (again solved by the agent illustrated in Fig. 1) as follows.

$$\underset{\mathbf{u}_t, \mathbf{u}_{t+1}, \ldots, \mathbf{u}_{t+T}}{\text{minimize}} \quad C\left(\hat{\mathbf{x}}, y\right) = \sum_{\mathcal{T}=t}^{t+T} J(\hat{\mathbf{x}}_{\mathcal{T}}, \mathrm{y}_{\mathcal{T}}) \tag{16}$$

$$\mathbf{s.t.} \quad y_{\mathcal{T}} = f(\hat{\mathbf{x}}_{\mathcal{T}-\mathrm{n}_w}, \hat{\mathbf{x}}_{\mathcal{T}-\mathrm{n}_w+1}, \ldots, \hat{\mathbf{x}}_{\mathcal{T}}) \quad \forall\ \mathcal{T} \in [\mathrm{t}, \mathrm{t+T}] \tag{17}$$

$$\mathbf{s}_{\mathcal{T}} = \mathrm{G}(\hat{\mathbf{x}}_{\mathcal{T}-\mathrm{n}_w}, \hat{\mathbf{x}}_{\mathcal{T}-\mathrm{n}_w+1}, \ldots, \hat{\mathbf{x}}_{\mathcal{T}-1}, \hat{u}_{\mathcal{T}}) \ \forall\ \mathcal{T} \in [\mathrm{t}, \mathrm{t+T}] \tag{18}$$

$$\hat{\mathbf{x}}_{\mathcal{T}} = \begin{bmatrix} \mathbf{s}_{\mathcal{T}} \\ \hat{\mathbf{u}}_{\mathcal{T}} \end{bmatrix}, \hat{u}_{\mathcal{T}} = \begin{bmatrix} \mathbf{u}_{\mathcal{T}} \\ \mathbf{v}_{\mathcal{T}} \end{bmatrix}, \forall\ \mathcal{T} \in [\mathrm{t}, \mathrm{t+T}] \tag{19}$$

$$\mathbf{v}_{\mathcal{T}} = -\mathbf{u}_{\mathcal{T}}, \forall\ \mathcal{T} \in [t, t+T] \tag{20}$$

$$\mathbf{s}_{\mathcal{T}} \in \mathcal{S}_{feasible}, \forall\ \mathcal{T} \in [t, t+T] \tag{21}$$

$$\mathbf{u}_{\mathcal{T}} \in \mathcal{U}_{feasible}, \forall\ \mathcal{T} \in [t, t+T] \tag{22}$$

where a new variable representing system inputs $\hat{\mathbf{x}} = [s_{\mathcal{T}}, \hat{u}_{\mathcal{T}}]$ was introduced for notational simplicity. $s_{\mathcal{T}}$ represent system states and $\mathbf{u}_{\mathcal{T}}$, $-\mathbf{u}_{\mathcal{T}}$ are duplicated control actions. $J(\hat{\mathbf{x}}_{\mathcal{T}}, \mathrm{y}_{\mathcal{T}})$ is the control system cost incurs at time $\mathcal{T}$, that is a function of both the system inputs $\hat{x}_{\mathcal{T}}$ and output $y_{\mathcal{T}}$. The functions $f(\cdot)$ and $G(\cdot)$ from above Eqs. (17), (18) are parameterized as ICLSTMs, to capture or model: (1) the system dynamics from the sequence of inputs $(\hat{\mathbf{x}}_{\mathcal{T}-\mathrm{n}_w}, \hat{\mathbf{x}}_{\mathcal{T}-\mathrm{n}_w+1}, \ldots, \hat{\mathbf{x}}_{\mathcal{T}})$ to the system output $y_{\mathcal{T}}$, and (2) the system dynamics from control actions to system states, respectively. n_w is the memory window length of the ICLSTM model. Eqs. (19), (20) duplicate the input variables $\mathbf{u}$ and enforce the consistency condition between $\mathbf{u}$ and its negation $\mathbf{v}$. Lastly, Eqs. (21) and (22) are constraints on feasible system states and control actions, respectively.

Note that as a general formulation, we do not include the duplication tricks on state variables, so the dynamics fitted by Eqs. (17), (18) are non-decreasing over the state space. Because we do not restrict the control space and explicitly include multiple previous states in the system transition dynamics, the non-decreasing constraint over state space should not restrict representation capacity. This is similar to the study in [24], where similar input convex networks were demonstrated theoretically to be representable and efficient.

The optimization problem defined by Eqs. (16)–(22) is a convex optimization w.r.t the inputs $\mathbf{u} = [\mathbf{u}_t, \ldots, \mathbf{u}_{t+T}]$ provided that the cost function $J(\hat{\mathbf{x}}_{\mathcal{T}}, \mathrm{y}_{\mathcal{T}}) = J(\mathbf{s}_{\mathcal{T}}, \hat{\mathbf{u}}_{\mathcal{T}}, \mathrm{y}_{\mathcal{T}})$ is convex w.r.t. $\hat{\mathbf{u}}_{\mathcal{T}}$, and convex, nondecreasing w.r.t. $\mathbf{s}_{\mathcal{T}}$ and $y_{\mathcal{T}}$. A problem is convex if both the constraints and the objective function (that must be minimized) are convex; (objective function must be concave if the problem is a maximization problem). In the above problem, $J(\mathbf{s}_{\mathcal{T}}, \hat{\mathbf{u}}_{\mathcal{T}}, \mathrm{y}_{\mathcal{T}})$ is convex and nondecreasing w.r.t. $\mathbf{s}_{\mathcal{T}}$ and $y_{\mathcal{T}}$; $\mathbf{s}_{\mathcal{T}}$ and $y_{\mathcal{T}}$ are parameterized as ICLTMs, such that they are convex w.r.t. $\hat{\mathbf{u}}_{\mathcal{T}}$. Therefore following the composition rule of convex functions, the objective function is convex w.r.t. inputs $\mathbf{u} = [\mathbf{u}_t, \ldots, \mathbf{u}_{t+T}]$. Besides, all the equality constraints (19) and (20) are affine. Assuming that both the states feasibile set (21) and action feasibile set (22) are convex, the overall optimization is convex.

The convexity of the problem in Eqs. (16)–(22) guarantees that it can be solved efficiently and optimally using thr gradient descend method, which is tractable and has optimality guarantees. Since both the objective function (16) and the constraints (17)–(18) are parameterized as input convex LSTM networks, their gradients can be calculated via back-propagation with the modification where cost is propagated to the input rather than the weights of the network. For implementation, the gradients can be conveniently calculated via existing modules such as Tensorflow via back-propagation. Let $\mathbf{u}^* = \left\{\mathbf{u}_t^*, \mathbf{u}_{t+1}^*, \ldots, \mathbf{u}_{t+T}^*\right\}$ be the optimal solution found for the optimization problem at time t. Then the first element of $\mathbf{u}^*$ gives the system control, that is $\mathbf{u}_t^*$. The above optimization problem is then repeated at time $t+1$, based on the updated state prediction using $\mathbf{u}_t^*$ yielding thus a model predictive control strategy.

5.2.1. Demonstration of convexity in the proposed ICLSTM-based MPC

To formally show that our closed-loop MPC problem is convex, consider the following optimization at each time step (or epoch). In summary, the decision variables are the HVAC control inputs (i.e., temperature setpoints) over a prediction horizon, and we aim to minimize a cost function subject to constraints derived from the learned building dynamics and feasible sets. In Eqs. (16)–(22), $s_{\mathcal{T}}$ denotes the system state, $\mathbf{u}_{\mathcal{T}}$ is the control input, and each $\hat{\mathbf{x}}_{\mathcal{T}}$ bundles $(s_{\mathcal{T}}, \mathbf{u}_{\mathcal{T}}, -\mathbf{u}_{\mathcal{T}})$.

(1) ICLSTM convexity. We employ Input Convex LSTM (ICLSTM) networks for multi-step prediction of the building's indoor temperature and HVAC power usage. The functions $f(\cdot)$ and $G(\cdot)$ are given by *input convex LSTM* (ICLSTM) networks. As shown in, e.g., [23], restricting the cell's weight matrices to be nonnegative and using monotonic (e.g., ReLU) activations guarantees that $f(\cdot)$ and $G(\cdot)$ are convex in their inputs $\hat{\mathbf{x}}_{\mathcal{T}}$. Because each gating equation is a nonnegative, monotonic transformation, repeated cell-by-cell composition preserves convexity [50].

(2) Convex cost function. The stage cost in our MPC design, denoted by $J(\hat{\mathbf{x}}_{\mathcal{T}}, \mathrm{y}_{\mathcal{T}})$ is designed to be convex and nondecreasing in both $\hat{x}_{\mathcal{T}}$ and $\mathrm{y}_{\mathcal{T}}$ (indoor temperature and the HVAC energy usage). Since $\mathrm{y}_{\mathcal{T}} = f(\hat{x}_{\mathcal{T}})$ is itself convex in $\hat{\mathbf{x}}_{\mathcal{T}}$ (as guaranteed by the ICLSTM), $J(f(\hat{x}_{\mathcal{T}}))$ remains convex by standard composition rules of convex analysis. Thus, the objective function to be minimized is convex with respect to the decision variables (the temperature setpoints over the horizon).

(3) Convex feasible sets. The sets $\mathcal{S}_{feasible}$ and $\mathcal{U}_{feasible}$ are modeled as intervals or polyhedral sets, which are convex. Furthermore, the system dynamics fitted by the ICLSTM yield affine or convex constraints when carried to the input space, assuming we keep the feasible ranges of states and actions convex.

(4) Affine state-transition and equality constraints. Our MPC formulation includes equality constraints relating states at time $t+1$ to states and control actions at time t. Because the ICLSTM effectively provides convex (and in many places, affine) transformations of the control inputs and states, these constraints remain convex/affine in the decision variables. Hence, from a mathematical perspective, they do not break the convexity of the overall problem. In another words,because $\hat{x}_{\mathcal{T}}$ is an *affine* mapping of $(s_{\mathcal{T}}, \mathbf{u}_{\mathcal{T}}, -\mathbf{u}_{\mathcal{T}})$, the ICLSTM constraints in Eqs. (17)–(18) ultimately translate to convex functions of $\mathbf{u}_{\mathcal{T}}$. Hence, every link in the chain the neural dynamics $f(\cdot), G(\cdot)$, the cost function $J(\cdot)$, and the feasible sets obeys convexity principles. This ensures that the entire MPC problem in Eqs. (16)–(22) is a *convex optimization problem* in the decision variables $\{\mathbf{u}_{\mathcal{T}}\}_{\tau=t}^{t+T}$, solvable by standard convex optimization methods.

(5) Overall convexity and computational tractability. Given that: (a) the ICLSTM ensures convex parameterization of temperature and energy usage predictions, (b) the cost function $J(.)$ is convex and nondecreasing in these predicted outputs, and (c) the feasible sets for states and actions are convex, the resulting MPC problem is convex with respect to the decision variables (HVAC setpoints). Consequently, standard convex optimization solvers (e.g., gradient-based methods) can find the global optimum efficiently. The gradients of the objective function and constraints with respect to the inputs can be computed through back-propagation, treating the ICLSTM parameters as fixed during control execution.

5.3. Q-Learning algorithm

5.3.1. Basics and the update rule

In reinforcement learning (RL), an agent interacts with the environment with the goal to maximize the total accumulated reward. Model-free RL does not require explicit knowledge or assumptions about the underlying system dynamics and can learn directly from interactions with the environment. Observations and rewards are inputs into the agent, which then generates outputs in the form of actions sent back to the environment. In our case, the environment is the HVAC system. A popular type of RL is Q-learning - in which the learning agent maintains a so called Q-Table. The entries in this table are called Q-values, and they correspond to state–action pairs. In other words, Q-values represent the expected rewards for taking specific actions in given states. In this paper, we use Q-Learning as an algorithm to learn the relation between HVAC setpoints on one hand and thermal comfort and energy efficiency on the other hand.

The Q-Learning algorithm is characterized by the following update rule equation [51]:

$$Q(s_t, \mathbf{u}_t^*) \leftarrow (1-\alpha)\cdot Q(s_t, \mathbf{u}_t^*) + \alpha \cdot \left[r + \gamma \cdot \max_{\mathbf{u}'_{t+1}} Q(s'_{t+1}, \mathbf{u}'_{t+1}) \right] \quad (23)$$

where $Q(s_t, \mathbf{u}_t^*)$ is the estimated action-value function for state–action pair $(s_t, \mathbf{u}_t^*)$. α is the learning rate, which controls the step size of updates. r is the immediate reward received after taking action $\mathbf{u}_t^*$ in state s_t. γ is the discount factor and represents the importance of future rewards. s'_{t+1} is the next state reached after taking action $\mathbf{u}_t^*$ in state s_t. $\mathbf{u}'_{t+1}$ is next action chosen using the current Q-values. $Q(s'_{t+1}, \mathbf{u}'_{t+1})$ is action-value function for state–action pair $(s'_{t+1}, \mathbf{u}'_{t+1})$. In this paper, $s_t = (\mathbf{s}_t^1 \cup \mathbf{s}_t^2)$ and $s'_{t+1} = (\mathbf{s}_{t+1}^1 \cup \mathbf{s}_{t+1}^2)$, where subscript t, represents the first 20 min of each given hour, while subscript $t+1$ represents the remaining 40 min of the given hour. Also, the superscript 1 refers to the dataset used for HVAC energy usage prediction, while superscript 2 refers to the dataset used for indoor temperature prediction. Therefore, the superscript 1 (i.e., $\mathbf{s}_t^1$) denotes the data set for HVAC energy usage prediction in the first 20 min, and the superscript 2 (i.e., $\mathbf{s}_t^2$) denotes the dataset used for indoor temperature prediction in the first 20 min. These two datasets are different by only one column, as illustrated in Fig. 4.

5.3.2. Implementation details: Integrating Q-Learning with the MPC

Fig. 5 presents the overall flow for how the Q-Learning agent cooperates with the ICLSTM-based MPC to refine HVAC setpoints (i.e., the local DR signals). Below is a step-by-step outline:

(1) ICLSTM-based MPC as a "greedy" action generator.

- At each iteration, the Q-Learning agent examines its current Q-table to decide whether to
  1. *exploit* by using the best-known action (the setpoints provided by the MPC), or
  2. *explore* by randomly sampling an alternative set of setpoints near a rational range (the ϵ-greedy policy).
- When exploiting, the algorithm *solves* the ICLSTM-based MPC to produce the control action u_t^*. This action is optimal according to the MPC's convex optimization framework and takes into account aggregator offers, user comfort constraints, and other known conditions.

(2) State transition & environment feedback.

- Once the control action (u_t^*) or a random exploration setpoint is chosen, it is applied to *the environment* (i.e., the HVAC system).
- After a short time window (e.g., a few minutes), new measurements indoor temperature, HVAC power draw become the *next state* for the Q-Learning agent. These measurements reflect how well the chosen action performed under operating conditions.

(3) Reward computation & Q-value update.

- A reward is computed based on how well the resulting temperature or power usage aligns with aggregator goals (peak reduction, cost minimization) and occupant comfort.
- Using this reward and the Q-Learning update rule (eq (23)), the agent *updates* its Q-table to capture the relative effectiveness of the chosen action.

(4) Iterate & refine.

- The process repeats (Fig. 5's loops). Each new scenarios/states triggers another decision:
  1. either *exploitation* of the best MPC action,
  2. or *exploration* of possible acceptable actions.
- By continuously updating Q-values after each reward, the agent gradually *refines* the MPC-generated setpoints. This convergence yields improved temperature regulation, reduced energy costs, and better alignment with aggregator objectives over time.

In summary, Q-Learning agent "wraps around" the MPC solution, selectively (base on (epsilon)) trusting MPC's optimal action (via exploitation) or testing acceptable alternatives (via exploration). This dual mechanism leads to ongoing refinements of the MPC action, ensuring robust and adaptive HVAC control even under non-ideal or changing real-world.

5.3.3. Specific implementation

The Q-learning algorithm proposed in this paper is briefly described next. Its specific implementation details are presented with the help of the pseudocode description shown in Fig. 5. There are basically three distinct phases. Initially, we locally generate a daily dataset, encompassing various scenarios/states, specifically simulate runtime or real-time applications. This dataset is called $\mathbf{D}_{Rand}$; the process of this generation is described in the next subsection. Then, we use a portion of this dataset to retrain the ICLSTM models. This portion is 70% and is denoted as $\mathbf{D}_{ICLSTM}$. The remaining 30% portion of the generated dataset is denoted as $\mathbf{D}_{RL}$ and is used in third phase, where we construct and continually update the Q-table. The steps of this Q-learning RL phase are described in more details next.

First, the Q-table is initialized. The algorithm is an iterative process involving several **For** loops as shown in Fig. 5. In each iteration the model predictive control (MPC) problem is formulated and solved. In a first step, an observational one, the algorithm formulates constraints in relation to comfort and energy efficiency within the MPC problem, as described by Eqs. (16)–(22). Next, the algorithm selects an action according to the exploration–exploitation strategy. In our case, the proposed ϵ-MPC (which is based on concept of ϵ-greedy) selects actions in a way that strikes a balance between exploring new actions within a rational region and exploiting the best-known/optimal actions as determined by their Q-values (i.e., rewards). In other words, the algorithm chooses with probability (1 - ϵ) the action with the highest Q-value (exploitation by direct use of MPC) and with probability ϵ a random action (exploration) close to rational setpoints. Afterwords, the next step is an interaction and learning one, in which the selected action is applied to the environment. Upon observing the resulting state, it assesses comfort and energy efficiency, and receives the appropriate reward. The expected cumulative reward is calculated for each simulated action trajectory. The updated Q-values calculate the immediate reward and estimated future reward (based on the Q-value of the next state) for the action chosen through MPC. Q-value of the current state–action pair is updated using the Q-learning update rule from Eq. (23). The exploration parameter ϵ decays over time in order to gradually shift from exploration to exploitation. The interact and learning step evaluates the Q-learning performance and fine-tunes hyperparameters by repeating the basic steps (choose an action, calculate Q-values, update Q-table) for a specified number of episodes or until convergence. Once training is finished, the house's SHEM uses the updated Q-table to select actions.

5.3.4. Model development

The custom simulation tool used in this paper (described later in the next section) is also used for ICLSTM models development. That is, the simulation tool is used to generate training datasets that are needed to train the proposed ICLSTM models. To achieve that, the simulation tool was instrumented to conduct multiple simulation scenarios and to generate the dataset files with necessary input–output pair values. These scenarios are generated during the simulation of the testcase used in this paper inside GridLab-D. As mentioned before, we generate 1000 different scenarios (states) by randomly sampling

Algorithm 1: Q-Learning RL optimization algorithm - combines ICLSTM models and MPC

1: Generate locally dataset $\mathbf{D}_{Rand}$ for selected day based on 1000 random but rational different senarios/states
2: Split dataset $\mathbf{D}_{Rand}$ into $\mathbf{D}_{ICLSTM}$ (70%) and $\mathbf{D}_{RL}$ (30%)
3: Use $\mathbf{D}_{ICLSTM}$ to train ICLSTMs $(f_1(\mathbf{s}_t^1, \mathbf{u}_t), G_1(\mathbf{s}_t^2, \mathbf{u}_t))$ first 20 min and ICLSTMs $(f_2(\mathbf{s}_{t+1}^1, \mathbf{u}_{t+1}), G_2(\mathbf{s}_{t+1}^2, \mathbf{u}_{t+1}))$ for the rest 40 min of each hour
4: Use dataset $\mathbf{D}_{RL}$ and initialize (ϵ, episodes, attemts, epochs, discont factor, learning rate)
5: Import or initialize the Q-table
6: **For** iter=1 to max episodes **Do**
7: **For** state=1 to max states **Do**
8: **For** attempt=1 to max attempts **Do**
9: **For** epoch=1 to max epochs **Do**
10: Solve ICLSTM-based MPC to find $\mathbf{u}_t^*$
11: **End**
12: **Generate** new $(\mathbf{s}_t^1, \mathbf{u}_t^*)$ do prediction by $f_1(\mathbf{s}_t^1, \mathbf{u}_t^*)$
13: **Generate** new $(\mathbf{s}_t^2, \mathbf{u}_t^*)$ do prediction by $G_1(\mathbf{s}_t^2, \mathbf{u}_t^*)$
14: **Generate** Q-learning punishment and reward
15: **If** best reward repeated: save action and reward of attempt as $Q(s_t, \mathbf{u}_t^*)$
16: **End**
17: **For** attempt=1 to max attempts **Do**
18: **For** epoch=1 to max epochs **Do**
19: Solve ICLSTM-based MPC to find $\mathbf{u}'_{t+1}$ as max
20: **End**
21: **Generate** new $(\mathbf{s}_{t+1}^1, \mathbf{u}'_{t+1})$ do prediction by $f_2(\mathbf{s}_{t+1}^1, \mathbf{u}'_{t+1})$
22: **Generate** new $(\mathbf{s}_{t+1}^2, \mathbf{u}'_{t+1})$ do prediction by $G_2(\mathbf{s}_{t+1}^2, \mathbf{u}'_{t+1})$
23: **Generate** Q-learning punishments and Reward
24: **If** best reward repeated: save action and reward of attempt as $Q(s'_{t+1}, \mathbf{u}'_{t+1})$
25: **End**
26: **End**
27: **Generate** q-table by using eq. (23)
28: **For** state=1 to max states **Do**
29: **If** (Reward of Q-table[state] < Reward of q-table[state])
30: **Update** Q-table
31: **End**
32: **End**
33: **Read** Q-table, then calculate the most probable intersection between all actions in Q-table and occupancy preference to identify new setpoints for HVAC
34: **End**

Fig. 5. Pseudocode description of proposed algorithm to solve optimal demand response for HVAC systems.

setpoint values from a *rational* setpoints range for a given house. This is illustrated in Fig. 6. This range of setpoints is set such that they reflect rational intention of the house owner with active HVAC controllers — which means the range allows reasonable values that can lead to cost reduction while still satisfying comfort levels. While simulations are conducted with a time step of 1 min, we collect data points with a sampling time of 3 min; therefore, each hour is represented with 20 points in the datasets created for ICLSTMs training. Such training datasets are generated for the training of four ICLSTM models — that we use for prediction of HVAC energy usage and of indoor temperature during the first 20 min of each hour and for the remaining 40 min of each hour. After datasets are generated, the ICLSTM models are trained (as listed in the pseudocode from Fig. 5). During this process several model hyperparameters need to be selected. In our case, model hyperparameter values are selected based on a simple trial and error procedure guided by our experience with LSTM model development that we reported in our recent work [52]. The final specific values of these hyperparameters are: input size 5, hidden layers size 72, sequence length 9 or 20, output size 1, and batch size 24.

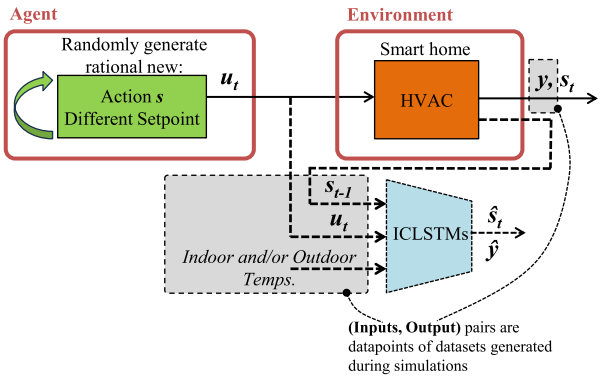


Fig. 6. Illustration of dataset generation: the Agent generates rational new states and the Environment (i.e., the HVAC system) is simulated by the simulation tool. Input–output pairs are collected as shown and saved into the dataset file. The dataset file will be used for ICLSTMs training.

Part of our model development is also the learning necessary to be done for the Q-learning component of our optimization, as described in the previous subsection. The result of this is a preliminary Q-table that stores the expected return or reward for each state–action pair as Q-values. In our case, *states* are defined by the different scenarios while *rewards* are calculated based on: (1) the *distance* between the output of the simulation tool and the home owner (i.e., user) preference, (2) the distance between HVAC energy usage resulted from simulation and the energy usage offered by the aggregator, and (3) the maximum and minimum indoor temperatures, as well as the indoor temperature variation slope. The home user preference is captured by the min, max, and mean temperature setpoints (e.g., 70F, 74F, and 72F).

In our implementation, we work with a Q-table size of 300 corresponding to 300 different states (embodying as many state–action scenarios). For each given scenario, we systematically evaluated eight distinct attempts employing the MPC approach. As a result, the cumulative count of MPC optimization runs amounted to 2400 instances within each episode. In parallel, the total number of runs conducted through Q-learning (by using Eq. (23)) was 300 for each episode. The number of distinct attempts of the MPC approach is determined by the maximum number of epochs allowed. We note that during the Q-learning process, we do not have to check the maximum possible number of epochs (2000 times). Smartly, the algorithms tries out new ideas while also concentrating on actions with Q-values that are already known to be good. As soon as it finds good enough Q-values with the help of the ICLSTM-based MPC, it can move on to the next steps. The hyperparameter values that we use are: learning rate 0.1 (i.e., rate at which Q-values are updated), discount factor 0.95 (measures future rewards importance), and exploration parameter or $\epsilon = 0.05$ (controls exploration-exploitation trade-off).

5.3.5. Performance gain/motivation: Why Q-Learning is beneficial in tandem with MPC

The key motivation for integrating Q-Learning with an ICLSTM-based MPC framework is that each component compensates for the other's limitations, thereby enhancing overall performance:

1. **Adaptive Correction of MPC Decisions:** The MPC alone makes greedy decisions based on its learned ICLSTM model of the HVAC system. However, any model inaccuracy (e.g., mismatch between training data and real conditions) can degrade performance over time. Q-Learning wraps around the MPC solution to *continuously refine* or "correct" the MPC's setpoints. Through its exploration–exploitation (ϵ-greedy) strategy, Q-Learning tests alternative actions and *updates* a Q-table based on *actual* performance (rewards). Over multiple iterations, this process may discover improved control signals that deviate from purely model-predicted optima.

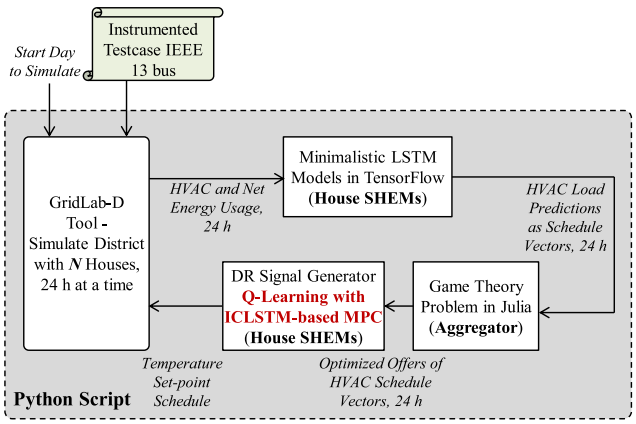


Fig. 7. Simplified diagram of the custom simulation framework used for simulation experiments.

2. **Handling Uncertainty and Non-Modeled Dynamics:** Real HVAC systems are often subject to changing occupant behaviors, varying aggregator signals, and other external factors not perfectly captured in the ICLSTM. While MPC optimizes over a finite horizon with known constraints, it is less robust when conditions fall *outside* the training distribution. Q-Learning, being model-free, learns from direct interaction with the environment. If new situations cause performance degradation, Q-Learning detects lower rewards and adapts future decisions accordingly, compensating for any shortfalls in the purely model-based approach.
3. **Improved Exploration for Long-Term Optimality:** Traditional MPC exploits its predictive model but typically lacks systematic exploration. If the model or cost function is imperfect, the MPC can become stuck in locally optimal solutions. Q-Learning's exploration ensures that *potentially better setpoints* (even if initially counterintuitive to the model) are tested. When these yield higher rewards, they update the Q-table and steer the system toward more globally optimal behaviors, rather than relying solely on local or short-term minima.
4. **Enhanced Robustness and Stability Over Time:** By periodically re-solving the MPC *and* updating Q-values, the control loop remains robust to changes such as weather fluctuations, shifts in comfort requirements, and aggregator scheduling changes. The synergy of MPC's multi-step lookahead and Q-Learning's adaptive feedback yields a solution less prone to large deviations or oscillations under nonlinear HVAC dynamics, resulting in more reliable peak shaving, reduced energy costs, and robust user comfort compliance.

6. Simulation experiments

6.1. Simulation tool

To conduct experiments, we have developed a custom simulation framework. The simulation framework is an updated version of the one that we proposed and described in great detail in our recent work [41]. The update consists of the addition of the RL algorithm (which integrates the MPC with ICLSTM models discussed earlier) as describe in Fig. 1. Here, we present only a brief overview of the updated simulation framework for completeness. Its simplified diagram is shown in Fig. 7. Please note that this simulation framework models both the distribution network with houses and the aggregator, and captures all actions indicated in Fig. 1 and highlighted again here in the context of the simulation tool whose actions/steps are shown in Fig. 7. The simulator integrates several software components including the GridLab-D tool [53], Julia optimization packages [54], and LSTM deep learning models, which we developed in Tensorflow and described in detail in our recent previous work [52].

These software components are integrated and executed from within a top-level Python script, which is in charge with running all the steps for a given experiment. A simulation experiment is run from within this script in several stages: In stage one, the script passes information about a specific day (dd-mm-yyyy) as input to the GridLab-D tool, which also reads in a specific testcase. In our experiments, we use an instrumented the IEEE 13 node testcase, which we also studied and reported in our recent previous work [52]. The GridLab-D tool simulates the testcase for 24 h; as result of the simulation, the hourly energy usage (both net and HVAC loads) for all houses is obtained. In stage two, the Python script passes the energy usage result to the LSTM prediction models of each house; in a practical deployment of our approach this would be done at the house sites, by their SHEM computers. The minimalistic LSTM models developed in [52] are used to predict the HVAC energy usage for the next 24 h. The predictions are passed in stage three to the aggregator component (see Fig. 7), which constructs a game theory based optimization problem for the entire distribution network testcase and solves it by means of the Clarabel package in Julia (i.e., interior-point solver for convex conic optimization problems). Once the game theory problem is solved by the aggregator, the new best HVAC schedule vectors are offered back to all house SHEMs.

At this point, on one hand, the SHEM system in each house could take those offered by the aggregator HVAC schedules and implement them throughout the respective 24 h period. This is what we did in our recent work [41], and use here as a base or reference case. On the other hand – and this is what we do in this paper as main contribution – the proposed Q-Learning algorithm running as an agent inside each house's SHEM (see Fig. 1) takes those aggregator-offered HVAC schedules and generates refined optimal DR signals for the local HVAC system, again throughout the respective 24 h period. At the end of the 24 h period the entire experiment is repeated for another 24 h period and so on.

6.2. Simulation results

6.2.1. The reference case

As mentioned earlier, we compare the results achieved with the proposed Q-learning algorithm against the approach that we presented in our previous work [41], and briefly reviewed in Section 6.1. The performance of this Demand Response (DR) generator model is evaluated by comparing it with the results of a previously published Demand Side Management (DSM) model [41]. The main idea of this reference case is that at the beginning of every 24 h period, the aggregator provides each house's SHEM a *offered* HVAC energy usage schedule, which was found by the aggregator by solving a game theory based optimization problem that involved all houses in the distribution network. The offered energy schedules are generated by the aggregator such that to reduce and shift at the same time the amount of HVAC energy usage during peak hours with a certain amount, which percent wise was fixed at 15%. For example, Fig. 8.a shows how HVAC energy usage is reduced and shifted for an arbitrary house 6 in the distribution system and the aggregated such impact from all houses in the system on the net load on the distribution system during the same 24 h period (Fig. 8.b). Table 2 lists the individual reduction in total HVAC energy usage during peak hours and the individual change in daily cost of HVAC energy usage for all 12 houses in the instrumented IEEE 13 bus testcase.

6.2.2. The new Q-Learning case

This is the case of having the optimization proposed in this paper applied to each house in the testcase. That is, the Q-Learning algorithm run by the agent inside each house's SHEM receives as input the offered energy schedule generated by the aggregator and refines it into a more optimal set of DR signals (i.e., temperature setpoints) for the local HVAC system. As a result of this refinement (i.e., local

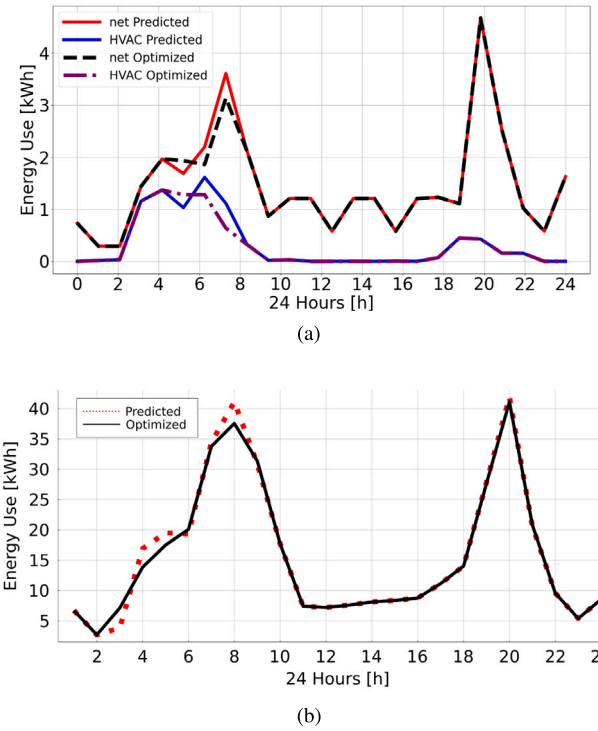


Fig. 8. Reference case: (a) Hourly HVAC and net loads for house 6 of the instrumented testcase. (b) Total aggregated net load in the entire system.

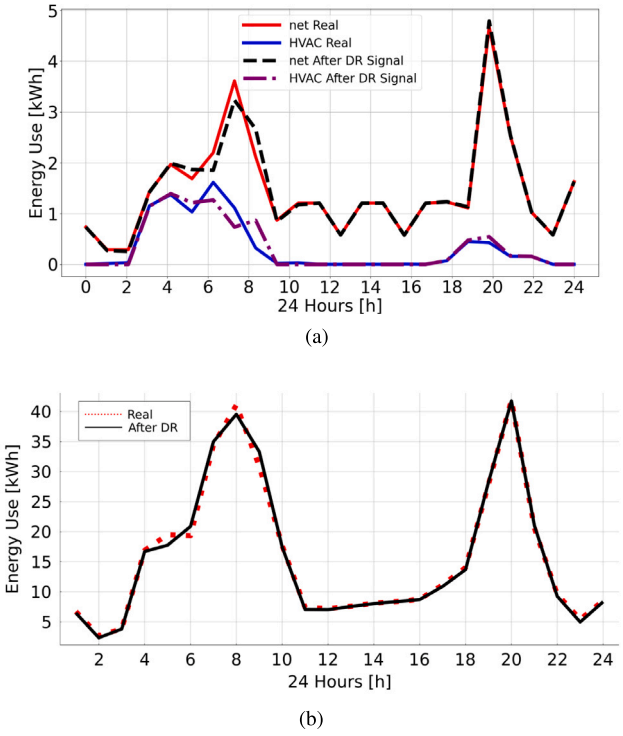


Fig. 9. Proposed Q-Learning based DR signals case: (a) Hourly HVAC and net loads for house 6 of the instrumented testcase. (b) Total aggregated net load in the entire system.

Table 2
Reference case: energy usage reduction in all houses of the instrumented testcase on a randomly selected day.

House	Reduction in HVAC energy usage during peak hours (%)	Reduction in daily HVAC energy usage cost ($)
House 1	15	0.60
House 5	15	0.45
House 6	15	0.20
House 7	15	0.15
House 8	15	0.10
House 9	15	0.15
House 10	15	0.30
House 11	15	0.10
House 12	15	0.15
House 13	15	0.20
House 14	15	0.15
House 15	15	0.10

Table 3
Proposed Q-Learning optimization algorithm case: energy usage reduction in all houses of the instrumented testcase on the same day as in Table 2.

House	Reduction in HVAC energy usage during peak hours (%)	Reduction in daily HVAC energy usage cost ($)
House 1	8	0.31
House 5	9	0.28
House 6	14	0.17
House 7	10	0.11
House 8	11	0.07
House 9	10	0.08
House 10	11	0.04
House 11	10	0.06
House 12	9.5	0.09
House 13	15	0.19
House 14	14.5	0.15
House 15	7	0.05

optimization), the aggregator-offered HVAC energy usage reduction of 15% may be altered and changed to a smaller value. For example, for house 6 discussed in Fig. 8.a this percentage is changed to 14% (while energy cost was reduced by 13%). The execution of the newly optimized controls preserve to a large extent the reduction and shifting of the HVAC energy usage obtained by executing the aggregator-offered HVAC energy usage schedule; that is shown for example for the same house 6 in Fig. 9. Table 3 lists the individual reduction in total HVAC energy usage during peak hours and the individual change in daily cost of HVAC energy usage for all 12 houses in the instrumented IEEE 13 bus testcase.

6.3. Comparison of ICLSTM and ICRNN models

In this section, we compare the performance of the proposed ICLSTM model versus the ICRNN model proposed in previous work. The comparison is done using a dataset formed by 1000 different scenarios/states which is split into train and test datasets as 0.7/0.3. The training set is used to train both models, while the testing set is used to evaluate their performance. We ensure that the hyperparameters, such as the learning rate, number of layers, and neurons, are the same for both models. Both models make predictions on the testing dataset of HVAC energy usage and the house indoor temperature dynamics. The entire process is systematically iterated 100 times in order to enhance precision and mitigate randomness.

The models' performance is evaluated using mean absolute percentage error (MAPE) and coefficient of variation of root mean squared error (CVRMSE), given by the following two equations:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right| \times 100 \tag{24}$$

$$CVRMSE = \frac{\text{RMSE}}{\text{Mean}(Y)} \times 100 \tag{25}$$

The results of this comparison are summarized in Table 4, where we observe that the proposed ICLSTM model provides better accuracy

Table 4
Qualitative comparison of proposed ICLSTM models to prior work ICRNN models for energy and temperature prediction.

Metric	ICLSTM model	ICRNN model	Diff. (%)
Energy MAPE (%)	inf	inf	–
Energy CV-RMSE (%)	10.94	241.21	2104.58
Temperature MAPE (%)	18.526	1.755	955.61
Temperature CV-RMSE (%)	0.204	0.0273	647.25
Model Trainable params	147 674	40 752	262.37
Memory footprint (KB)	606	167	262.87

for prediction of HVAC energy. However, the ICRNN model performs better in scenarios characterized by smoother patterns, such as indoor temperature.

Furthermore, we are interested in comparing how the proposed ICLSTM-based MPC Q-Learning optimization versus the ICRNN-based MPC presented in [24]. The comparison is done in terms of *sample efficiency*, which refers to how quickly an algorithm can learn a good policy with the least amount of interaction with the environment, or in other words, with the fewest number of training samples or episodes. In RL, measuring sample efficiency is critical because collecting real-world data can be costly or time-consuming, and efficient algorithms are desirable for practical applications. Sample efficiency and performance can be measured using metrics like the success rate or value function accuracy in solving the task. Such metrics help to determine whether an agent is capable of learning to perform a task while using as few interactions with the environment. The *success rate* is a metric that utilizes a predefined time or number of actions to determine how often the RL agent is successful in reaching the goal. The *time to convergence* metric is used to describe the amount of time it takes for an agent to learn a good policy or strategy.

Before calculating the success rate, we need to define first what the success criterion is, i.e., establish what constitutes a "successful" adjustment to the HVAC setpoint in a house. For the purposes of this experiment, we desire the setpoint to be within the interval 72-74F. Success is then defined a setpoint achieved in the middle of this interval with an acceptable 0.5F error threshold, that is any temperature setpoint between 72.5–73.5F is considered acceptable. Once the success criterion is defined, the success rate can be readily calculated based on the results of the 100 different runs of the experiment. In each of the 100 different runs, Q-learning was used to determine the setpoint for an HVAC system. The difference between the desired setpoint and the actual setpoint gives the associated error. The number of times this error falls within the acceptable error threshold is denoted as *Number of Successful Runs*. Then, *success rate* is calculated as:

$$\text{Success Rate} = \frac{\text{Number of Successful Runs}}{\text{Total Number of Runs}} \times 100\%. \tag{26}$$

The results of the above comparison experiments are summarized in Table 5, as average values over 100 repeats for robustness. We observe that the proposed ICLSTM-based MPC Q-Learning optimization is significantly better than the ICRNN-based MPC Q-Learning [24] in terms of success rate. In terms of time to convergence, the proposed algorithm is slower. However, the time to convergence of the proposed algorithm could be significantly reduced if it is run in the cloud on processors with GPU accelerators (currently, we have it run on a simple processor without any acceleration). In addition, the time to convergence can be decreased by decreasing the number of attempts and epochs discussed earlier in the paper (end user can easily change them).

7. Conclusion

We presented a new strategy to model-based Q-Learning reinforcement learning applied to the optimization of house HVAC system operation — whose objective is to: optimally reduce and/or shift peak energy usage, reduce electricity costs, minimize user discomfort, and honor in a best-effort way power consumption recommendations from the energy provider of a distribution network. The proposed algorithm combines input convex long short-term memory models with model predictive control (MPC) techniques. With this approach, we address one of the greatest challenges that limited the use of expressive and high-capacity machine learning models in model-based control algorithms. The proposed optimization is implemented as an agent that is executed by the smart house energy management (SHEM) system, which generates local demand response (DR) signals to directly control the HVAC system to strike an optimal balance between minimizing electricity bills and minimizing user discomfort. Using the proposed optimization framework, smart homes equipped with passive HVAC controllers, which are generally limited to responding to setpoints provided by the end user, can be transformed into smart homes equipped with active HVAC controllers. Through an external signal provided by the utility or aggregator, this advanced system goes beyond simple responsiveness to end-user preferences. User preferences are balanced with external control inputs, which represents a significant shift from a user-centric control model. Simulation experiments conducted with a custom simulation tool demonstrated that the proposed optimization framework provides 87% higher success rate of optimizing setpoints in the desired range, resulting in up to 15% energy savings and zero temperature discomfort.

Table 5
Qualitative comparison of the proposed Q-Learning versus prior MPC [24].

Metric	Model in this work	Model in [24]	Diff. (%)
Success rate (%)	60.9	32.5	87.38
Time to convergence (min)	41.1	16.5	149.1

CRediT authorship contribution statement

Rahman Heidarykiany: Writing – original draft, Visualization, Validation, Software, Investigation, Formal analysis, Data curation, Conceptualization. **Cristinel Ababei:** Writing – review & editing, Supervision, Resources, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: RAHMAN HEIDARYKIANY reports financial support was provided by National Science Foundation ECCF 1936494. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] Zeighami A, Kern J, Yates J, Weber P, Bruno A. US West Coast droughts and heat waves exacerbate pollution inequality and can evade emission control policies. Nature 2023;14(1):1415.
[2] PH. Shaikh NN, Nallagownden P, Elamvazuthi I, Ibrahim T. A review on optimized control systems for building energy and comfort management of smart sustainable buildings. In: Renewable and sustainable energy reviews. Vol. 34, Elsevier; 2014, p. 409–29.
[3] Zhang Z, Deng R, Yuan T, Qin S. Distributed optimization of multi-building energy systems with spatially and temporally coupled constraints. In: IEEE. 2017 American control conference (ACC). 2017.

[4] Ma Y, Kelman A, Daly A, Borrelli F. Predictive control for energy efficient buildings with thermal storage: Modeling, stimulation and experiments. IEEE control Syst Mag 2012;32(1):44–64.

[5] Meger D, Higuera J, Xu A, Giguere P, Dudek G. Learning legged swimming gaits from experience. In: 2015 IEEE international conference on robotics and automation (ICRA). IEEE; 2015.

[6] Mahé A, Richard A, Mouscadet B, Pradalier C, Geist M. Importance sampling for deep system identification. In: IEEE. international conference on advanced robotics (ICAR). 2019.

[7] Suryadevara N, Mukhopadhyay S, Kelly S, Gill S. WSN-based smart sensors and actuator for power management in intelligent buildings. IEEE/ASME Trans Mechatronics 2014;20(2):564–71.

[8] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[9] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A, Kaiser L, Jones L, Polosukhin I. Attention is all you need. Adv Neural Inf Process Syst 2017;30.

[10] Vicas C. Valuation of Deep Learning architectures for System Identification. Applications for a household heating system. In: 2020 international conference on intelligent computer communication and processing (ICCP). IEEE; 2020.

[11] Mavkov B, Forgione M, Piga D. Integrated neural networks for nonlinear continuous-time system identification. IEEE Control Syst Lett 2020;4(4):851–6.

[12] Revay M, Wang R, Manchester I. A convex parameterization of robust recurrent neural networks. IEEE Control Syst Lett 2020;5(1):1363–8.

[13] Bemporad A. Recurrent neural network training with convex loss and regularization functions by extended Kalman filtering. IEEE J Multisc Multiphys Comput Tech 2022.

[14] Wang Y. A new concept using lstm neural networks for dynamic system identification. In: IEEE. 2017 American control conference (ACC). 2017.

[15] Feng L. Predicting output responses of nonlinear dynamical systems with parametrized inputs using LSTM. IEEE J Multisc Multiphys Comput Tech 2023;8:97–107.

[16] Masti D, Bemporad A. Learning nonlinear state-space models using deep autoencoders. In: Conference on decision and control (CDC). IEEE; 2018.

[17] Gonzalez J, Yu W. Non-linear system modeling using LSTM neural networks. In: IFAC-PapersOnLine. Vol. 51, (13):Elsevier; 2018, p. 485–9.

[18] Skomski E, Vasisht S, Wight C, Tuor A, Drgoňa J, Vrabie D. Constrained block nonlinear neural dynamical models. In: 2021 American control conference (ACC). IEEE; 2021.

[19] Forgione F, Piga D. Model structures and fitting criteria for system identification with neural networks. In: International conference on application of information and communication technologies (AICT). IEEE; 2020.

[20] Kawaguchi K. Deep learning without poor local minima. Advances in neural information processing systems 2016;29.

[21] Drgona J, Tuor A, Vasisht S, Vrabie D. Dissipative deep neural dynamical systems. IEEE Open J Control Syst 2022;1:100–12.

[22] ul Abdeen Z, Yin H, Kekatos V, Jin M. Learning neural networks under input-output specifications. In: IEEE. American Control Conference (ACC). 2022.

[23] Amos B, Xu L, Kolter J. Input convex neural networks. In: International conference on machine learning. PMLR; 2017.

[24] Chen Y, Shi Y, Zhang B. Optimal control via neural networks: A convex approach. 2018, arXiv preprint arXiv:1805.11835.

[25] Bünning F, Schalbetter A, Aboudonia A, de Badyn M, Heer P, Lygeros J. Input convex neural networks for building MPC. In: Learning for dynamics and control. PMLR; 2021.

[26] Maddalena E, Müller S, dos Santos R, Salzmann C, Jones C. Experimental data-driven model predictive control of a hospital HVAC system during regular use. In: Energy and buildings. Elsevier; 2022.

[27] Lefebure N, Khosravi M, de Badyn M, Bünning F, J. Lygeros CJ, Smith R. Distributed model predictive control of buildings and energy hubs. In: Energy and Buildings. Elsevier; 2022.

[28] Patil M, Charuku B, Ren J. Long short-term memory neural network-based system identification and augmented predictive control of piezoelectric actuators for precise trajectory tracking. In: IFAC-PapersOnLine. Vol. 54, Elsevier; 2021, p. 38–45, no. 20.

[29] Blaud P, Chevrel P, Claveau F, Haurant P, Mouraud A. ResNet and PolyNet based identification and (MPC) control of dynamical systems: a promising way. IEEE Access 2022;11:20657–72.

[30] Sabahi F. Introducing convex BiLSTM-based controller applied to 3-PSP spatial parallel robot manipulator. In: Transactions of the institute of measurement and control. SAGE Publications Sage UK; 2023.

[31] Zarzycki K, Ławryńczuk M. Fast nonlinear model predictive control using LSTM networks: A model linearisation approach. In: Mediterranean conference on control and automation (MED). IEEE; 2022.

[32] Bünning F, Huber B, Schalbetter A, Aboudonia A, de Badyn M, Heer P, Smith R, Lygeros J. Physics-informed linear regression is competitive with two Machine Learning methods in residential building MPC. In: Energy and buildings. Elsevier; 2022.

[33] Elnour M, Himeur Y, Fadli F, Mohammedsherif H, Meskin N, Ahmad A, Petri I, Rezgui Y, Heer P. Neural network-based model predictive control system for optimizing building automation and management systems of sports facilities. In: Applied energy. 318, Elsevier; 2022, 119153.

[34] Bayer D, Pruckner M. Enhancing the performance of multi-agent reinforcement learning for controlling HVAC systems. In: Conference on technologies for sustainability (SusTech). IEEE; 2022.

[35] Langer L, Volling T. A reinforcement learning approach to home energy management for modulating heat pumps and photovoltaic systems. In: Applied energy. Vol. 327, Elsevier; 2022, 120020.

[36] Chen B, Cai Z, Bergés M. Gnu-rl: A practical and scalable reinforcement learning solution for building hvac control using a differentiable mpc policy. Front Built Environ 2020;6:562239.

[37] Chen L, Meng F, Zhang Y. MBRL-MC: An HVAC control approach via combining model-based deep reinforcement learning and model predictive control. IEEE Internet Things J 2022;9(19):19160–73.

[38] Zhang X, Chen Y, Bernstein A, Chintala R, Graf P, Jin X, Biagioni D. Two-stage reinforcement learning policy search for grid-interactive building control. IEEE Trans Smart Grid 2022;13(3):1976–87.

[39] Zhang H, Seal S, Wu D, Boulet B, Bouffard F, Joos G. Data-driven model predictive and reinforcement learning based control for building energy management: a survey. arXiv 2021.

[40] Al-Ani O, Das S. Reinforcement learning: Theory and applications in HEMS. In: Energies. Vol. 15, MDPI; 2022, p. 6392, no. 17.

[41] Heidarykiany R, Ababei C. HVAC energy cost minimization in smart grids: A cloud-based demand side management approach with game theory optimization and deep learning. In: Energy and AI. Vol. 65, Elsevier; 2023, p. 636–48.

[42] Cai W, Sawant S, Reinhardt D, Rastegarpour S, Gros S. A learning-based model predictive control strategy for home energy management systems. IEEE Access 2023;11:145264–80.

[43] Seel K, Kordabad A, Gros S, Gravdahl J. Convex neural network-based cost modifications for learning model predictive control. IEEE Open J Control Syst 2022;1:366–79.

[44] Gros S, Zanon M. Data-driven economic NMPC using reinforcement learning. IEEE Trans Autom Control 2019.

[45] Cai W, Kordabad A, Gros S. Energy management in residential microgrid using model predictive control-based reinforcement learning and Shapley value. In: Engineering Applications of Artificial Intelligence. Vol. 119, Elsevier; 2023, p. 105793.

[46] Adhau S, Gros S, Skogestad S. Reinforcement learning based MPC with neural dynamical models. In: European journal of control. Vol. 80, Elsevier; 2024, p. 101048.

[47] Kordabad A, Reinhardt D, Anand A, Gros S. Reinforcement learning for MPC: Fundamentals and current challenges. In: IFAC-PapersOnLine. Vol. 56, Elsevier; 2023, p. 5773–80.

[48] Zanon M, Gros S. Safe reinforcement learning using robust MPC. IEEE Trans Autom Control 2020;66:3638–52.

[49] Massaoudi M, Abu-Rub H, Refaat S, Chihi I, Oueslati F. Deep learning in smart grid technology: a review of recent advancements and future prospects. IEEE Access 2020;9:54558–78.

[50] Boyd S, Boyd S, Vandenberghe L. Convex optimization. Cambridge university press; 2004.

[51] Jang B, Kim M, Harerimana G, Kim J. Q-learning algorithms: A comprehensive classification and applications. IEEE Access 2019;7:133653–67.

[52] Heidarykiany R, Ababei C. Minimalistic LSTM models for next day hourly residential HVAC energy usage forecasting. In: IEEE electrical power and energy conference (EPEC). 2022.

[53] Chassin D, Schneider K, Gerkensmeyer C. GridLAB-D: An open-source power systems modeling and simulation environment. In: IEEE/PES transmission and distribution conference and exposition. 2008.

[54] Goulart P, Chen Y. Clarabel.jl is a Julia implementation of an interior point numerical solver for convex optimization problems using a novel homogeneous embedding. [Online]. Available: https://docs.juliahub.com/Clarabel/g676L/0.1.2/.