DEEP LEARNING METHODS APPLIED TO MODELING AND POLICY OPTIMIZATION IN LARGE BUILDINGS

By

Avisek Naug

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Computer Science

May 31, 2022

Nashville, Tennessee

Approved:

Dr Gautam Biswas, Ph.D.

Dr Abhishek Dubey, Ph.D.

Dr Matthew S Berger, Ph.D.

Dr Douglas H Fisher, Ph.D.

Dr Janos Sztipanovits, Ph.D.

Dr Dan Work, Ph.D.

ACKNOWLEDGMENTS

I want to thank Dr Gautam Biswas for his immense contribution to materialize the ideas of my proposal and his constant supervision while writing the same and publishing related papers. I also want to thank Dr Abhishek Dubey, Dr Matthew Berger, Dr Janos Sztipanovits, Dr Douglas Fisher and Dr Dan Work for serving in the committee. I want to thank Darren Beville from Plant Ops Vanderbilt and Dan Fink from Building Logix for their valuable and practical contributions in implementing a prototype controller at Alumni Hall. I also want to thank Jack King for coordinating the effort.I also want to thank Dr Marcos Quinones and Ibrahim Ahmed for their valuable feedback and constant support while discussing ideas. Finally, I would like to thank my parents, whose constant support during this journey helped me achieve my goals.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

Over the past few decades, energy usage has increased significantly, and this is causing major stresses on the energy generation and distribution systems, while contributing significantly to environmental deterioration and climate change [4]. In the US, 92.94 Quadrillion BTUs of energy was consumed in 2020 out of which roughly 29% or 27 Qd BTUs was consumed for operating commercial buildings. 40% of this, building energy consumption is attributed to Heating Ventilation and Air Conditioning(HVAC) systems [1], which amounts to billions of dollars in running costs.

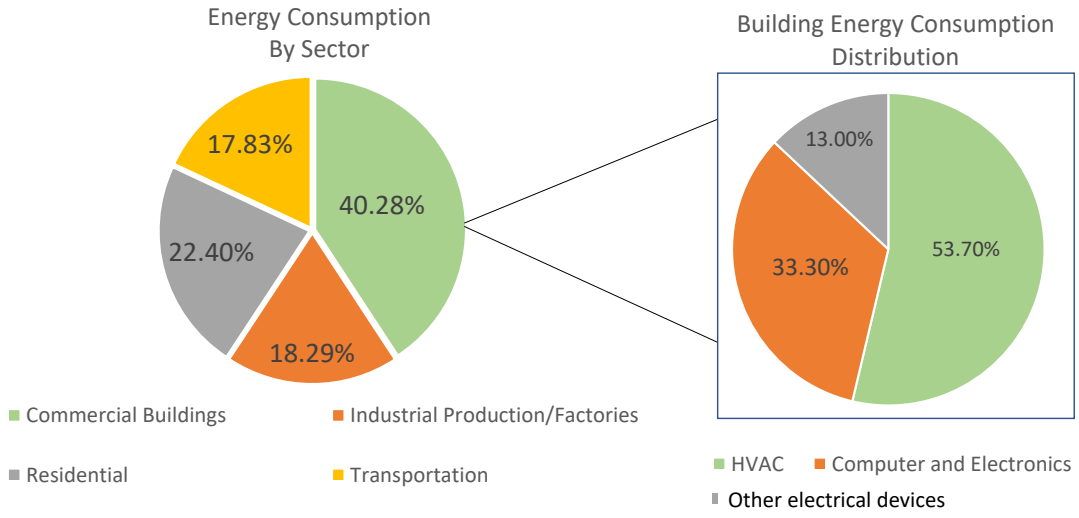

Figure 1.1: Distribution of Energy consumption across Different Sectors[1] and in Commercial Buildings[2](2019)

Hence, it is of primary interest to make HVAC operations energy and cost-efficient. Large commercial buildings are commonly equipped with Building Energy Management systems(BEMs) that control and monitor the energy use of different building HVAC components like Air Handling Units(AHU), Condensers, Cooling Towers, Variable Air Volume(VAV) units and Variable Refrigerant Flow(VRF) systems. These components are designed to maintain comfortable environments in buildings when they operate nominally. However, building energy operations are complex and non-stationary. The non-stationarity can be attributed to multiple causes: (1) unexpected changes in environmental conditions that includes weather; (2) sudden changes in building operations, such as changes in occupancy levels; and (3) faults in building HVAC components. Variables associated with these abrupt changes can include ambient temperature, relative humidity, zone setpoint

requirements, thermal loads *etc.* Even though current building energy control methods use a set of rules to adapt to these changes, they fall short when it comes to using closed loop feedback, for variables like energy consumption, occupant comfort, actuator safety from the building under non-stationary conditions. As such, these methods do not easily lend them to develop systematic adaptation and optimization, especially when unexpected conditions occur that affect building operations.

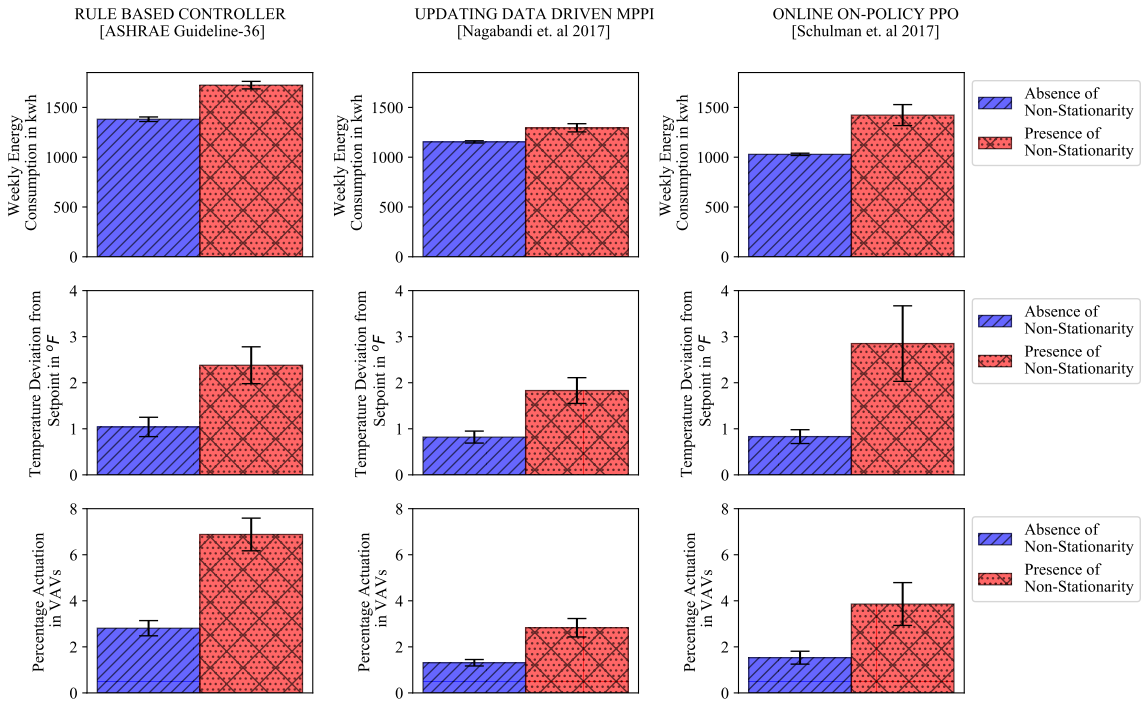

Figure 1.2: Performance Comparison in terms of Energy Consumption, Temperature Comfort, VAV Actuation across state-of-the-art control strategies like the Industrial Standard Rule Based Control based on ASHRAE Guideline 36, Data-Driven Model based MPC with weekly updates and online Deep Reinforcement Learning algorithm PPO demonstrated on a testbed simulating a 5 zone building setup.

The necessity of advanced adaptive control strategies that can adjust to building non-stationarities can be demonstrated by observing the performance of some existing traditional and state-of-the-art control techniques in building HVACs. Figure 1.2 compares the weekly energy consumption, comfort and VAV actuation rate in a building testbed across two different conditions: absence and presence of non-stationary behaviors due to exogenous variables like weather, zone setpoints and occupancy based thermal load. We independently apply an adaptive rule based control(Guideline-36[5]), Data-driven Model Predictive Path Integral(MPPI) Control which is based on models updated weekly[6, 7] and Online Proximal Policy Optimization(PPO)[8] under identical conditions to this building. We observed that, under the presence of non-stationary behaviors, energy consumption increases significantly(24%), occupant comfort is affected(59%), and VAV actuation rate increases(52%) causing increased for wear and tear for each of these control strategies. The MPPI, while

performing better than the rest due to the periodic updates of the data-driven models, can improve the performance by conditioning its updates based on detecting non-stationary building behavior changes. This makes it easier to understand the changed building conditions in a timely manner, instead of waiting until a periodic model update is performed.

However, these performance based updates bring some new challenges. Adaptive control needs accurate models of dynamic environments, and it is prohibitively difficult and expensive to build them using a physics based approach. While data-driven approaches can be used to partially mitigate this problem, there are other challenges associated with data scarcity, speed of adaptation, actuation wear and tear issues that need to be addressed. Numerous samples are needed to relearn these data-driven models, as well as sample trajectories to train the controller. Typically, the sampling rate of data in buildings are at least in 5 minute intervals, and any strategy relying on building data will have to wait for at least a day or two before a controller update occurs. This is not acceptable when the non-stationary behavior has to be immediately addressed. Hence, the adaptation must be fast. Finally, the controller training process might involve exploration with different actions to find the best possible strategy, and this can cause actuation related wear and tear issues if tested on a real system. Hence, the potential for improvement in adaptive control under these circumstances motivates us to develop our approach.

To develop control approaches that can accommodate building non-stationary behaviors, we review existing control techniques that apply to building energy management, and discuss their shortcomings in handling exhibiting non-stationary behaviors, which then leads to suboptimal building energy control. We then try to address these shortcomings inside a data-driven control approach.

Simple adaptive rule based controls(RBC) such as reset-schedule [9, 10, 11], Guideline-36[5] rely on feedback from a limited set of sensors like zone temperature, humidity, ambient weather conditions *etc.* and use some discrete if-then-else rules based on linear relationships for control. However, these rules do not adjust well to changing conditions triggered by exogenous variables, both, internal (e.g., occupancy changes, faults in equipment) and external (e.g., weather conditions) to the building. Besides, the rules used for building energy control are derived from simplistic models that often do not reflect the true conditions in the building.

Advanced control techniques like Model Predictive Control(MPC) and Reinforcement Learning(RL) based control are being increasingly applied to buildings research problems. These approaches generally need a model of the building where they can plan the best possible action *e.g.*: Model-Based Control in MPCs[12, 13], Model-Based Reinforcement Learning[14, 15] or learn the optimal control by interacting with the model *e.g.*: Model-free Reinforcement Learning[16, 17]). For either of these cases, the controllers need access to a sufficiently accurate model of the building that can simulate building non-stationary be-

havior. However, building accurate physics-based models for an entire building that can generalize well to non-stationary behavior is extremely difficult and time-consuming to create given the complex building geometries, and the inherently nonlinear energy flow models. Instead, simpler physics based models are used that approximate normal building behavior. While this works for nominal conditions, they fail to accurately simulate building non-stationarity, and controllers trained on them are not able to adapt well, leading to suboptimal performance.

An alternative to physics model based control is data-driven modeling and control, or commonly called data-driven control[18]. Here, real data from the building is leveraged to create data-driven energy models[19, 20, 21, 22, 23, 24, 25] of the building and a control strategy is learned to use those models[26, 27]. While existing works have applied them to learn control under nominal conditions, these models fail to accurately predict the building dynamics when it exhibits non-stationary behavior if it has not been encountered before in the training data. In the robotics domain, [7] suggested an iterative loop where the data-driven models are learned periodically and controllers are trained on them. But, for buildings, data sampling rate is not fast enough. Hence, scarcity of new data related to non-stationary behavior is an issue while relearning the models. Moreover, non-stationarities do not occur at regular intervals. They depend on certain exogenous variables. Hence, it is useful to update the models conditioned on the occurrence of non-stationarity.

1.1 Proposed Approach

Our proposed approach is a conditional deep-learning based modeling and policy optimization strategy, creating essentially an outer performance monitoring loop and an inner system-model and policy relearning loop.

The outer loop detects the onset of performance degradation due to non-stationarity in the real dynamic system using a performance metric and triggers the inner loop, constituting the offline relearning phase. Here, the dynamic system models for simulating the building are relearned on recent data with Elastic Weighted Consolidation based regularization [28] to prevent overfitting.

Next, the existing, deployed controller is retrained offline using these dynamic system models. To ensure a sufficient data for policy updates, the dynamic system model is simulated over a future horizon using forecasts of the exogenous variables. This is achieved using a set of exogenous variable predictors that forecast these variables over the required horizon length.

To reduce variance in expected return estimates, due to uncertainty in data-driven model predictions, we collect experiences using the agent on parallel copies of the environment. This provides an approach for generating diverse data and decorrelated transitions. This approach is particularly applicable to dynamic systems like buildings, where sequential transitions have a high degree of correlation between themselves

and do not convey important information over single trajectories. Once, the agent trains until convergence, it is re-deployed on the actual system.

1.2 Challenges in Data Driven Adaptive Control

Based on the preceding discussions, we realized that the process of deep learning-based modeling and control for complex building modeled as non-stationary systems has to address several important challenges, such as

- Achieving Energy Efficiency
- Maintaining Occupant Comfort
- Speed of Adaptation to Non-Stationary behavior
- Wear and Tear in System Components
- Achieving robustness

We discuss each of these challenges in greater detail below.

Achieving Energy Efficiency: The adaptive controller needs to ensure that the system behaves efficiently in terms of energy consumption, in response to its control actions, even when the building exhibits non-stationary behavior. This efficiency can be achieved in a piecewise manner, *i.e.* over intervals of time, as non-stationarity changes need to be detected during operations and adapted to. This entails the challenge of how to retain energy-efficient performance under changing building behavior. From practical experience of working on HVAC control, this can be an issue in complex structures like buildings, where non-linear relations between different variables keeps evolving under non-stationary conditions and become hard to model accurately. Hence, discerning the sole effect of control actions on energy consumption can be difficult.

Maintaining Occupant Comfort: The adaptive controller needs to ensure that the building conditions are comfortable for the occupants, even when the building exhibits non-stationary behavior. Similar to energy efficiency, this may be achieved in a piece wise manner. While occupants are not immediately aware of energy demands, they are highly aware of sudden changes in comfort, especially temperature fluctuations in buildings, during high occupancy periods as controllers have been known to not provide sufficient heating or cooling under such periods, leading to uncomfortable conditions. Hence, a balance needs to be attained between energy usage and comfort that is not properly addressed in previous work.

Speed of Adaptation to Non-Stationary behavior: The adaptive control needs to learn and decide the best control actions even as the building behavior is changing. This involves two distinct phases: (1) quickly

identifying the changes in the building dynamics, and (2) adapting to respond to those changes as soon as possible. Delays in detection and adaptation can significant affect energy use and occupant comfort.

Wear and Tear in System Components: We use the term *safety* to describe critical boundary requirements of the system. For the building, we are concerned with the wear and tear that might happen as a result of frequent adaptation and changes to the operative settings of the HVAC equipment in the buildings (see Figure 1.1 3rd column for PPO). These include damage to the plant actuator due to over actuation and internal wear and tear due to stress on certain system components. Model-free controllers usually interact with the system to learn better strategies when non-stationarities initiate the adaptation process. During this phase, exploratory actions of the controller can strain the actuators of the system and cause severe wear and tear[29, 30].

Achieving robustness: For a controller to be adaptive, it must be able to adjust to non-stationary behavior, which can be minimal to severe. During its adaptation phase, noise due to uncertainty in sensor data and subsequent data-driven model prediction limitations should not prevent the adaptive controller from learning optimal actions for the system.

1.3 Contributions of the Thesis

Our approach addresses the challenges that we outlined in Section 1.2.

We develop a deep RL approach for efficient control of complex, dynamic systems, in particular large building systems, that accommodates non-stationary behavior in the system under some continuity assumptions. Our approach includes two primary components: (1) Monitoring the building performance using a single metric like the deployment phase reward simplifies the requirement of tracking building non-stationarity due to unknown exogenous variables to monitoring a scalar combination of metrics like efficiency, safety, comfort *i.e.* the variables which we are of primary concern. (2) The performance based incremental updates to the data-driven models help the controller relearn on the most recent behavior of the building, thereby maintaining efficiency, ensuring comfort and safety of the system.

We design and develop a computational architecture that combines data-driven model learning and efficient controller retraining for non-stationary systems. It helps us address the data scarcity issues, learn the data-models relevant to the controller design without overfitting on limited data, and make the overall approach more robust to observation noise and a range of non-stationary behaviors.

We design and develop a reward function to support ”optimal” RL-based controller design. The reward function can be used to implicitly enforce the optimality conditions for the system without putting hard external constraints, which is typically used in optimization processes. This helps prevent the creation of

possibly non-convex optimization surfaces for the problem. For the building, it helps us incentivize lower energy consumption, better occupant comfort and lower wear and tear due to exploratory actions during controller retraining.

We have developed a hyperparameter optimization framework using a Bayesian Net approach that captures the relation between global and local hyperparameters of the approach and simplifies the optimization process by creating conditionally independent subsets of the search space. This helps reduce computational complexity for the tuning of the large number of hyperparameters associated with the approach.

We propose a systematic experimental framework using a standard 5-zone building testbed to study the effectiveness of the trigger-based RL control approach, and then extending our experimental studies to a real building environment. The testbed is parameterized and has well-defined interfaces to allow us access to energy related variables from the building and run a variety of experiments to empirically validate our approach. We extend our experimental studies to a real building, where we deploy it to control the whole building energy consumption and temperature comfort. We study these metrics along with the fan actuation of the zones to analyze actuation safety of our approach.

1.4 Organization of the Thesis

The rest of the document is organized as follows:

In Chapter 2 we review the existing approaches to modeling Building Systems for energy management, with greater focus on data-driven models and its potential application in control. We then review the existing supervisory control strategies actively described in buildings energy optimization research and critically review how they fall short under the non-stationarity assumption. We then discuss different continual learning paradigms that we wish to incorporate into our approach.

In Chapter 3, we formally state the problem we are trying to solve in the form of a Lipschitz Continuous Non-stationary MDP and motivate our solution approach.

In Chapter 4, we provide our solution architecture motivating the necessity and describing in details the individual components and how they combine to form the entire solution in the form of an inner and outer loop.

In Chapter 5, we describe the approach for systematic hyperparameter optimization for an approach with such a complex hyperparameter space.

In Chapters 6 and 7, we describe the application of our approach to a 5-zone building testbed and a real building, respectively. We map the different steps of our approach and discuss how they apply to the given application. Then, we demonstrate the results of the application of our approach. We discuss the results of the hyperparameter tuning and the resulting model architectures with their associated performance in terms

of common evaluation metrics. We then provide the results of benchmarking experiments where we compare performance of our relearning approach to existing state-of-the-art approaches in supervisory control. We conclude with the sensitivity studies for the hyperparameters.

In Chapter 8 we discuss the overall contribution, limitations and potential extensions of our approach.

CHAPTER 2

Literature Review

Building Energy Management(BEM) is the primary tool for controlling energy consumption while also ensuring the thermal comfort of a building. It does this by analyzing the weather conditions and building energy state using an array of sensors located throughout the building and using the embedded supervisory control logic that generates appropriate control signals and sends them to the different low-level PID controllers of the HVAC components (*e.g.:* Air Handling Units, Thermostats and VAV systems). In this chapter, we analyze the existing methodologies applied in supervisory control for buildings and motivate how deep reinforcement learning approaches can be developed to improve certain aspects of supervisory control as environmental and building operational conditions change over time.

In order to motivate our approach, we need to review the existing applications of supervisory control strategies in buildings which are commonly practiced in HVAC industry as well as being actively developed in the latest research. These include

1. Traditional Control Strategies commonly called Rule-based control
2. Advanced Adaptive Control Strategies like
   (a) Model Based Approaches where we rely on complete knowledge of the MDP to plan ahead
   (b) Model Free approaches where the strategy is gradually learned during a training phase by interacting with the system commonly using Deep Learning based Approaches

   Each of these approaches can learn on both physics based and data-driven models.

Based on the classification above, the first step in developing supervisory control for buildings is to plan or learn the strategy on a real building or an energy model that simulates the energy behavior of the building. Hence, we look at existing methodologies for modeling buildings and discuss their limitations when they are used to learning control strategies for a real building that exhibits non-stationary behavior. Next we discuss, existing control strategies that have been widely used in the literature and their limitations when applied to building control over extended periods of time. Finally, we review the state of the art in continual reinforcement learning, from where we borrow certain ideas to advance the application of supervisory control in buildings.

2.1 Modeling of Building Systems for Energy Management

Researchers often use different modeling techniques to analyze and control the energy behavior of building HVAC systems and their effects on the energy use and air quality in buildings. Modeling methods fall into three primary categories: (1) physics-based models, (2) data-driven models and (3) gray box models.

We briefly review existing methods for physics based and gray box approaches, and then dive deeper into data-driven approaches, since it is mainly used in our approach.

2.1.1 Physics Based Modeling of HVACs

Physics-based models rely on detailed physics-based or empirical equations derived from the conservation of mass and energy principles, as well as the geometry of the building structure. Since they are based on first principles and laws of thermodynamics, they tend to generalize well for most conditions. But building sufficiently accurate and detailed models is very time-consuming, complex and an expensive process. Hence, approximations are made to achieve tractability, which may result in decreased accuracy and applicability for these models.

While there is a large volume of literature [31] on HVAC modeling using energy based physics models, they have been primarily directed towards capturing behavior of individual components and subsystems of buildings as opposed to whole building energy models. For example, physics based approach have been used to model single zone-thermodynamics [32, 33, 34]. Physics based models have also been developed for heat exchangers [35, 36] to study cooling and heating efficiency in building zones, for chiller components like boilers [37, 38] and heat-pumps [39] to understand the heat transfer process between the buildings and exterior components that supply/remove heat from the building.

The absence of an ensemble model that can simulate whole building energy dynamics with sufficient details and accuracy is primarily due to the complexity associated with modeling all the details of the building HVAC thermodynamics. This requires thorough knowledge about the behavior of the building HVAC system, energy flow between the exterior and interior of the building, as well as energy flow between zones. This in turn involves collecting detailed information about the building thermodynamics and geometry. The second issue is knowing the correct model parameters and then simulating every component using these models synchronously with sufficient speed. Most applications where simulators like Energy Plus [40], TRYNSYS [41] or whole building energy models using Modelica [42] have been built, describe simple buildings with simple structures and ignore complex interactions present in a real building involving non-uniform geometry across zones, temperature preferences among building occupants, thermal loads during events *etc*. These limitations have prevented their application in learning optimal control for real buildings.

Another class of model based approaches commonly used to learn simple control problems are based

on the State Space modeling approach. If the data acquired from the system does not show much variation with respect to the process noise, then State Space Models can simulate the system behavior very well. The subspace state-space(4SID) modeling technique uses the system input and output data to derive the system and state matrices in a deterministic manner does not depend on the complexity of the system to be modeled. This method has been demonstrated for radiant cooling in buildings in [43]. However, there has not been much application of this technique due to the prohibitive requirement of a large amount of data that is needed to tune the matrices, especially for systems with large state-spaces such as buildings.

2.1.2 Gray Box Modeling of HVACs

Grey box models combine the use of physics-based models to model the system equations, but the model parameters are estimated from data collected from the system. In some ways they combine the advantages of physics and data-driven approaches, but in other ways, they also face the limitations of the two approaches.

Physics based equations have been used to create zone thermal models, but the parameters for these models are estimated from an actual system using methods like Genetic Algorithm in [44], system identification methods [45], the Least Square estimates [46]. Sequential Quadratic Programming was used in to estimate the parameters of an R-C circuit model in [47]. Other components, like chilled water coils [48], expansion valves [49], were developed using a combination of the least squares and simplex search processes.

Grey box models provide good accuracy than physics based models and better generalization capability compared to data-driven models, but they are also the hardest to develop. In order to develop gray box models, both the knowledge of underlying physical phenomenon and input-output data of the system is required. In the context of non-stationary building operation, the parameters for gray box models also need retuning when the operating conditions deviate from the training data in order to ensure higher accuracy.

2.1.3 Data-Driven Modeling of HVACs

Data-driven models employ statistical methods on data collected from the system to build operational models of system behavior. These models require little domain knowledge, and depending on the nature of the data collected, they may achieve good accuracy in modeling operating modes of the system. However, they suffer from generalization problems [50] and may have to be updated periodically to maintain their effectiveness. They can be broadly classified into the different categories as described below:

2.1.3.1 Statistical Regression Models

The statistical black box models consist of single and multivariate regression, autoregressive exogenous (ARX), autoregressive moving average exogenous (ARMAX), autoregressive integrating moving average(ARIMA),

finite impulse response (FIR), Box Jenkins (BJ), and output error (OE) models. These have been commonly used to create high level data-driven models for air-conditioning systems [51, 52], predict days ahead room temperature and humidity [53], model the behavior of heat pumps [54], compressor capacity and power consumption [55].

Since the dynamics in an HVAC system depend on their previous values, a time series regression model (i.e., ARX, ARMAX, and ARIMA) captures these correlations by including the process variables from the previous sampling times. This results in a very accurate model of the process dynamics, but the memory requirements increase in order to save the previous values. MIMO model identification using these methods requires many parameters to be determined, and this model tuning process requires experience as brute force tuning takes a long time to tune the model parameters, and this becomes an issue for non-stationary systems as they need to be retuned very frequently. Also, they fail to capture strong nonlinear dynamics in buildings like Neural Networks.

2.1.3.2 Data Mining Algorithms

HVACs systems and building energy flow models have highly non-linear dynamics. As number of advanced non-linear modeling techniques have been extensively used to model building energy like Support Vector Machines in [56, 57, 58], Classification and Regression Trees in [59, 60], Artificial Neural Networks in [19, 61, 62, 63]. While these methods have been shown to work well on simpler building configurations, neither of these models are able to perform well on complex building data. Hence, researchers started using ensemble models to capture building energy dynamics. Ensemble models have been developed to simulate HVACs in [20] and have been able to outperform individual models in capturing the energy behavior.

However, most of the above-mentioned techniques have assumed that there does not exist a time correlation between the variables and considered it simply as a prediction problem on tabular data. But HVACs are dynamic systems that incorporate the notion of state to reflect the dynamics of the system behaviors.Hence, models like Long Short-Term Memory Networks(LSTM) [64] must be used to develop data-driven models that truly capture the behavior of these dynamic systems. Also, with the advent of deep learning, Deep Neural Networks gained popularity in modeling building energy. In [21, 22], the authors abstracted multiple HVAC components and zone thermal models using deep neural networks to learn the HVAC dynamic behaviors. Deep neural networks have also been used to perform feature engineering to improve the prediction accuracies in benchmark building energy datasets in [25].

There has also been some recent work on data-driven models for building energy forecasting [23] and [65] where they predict days ahead energy demand using sequence to sequence based LSTM models and combination of CNN-LSTM architectures respectively. These approaches demonstrate that it is possible to

forecast building energy over a longer horizon and can be leveraged to our advantage when we need to predict future behavior after the occurrence of non-stationarities.

Most of the work discussed, assume that the building energy behavior is stationary and a model, once learned can continuously predict energy behavior accurately. But depending on weather conditions and occupant preferences, the building can exhibit non-stationary behavior due to change in its control logic. Hence, it is important to update these models whenever the non-stationary building behavior causes a distributional shift in the data. However, there has been very little work in literature that have tackled this issue. Adaptive LSTM models have been constructed in [66] to update LSTM models at regular intervals. While regular updates work in theory, they need to be updated mostly when the building exhibits non-stationary behavior that hasn't been modeled before. Hence, it is desirable to update these models based on some conditions.

2.1.4 Summary

Overall, we observed that data-driven models are better suited for modeling real building energy dynamics to capture the non-linearity and other complexities. However, a 2021 study [67] found that most control research performed on buildings still using use reduced order state space models.One explanation for this is that the theory around state-space models is well established, and they are commonly used in MPC applications in many industries, and in particular the process industry[68, 69]. However, data-driven approaches aimed at capturing the building thermal dynamics are seen to be increasing in number over the last few years. But they claim that, most work, are not able to present the results of the control approach in the context of error in these data-driven models over longer horizons [70] and the data-driven models are rarely validated on real buildings. In several approaches related to modeling neural networks [71, 72] for the purposes of designing control applications, it was found that model pruning and regularization is also an essential step to ensure that the model does not overfit (*i.e.:*, generalizes poorly to unseen data). In our thesis research, we study the prediction accuracy of extended data-driven models that accommodate non-stationary behaviors.

2.2 Supervisory Control Strategies for Buildings

Control strategies in buildings can be classified into two groups [73]: Traditional Control Strategies(TCS) and Advanced Adaptive Control Strategies(ACS). The focus of our research is on supervisory control. Therefore, we review the literature in this area. We assume the presence of traditional lower level PID control for all of our systems.

2.2.1 Traditional Control Strategies

Supervisory control under TCS is usually called sequence-based-control [74] where the controller uses feedback from the environment and building sensors like temperature, humidity etc. and generates on-off signals. Due to its limited expressiveness, it has been used very little, with only applications in [75] to control room heating and cooling coils in small buildings. According to [75, 76] these control strategies have simple structures that are easy to implement, quick response times and low initial costs. But, they lack in terms of accuracy, energy-efficient performance, ability to control non-linear moving processes with time delays, complex system with uncertain information systems. Also, they are not capable of handling dynamic information/input [77] and most importantly, they have been found to be unable to adjust to changing building conditions as do not interact with the external environment (i.e. user/ grid/ district/ city) [78] which precludes the application of high-efficiency control. Hence, it is imperative to use more advanced control frameworks for large, complex buildings.

2.2.2 Advanced Control Strategies: Model Predictive Control and Reinforcement Learning

Adaptive Advanced Control Strategies, is defined as a group of control techniques which can adjust to unforeseen conditions in the system at run time [73]. They typically employ monitoring of system data to ”re-calibrate their model of the system” and ”update their control logic or parameters”. Different categories of ACS have been proposed in literature. 1) *Soft Computing Techniques* like Reinforcement Learning [79], Artificial Neural Networks [80], Fuzzy Logic [81] used agent based control and have inherent ability to deal with uncertainties, noisy data in the system and are applicable to complex HVAC system control. 2) *Hard Computing Techniques* like Model Predictive Control [74, 82]. While these models provide quick accurate response, they can suffer from model uncertainties and need an accurate prediction model for all purposes. 3) *Hybrid Control Techniques,* which rely on a combination of the above two techniques for better control. While there are not many applications of hybrid techniques of this type in HVAC system, similar applications have been found in the domain of vehicle automation in [83, 84] and in robotics in [6].

According to the survey conducted in [73] MPCs account for more than 50% of the research applications in HVAC control literature with Deep Reinforcement Learning based approaches gradually gaining prominence in the domain. So we mainly review the existing literature having application of MPCs and Reinforcement Learning based Control to buildings.

2.2.2.1 Model Predictive Control: Approaches based on Physics-based Models

Most applications of Model Predictive Control on building HVACs have focused on achieving energy savings and optimal comfort conditions by using physics based models to plan and find the best control strategy.

Hence, accuracy and generalization of the model is important, as it needs to simulate it over long prediction horizons. In the literature, MPCs have been mostly applied in simple configurations in physics based models of small buildings and have demonstrated significant energy savings by utilizing the thermal capacity of these buildings as passive storage of thermal energy [85, 86, 87, 88, 89]. They heat or cool the buildings during off-peak hours and try to store that energy for the entire duration of the building operation [90, 91]. MPCs can also handle ad-hoc preference changes in the zone comfort requirements [92] and based on this an interactive Building Automation and Control System(BACS) was proposed in [93] by demonstrating it on Energy Plus models. However, in experiments conducted by [70, 94], it was observed that MPCs can sometimes be difficult to implement online for large-scale buildings since building accurate physics based models on a large scale can be difficult and solving a large-scale problem at each time step can lead to delay in issue of control action.

2.2.2.2 Model Predictive Control: Approaches based on Data-driven Models

In many cases, the performance of MPCs have been demonstrated using data-driven models. Models based on neural networks have been used for planning by MPCs to control and optimize energy consumption in buildings in [95, 96, 97]. They mainly assumed that the data-driven models need only be trained once and can be used for reasonable horizons of up to 1 or 2 days to plan and find the best control actions to save energy. Similar applications in optimizing comfort with neural network models was performed in [98, 99]. Other MPC applications based on data-driven models involving ensemble of Regression Trees [100, 101], Gaussian Processes [102] also showed that MPCs tend to perform better compared to applications involving a single data-driven model described above. However, most of these applications have been restricted to modeling simple and uniform building architectures. They have been shown to perform suboptimally when applied to real buildings with complex structures where the data-driven models are not generalized. This happens due to the fact that in large buildings, increase in the state space can lead to data-driven models not capturing the behavior of the building across a large state space properly, leading to the degradation in control performance and issues in convergence of the optimization as shown in [103, 104]. For real buildings, having well calibrated models can be infeasible, as discussed before, and limits extensive application of MPCs as a supervisory controller for a large HVAC component like Air Handling Units. These existing issues have limited the application of MPCs to small-scale optimization problems for building HVACs.

To adopt wide scale application of Adaptive Advanced Control Strategies to real buildings, researchers need control strategies which can deal with non-stationarities, modeling accuracy issues and limited domain knowledge about the system. Hence, we tend to rely on model-free control strategies, *i.e.* the controller does not plan using the model over a long horizon. Instead, it interacts with the model(data/physics based) and

iteratively learns an efficient control strategy and this is called Model-Free Reinforcement Learning [105]. Deep Reinforcement Learning(DRL) based model-free control have seen increased use in research problem involving supervisory control for buildings. This started mostly when simple function approximations in reinforcement learning started to be replaced by deep neural networks, leading to the application of deep reinforcement learning in buildings. The use of neural networks made the problem partially simpler because buildings are non-linear and usually continuous dynamics systems and previous tabular RL techniques could not scale well for such domains. We review the current literature and identify the key points that enable us to extend the existing RL applications to our relearning framework.

2.2.2.3 Reinforcement Learning based approaches using Physics Based Models

Until now, most of the works in the application of DRL have been demonstrated on Physics Based Models of the building. One of the earliest Q-learning [106] neural network-based function approximations for HVAC thermostat control was developed in [107] using *EnergyPlus* where the thermostat decided whether the heater should be turned on or off in response to occupancy and room temperature conditions. This demonstrated an energy savings of 10% over a programmable thermostat whose schedule was developed by the building manger. Based on the success of Deep Q-Learning (DQN) [108], a number of methods were proposed that used off-policy DQN and its variants to create a supervisory controller for buildings. DQN was used to learn a supervisory controller that can actuate VAV dampers in [16] to save energy and ensure comfort in an *EnergyPlus* model of a multizone building. They obtained energy savings ranging from 22% to 71% compared to the baseline controllers under different building configurations. A supervisory controller scheme based on DDPG [109], a variant of DQN, was developed for cooling a datacenter in [17] which obtained an energy savings of 11% compared to two baseline controllers called DefaultE+ which uses some manual reset schedules to attain zone set point requirements and the TS control optimization algorithm which uses genetic algorithm to find the best control scheme. A completely off-policy form of RL called Batch Reinforcement Learning was used in [110] to maximize energy savings and maximize costs in a physics based simulation of a smart grid setting. It reduced peak energy demand in a smart building by 30% and gained approximately 15% more profit by selling energy during peak hours when compared to a naïve, greedy policy. Asynchronous Advantage Actor Critic(A3C) was applied in [111] to control the hot water supply temperature in discrete ranges from no heating to 65^oC to heat an office building. They learned the control on an *EnergyPlus* and then deployed it on a real building and obtained 16.7% when compared to a rule-based controller. One of the latest works in creating an RL framework for whole building energy control was tackled in [112]. Here the authors applied the state-of-the-art policy gradient based RL method called PPO [8] to a physics based model of a 5-zone building which uses the Standard VAV system with a central boiler and a central chiller for

providing the necessary zone temperature requirements. Overall, the authors show that using RL, they attain 22% lower energy consumption compared to the baseline Energy+ default controller.

2.2.2.4 Reinforcement Learning based approaches using Data Driven Models

There have been very little applications of data-driven models being used to train reinforcement learning agents in the literature. Gaussian process models were used to simulate building energy consumption in [113] and learn a control strategy using Dyna-Q. They obtained comparable energy savings to a MPC based approach. In [114], Q-learning is applied to a data-driven model of a building created using a combination of Neural Networks and an ensemble of Extreme Learning Machines(ELMs) [115] models. The proposed approach was able to achieve results within 10% of the true optimum.

2.2.3 Summary

The literature review covered a variety of approaches from heuristic methods to model-based and data-driven methods that have been applied to supervisory control in buildings.

Traditional control strategies are not sustainable for efficient operation because of the complex dynamics of building systems, as they are only guided by simple rules that cannot adapt to the dynamics of the building behavior. Rule Based Adaptive Control(RBC) strategies improve upon them by including more feedback variables, but such rules fail to respond to feedback components like energy, comfort, actuation safety which are non-linearly related to the building behavior. Hence, more advanced control strategies are used, out of which Model Predictive Control and now Reinforcement Learning are most prominent and have achieved similar levels of success. Both MPC and RL have great potential for application in buildings, with each having some edge over the other depending on the circumstances.

MPCs cannot tolerate imperfections in the model, which it uses for predicting future behavior and then determining optimal actions. The longer the horizon, the more pronounced is the issue. For complex systems involving high dimensional state space, MPC has been observed to suffer from divergence issues [116].

Reinforcement Learning on the other hand bootstraps multiple experiences to update its control policies and with appropriate techniques(like implementing advantage estimation, discount factor, parallel environments with multiple agent workers). Therefore, it has demonstrated better success in handling complex dynamics, imperfections like noise(variance) in the data-driven model predictions while estimating the ”optimal” control action. Experimentally, it has also been observed that MPCs can be less robust than simple Q-learning algorithms [117]. However, applications of Reinforcement Learning to buildings have highlighted some potential drawbacks. These can be attributed to simplifying assumptions like uniform temperature requirement, no unexpected events that reduce model accuracies even while learning online. Moreover, most

methods have completely ignored the need to relearn iteratively due to the non-stationary behavior in buildings [111].For most of the reported work, the results have been demonstrated on *EnergyPlus* models without extending results to real building environments. Moreover, the results reported consider year-long aggregate behavior when showing improvements, but non-stationarities affect building behavior over shorter durations, and it is important to compare performance during these intervals to the same during nominal building operations. Also, some approaches consider learning directly online [16, 17]. There is a valid concern for effects of exploratory actions, and this has not been addressed in any of these papers. In some cases, the reward functions are highly reactive and clearly dictate the control policy without letting the agent explore much. In [110] it is evident that the reward function directly helped the agent to maximize the profit and on top of that helped it reduce peak energy demand even though that component was not directly part of the reward.

Apart from the drawbacks, we made one additional observation regarding the components of the reward function used in these papers. Most of the reviewed literature have a common set of metrics as a part of the reward function. These include energy consumption, comfort either as deviation from set point or using PMV index. Only [16] seems to use actuation constraints, but it is introduced artificially outside the RL framework. These provide us some fair idea as to the possible candidates for the reward function in our framework.

The above examples on the application of supervisory RL control to buildings demonstrate that adapting to non-stationarity in the system behaviors has not been fully tackled unless the control is learned online, which is infeasible in most real buildings due to the safety concerns associated with exploratory actions and sample inefficiency faced by DRL approaches. Hence, we need to explore ways in which data-driven control for reinforcement learning can be used in an adaptive manner. Continual learning, and it's relevance in some Reinforcement Learning applications, has been a well-studied topic. We review some ideas, that we wish to use, in advancing existing RL techniques for buildings so that they can be applied to real buildings in a life-long fashion.

2.3 Continual Reinforcement Learning in Non-Stationary Environments

Most of the research in reinforcement learning literature demonstrate performance on tasks where the system is assumed to be stationary, *i.e.*, the RL agents are trained in environments where observations can be considered to be sampled from the entire possible spectrum of the observation space during training itself. This helps in generalization. But achieving generalization becomes an issue in real-world conditions where the environment can change in unexpected ways causing non-stationary behaviors in the system [118] leading to unaccounted changes in the goals and dynamic behavior [119]. To this end, researchers have proposed the continual learning paradigm [120] for Non-Stationary Markov Decision Processes(NSMDP). We investigate Reinforcement Learning for NSMDP systems in this thesis research.

2.3.1 Classical Approaches in Continual Learning

One of the earliest continual learning agents CHILD(continual, hierarchical, incremental learning and development) was developed by [120] which could learn existing tasks using neural networks and transfer that knowledge to newer tasks by augmenting and sometimes amending its policy to learn faster than an agent from scratch on that new task. Other classical approaches to the continual reinforcement learning problem involve detecting the changes in the environment behavior and fine tune the target policy after changes have been detected [118, 121]. Still, others [122, 123, 124, 125] have introduced a context-based regret minimization approach where they have access to the time varying MDP with its corresponding reward functions and transition probability matrices, and the goal has been to minimize the sum of missed rewards over a finite horizon. Here, the missed rewards are obtained by comparing a stationary policy to the best time-dependent policy for each context. Naturally, the overall best policy is only known in hindsight and not useful for future applications because future contexts are not known beforehand.

2.3.2 Continual Learning in a DRL setting

With the advent of deep reinforcement learning, several new problems arose when applying neural networks as function approximations for non-stationary environments. Neural networks tend to suffer from catastrophic forgetting when subjected to continual learning when trained on environment state distributions that are distinct from each other. The notion of detecting changes in behavior, beforehand, for high dimensional continuous non-stationary environments is also a problem. The research on deep reinforcement learning in non-stationary environments is still in an early-stage and new methods are being proposed by using ideas from continual learning. One of the common methodologies advocated for continual learning in non-stationary environments is through meta-learning. It has been used in combination with importance sampling in [126] to demonstrate fast few-shot adaptations between distinct and sequential pairs of environment changes with distinct boundaries. The proposed method seems to perform well on non-stationary environments, but when the non-stationary behavior evolves over a large time period(samples), the meta-learning methods lag on-policy methods, such as PPO [8]. Also, they rely on knowing clear boundaries between unique tasks, which requires identifying task labels or contexts. Schulman et al. [127] efficiently borrow ideas of model-free meta reinforcement learning and incorporate it into mode-based reinforcement learning. Model based RL is more sample efficient and the incorporation of meta learning into dynamic models helps in fast adaptation to local changes in the environment. They also make the model learning more robust by ensuring that corrupt data does not degrade the trained policy. The approach, however, still relies on the distinct boundaries between unique environment conditions.

Another group of methods selectively updates the agent policy network between changes in the environ-

ment by importance weighting the policy network using the idea of Memory Aware Synapses [128] which uses the Fisher Information(FI) based on the data from the changed environment to decide which parts of the policy network to update when the agent encounters a change in the environment effectively creating a regularization for incremental updates to neural network. This update could happen when the boundaries between these changes are distinct as shown in [129] or when the agent is focused on a single task that changes slightly with time and the goal is to identify it as feedback in the form of the supervised signals as in [130] or in the form of rewards as shown in [131]. In our opinion, [131] represents a truly continual learning paradigm as it does not distinguish between tasks using any other parameterization other than the reward itself and has demonstrated this on non-distinguishable task boundaries which they claim to be the hardest problem to solve. For our work on buildings, we also plan to use the reward as a metric to determine the occurrence of non-stationarities due to exogenous variables. Also, [132] proposed a continual learning framework where they retain a history of policies and constrain the policy optimization by keeping the new policy as close as possible to the discounted weighted sum of the older policies. They show that this achieves stability as well as prevents catastrophic forgetting compared to the baseline clipped PPO implementation. We wish to achieve this quality of stable behavior and prevent catastrophic forgetting by reusing some past data along with the new data along with the Elastic Weighted Consolidation(EWC) [129] regularization while learning our data-driven models as well as policies if needed for data-driven control.

Thus, we realize that continual learning provides several methods to improve existing RL methods in non-stationary systems. We are mainly interested in learning quickly *i.e.* not training from scratch and do so from limited demonstrations *i.e.* without over-fitting to new behavior. Hence, we plan to borrow the ideas of Memory Aware Synapses to update our data-driven models simulating the dynamic system faster without catastrophic forgetting. Based on existing gaps in literature to address the non-stationarity problem and ideas from continual learning, we propose our research plan next for real world data-driven control problem.

2.4 Overall Summary

Based on our literature review of the three research areas, we summarize the issues that we wish to address in our approach.

We aim to maintain energy efficiency and occupant comfort, minimize wear and tear of HVAC components during occurrence of non-stationary changes in the buildings. We wish to ensure quick online detection of the onset of building non-stationarity.

We have to model environments where Dynamic Systems exhibit non-stationarities. During this modeling process, we have to address certain issues. These include: (1) Addressing the complexity in data-driven modeling of large HVAC systems by considering the data as dynamic rather than tabular. (2) Handling data

scarcity issues when non-stationarities occur, and prevent overfitting. (3) Ensuring sustainable accuracy for data-driven model predictions over long prediction horizons.

Finally, we have to design Supervisory RL Controllers for Dynamic Systems. This would involve calculating optimal control given the complex interactions between system components, handling errors in the environment models, and generating enough samples/experiences from the environment so that the controller does not suffer from sample inefficiency.

CHAPTER 3

Approach

Our overall DRL approach to relearning control for large buildings that exhibit non-stationary behaviors combines an outer loop and an inner loop scheme. The outer loop monitors building performance, and when the performance shows a consistent downward trend, in particular, due to non-stationarities, the inner loop is activated offline to relearn the building models and subsequently retrain the control policy, so that it is a better match to current building conditions. The inner loop includes two primary steps: (1) Build predictive, dynamic data-driven models for building energy consumption from relevant experiences of building energy data; and (2) Use a Deep RL methodology to learn a control policy by interacting with the dynamic data-driven models and deploy this controller for building energy management.

Unlike previous work, the proposed approach decouples the controller training and deployment problem by relearning offline using the inner loop. This reduces safety issues that may arise because of exploratory behavior online to learn controller behaviors. These are exhibited in common RL approaches like PPO [8] and DDPG [133]. The offline relearning framework allows us to simulate relevant behavior of the building under non-stationarities by using relatively higher sampling rate of the data-driven components constituting the system model and thereby relearning much faster. On real buildings, learning will be severely limited by the sampling frequency, which tends to be much lower. Furthermore, as discussed later, use of a forecast horizon to generate enough trajectories and multienvironment training helped generate diverse samples for better training of the RL controller. The outer loop reduces the constant overhead associated with lifelong reinforcement learning commonly associated with previous approaches.

Next, we formally state the problem by modeling non-stationary building behaviors as a Lipschitz Continuous Non-stationary Markov Decision Process. We state our assumptions to solve the problem and justify these assumptions in the context of building energy management.

Before we introduce Lipschitz Continuous Non-stationary Markov Decision Process, we define a Non-stationary Markov Decision Process.

Definition 1 (Non-Stationary Markov Decision Process (NS-MDP)) *A non-stationary Markov decision process is defined by a 5-tuple: $M = \{S, A, \mathscr{T}, (p_t)_{t\in\mathscr{T}}, (r_t)_{t\in\mathscr{T}}\}$. S represents the set of possible states that the environment can reach at decision epoch t. A is the action space. $\mathscr{T} = \{1, 2, ..., N\}$ is the set of decision epochs with $N \leq +\infty$. $p_t(s'|s,a)$ and $r_t(s,a)$ represent the transition function and the reward function at decision epoch $t \in \mathscr{T}$, respectively.*

The above formulation differs from a standard MDP, where the transition dynamics and the reward function do not change with time or decision epochs. Hence, the best policy π^* for an MDP would be stationary. In the NSMDP case, the optimal policy for a NSMDP, π_t^* will be non-stationary as it depends on the dynamics and rewards of each epoch. Now, learning optimal policies from Non-Stationary MDPs is particularly difficult for non-episodic tasks because the agent is unable to explore the time axis at will. However, if the non-stationary system has slow time constants and does not change arbitrarily fast over time, its behavior can be formulated by applying the *regularity hypothesis* that can be formalized using the notion of Lipschitz Continuity (LC) applied to the transition and reward functions of a non-stationary MDP [134]. This results in the definition of Lipschitz Continuous NS-MDP (LC-NSMDP) as:

Definition 2 ((L_p, L_r) -LC-NSMDP) *An (L_p, L_r) -LC-NSMDP is a NSMDP whose transition and reward functions are Lipschitz Continuous with respect to time and are constrained by*

$$W_1(p_t(s'|s,a), p_{\hat{t}}(s'|s,a)) \leq L_p|t-\hat{t}|, \ \forall\ (t,\hat{t},s,s',a) \tag{3.1}$$

$$|r_t(s,a,s') - r_{\hat{t}}(s,a,s')| \leq L_r|t-\hat{t}|, \ \forall\ (t,\hat{t},s,s',a), \tag{3.2}$$

where W_1, the Wasserstein distance is used to quantify the difference between two distributions. L_p and L_r and the associated Lipschitz bounds on the change.

$t, \hat{t} \in \mathscr{T}$, where $\mathscr{T} = \{1, 2, ..., N\}$ is the set of decision epochs with $N \leq +\infty$. $s, s' \in S$. S represents the set of possible states(possibly continuous and infinite) that the environment can reach.

Although the agent does not have access to the true NSMDP model, it is possible for it to learn a quasi-optimal policy by interacting with temporal slices of the NSMDP assuming the LC-property. This means that the agent can learn using a stationary MDP of the environment at epoch t[134]. Therefore, the trajectory generated by an LC-NSMDP $\{s_0, r_0, ..., s_k\}$ is assumed to be generated by a sequence of stationary MDPs $\{MDP_{t_0}, ..., MDP_{t_0+k-1}\}$. The LC assumption translates to our relearning framework as the following condition, i.e., the temporal distances of the transition function and reward in equations 3.1 and 3.2 are bounded between time slices.

Buildings exhibit slow temporal dynamics with large time-constants that are associated with most thermodynamic processes in the HVAC systems and building energy flow. This is generally assumed to be true under the influence of exogenous variables like ambient temperature, relative humidity, setpoint preferences and thermal loads. As such, the changing behavior due to non-stationarity can be modeled as continuous mode changes due to exogenous variables that occurs over a relatively large time duration δ. This allows

us sufficient time to detect the onset of non-stationarity using the outer loop, followed by the relearning of the models and subsequently the RL agent in the inner loop. Hence, we formulate the building as a Lipschitz Continuous non-stationary Markov Decision Process(LC-NSMDP) that can be learned by RL based model-free interactions in the duration δ of the mode changes. The existence of Lipschitz Bounds are reasonable assumptions for building transition function T under the influence of the exogenous variables except faults. Building thermodynamic behavior does not undergo unbounded changes within short periods of time. Hence, the "incremental" learning of building energy models is feasible between changes given sufficient information about the new behavior.

Next, in Chapter 4, we present the implementation details for the individual components of this approach in the context of the loops defined above.

CHAPTER 4

Solution Architecture

Based on our approach, we shall now develop the solution architecture for a condition-based modeling and control approach for a Lipschitz Continuous Non-stationary Markov Decision Process. As discussed before, there are two loops in the solution: an outer performance monitoring loop and the inner model and control relearning loop. A simplified overview of the scheme is provided in figure 4.1. The outer loop comprises a monitoring module that will track the relevant performance to determine whether relearning is required. The inner loop will use recent data along with an array of techniques described later in this chapter to perform system model relearning and control adaption in an optimal and time efficient manner.

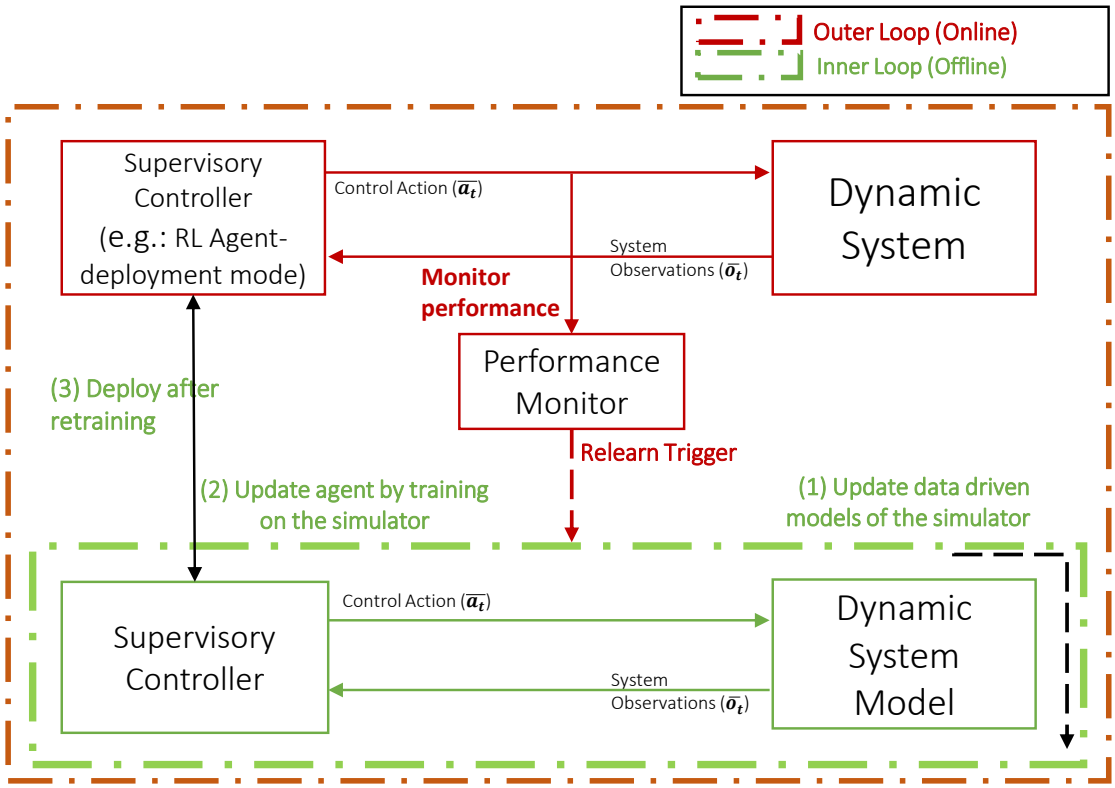

Figure 4.1: Overview of the solution using an inner and outer loop schema

4.1 Outer Loop

We first describe the operation of the outer loop where we briefly talk about the interaction between the *Supervisory Controller* and the *Actual Dynamic System* online and how the performance monitoring module is used to monitor performance.

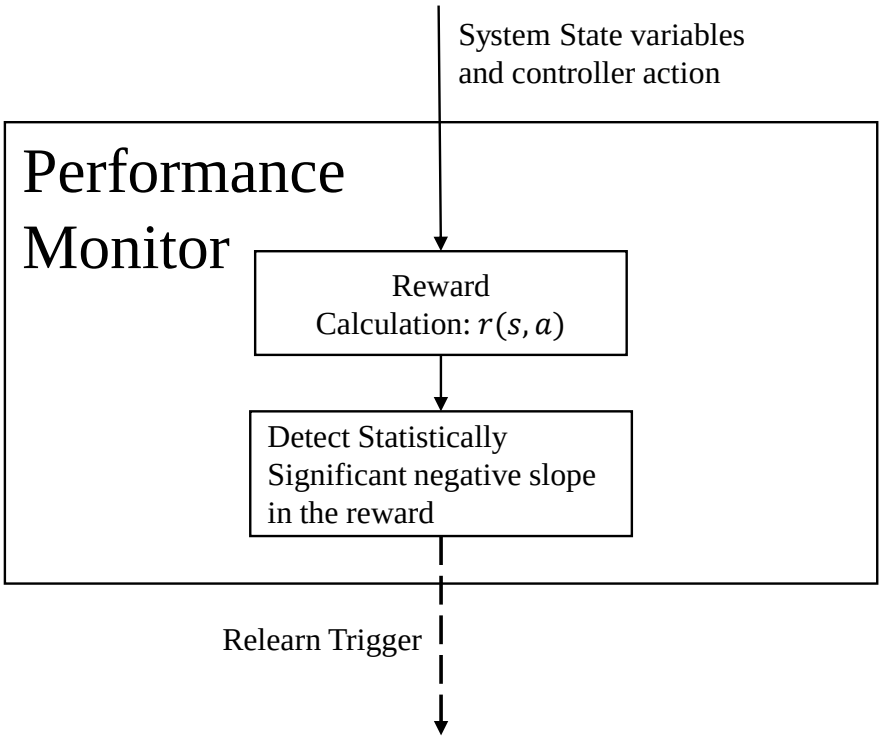

Figure 4.2: Performance Monitor Module

4.1.1 Performance Monitor Module

The reinforcement learning *Supervisory Controller*, which is used to control a real system, needs to train, on a *Dynamic System Model* in order to perform optimally. But, since, the real system exhibits non-stationarity, the controller policy needs to adapt from time to time.

Hence, it is of paramount importance to identify system non-stationarity in a fast and efficient manner so that the policy can adapt to the changes as soon as possible. In this work, we use the *Performance Monitor Module* that tracks a calculated reward signal when the *Supervisory Controller* is interacting with the real system, as shown in Figure 4.2. For simplicity, this reward formulation can be augmented or exactly identical to the reward function used as a part of the Lipschitz Continuous NS-MDP formulation. We assume that whenever the controller performance deteriorates, it is being reflected in this reward signal by an overall negative trend. Since it is a simpler scalar time-series signal, we estimate the fit of the negative trend using the technique described in [135]. It states that the trend is statistically significant if the R^2 coefficient of determination is greater than 0.65 and the $p - value$ is less than 0.05. More complex non-linear trend detection methods may also be used and has been recently demonstrated in a similar application in [136].

The performance of this module depends on the window over which we track the reward. As such, we introduce window sizes W_{pm_i} where $i \geq 1$ depending on the specific application. This is done in order to track

slow moving as well as fast moving non-stationarities.

Once, the performance monitor module is triggered, it initiates the inner loop which is discussed next.

4.2 Inner Loop

Our solution architecture for the inner loop includes simulating the real system using multiple data-driven models that predict relevant state variables in the LC-NSMDP. These models are relearned whenever the outer loop triggers the relearn phase. Subsequently, a copy of the deployed reinforcement learning based controller is retrained on the system model(composed of the above data-driven models) to adapt to the latest behavior. This concludes the inner loop, after which the controller is redeployed on the actual system.

We now motivate and provide a detailed description of the different components of the solution to the problem and how they are tied up into a relearning architecture.

4.2.1 Data-driven Modeling of the Dynamic System

When advanced control strategies are applied to real world systems, a predictive model, whether model-based or data-driven, is required to select appropriate control actions to optimize system performance. The complexity of large building systems makes it prohibitively expensive to develop accurate physics-based models [99]. Therefore, data-driven techniques are being increasing used to develop dynamic models for large, complex systems.

Therefore, an important component of our approach is to develop *Dynamic System Models* (*DSM*) of the system using one or an ensemble of data-driven models that estimates the next state $\bar{s}_{t+1}$ of the system provided the current state$\bar{s}_t$ and inputs $\bar{u}_t$ in the form of actions from a supervisory controller $\bar{a}_t$ and exogenous variables $\bar{d}_t$. The purpose of this model is to simulate the transition dynamics $p_t(s'|s,a)$. The general schematic, of our data-driven approach to construct a state space model of the dynamic system, is shown in Figure 4.3. Since, these models simulate state variables in a dynamic system, we develop recurrent neural network Long Short Term-Memory networks(LSTMs) to capture the time-correlation between inputs and outputs and Fully Connected/Dense Neural Networks(FCNN) to capture non-linearity of the processes. The specific network architecture of the deep learning models are obtained by a hyperparameter search process associated with the models.

Furthermore, given the non-stationary behaviors in the operating regimes of our system, it is necessary to retrain the deep learning networks, when non-stationary changes occur in the system. Retraining with limited data available after the non-stationary change may cause *overfitting*, and as a result, *catastrophic forgetting* [129]. To prevent this, we adapt a regularization process termed Elastic Weight Consolidation (EWC) proposed [129]. This process is effective in retaining previously learned behavior from older data, while

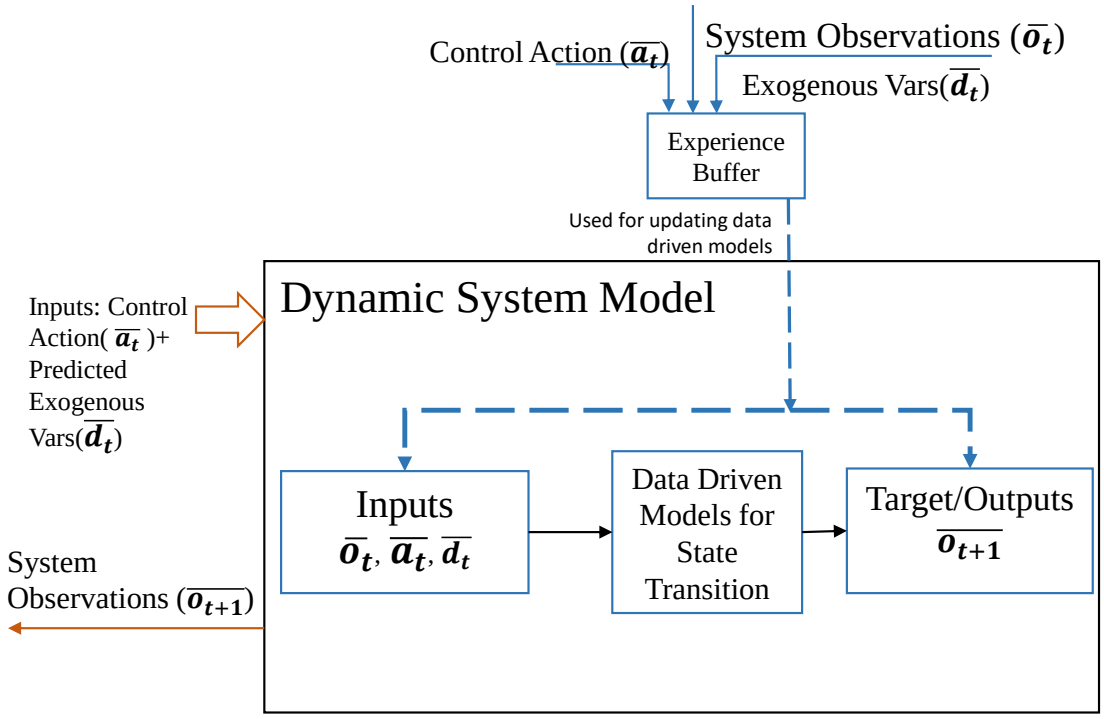

Figure 4.3: Dynamic System Models

avoiding long learning times. The loss function for the network training updates is modified to incorporate the regularization term:

$$\mathscr{L}(\theta) = \mathscr{L}_{old+newdata}(\theta) + \sum_{i \in \theta} \frac{\lambda}{2} \mathscr{F}_i(\theta_i - \theta_i, old)^2 \tag{4.1}$$

The first loss term simply indicates the training loss on the old and new data (from the Experience Buffer) after the non-stationary change. The second term penalizes weight changes scaled by the Fisher Information Matrix ($\mathscr{F}_i$) on the new data for each parameter i of the network θ. λ is the hyperparameter that determines how much weight we want to assign to learning the new behavior. In our work, we use the value of $\lambda = 400$. Since we need to evaluate the Fisher importance matrix for all the data points, we use a small data set for these incremental updates for faster computation. This requirement aligns with the provision of limited data available during relearning.

4.2.2 Experience Buffer

Since we consider data-driven models, we need to provide training data containing the input variables for the models. This includes variables, like $\bar{a}_t$, $\bar{s}_t$ and $\bar{d}_t$ and the target data to estimate $\bar{s}_{t+1}$. We collect and use

real data from the actual pre-existing controller-system interactions. This data is continuously collected and stored in an *Experience Buffer* modeled as a FIFO queue with queue length M_e. The *Experience Buffer* helps us retain the latest data from the real system so that the models can be adapted to the latest system behavior if needed. Choosing an optimal value for M_e is an important step in the overall approach.

4.2.3 Supervisory Controller: Deep Reinforcement Learning Agent

In order to have an optimal policy π_t^* for each epoch of the LC-NSMDP, we need a controller that can interact with the real system in an efficient, fast and safe manner and, when needed, adapt to the changes in the real system transition dynamics. It needs to utilize feedback from multiple indicators of non-stationarity as a part of the state space $\bar{s}_t$. In order to adapt safely, it needs to retrain offline on a *Dynamic System Model* and handle a certain degree of inaccuracy commonly associated with data-driven *Dynamic System Models*. Furthermore, it needs access to numerous and a diverse set of sample transitions in order to converge faster to an optimal policy. Based on these requirements, we choose to use policy gradient based reinforcement learning algorithm as a *Supervisory Controller* with some augmentations, as we shall discuss now in details.

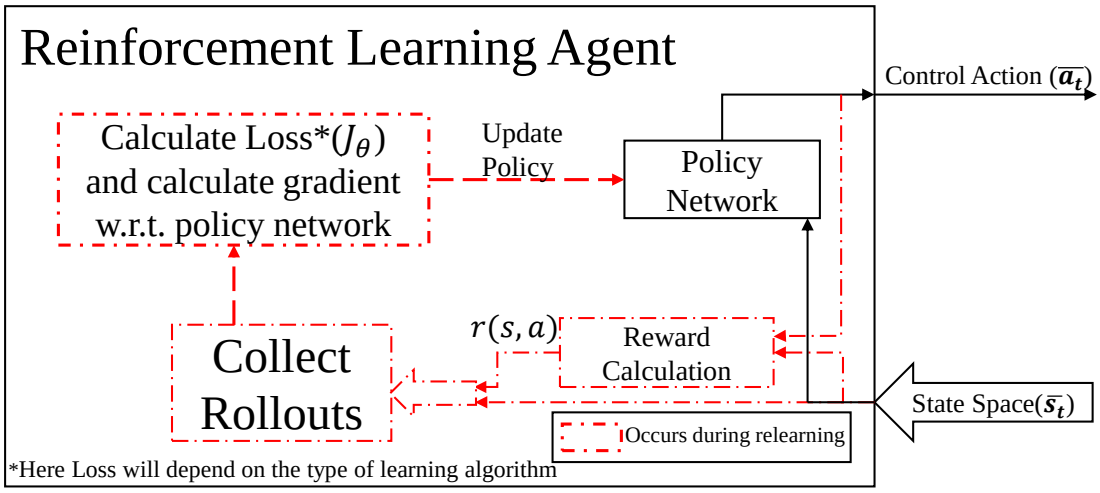

Figure 4.4: Training of the *Supervisory Controller* implemented as an RL agent

For our work, we consider a policy-gradient [137] based Deep Reinforcement Learning [105] *Supervisory Controller*, as shown in Figure 4.4. It has a Policy Network parameterized by θ. It takes as input the current state of the MDP, $\bar{s}_t$ which comprises observations $\bar{o}_t$ from the system and certain exogenous variables in $\bar{d}_t$ that aid in the decision-making process. In response, the network outputs an action $\bar{a}_t$. Additionally, when the agent is performing model-free training, it either receives from the environment or generates its own reward feedback signal using the reward function $r(s_t, a_t)$. It then collects the tuple of information ($\bar{a}_t$, $\bar{a}_t$, $\bar{a}_{t+1}$, $r(s_t, a_t)$). These tuples are then used to optimize the network loss function J_θ. Depending on the

algorithm used to update the loss function, the DRL agent might deploy a simple policy gradient network like REINFORCE [138], a simple Actor-Critic network like A2C [139], or an advanced actor critic algorithm like Proximal Policy Optimization(PPO) [127] or Deep Deterministic Policy Gradient(DDPG) [133].

Since the measurements from the real environments are prone to contain sensor noise, the data-driven models forming the *Dynamic System Model* exhibit variance due to inaccuracies. This can lead to convergence issues during the agent network training process due to inaccurate gradient estimates during backups of returns. This is commonly described as the Out of Distribution Error(OOD) [50, 140] in data-driven reinforcement learning. Hence, during model-free training, we run multiple "worker" Agents on different initialization or seeds of the *Dynamic System Model* to generate diverse experiences in parallel so that we can reduce the variance due to the data-driven models and make the agent learning process is stable via bootstrap aggregation.

4.2.4 Exogenous Variable Predictors

Our approach aims to adapt the reinforcement learning based *Supervisory Controller* to the system non-stationary changes immediately after they are detected. RL controllers based on policy gradient approaches are typically sample inefficient. As such, we need sufficient data to simulate the behavior of the *Dynamic System Model* under those conditions. For slow moving systems like buildings, access to sufficient real data may not be possible. Hence, we have developed an approach to forecast the future behavior of exogenous variables that serve as inputs to the *Dynamic System Models* so that we can simulate the model into the future and generate enough data for faster updates.

Hence, we introduce a class of predictor models called *Exogenous Variable Predictors* that learn to forecast behavior of exogenous variables affecting a dynamic system. The schematic of a generic exogenous variable predictor module is shown in Figure 4.5 We demonstrate the development of *Exogenous Variable Predictors* in the context of weather, thermal load and occupancy schedules when we apply our approach to buildings. Given the current sequence of inputs of length K for the exogenous variables $\bar{d}_{t-K:t}$, these models predict the value of the exogenous variable at the next time instant as $\hat{d}_{t+1}$. For forecasting over a horizon of length N, we use the output at the $t+1^{th}$ instant and append it to the input sequence. Both K and N are hyperparameters that need to be fine-tuned based on the specific application. We mostly use LSTM models for capturing time-correlation in these variables and Fully Connected/Dense Neural Networks(FCNN) to capture non-linearity of the processes. The specific network architecture is obtained by a hyperparameter search process associated with the models. Further, depending on the domain of application, it can keep training continuously with newer batches of data or train only when certain system non-stationary behavior is detected.

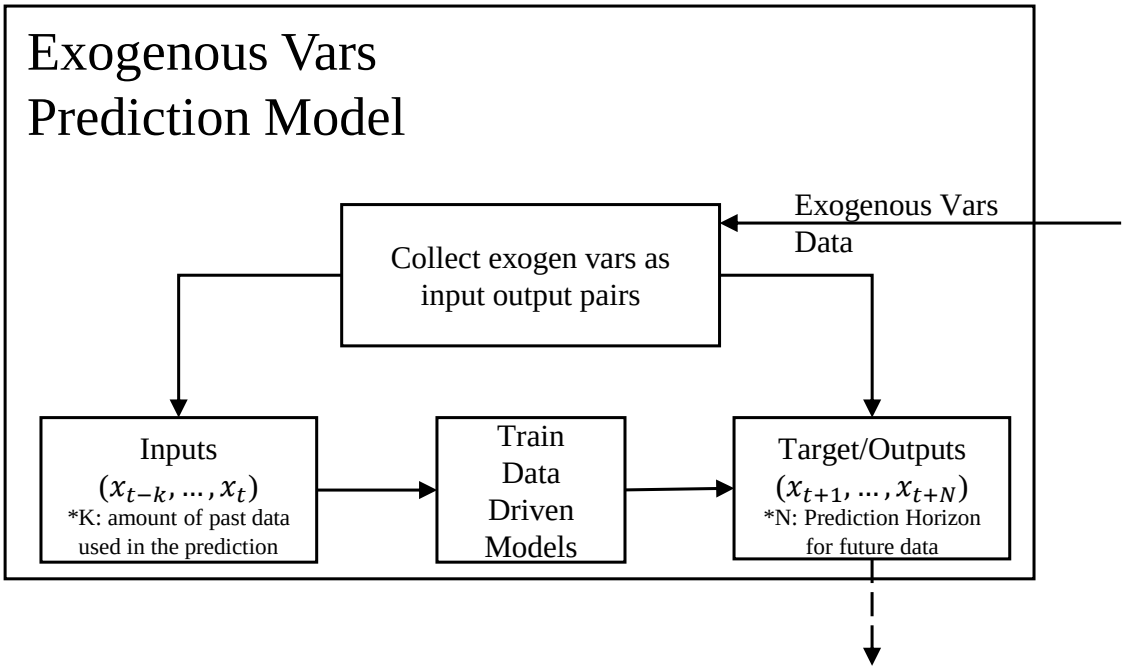

Figure 4.5: Exogenous Variable Prediction Module

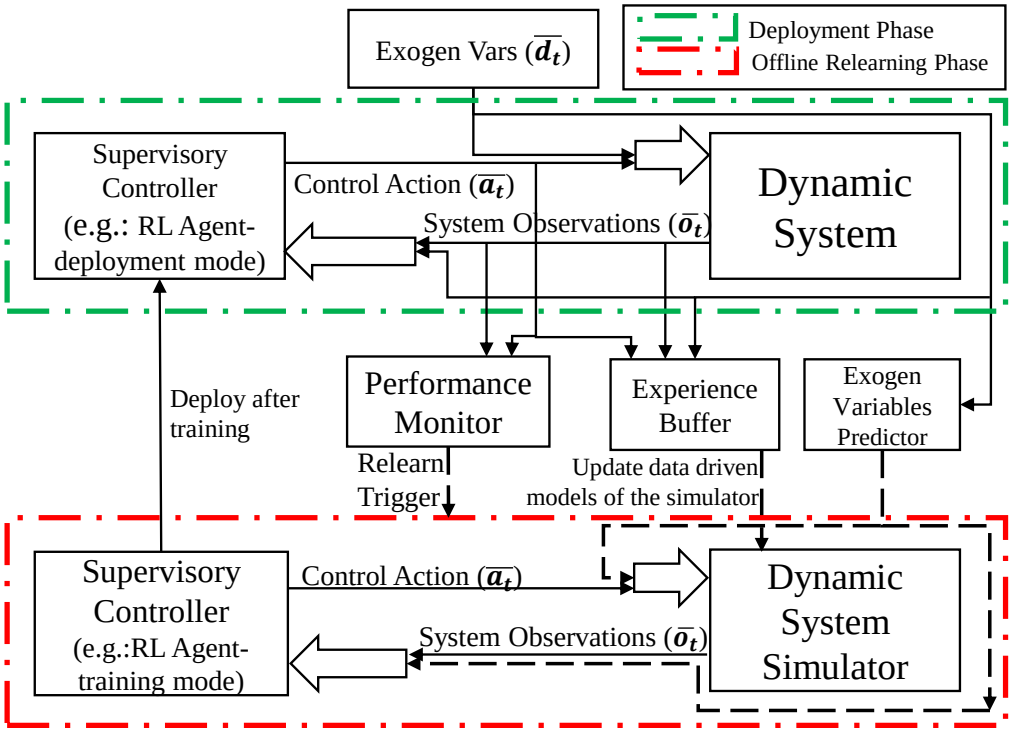

Figure 4.6: Proposed architecture for building energy control

The overall approach is now be represented using the schematic shown in Figure 4.6. There are two distinct phases annotated in this schematic. During the *Deployment Phase*, a pre-trained *Supervisory Controller* interacts with the real *Dynamic System*, with the interactions being collected in the *Experience Buffer*. The *Performance Monitor Module* tracks the performance of the controller and whenever it deteriorates, it generates a *Relearn Trigger* signal to start the *Offline Relearn Phase*. The *Dynamic System Model* is updated using data from the *Experience Buffer*, and then the *Supervisory Controller* adapts itself by model-free learning on parallel copies of the *Dynamic System Model* using a multi Actor Critic approach. The *Exogenous Variable Predictors* are used to forecast the exogenous variables over a horizon so that the agent training process can gather numerous data points during the model-free training process for faster updates.

We summarize our approach in Algorithm 1.

Algorithm 1: Relearning Approach for end-to-end data-driven DRL

Input $\leftarrow \mathscr{D}$; `Exogenous Variables Source`
Input $\leftarrow \mathscr{M}$; `Real Dynamic System`
Input $\leftarrow P_\theta$; `Pre-trained RL Controller`
Input $\leftarrow PM$; `Performance Monitor:` $Window = W_{pm}$
Input $\leftarrow EB$; `Experience Buffer:` $MemorySize = M_e$
Input $\leftarrow D_p$; `Exogenous Var Predictor:` $OutputSequenceLength = N$
Input $\leftarrow DSM$; `Dynamic System Model`
while *True* **do**
    Deploy P_θ on $\mathscr{M}$
    Monitor reward $r(s_t, a_t)$ using PM
    Update D_p models using $\mathscr{D}$ and Online Learning
    Add current (s, a, s') to EB
    if *PM triggers* **then**
        Offline Relearn Phase() ; `function to run as a independent parallel thread`
Offline Relearn Phase
    update DSM models using EB using EWC regularization
    Create a copy of P_θ as $P_\theta^{'}$
    Optimize $J_\theta^{'}$ using parallel roll-outs on DSM model
    Update deployed controller: $P_\theta \leftarrow P_\theta^{'}$

4.3 Summary

In this chapter, we provided a detailed overview of our approach to address the issues faced by data-driven deep reinforcement learning based control. The combination of detection of non-stationarity, data-collection, data-driven modeling for the exogenous as well as dynamic system variables, long-term predictions, future trajectory simulations, reward formulation and finally bootstrapping using a multi-actor-critic approach helps us advance the state of the art in designing optimal controllers under non-stationarities. Specifically, we aim to make the following contributions:

We extend the application of DRL in non-stationary MDPs from a hybrid system approach where knowledge of modes is assumed to one where DRL can be applied without preexisting knowledge of operating conditions. (2) A unified metric to monitor performance of controller on the real system and quickly trigger adaptation due to non-stationary building behavior. Earlier approaches to data-driven control [7, 141] have specified periodic updates, which does not consider any information about when the non-stationarity occurs.

In our approach, the data-driven models help relearn relevant system dynamics using real data from the building, much faster, compared to the time spent in developing whole building physics based models and needing detailed domain knowledge to update them during non-stationary behavior. This has been shown in [141] to improve controller performance.

Since we relearn offline on a data-driven dynamic system model, we are able to speed up the training process(2.4 millisecond/sample) compared to learning online where sampling rate is much slower(5min/sample). Learning online by planning as in [141] involves costly computations at every time instant, which we avoid by adapting offline.

The use of forecasts to generate more data helps generate more samples for the offline controller training and helps stabilize the training process, thereby speeding up the convergence.

The application of regularization using Elastic Weighted Consolidation [28] during the updates helps prevent overfitting on limited data available immediately after detection of non-stationarity. Unlike our work, [141] uses warm starts to retrain the data-driven model neural network weights from previous iterations during model updates.

By virtue of the design and implementation of the reward function, we are able to support a controller design that takes multiple feedbacks.

To make the approach more robust under varying amount of noise in the observations and resulting errors in the data-driven models, we run multiple simulations in parallel during the relearning phase of the controller and bootstrap aggregate these experiences during the controller updates. This helps reduce the variance, thereby stabilizing the controller updates and reducing the effects of Out of Distribution Error(OOD) [50] estimates while calculating returns in data-driven RL. Existing relearning approaches in data-driven RL [141] have not considered addressing this problem.

4.4 Limitations

It is important that we address the limitations of this approach before we apply it to different problem domains. Since we perform the relearning conditioned on a trend detection using a window, we are not able to tackle sudden changes in building performance that may arise due to sudden faults in system components or sudden occupancy changes, as they violate the Lipschitz assumption. Also, the learning process of the data-driven models for the dynamic system assume that the non-stationary behavior continues for a certain duration of time into the future as it gives us enough time to update and redeploy. This assumption might not hold for an exhaustive set of non-stationary behavior in buildings. Hence, in the application, we restrict ourselves to classes of non-stationary behavior caused by changes that are sustained over time periods longer than it takes to relearn. The performance of our approach also depends on its sensitivity to different hyperparameters and the amount of noise in the observations. Hence, optimal hyperparameter estimation and robustness to noise has to be appropriately tackled in different applications.

CHAPTER 5

Hyperparameters: Tuning and Sensitivity Study

In our approach, there are certain important hyperparameters associated with individual or multiple components. These hyperparameters affect the overall performance of the approach. Hence, we need to systematically study the effects of these hyperparameters on our approach to understand how well or worse our approach generalizes under different conditions with varying severity of non-stationary. This requires us to 1) formalize the search process for the best set of hyperparameters and 2) study the sensitivity of the approach (w.r.t. the performance metrics) to these individual hyperparameter variations.

5.1 Hyperparameter Optimization

To search for the best set of hyperparameters, we undertake an optimization problem. First, we construct a Bayes net structure of the set of hyperparameters in a complex hyperparameter space, and use *d-separation* methods to decompose the large space into smaller subsets of hyperparameters that can be optimized independently, thus making the optimization problem feasible in terms of time and memory requirements.

We formulate the hyperparameter optimization problem and discuss the set of steps to reduce the complexity of the tuning process.

5.1.1 Problem Formulation

Let us assume that an ensemble A of models (not limited to machine learning models) is used to solve a problem associated with a complex CPS, D. The ensemble has an associated set of hyperparameters $H = (H_1, H_2, \ldots, H_j)$. The performance of the ensemble associated with D is defined by multiple metrics, $L = (L_1, L_2, \ldots, L_k)$. Our task is to identify the values for the set of hyperparameters, H^*, that provides the best performance for the ensemble in terms of the metrics in L. This is mathematically formulated as the hyperparameter optimization of the ensemble model A with respect to the metrics in L:

$$H^* = argmin_{h \in H} \mathscr{L}(A, D; h), \tag{5.1}$$

where $\mathscr{L}$ can be a single scalar evaluated as a weighted combination of the metrics in L or a set of individual equations if the elements in L depend on different subsets of H. If the cardinality of H, i.e., $|H|$ is large, the joint hyperparameter optimization problem can become computationally very complex, and sometimes, intractable. We develop an approach for systematically decomposing the hyperparameter space to independent

subsets of hyperparameters, so that each subset can be optimized individually.

5.1.2 Approach

We adopt a Bayes Net for characterizing the hyperparameter space as a graphical structure, where the links in the graph capture the directed relations between corresponding hyperparameters and evaluation metrics they influence. Then we use concepts of d-separation to determine the conditionally independent subsets of hyperparameters that can be optimized independently.

5.1.2.1 Creation of the Bayes Net

Using our knowledge of the ensemble of models and the relations between the performance metrics in the application domain, we derive a Bayes Net that captures the relations between the model hyperparameters and their corresponding evaluation metrics.

Adopting the method discussed in [142], we construct a Bayes Net using a topological order (cause precedes effects) to link all $H_i \in H$ to the corresponding performance variables, $L_k \in L$, such that the order is consistent with the directed graph structure. This helps establish the Markov condition, i.e., $P(X_i|X_1, X_2, X_n) = P(X_i|Parents(X_i)$, where $X = \{X_1, X_2, \cdots X_n\} = H \cup L$.

5.1.2.2 Decomposition of Hyperparameter Space

The hyperparameter optimization problem is further simplified by classifying the hyperparameters in the ensemble models into two primary groups: (1) *local* and (2) *global*, based on which evaluation metrics they are linked to. This grouping starts with characterizing local and global metrics.

Definition 3 (Local and Global Metrics) *For an ensemble of data-driven models A associated with a system D, the evaluation metrics in L may be decomposed into two classes. The metrics that are only affected by an individual data-driven model performance are called local metrics L_l. The metrics that are affected by multiple models, are called global metrics L_h.*

Accordingly, let us now divide hyperparameters based on which metrics they influence. Corresponding to global and local metrics, we decompose the hyperparameter space.

Definition 4 (Decomposition of Hyperparameter space) *For a data-driven ensemble of models, A applied to a system D, the existing hyperparameter set H can be decomposed into two main subsets H_h and H_l using the causal structure [143] of the derived Bayes net. Hyperparameters that are linked to at least one global metric, L_h are termed global hyperparameters, H_h. Hyperparameters that are only linked to the local metrics L_l are called local hyperparameters: H_l.*

5.1.2.3 Separation of Hyperparameters: Connected Components and D-separation

Given our Bayes Net that captures the relations between hyperparameters and the metrics, we apply the concepts connected components and d-separation [144] to identify independent sets of hyperparameters.

5.1.2.3.1 Connected Components

A graph, G, is said to be connected if every vertex of the graph is reachable from every other vertex. For our Bayes Net, which is a directed acyclic graph (DAG), we can use a simple Breath First Search to first identify the connected components in the Bayes Net. Once, we identify the connected components, we use the concept of d-separation by converting all directed links to undirected links, and then establishing if a set of variables, X in the graph is independent of a second set of variables, Y, given a third set, Z.

5.1.2.3.2 D-separation

D-separation encompasses a set of well established rules that determine whether two sets of nodes X and Y are mutually independent when conditioned on a third set of nodes, Z. There are four primary rules that can be used to test the independence relation.

1. Indirect Causal Effect $X \rightarrow Z \rightarrow Y$: X can influence evidence Y if Z has not been observed.
2. Indirect Evidential Effect $Y \rightarrow Z \rightarrow X$: Evidence X can influence Y if Z has not been observed.
3. Common Cause $X \leftarrow Z \rightarrow Y$: X can influence Y if Z has not been observed.
4. Common Effect $X \rightarrow Z \leftarrow Y$: X can influence Y only if Z has been observed. Otherwise, they are independent.

In our formulation of Ensemble models, we frequently encounter the following examples based on the above rules.

1. The evaluation of local metrics blocks the effect of the local hyperparameters on global metrics (Rule 1). Therefore, any two models affecting the same global metric will not have their hyperparameters related via a common effect (Rule 4).
2. Assume that Z is the set of global hyperparameters, while X and Y represent local hyperparameter sets associated with individual models. If they belong to the same connected component (sub-graph), we will have to jointly optimize hyperparameters of the individual models, thus increasing the computational complexity of the optimization process. Instead, we can consider a two level optimization process: First, we chose candidate values for the global hyperparameters in Z. Given the values of the

variables in Z, we can independently optimize the hyperparameters connected components X and Y, thus decomposing the hyperparameter space further by applying Rule 3.

3. In many cases, all the global hyperparameters are linked to all the global metrics, and then they have to be jointly optimized in accordance with Rule 4.

Overall, applying the rules to the ensemble models and their performance metrics, helps us optimize the hyperparameters of most of the models independently. In certain cases, as we demonstrate later, the global hyperparameters may have to be optimized jointly.

During each trial of the hyperparameter optimization process, we follow the following steps:

1. Choose candidate values of the global hyperparameters H_h
2. With the chosen values of H_h, optimize the hyperparameters in H_l associated with each model in the ensemble by using the corresponding model performance as the objective function.
3. Evaluate the performance of subsets in H_h on L_h using the set of equations in 5.1

5.1.2.4 Bayesian Optimization for Hyperparameter Tuning

Once we have obtained the independent subsets of the hyperparameters, we optimize them using the performance metrics which they influence. In this work, we consider a model based black box optimization approach called Bayesian Optimization or Sequential Model Based Optimization(SMBO) [145]. We chose this process due to several advantages compared to other gradient based, derivate-free optimization processes. Unlike gradient-based optimization, SMBO can optimize on non-convex surfaces. This becomes a deciding factor in choosing SMBO over gradient based optimization when the hyperparameter space includes discrete variables. Also, unlike other derivative free methods, SMBO keeps track of the history of past evaluations of candidate hyperparameter settings and uses that information to form update a prior, which it uses to choose for hyperparameter values with potential improvements to the best choice until that point.

The Bayesian Optimization process starts with the initialization of a surrogate probability model. Let us denote this by $P(score|h)$. Here, *score* represents the result of evaluating the fitness metric $\mathscr{L}(\mathbf{A},\mathbf{D};\mathbf{h})$ defined in equation 5.1 for a given choice of the hyperparameters $h \in H$. This model is continuously updated to create a better estimate of the hyperparameter choices mapped to scores across the search space and make informed decisions when choosing candidate hyperparameter values. For our work, we apply Bayes rule on the surrogate probability model and then use the Tree Parzen Estimator(TPE)[145] to create a generative

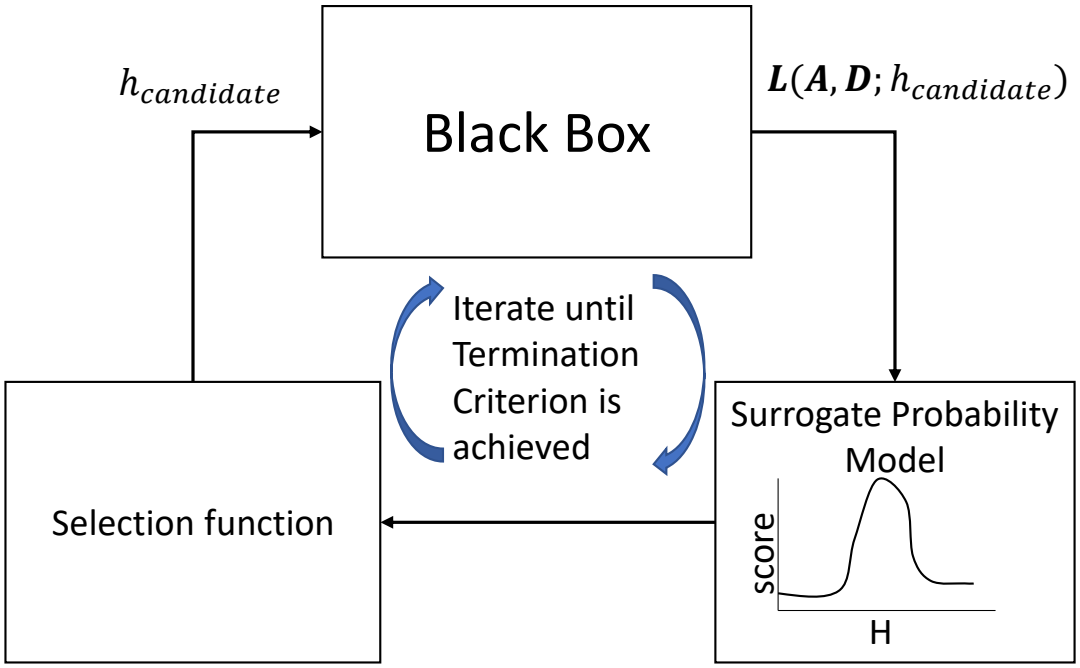

Figure 5.1: Schematic of Hyperparameter Optimization for our Approach based on Bayesian Optimization

model for creating the hyperparameter distribution conditioned on a threshold($score^*$) value of the score.

$$P(\theta|score) = \begin{cases} l(\theta), & \text{if } score \geq score* \\ g(\theta), & \text{otherwise} \end{cases}$$

where either of $l(\theta)$ or $g(\theta)$ are a combination of Gaussian Kernel Density Functions [146], which gradually estimate the true probability model as more evaluations of the hyperparameters become available.

We also initialize a Selection Function(also called the Acquisition Function) which decides the next candidate hyperparameter instant θ to choose for the potential improvement. We use the Expected Improvement Metric[145]

$$EI_{score^*} = \int_{-\inf}^{score^*} (score^* - score) p(score|\theta)\, d(score)$$

evaluated using the surrogate probability model to calculate the candidate improvements and choose the most promising candidate hyperparameter instantiation. The aim is to maximize the Expected Improvement with respect to θ.

The optimization process itself then includes three iterative steps that we run until some termination

criterion is achieved. These are enumerated below and schematically presented in Figure 5.1:

- Using the surrogate probability model and the selection function, choose the best performing hyperparameter
- Apply the hyperparameters to the fitness function **L**
- Update the surrogate model incorporating the new trial results.

Once, this process terminates, we obtain a locally optimal estimate of the hyperparameters that maximize the performance metric under consideration and use these values for evaluation of our approach when performing benchmarking experiments or for deployment.

5.2 Hyperparameter Sensitivity Study

Apart from hyperparameter tuning process, we also study the generalization of our approach across the different hyperparameters to understand the limits of our approach, i.e., regions in the hyperparameter space where it does not perform as intended. We run a standard One-at-a-Time(OAT) hyperparameter sensitivity study, where we fix the other hyperparameters at default values [147] and study the effect of varying a single hyperparameter. Let us assume that we obtain different values of the fitness function L with different values of the hyperparameter θ_i. Then we evaluate the first order Sobol Sensitivity Index(SI)[148] for that hyperparameter θ_i as:

$$S_i = \frac{V_{\theta_i}[E_{\theta_i}(L|\theta_i)]}{V(L)}$$

The SI value ranges from 0.0 to 1.0 and empirically, an SI value of greater than 0.05 (with greater than 95% confidence)[149] indicates that hyperparameter θ_i exhibits significant first-order sensitivity on the fitness values. The above Sensitivity Index measures the reduction in the Variance of the fitness measure L if we fix the value of θ_i. The use of the OAT condition lets us restrict ourselves to first order SI. In case, we plan to study the effects of cross interactions, across hyperparameters, we need to evaluate second and total order SIs. We chose the Sobol sensitivity analysis as it is considered one of the most powerful Sensitivity Analyses techniques compared to other methods[149] like weighted average of local sensitivity analysis (WALS), Partial rank correlation coefficient(PRCC), Multiparametric sensitivity analysis (MPSA) due to its assumption of model independence and taking into account higher order interactions among hyperparameters.

5.3 Summary

In this chapter, we provided a detailed overview of optimizing the hyperparameters of our approach, which can span across multiple dimensions. We simplified the computational complexity of the optimization process by creating a Bayes Net relating these hyperparameters and identifying conditionally independent subsets which can be tuned separately. Further, we proposed a set of sensitivity studies that can provide us useful insights on the generalization of our approach across a range of hyperparameter values. This comes handy when certain limitations like accuracy or computational complexity prevent us from choosing the most optimal hyperparameter values and, instead, rely on a computationally feasible value.

5.4 Limitations

While, we are able to break down the hyperparameter space into multiple subsets, the possibility of this decomposition depends on the problem formulation itself. In certain, cases it may not be possible to find any decomposition and can lead to computationally infeasible joint hyperparameter optimization. To mitigate such problems, we rely on distributing the optimization candidate trials across multiple workstations to parallelize the number of trials that can be performed parallelly. We use a distributed optimization framework called Ray Tune [150] to achieve this, where we can use local servers, remote high performance computing resources to communicate with each other about trial performance and update the surrogate model.

CHAPTER 6

Experimental Studies on a 5-zone Buildings Testbed

In this chapter, we describe the application of our proposed approach to the ASHRAE 5-zone testbed. We describe the system and the solution architecture and evaluate our approach on the testbed using multiple experiments. One of our goals in this study is to benchmark our approach against other approaches like Guideline36 [5], PPO [8], DDPG [133] and MPC [141]. To evaluate the performance, we use multiple standard building controller performance evaluation metrics under a variety of non-stationary conditions that occur over a period of one year. We report the average and standard deviation of these performances across multiple experiments using different performance metrics. We discuss the significance of the results in the context of the external conditions under which the experiments were performed. Finally, we perform the sensitivity analysis across different hyperparameter values to analyze the importance of different hyperparameters in our approach.

We first map the problem for the testbed in to an LC-NSMDP formulation introduced in Chapter 3. Here we describe: (1) the state and action variables, (2) the reward function for the application, (3) the transition function in terms of multiple component (data-driven) models. Thereafter, we outline the implementation of the solution architecture based on Chapter 4, where we go in to the details of (1) the data-driven models for the transition functions and the experience buffer, (2) the supervisory controller implemented as a DRL agent, (3) the components of the exogenous variable predictors and the (4) performance monitor module. For each, we also highlight the associated hyperparameters to motivate the next step of hyperparameter separation and optimization procedure. Next, we apply the hyperparameter separation procedure, described in Chapter 5, whereby we discuss the (1) Bayes Net, (2) classification of hyperparameters and (3) subsequent application of d-separation to get the independent subsets of hyperparameters for faster optimization.

Next, we report the results of the different experiments we performed. First, we provide the tuned values for the hyperparameters and the optimal architectures for the data driven components of the solution architecture. We also outline the evaluation procedure and results of these individual components under multiple types of building non-stationarities.

6.1 System Description

The five-zone testbed, developed by Michael Wetter and team at the Lawrence Berkeley National Labs, is a frequently used physics-based dynamic simulation model comprising an HVAC system, a building envelope model that includes air flow and leakage through open doors and other parts of the building [3]. It has an

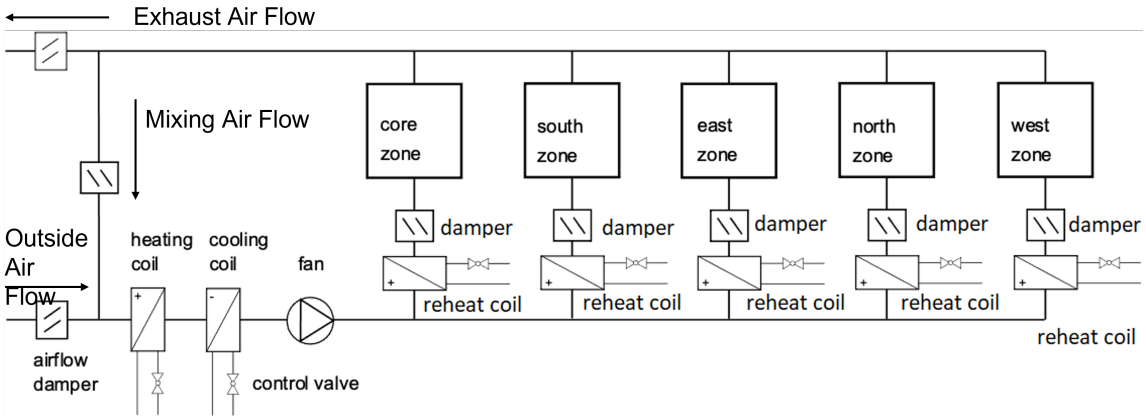

Figure 6.1: Schematic of the Five Zone Testbed. Source: [3]

Air Handling Unit(AHU) that handles heating and cooling, with a return air economizer that conditions the air temperature before releasing it into the building. The air is then ducted to the individual zones of the building using forced draft created by a VFD fan. There are 5 zones in the building oriented along the North, South, East and West direction with a central Core zone which connects to the other four zones. Each zone is equipped with a standard Variable Air Volume(VAV) unit comprising a damper and reheat coil. The purpose of the damper is to modulate the amount of heated air that is allowed to enter to be reheated, and also control how much air enter the zone. This depends on the cooling and heating temperature set points for the zone. It does this by using the centrally supplied air temperature and the zone air temperature as feedback signals. It is implemented using a low-level PID controller. For a further detailed description involving the equations for modeling of this building testbed, we refer the reader to [3].

In this work, we adapt an open-source implementation of this building testbed in Modelica [151] for our experiments. We add all the necessary input and outputs signals for the model and modify the setup to simulate non-stationary behavior in the system. Next, we compiled the model into a Functional Mock-up Unit(FMU) using a *Jmodelica* compiler. We host this model [152] using a Docker and then use *PyFMI* to interact with the building model through simple API calls using an approach we have developed in Python. The goal of the controller deployed using our approach is to ensure efficiency in terms of energy as well as to maintain comfort, safety and robustness when the building exhibits non-stationary behavior. We tune the Air Handling Unit discharge air temperature set point of the testbed under varying conditions of exogenous variables like weather, thermal load/occupancy schedule and zone temperature setpoint preferences, which cause the building to exhibit non-stationary behavior. We evaluate the energy performance, occupant comfort, system actuation safety, speed of adaptation and robustness to the non-stationarities. The purpose of these evaluations is to investigate whether our approach can address some common challenges in this domain.

Our goal is to develop a supervisory controller for the AHU discharge setpoint to ensure energy efficiency, comfort, and reduced VAV damper actuation under building non-stationarity due to changing 1) weather, 2) zone temperature requirements and 3) thermal load(occupancy). Accordingly, we perform four sets of experiments where we evaluate performance of our approach separately under each type of building non-stationarity and a fourth type where the non-stationarities are combined. We further benchmark the performance against other state-of-the-art supervisory controllers in current literature.

6.2 Problem Formulation

Table 6.1 describes different components of the testbed mapped onto the MDP. The choice of variables is based on suggestions from building managers and partly inspired by relevant work in Section 2.

Table 6.1: Description of the testbed *Dynamic System Model* in terms of an MDP

Component	Variables
State $\bar{s_t}$	1.Outside Air Temperature(oat) 2.Outside Air Relative Humidity(orh) 3.Five Zone Temperatures($T_z, z = 1 \dots 5$) 4.Total Energy Consumption(E_{tot}) 5.Air Handling Unit Discharge Air Temperature (T_{disch})
Action: $\bar{a_t}$	Air Handling Unit Discharge Temperature Set-point(T_{disch_stpt})
Reward $r(s_t, a_t)$	$r_{energy} + r_{cmft} + r_{vav\_actuation}$ where $r_{energy} = -E_{tot}$ $r_{cmft} = -\sum_{z \in zones} max((T_z - ub_{z,t}), 0, (lb_{z,t} - T_z))$ $r_{vav\_actuation} = -\sum_{z \in zones} \lvert v_{\%,z,t} - v_{\%,z,t-1} \rvert$
Transition Model($p(s', s_t, a_t)$)	1. Total Energy Consumption Model(M_{energy}) 2. Zone Temperature Model($M_{T,zone}$) 3. VAV Damper Percentage Model($M_{vav\%}$)

In the reward function, r_{energy} incentivizes energy efficiency, r_{cmft} penalizes zone comfort violations and $r_{vav\_actuation}$ penalizes frequent changes in the controller actions T_{disch_stpt}, as the zone VAV dampers tend to aggressively actuate in response. Here, $ub_{z,t}$ and $lb_{z,t}$ represent the upper and lower comfort bounds for the temperature in each zone z at time instant t. $v_{\%,z,t}$ denotes the VAV damper valve percentage in zone z at time instant t. The reward components are assumed to be locally Lipschitz Continuous due to large time-constants in building thermodynamics.

6.3 Implementation of the Solution

Here we discuss the different components of our approach specific to the testbed where we apply it. The details of the architecture of the individual models, the hyperparameter values and performance evaluation procedures and corresponding results are provided in the results subsection in this chapter.

6.3.1 Dynamic System Model

The Transition Model ($p(s', s_t, a_t)$) needs to provide the values of the following set of observations ($\bar{o}_t$) at the next time step: *Total Energy Consumption* (E_{tot}), the zone temperatures ($T_z, z = 1 \dots 5$), the VAV damper percentages ($v_{\%,z}, z = 1 \dots 5$). Accordingly, we create a model M_{energy} for predicting energy consumption, a model $M_{T,zone}$ for predicting all the zone temperatures and 5 individual models($M_{vav\%,z}, z = 1 \dots 5$) each of which predicts the VAV damper percentage. The ensemble of these models constitute the *Dynamic System Model*. The discharge temperature model for predicting $T_{disch,t}$ is modeled as

$$T_{disch,t} = T_{disch\_st\_pt,t-1} + N(0, \sigma_{disch})$$

because we observed that the discharge temperature was always close to the setpoint specified by the Action: $a = T_{disch\_st\_pt}$. σ_{disch} is chosen to be 0.1^oF o account for any non-deterministic effects.

6.3.2 Experience Buffer

The *Experience Buffer* was initiated with an optimal memory size M_e.

6.3.3 Supervisory Controller

The DRL based *Supervisory Controller* was implemented using a simple Multi-Actor Critic framework [153, 154]. We chose this formulation as we wanted to see whether there is an improvement due to our framework rather than inherent efficiency of a standalone state-of-the-art algorithm like PPO [127]. Given the training resources, we chose to train in $n = 10$ parallel environments to generate numerous diverse samples similar to [26] to account for the measurement noise and resulting model inaccuracies in the *Dynamic System Models*. We did not specify max training steps, as we are incrementally training when performance degradation is detected and used callbacks to stop training when the reward did not improve further or converged.

6.3.4 Exogenous Variable Predictors

For the testbed, we needed to predict the values of ambient dry bulb temperature(oat), the relative humidity(orh). For this problem, the outside air temperature and humidity prediction models are based on Long Short-Term Memory Neural Networks(*LSTMs*) adapted from [154, 155] with some fine-tuning to adjust to the data for our specific location. The *Exogenous variable predictor* models are continuously learned in a batched online fashion. For the zone setpoints schedules and thermal loads based exogenous variables, models were based on simple rules which look up the current schedules or values for the variables in the real system and set them to those schedules or values for the required prediction horizon N.

6.3.5 Performance Monitor Module

For the *Performance Monitor*, we tracked the presence of a negative trend in the deployment phase using $i = 2$ windows W_{pm1} and W_{pm2} in parallel. The purpose is to track slow moving non-stationarities due to weather using W_{pm1} and fast moving non-stationarities due to zone temperature or load changes.

6.4 Hyperparameter Optimization

We now identify the hyperparameters of the approach applied to the testbed and demonstrate the methods developed in Chapter 5 to separate and simplify the optimization procedure. We create the Bayes Net, classify the set of hyperparameters as global and local, and finally test for mutual independence. We conclude this section with the framing of the hyperparameter optimization problem and provide the results in the next Chapter.

6.4.1 Bayes Net

We identify the hyperparameters and metrics of the ensemble models described in the last section to build the Bayes Net. The set of hyperparameters is broadly summarized in the first column of Table 6.2. The set of performance metrics included in the Bayes Net are: (1) time to detect non-stationarity after occurrence($L_{\Delta t_{detection}}$), (2) energy consumption(L_{energy}), (3) comfort($L_{comfort}$), (4) actuation safety($L_{actuation}$), (5) total relearning time($L_{\Delta t_{relearning}}$), (6) individual data-driven models errors($Lmodel, error$), and (7) RL agent losses($L_{Rlagent,reward}$). We create the Bayes Net shown in Figure 6.2 using the two sets of variables. The connections are based on the cause effect relations, assuming knowledge of the ensemble structure, which is further explained in the second column of Table 6.2.

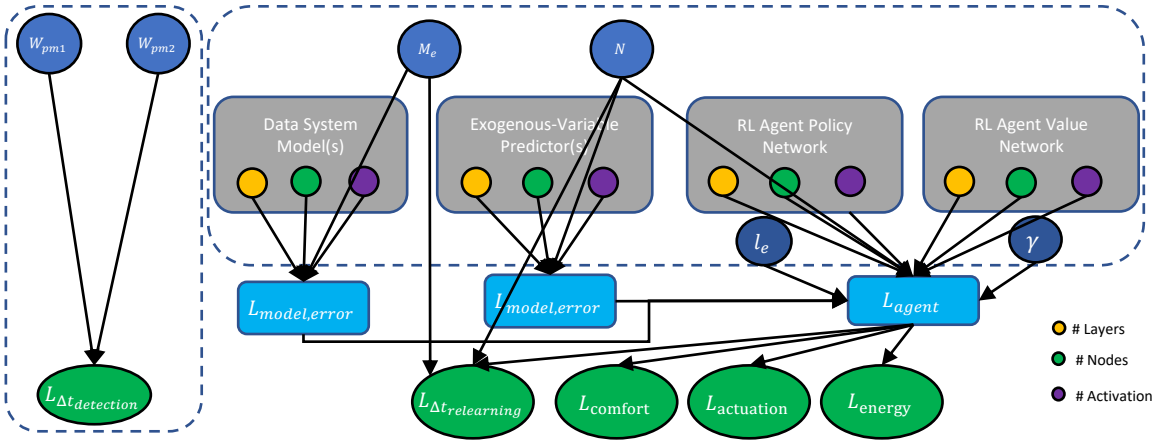

Figure 6.2: Bayes Net for the Relearning Approach applied to the 5 Zone Testbed

Table 6.2: Hyperparameters of the ensemble-CPS pair, the metrics they affect and the classification of the hyperparameters as global or local

Hyperparameter(s)	Metric Affected	Global/Local
Performance Monitor Module Window sizes: W_{pm1}, W_{pm2}	$L_{\Delta t_{detection}}$	Global
Memory Buffer Size: M_e	$L_{model,error}$, $L_{\Delta t_{relearning}}$	Global
Forecast Horizon Length: N	$L_{model,error}$, $L_{RLagent,reward}$. $L_{\Delta t_{relearning}}$	Global
Exogenous Variable Predictor: *Nodes, Layers, Activations*	$Lmodel, error$	Local
Dynamic System Model(s): *Nodes, Layers, Activations*	$Lmodel, error$	Local
RL Agent Policy Network: *Nodes, Layers, Activations*	$L_{RLagent,reward}$	Local
RL Agent Value Network: *Nodes, Layers, Activations*	$L_{RLagent,reward}$	Local
Discount Factor: γ	$L_{RLagent,reward}$	Local
Episode Length: l_e	$L_{RLagent,reward}$	Local

6.4.2 Identification of global and local hyperparameters

To identify the set of global and local hyperparameters, we first classify the set of metrics as global and local. For the ensemble approach and the 5-zone testbed, time to detect a non-stationary change after occurrence($L_{\Delta t_{detection}}$), energy consumption(L_{energy}), comfort($L_{comfort}$), actuation safety($L_{actuation}$), total re-learning time($L_{\Delta t_{relearning}}$) are the global metrics as they are affected by multiple models and indicate overall ensemble performance. The individual data-driven models errors($Lmodel, error$) and RL agent losses($L_{RLagent,reward}$) are the local performance metrics.

Accordingly, Table 6.2 explains the metrics that are affected by the hyperparameters and classifies them as Global and Local hyperparameters in the last column.

6.4.3 Separation of Hyperparameters

Now that we have a Bayes Net relating the hyperparameters, we apply the concept of connected components and d-separation to identify potential mutual independence between hyperparameters.

Given the Bayes Net in figure 6.2, we can easily identify that W_pm1 and W_pm2 are related via Common

Effect (Rule 4). However, they are independent of the other variables since they are not connected to the rest of the nodes. The nodes M_e and N are connected via the Common Effect rule (Rule 4) at the node $L_{\Delta t_{relearning}}$. The individual model hyperparameters in the dynamic systems, exogenous variable predictors, the agent actor and critic networks are independent of each other via the common cause(Rule 3) due to the two-level hyperparameter optimization we discussed in Section 5.1.2.2. The hyperparameters inside individual models are dependent on each other simply due to common effect Rule 4. The nodes l_e and γ are jointly optimized with the agent hyperparameters due to common cause Rule 4 since they affect $L_{RLagent,reward}$. Finally, the individual model hyperparameters do not affect the global metrics due to the blocking effect of the local metrics $Lmodel, error$.

Hence, the hyperparameter space can be decomposed as follows:

1. (W_{pm1}, W_{pm2})
2. (M_e, N, H_l) where subsets of H_l can be optimized independent of each other.

6.4.4 Two-Step Hyperparameter Optimization

Given the separation of hyperparameters, we independently optimize the values of (W_{pm1}, W_{pm2}) based on the metric $L_{\Delta t_{detection}}$. For the other subset including (M_e, N, H_l), we perform of two-level optimization, where we choose candidate values of (M_e, N) and then optimize the subsets of H_l independently. The independent subsets include hyperparameters of the following:

1. Exogenous Variable Predictor
2. Dynamic System Model(s)
3. RL Agent Policy Network, Value Network, l_e and γ together

In our work, we consider four types of building non-stationarity: 1) Weather based: ambient temperature and ambient relative humidity, 2) Changes in zone setpoints 3) Changes in thermal load due to changes in occupancy 4) combination of the first 3 cases. It should also be kept in mind that for cases 2 and 3, weather related non-stationarity is also present naturally. This two-step optimization approach is thus performed across all possible non-stationarities and the corresponding optimal architectures and performance are reported in the Result section of the chapter. The over hyperparameter optimization is thus denoted in the Figure 6.3. For the hyperparameter optimization stage, we do not introduce noise($\sigma = 0$) into the measurements to get the best architecture possible.

Figure 6.3: Two-Step Hyperparameter Optimization for the testbed

6.4.5 Global and Local Hyperparameter Choices

The candidate choices for the global and local hyperparameters are described in Tables 6.3 and 6.4 respectively.

Table 6.3: Ranges of the global hyperparameters

Hyperparameter	Search Space (Range)
W_{pm1} (Performance Monitor Module)	$\{1, 2, \ldots, 12\} hrs$
W_{pm2} (Performance Monitor Module)	$\{15, 30, \ldots, 165, 180\} min$
M_e (Experience Buffer)	$\{24, 48, \ldots, 144, 168\} hrs$
N(Exogenous Variable Predictor and Dynamic System Model)	$\{24, 48, 72, 96\} hrs$

We use these ranges based on existing applications of similar models in building energy optimization problems in the literature. While we are mostly working with discrete choices for the values of the hyperparameters, the separation of the hyperparameters indeed helps in simplifying the search space, as indicated by the calculations below. For the neural network based models associated with the dynamic systems, the exogenous variable predictor models and the actor critic networks, we restricted ourselves to 4 hidden layers by default, otherwise the problem became further intractable.

6.5 Individual Component Architecture and Performance Evaluation

Here we provide the details of the tuned model architectures resulting from hyperparameter optimization process and the associated evaluation procedures and model accuracies. We ran distributed hyperparameter optimization using the Ray-Tune library [150]. We performed this tuning process across 4 types of building non-stationarities: 1) weather, 2) zone setpoint, 3) thermal load and 4) combination of previous three. Accordingly, tuned architectures are reported across each condition.

6.5.1 Dynamic System Model

The network architecture for the different data-driven models in the *Dynamic System Model* are show in Table 6.5.

These models were initially pre-trained on 6 months of data from the testbed. During evaluation, we fed the predicted output from $t-1$ as input for predicting at time step t. This helped us evaluate the model error over the prediction horizon N. We evaluated the models across 100 $N = 48$ hour horizons from different parts

Table 6.4: Ranges of the local hyperparameters

Hyperparameter	Search Space (Range)
Dynamic System Model Weights (M_{energy}, $M_{T,zone}$, $M_{vav\%,z}$)	$W \in \{8, 16, 32, \ldots, 256\}$
Dynamic System Model Activation (M_{energy}, $M_{T,zone}$, $M_{vav\%,z}$)	$a \in \{linear, relu, sigmoid, tanh\}$
Exogenous Variable Model Weights	$W \in \{8, 16, 32, \ldots, 256\}$
Exogenous Variable Model Activation	$a \in \{linear, relu, tanh\}$
Actor(π_θ) Critic (V_ϕ) Network weights	$W \in \{64, 128, 256, 512\}$
Actor(π_θ) Critic (V_ϕ) Network Activation	$a \in \{linear, relu, tanh\}$
γ	$\{0.99, 0.95, \ldots, 0.80\}$
l_e	$\{1, 2, 3\}$ days

of the year. The energy models were evaluated using CVRMSE error, while the temperature and zone VAV predictors were evaluated using MAE. The results are show in Table 6.6

6.5.2 Experience Buffer

The size of the Experience Buffer M_e is tuned using hyperparameter optimization and we obtain the following values show in Table 6.8

6.5.3 Supervisory Controller

The *Supervisory Controller* consists of the Actor Critic framework [153, 154]. The inputs to the actor network are the state variables in $\bar{s}_t$ described in Table 6.1 and the output is the mean a_μ and a_σ from which an action $\bar{a}_t$ is sampled. The critic network in this case is a Q-network that takes as input the current state $\bar{s}_t$ and action $\bar{a}_t$. It is then used to estimate the Advantage using Equation 6.1.

$$
\begin{aligned}
A(s_t, a_t) &= Q(s_t, a_t) - V(s_t) \\
&= Q(s_t, a_t) - \sum_{a_t \in A} \pi(a_t|s_t) * Q(s_t, a_t)
\end{aligned}
\tag{6.1}
$$

Typically, advantages are normalized for better convergence during agent training. The tuned network architecture is shown in Table 6.7

Next, we look at the values of the episode length l and discount factor γ in Table 6.9. We believe that for slower non-stationarities with high degree of uncertainty like weather, there is less faith in future estimated returns. Hence, we have low values of γ for weather while the opposite is true for non-stationarities related to zone setpoint and thermal load changes as they tend to sustain values or maintain the same schedule in the future and not change very often.

6.5.4 Exogenous Variable Predictors

The *Exogenous Variable Predictors* models for this problem involved the prediction of outside air temperature and relative humidity. We formulated this a sequence prediction problem where, given K past temperature or relative humidity inputs, we predict the corresponding values over a horizon N. The model architectures were adapted from the paper [154, 155] and fine-tuned for the weather data of Nashville, TN, USA. However, it was difficult to perform sequence to sequence prediction using the method outlined in that paper because we did not have labeled data for the future when we use these models for forecasting purposes. Hence, during prediction, we implemented a for loop where the previous predicted output($\bar{o}_t$) cell state($\bar{c}_t$) and the network state($\bar{h}_t$) of the last LSTM is stored and fed as input during prediction of the temperature at the next time step($\bar{o}_{t+1}$). This is a standard method in sequence based problems in recurrent neural network based machine translation methods, as shown in [156]. Based on this, the network architecture is formulated as an encoder decoder network as shown in Table 6.10

Based on this architecture, we pre-trained the models using 1 year of data from Nashville and then performed prediction over a horizon of length N using past $K = 12$ hour inputs sampled at a half-hour interval. The test data in this case comprised 100 samples of data, where each sample had over $N = 48$ hour horizon sampled at a half-hour interval. The resulting modeling errors are reported in Table 6.11

Finally, the tuned values of the input sequence length K and the output horizon N under different conditions of non-stationarity are shown in Table 6.12

The output horizon length is tuned based on a weighted metric comprising the model error, time to relearn, energy, actuation and comfort performance. The value of N has to balance between generating numerous

samples via a larger future horizon to facilitate faster agent training and convergence and the effects of modeling errors across due to uncertainty in future estimates G_t.

6.5.5 Performance Monitor Module

For the performance monitor module, we tuned the values of the two windows for detecting negative trends over both small intervals due to faster relatively non-stationarities and large intervals for slow-moving non-stationarities. The tuned values for W_{pm1} and W_{pm2} are shown in Table 6.13.

6.6 Sensitivity Analysis of global hyperparameters

Given the large number of hyperparameters of the approach, we wanted to address their effect on the overall approach. For this, we ran sensitivity analysis using the guideline provided in [149] where we simulate the system and collect performance metrics mentioned in 6.7.1. We report the first order and total order Sobol sensitivity values for the global hyperparameters : W_{pm1}, W_{pm2}, M_e, N. We ran experiments with the all possible choice of values for these hyperparameters, as provided in Table 6.3. The results are demonstrated in Figure 6.4.

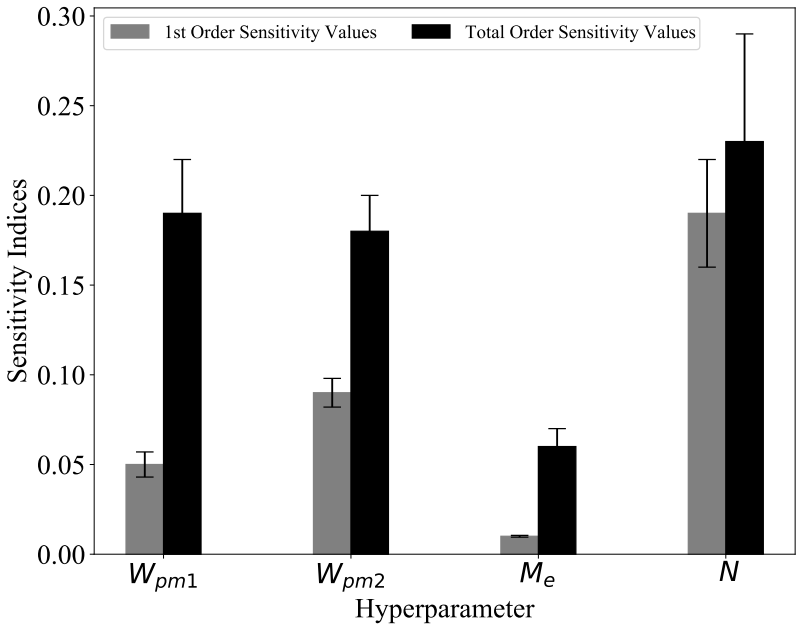

Figure 6.4: Sobol sensitivity indices of Four Global hyperparameters are shown in this graph. Total-order sensitivity indices (black bar) and first-order sensitivity indices (gray bar) are shown, respectively. The greater the sensitivity indices are, the more critical parameters are for the model. Complex models usually have more than one parameter that are critical and often vary at the same time with external conditions.

The results indicate that hyperparameter N is the most important parameter contributing to 20% of the approach performance variability, followed by the important parameters W_{pm1}, W_{pm2} and M_e. Both total-order

sensitivity indices and first-order sensitivity indices have similar values for N, which indicate no significant second-order interaction between the N and other hyperparameters. The error bars in the figure represent the bootstrap confidence intervals (1.96*standard error) and 95% confidence intervals, which can be calculated by using the formula (sensitivity indices ± the bootstrap confidence intervals).

6.7 Benchmark Experiments

6.7.1 Evaluation Metrics

We use standard metrics in building literature to evaluate our approach.

1. Energy Consumption E_{tot}: A lower energy consumption under similar conditions indicates better energy efficiency.

2. Average Zone Temperature Deviation: $T_D = \sum_{z \in Z} max(T_z - ub_z, 0, lb_z - T_z)$ or $\sum_{z \in Z} |T_z - T_{z,stpt}|$. A lower value indicates better thermal comfort. Here, Z represents the total number of zones, T_z is the temperature in zone z, $T_{z,stpt}$ is the set-point for zone z, and ub_z and lb_z are a lower and upper comfort bounds. One of the two choices can be selected as a proxy for comfort, depending on whether the bounds are predefined or not for each zone of the building.

3. Actuation Rate: $A_s = \sum_{t=0}^{T} \sum_{z \in Z} |A_{t+1,z} - A_{t,z}|$ where A is the VAV damper opening state (percentage) for the testbed or fan switching state (on/off) for the VRF cassettes fans. Lesser actuation on aggregate implies that the controller is able to enforce smoother control such that the actuators wear less.

6.7.2 Results

We designed experiments considering combinations of changes in the weather conditions, zone set-point values and thermal load. We considered 3 months of data in a pre-training period for learning the building dynamics and initial supervisory controller, and then the proposed framework is evaluated over a period of one year using weather data from the city of Nashville. Weather-related changes include sudden increase and decrease in the outside air temperature and relative humidity, *e.g.: sudden warm spells during spring mostly occurring in February and March 2021 or sudden cold spells in August and October 2021*. Zone set-point changes modify the upper and lower bounds for temperature in certain zones of the test-bed building from 4^oF to 6^oF. Finally, sudden increase in the thermal load of a zone simulates the occurrence of events with large gatherings. There were 24 time-points where weather-related non-stationary changes were detected, 14 time-points where zone-set point changes were introduced artificially and successfully detected, 11 time-points where thermal load changes were introduced artificially and successfully detected, and 26 time-points

where a combination of all were detected.

We compared the performance of the proposed framework against state-of-the-art in Rule-Based Control Guideline-36 [5], in Deep Reinforcement Learning(DRL) Control PPO(online and on-policy) [8], DDPG(online and off-policy) [133] and a data-driven MPC approach described in [157] which is based on Path Integral based Model Predictive Control (MPPI) with periodic updates to the model (uses data-driven models to perform planning and choose the best action at each time step). The reward function, state space, and control actions are the same for each algorithm. The DRL approaches are allowed to run online[8, 133] as outlined in the original paper and their implementation in related reinforcement-learning based buildings control literature [16, 107, 110, 111]. PPO algorithm updates its policy network when a batch of continuously collected experiences comprises a period of 12 hours. Since DDPG updates off-policy, it is allowed to update using its Replay Buffer after 1-week intervals from where it samples experiences. The prediction horizon considered for MPPI is similar to ours under each case of non-stationarity to ensure fairness in the comparison. Similar to [26], we consider periodic(weekly) updates for the data-driven models used for MPPI.

To account for the effect of random seed initialization in recurrent neural networks used in our approaches, and based on available resources for computation, we ran 15 experiments for each one of the controllers under each type of non-stationarity. We highlight that the PPO and DDPG learn directly from interacting with the building. Since we were interested in comparing the performance of our approach once a non-stationary change occurs, we analyze the evaluation metrics over a 1-week period after non-stationary changes are detected, as it can safely be assumed as the maximum duration within which the building response changes.

The results from the benchmarking are presented in Table 6.14 with the best result in each case highlighted in bold. The performances in these experiments are further described in terms of the bar plots in Figure 6.5.

Table 6.14 shows that our approach is able to perform significantly better than the rule-based control and the online deployment of PPO and DDPG. Our performance is at par or often better than MPPI. Next, we study the performance in terms of the individual metrics.

The savings in energy consumption E_{tot} using our approach compared to the others is significantly better when non-stationarities occur from more abrupt changes due to the zone setpoints, thermal-loads or combination thereof. We believe that this happens mainly due to the speed-up in the adaptation process under our approach, which simulates the changes on the model with multiple actors generating more experiences. This helps overall in faster convergence, which is discussed a little later when comparing time required for adaptation. MPPI comes at a close second on most of the cases, since it also uses a model of the system to plan offline and generate better results. However, as discussed before, MPPI would still suffer from unreliable state estimates the further in the horizon it simulates. The uncertainty associated with the transition function could have possibly led to suboptimal results. Moreover, in several trials/experiments, we had to restart the

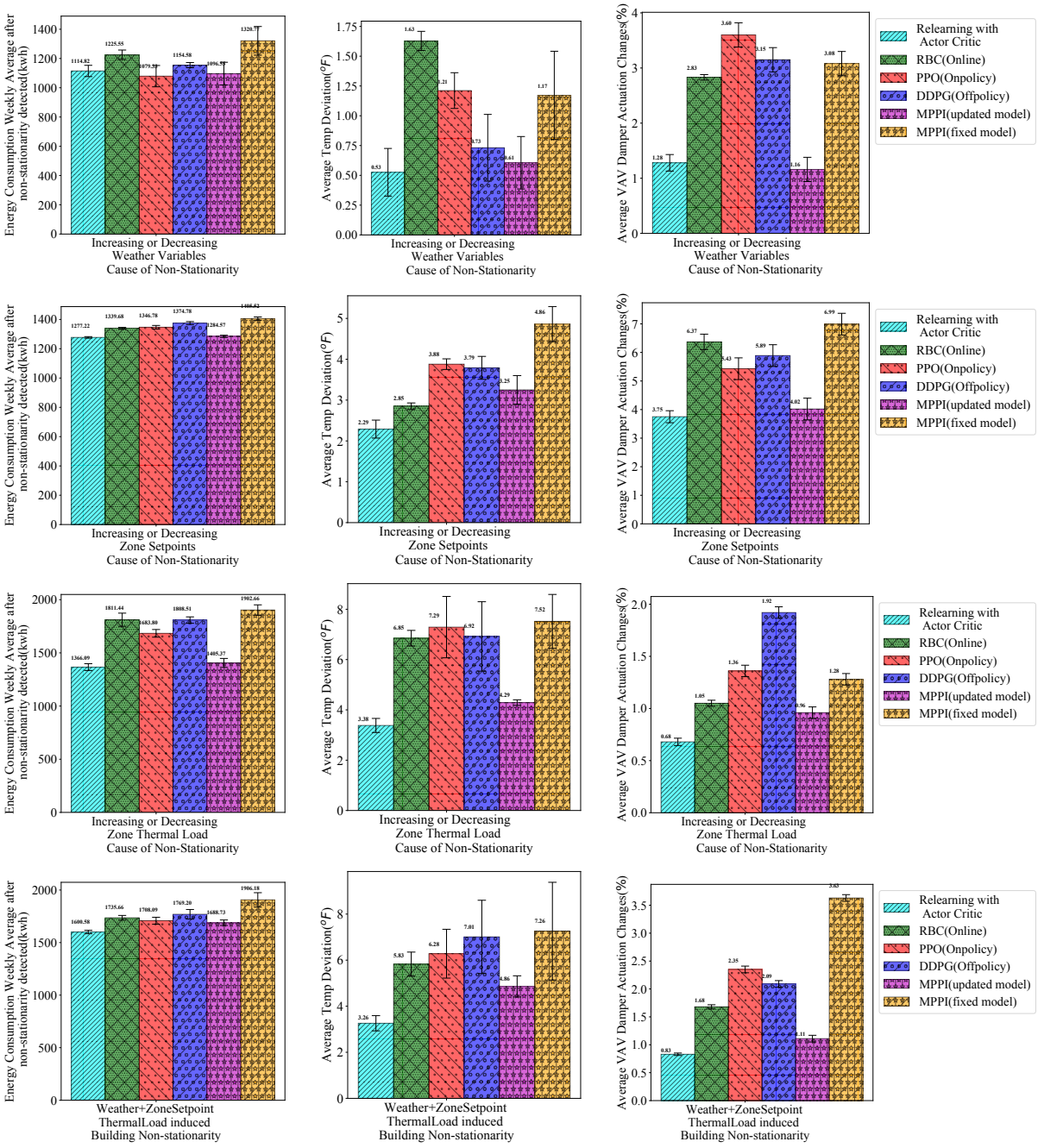

Figure 6.5: Performance of Rule-Based, PPO, DDPG, MPPI, Relearning Approach deployed on the testbed and simulated for a period of 1 year. Benchmarking is done under four conditions of non-stationarity. Metrics are recorded for a week after detection of non-stationarity using the *Performance Monitor Module*. Energy performance is aggregated for a week. Temperature deviation and Actuation rates are aggregated on a per-hour basis.

MPPI as it was facing convergence issues during the planning stage under non-stationarities.

With respect to comfort measurement indicated by the temperature deviation metrics, T_D, our approach performed significantly better than the other approaches. Here, the performances of PPO and DDPG were worse compared to even rule-based control under non-stationarity. After running the experiments, we studied the setpoints issued by either of these algorithms under non-stationarity and noticed that the setpoints generated by them were significantly diverse, indicating exploratory behavior. Both of them were trying to move from the existing policy to explore and look for better operational conditions. The effects under PPO were sometimes less significant, since it adapts using a conservative policy update-rule, while DDPG doesn't. Guideline 36 also showed worse performance compared to our approach, since it abruptly changed the operational rules based on the temperature reset requests from the zones, leading to both increased energy consumption and suboptimal temperature control. MPPI has a higher discomfort metric too. Upon investigation of the setpoints generated during the planning stage, we noticed that they were very different from one time step to the next, causing the zone VAVs to go into a hunting mode to adjust the temperature. Hence, even though it chose optimal setpoints, the frequent changes led to overall suboptimal performance. However, in our approach, the reward function ensured that sudden changes were penalized and hence ensure consistent temperatures for the zones.

For similar reasons as above, we observed that our approach and MPPI had comparatively similar performance with respect to the actuation metric A_s while performing significantly better than Guideline-36, PPO and DDPG. Again, the reward component in our approach had ensured the setpoints did not trigger large actuation signals for the VAV dampers under different conditions.

Thus, we observed that our proposed augmentations to existing DRL approaches helped it perform significantly better with respect to the energy consumption, thermal comfort and system actuation.

Next, we compare the training time of the different reinforcement learning algorithms used in this problem and highlight the advantage to learning offline for buildings. Figure 6.6 compares the time in hours it takes to adapt to the non-stationarity after its detection. Our approach is able to adjust to the non-stationarities much faster than the other reinforcement learning based approaches since it is able to simulate the changing conditions much faster on the data-driven models and collect experiences across multiple environments in parallel. Hence, the training duration is significantly lower while the number of steps simulated is much higher.

6.8 Summary

In this chapter, we demonstrated the effectiveness of our proposed approach in comparison with other well-known control approaches using a 5-zone testbed.

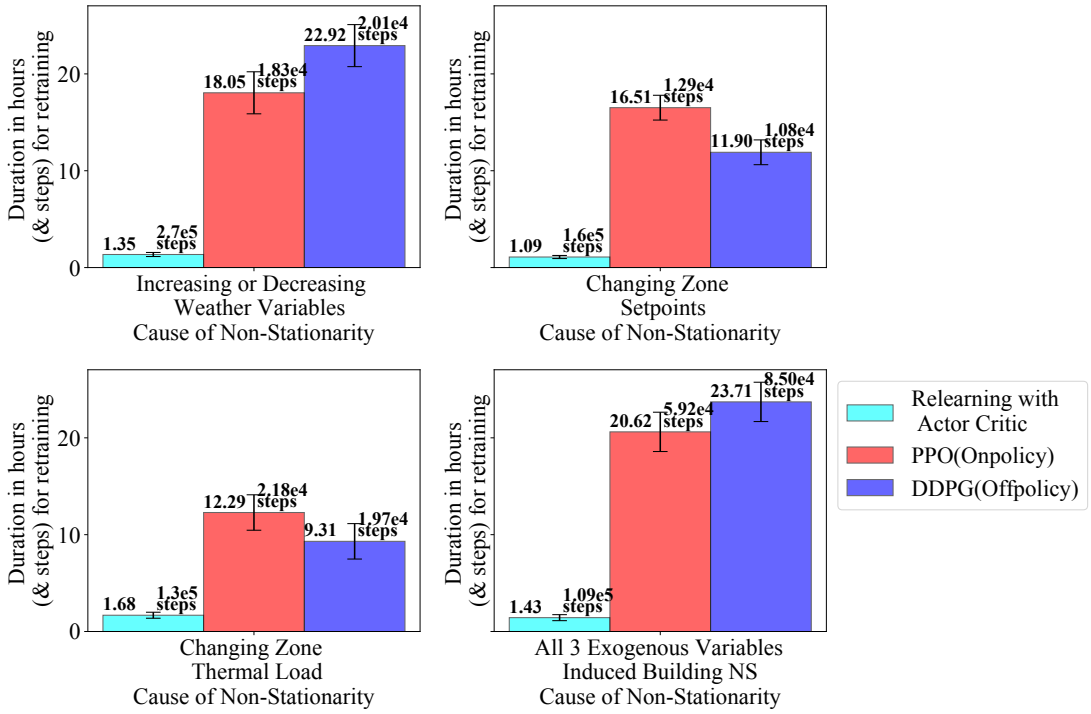

Figure 6.6: The time in hours and the number of training steps needed for adjusting to the building non-stationarities. We compare our approach with PPO and DDPG, both of which are running online on the real system

We described the approach, the implementation of the solution and the simplification of the hyperparameter optimization problem. Based on that, we found the optimal architecture of the different components of the approach.

Next, we studied the sensitivity of the approach to some important hyperparameters and saw that the forecasting horizon and the performance detection windows play a significant role in improving the performance of our approach. Hence, we had to maintain a careful balance of the availability of data and the uncertainties in model estimates based on the horizon length. We then ran benchmark experiments where we compared our approach to state-of-the-art supervisory controller strategies like Guideline-36, PPO, DDPG and MPPI. We observed that by virtue of the offline relearning technique, system model, forecasting and reward design, we were able to perform significantly better than these approaches under the conditions we tested.

We thus showed that a deep-learning based modeling and performance optimization approach for buildings based on our framework can address most of the challenges discussed in Chapter 1. These include faster adaptation using the system model for faster sampling rate, multi-actor training for more samples with decorrelated transitions, use of exogenous variable predictors for generating longer trajectories to get better idea of the system behavior under changed conditions. Our conditional relearning trigger helped us avoid continuous

expensive computation by utilizing the fact that buildings do not undergo non-stationary changes all the time.

However, this performance improvement comes at the cost of increased complexity of the approach. Each of the individual components have to perform optimally for the entire approach to work well.

Table 6.5: Tuned network architecture of the Data-driven models under four possible conditions of non-stationarity. M_{energy} predicts heating and cooling energy, M_T predicts zone temperature across all zones. Each of $M_{vav\%,z}$ predicts valve percentage in zone z.

Non-stationarity	Weather			Zone Set Point			Thermal Load			Combined		
Models	M_{energy}	M_T	$M_{vav\%,z}$	M_{energy}	M_T	$M_{vav\%,z}$	M_{energy}	M_T	$M_{vav\%,z}$	M_{energy}	M_T	$M_{vav\%,z}$
Network Inputs	oat_{t-1},orh_{t-1}, $E_{cooling,t-1}$, $E_{heating,t-1}$, $T_{disch,t-1}$	oat_{t-1},orh_{t-1}, $T_{disch,t-1}$,$T_{z,t-1}$, $T_{z,spt,t-1}$	oat_{t-1},orh_{t-1}, $T_{z,spt,t-1}$,$T_{z,t-1}$, $vav_{z,\%,t-1}$	oat_{t-1},orh_{t-1}, $E_{cooling,t-1}$, $E_{heating,t-1}$, $T_{disch,t-1}$	oat_{t-1},orh_{t-1}, $T_{disch,t-1}$,$T_{z,t-1}$, $T_{z,spt,t-1}$	oat_{t-1},orh_{t-1}, $T_{z,spt,t-1}$,$T_{z,t-1}$, $vav_{z,\%,t-1}$	oat_{t-1},orh_{t-1}, $E_{cooling,t-1}$, $E_{heating,t-1}$, $T_{disch,t-1}$	oat_{t-1},orh_{t-1}, $T_{disch,t-1}$,$T_{z,t-1}$, $T_{z,spt,t-1}$	oat_{t-1},orh_{t-1}, $T_{z,spt,t-1}$,$T_{z,t-1}$, $vav_{z,\%,t-1}$	oat_{t-1},orh_{t-1}, $E_{cooling,t-1}$, $E_{heating,t-1}$, $T_{disch,t-1}$	oat_{t-1},orh_{t-1}, $T_{disch,t-1}$,$T_{z,t-1}$, $T_{z,spt,t-1}$	oat_{t-1},orh_{t-1}, $T_{z,spt,t-1}$,$T_{z,t-1}$, $vav_{z,\%,t-1}$
Network Architecture	lstm(32)→ lstm(32)→ fnn(16)→ fnn(16)→ fnn(2)	lstm(32)→ lstm(32)→ fnn(16)→ fnn(16)→ fnn(5)	lstm(32)→ lstm(64)→ fnn(64)→ fnn(64)→ fnn(1)	lstm(64)→ lstm(64)→ fnn(32)→ fnn(32)→ fnn(2)	lstm(64)→ lstm(64)→ fnn(32)→ fnn(8)→ fnn(5)	lstm(32)→ lstm(32)→ fnn(64)→ fnn(64)→ fnn(1)	lstm(64)→ lstm(64)→ fnn(64)→ fnn(32)→ fnn(2)	lstm(32)→ lstm(64)→ fnn(64)→ fnn(8)→ fnn(5)	lstm(32)→ lstm(32)→ fnn(32)→ fnn(32)→ fnn(1)	lstm(64)→ lstm(64)→ fnn(128)→ fnn(64)→ fnn(2)	lstm(32)→ lstm(64)→ fnn(64)→ fnn(8)→ fnn(5)	lstm(64)→ lstm(32)→ fnn(32)→ fnn(32)→ fnn(1)
Network Outputs	$E_{cooling,t}$, $E_{heating,t}$	$T_{z,t}$	$VAV\%_{z,t}$	$E_{cooling,t}$, $E_{heating,t}$	$T_{z,t}$	$VAV\%_{z,t}$	$E_{cooling,t}$, $E_{heating,t}$	$T_{z,t}$	$VAV\%_{z,t}$	$E_{cooling,t}$, $E_{heating,t}$	$T_{z,t}$	$VAV\%_{z,t}$

Table 6.6: Error in the prediction models for the *Dynamic System Models*.

Non-stationarity	Weather			Zone Set Point			Thermal Load			Combined		
Model Type	M_{energy} (CVRMSE)	M_T (MAPE)	$M_{vav\%,z}$ (MAPE)	M_{energy} (CVRMSE)	M_T (MAPE)	$M_{vav\%,z}$ (MAPE)	M_{energy} (CVRMSE)	M_T (MAPE)	$M_{vav\%,z}$ (MAPE)	M_{energy} (CVRMSE)	M_T (MAPE)	$M_{vav\%,z}$ (MAPE)
Error	23.18% (3.67%)	14.29% (2.38%)	16.09% (3.08%)	19.06% (2.18%)	15.80% (1.66%)	13.88% (1.86%)	18.59% (3.91%)	17.92% (4.08%)	14.80% (2.47%)	24.86% (3.87%)	21.36% (4.58%)	18.91% (4.71%)

Table 6.7: Tuned network architecture of the Actor-Critic Network under four possible conditions of non-stationarity.

Non-stationarity	Weather		Zone Set Point		Thermal Load		Combined	
Agent Component	Actor Network	Critic Network	Actor Network	Critic Network	Actor Network	Critic Network	Actor Network	Critic Network
Network Architecture	fnn(256)→ fnn(256)→ fnn(256)→ fnn(2)	fnn(64)→ fnn(64)→ fnn(64)→ fnn(1)	fnn(64)→ fnn(256)→ fnn(128)→ fnn(2)	fnn(128)→ fnn(128)→ fnn(16)→ fnn(1)	fnn(64)→ fnn(64)→ fnn(64)→ fnn(2)	fnn(128)→ fnn(64)→ fnn(32)→ fnn(1)	fnn(256)→ fnn(256)→ fnn(64)→ fnn(2)	fnn(128)→ fnn(64)→ fnn(64)→ fnn(1)

Table 6.8: Tuned size of the Experience Buffer in hours under different non-stationarities

Non-stationarity	Weather	Zone Set Point	Thermal Load	Combined
M_e (hrs)	36	18	18	24

Table 6.9: Tuned Values of Episode Length and Discount factor under four possible conditions of non-stationarity

Non-stationarity	Weather	Zone Set Point	Thermal Load	Combined
Episode Length in days (l)	2	1	1	2
Discount Factor (γ)	0.85	0.92	0.95	0.95

Table 6.10: Model Architecture of the Exogenous Variable Predictor for Outside Air Temperature and Outside Air Relative Humidity

Model Name	Predictor of Outside Air Temperature	Predictor of Outside Air Relative Humidity
Network Inputs	$oat_{t-K:t-1}$	$orh_{t-K:t-1}$
Network Architecture	lstm(K,128)→ lstm(K,128)→ lstm(1,128)→ fnn(1,1)	lstm(K,128)→ lstm(K,128)→ lstm(1,128)→ fnn(1,1)
Network Outputs	oat_t	orh_t

Table 6.11: Error in the Exogenous Variable Predictor Models for the *Dynamic System Models*.

Model Type	OAT Predictor	ORH Predictor
MAPE	5.47% (1.36%)	11.69% (3.11%)

Table 6.12: Tuned values of the input sequence length K and the prediction horizon N under different conditions of non-stationarity.

Non-stationarity	Weather		Zone Set Point		Thermal Load		Combined	
Hyperparameter	K	N	K	N	K	N	K	N
Value (hrs)	18	72	6	72	6	72	12	48

Table 6.13: Tuned values W_{pm1} and W_{pm2} associated with the performance monitor module under different conditions of non-stationarity.

Non-stationarity	Weather		Zone Set Point		Thermal Load		Combined	
Hyperparameter	W_{pm1}	W_{pm2}	W_{pm1}	W_{pm2}	W_{pm1}	W_{pm2}	W_{pm1}	W_{pm2}
Value (hrs)	6	2	3	2	3	2	6	3

Table 6.14: Performance of Rule-Based, PPO, DDPG, MPPI, Relearning Approach deployed on the testbed and simulated for a period of 1 year. Benchmarking is done under four conditions of non-stationarity. Metrics are recorded for a week after detection of non-stationarity using the *Performance Monitor Module*. Energy performance is aggregated for a week. Temperature deviation and Actuation rates are aggregated on a per hour basis.

Metric	Cause of Non-stationarity	Guideline-36	PPO	DDPG	MPPI	Our Approach
E_{tot} (kwh)	Weather	1225.55 (32.24)	1079.33 (72.9)	1154.58 (17.62)	**1096.55 (32.2)**	1114.82 (77.72)
	Zone-Setpoint	1339.68 (6.08)	1346.78 (12.22)	1374.78 (10.53)	1284.57 (7.51)	**1277.22 (5.33)**
	Thermal Load	1811.44 (62.18)	1683.80 (35.82)	1808.51 (29.92)	1405.37 (42.28)	**1366.09 (32.15)**
	Combined	1735.66 (22.60)	1708.09 (32.82)	1769.20 (45.63)	1688.73 (26.22)	**1600.58 (16.19)**
T_D (^oF)	Weather	1.63 (0.08)	1.21 (0.15)	0.73 (0.28)	0.61 (0.22)	**0.53 (0.20)**
	Zone-Setpoint	2.85 (0.08)	3.88 (0.13)	3.79 (0.28)	3.25 (0.35)	**2.29 (0.22)**
	Thermal Load	6.85 (0.31)	7.29 (1.22)	6.92 (1.38)	4.29 (0.11)	**3.38 (0.28)**
	Combined	5.83 (0.52)	6.28 (1.06)	7.01 (1.59)	4.86 (0.46)	**3.26 (0.33)**
A_s (%)	Weather	2.83 (0.05)	3.6 (0.22)	3.15 (0.28)	1.16 (0.36)	**1.28 (0.15)**
	Zone-Setpoint	6.37 (0.27)	5.43 (0.38)	5.89 (0.73)	4.02 (0.28)	**3.75 (0.21)**
	Thermal Load	1.05 (0.03)	1.36 (0.06)	1.92 (0.08)	0.96 (0.08)	**0.68 (0.04)**
	Combined	1.68 (0.04)	2.35 (0.06)	2.09 (0.09)	1.11 (0.03)	**0.83 (0.02)**

CHAPTER 7

Experimental Studies on a Real Building

In this chapter, we describe the application of our proposed approach to a real building. We wish to compare the performance of our approach to the currently deployed reset-schedule dependent Rule-Based Control. The goal of this experiment would be to validate our approach on a real building and see whether it can perform better than rules developed by system experts while satisfying actual occupant comfort and system actuation related wear and tear. Unlike the testbed, we are limited in the number of sensor information we can collect from the building, requiring us to model more components of the building with higher fidelity. Also, we cannot separate the experiments in terms of the non-stationarities, and we have to analyze the performance under the effects of all the non-stationarities combined. Furthermore, the operational conditions of the building under which we analyze the performance are dependent on wet bulb temperatures, pre-defined cooling setpoints, zone temperature requirements and other limitations with respect to the range of the action space. We try to enforce these constraints naturally inside the reward function to make our approach more generalized.

Similar to the testbed in the previous chapter, we first map the problem for the testbed in to an LC-NSMDP formulation where we describe: (1) the state and action variables, (2) the reward function for the application, (3) the transition function in terms of multiple component (data-driven) models. Thereafter, we outline the implementation of the solution architecture, where we go in to the details of (1) the data-driven models for the transition functions and the experience buffer, (2) the supervisory controller implemented as a DRL agent, (3) the components of the exogenous variable predictors and the (4) performance monitor module. We do not go into the details of the hyperparameter separation since the entire process is identical.

Next, we report the results of the different experiments performed. First, we provide the tuned values for the hyperparameters and the optimal architectures for the data driven components of the solution architecture. We outline the evaluation procedure and results of these individual components under combined conditions of building non-stationarity since we cannot control exogenous variables in the real building.

Thereafter, we report the results for the tuned architecture, the evaluation procedures for the individual components and their corresponding results. Finally, we provide the results of the benchmarking experiments where we compare our approach to a previously deployed reset-schedule [11] Rule-based controller. Since we cannot have identical conditions for comparison in a real building, we outline a method where we established similar weather conditions to make the comparison as fair as possible.

7.1 System Description

The system under consideration is a large three-story building on our university campus. Its climate is controlled by a combination of central Air Handling Units (*AHU*) and Variable Refrigerant Flow (*VRF*) systems.The configuration of the HVAC system is shown in Figure 7.1. The AHU is a DOAS (Dedicated Outside Air Unit) system also allowing for humidity control. It has two operating modes, depending on the wet bulb temperature (t_{wb}). When $t_{wb} > 52^oF$, the cooling and the reheat coils operate to first dehumidify the air by cooling it to 52^oF close to its condensation temperature, and then reheating the air back up to a neutral relative humidity (50%). The discharge temperature as determined by the original rule-based controller was either 68^oF or 72^oF. When $t_{wb} < 52^oF$, only the preheat coil operates to heat the incoming cold air to the neutral temperature. Typically, the temperatures for the individual zones are set by the occupants, and the VRF units heat or cool the incoming air to maintain the comfort level. We propose a RL controller to discharge air into the building at a temperature that minimizes the total AHU and VRF heating and cooling energy consumption. We hypothesize that this happens when the AHU discharge temperature is such that the VRF units consume a minimum amount of heating and cooling energy to maintain the comfort levels in the individual zones. This also ensures that the VRF fans switch on and off as little as possible (minimum actuation).

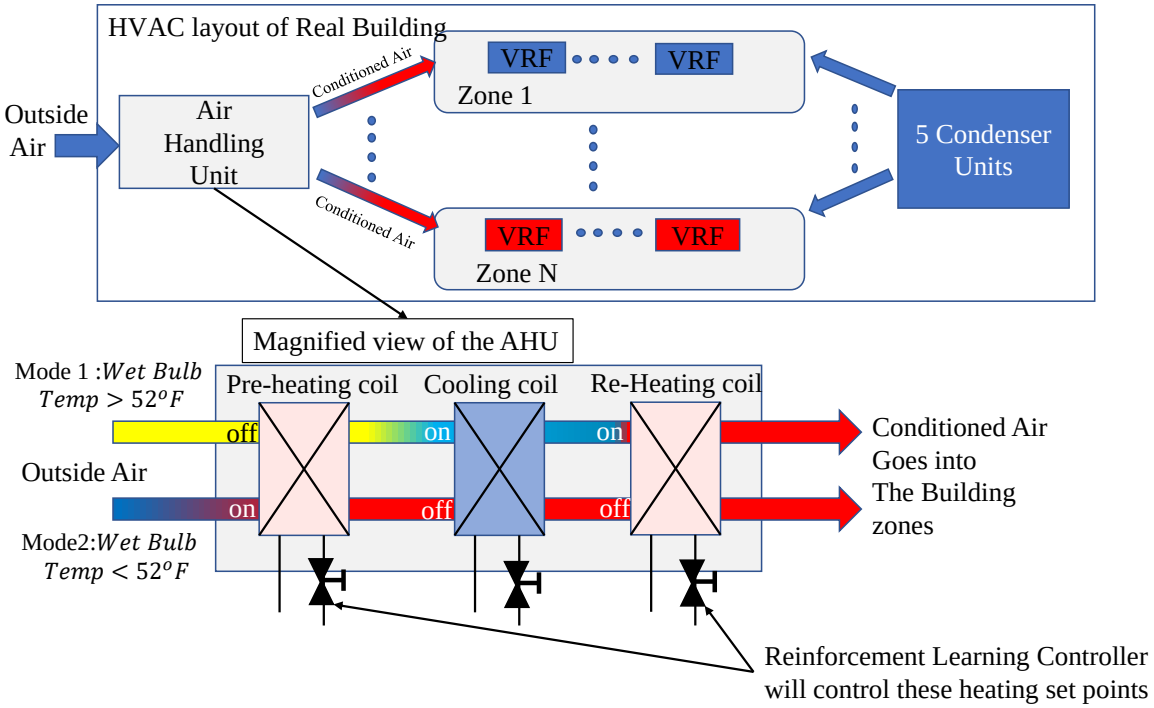

Figure 7.1: Simplified schematic of the HVAC system under Study

7.2 Problem Formulation

Similar to the testbed, Table 7.1 describes different components of the real building mapped onto the MDP.

Table 7.1: HVAC supervisory control problem for multi-zone building

Component	Variables
State $\bar{s}_t$	1.Outside Air Temperature (oat) 2.Outside Air Relative Humidity (orh), 3.Total Energy Consumption(E_{tot}), 4.AHU Discharge Air Temperature (T_{disch})
Action $\bar{a}_t$	AHU Preheat/Reheat Discharge Temperature Set-point(T_{disch_stpt})
Reward $r(s_t, a_t)$	$r_{energy} + r_{cmft}$ where $r_{energy} = -E_{tot}$ $r_{cmft} = -\sum_{z \in zones} \lvert T_{disch\_stpt} - T_{z,stpt} \rvert$
Transition Model $p(s', s_t, a_t)$	1. Total Heating Energy Consumption (M_{heat}) 2. Total Cooling Energy Consumption (M_{cool}) 3. Steam/Heating Valve State Model (M_{vlv})

In the reward function, r_{energy} incentivizes energy efficiency, r_{cmft} penalizes zone comfort violation. The total energy consumption (E_{tot}) includes Heating, and Cooling energy components. $T_{z,stpt}$ represents the temperature set-point in each zone z. The reward has been formulated after consulting the building manager who stressed on energy savings also at the zone level and suggested that if we can ensure that discharge air temperature (T_{disch_stpt}) lies close to the zone setpoints then we can save a significant amount of energy at the zone level which according to them costs more than centralized AHU energy. Hence, we formulated the second term so that T_{disch_stpt} is close to set-point for all the zones and the VRFs have to do minimum decentralized cooling or heating.

7.3 Implementation of the Solution

7.3.1 Dynamic System Model

For the Transition Model ($p(s', s_t, a_t)$) we need to evaluate the Energy (E_{tot}) using the models M_{heat} for heating energy, M_{cool} for cooling energy. In this building, heating system operates in a hybrid mode(steam heating valve on/off). So, we had to create a separate model M_{vlv} which predicts the valve behavior as on or off and is multiplied with the output of the M_{heat} model to get actual heating energy. We had to do this because the LSTM models by themselves cannot model such discrete changes. Hence, $E_{tot} = M_{heat} * M_{vlv} + M_{cool}$.

7.3.2 Experience Buffer

The *Experience Buffer* was initiated with an optimal memory size $M_e = 24$ hours.

7.3.3 Supervisory Controller

The DRL based *Supervisory Controller* was implemented using a simple Multi-Actor PPO algorithm [127]. We chose PPO because the policy updates in PPO are very conservative and there were fewer chances of the policy diverging and causing safety critical issues with the system. Further, we wanted to get the best performance for the actual deployment and PPO, being the state-of-the-art Actor Critic algorithm, was a natural choice for deployment purposes. [1] Similar to the testbed, we chose to train in $n = 10$ parallel environments. The tuned model architecture for the Actor Critic networks, the length of an episode (l) and the discount factor (γ) are summarized in Appendix B2.

7.3.4 Exogenous Variable Predictors

Similar to the testbed experiments, for the real building, we needed to predict the values of ambient dry bulb temperature(oat), the relative humidity(orh). We used the same models that we had developed in the testbed experiments. For the zone setpoints schedules based exogenous variables, models were based on simple rules which look up the current schedules in the real building and set them to those schedules for the required prediction horizon N. The optimal values for K and N in this case were $K = 12$ hours and $N = 48$ hours, respectively.

7.3.5 Performance Monitor Module

Similar to the testbed, we tracked the presence of a negative trend in the deployment phase using $i = 2$ windows W_{pm1} and W_{pm2} in parallel. The optimal values for W_{pm1} and W_{pm2} were $W_{pm1} = 6$ hours and $W_{pm2} = 2$ hours, respectively.

7.4 Hyperparameter Optimization

The hyperparameter optimization problem for the approach in case of the real building is totally identical to the approach for the testbed. We identify, separate and independently optimize subsets of hyperparameters using the same strategy. There are however two main differences. We cannot run multiple tuning trials on the real building, and the evaluation had to be performed by assuming the data-driven model ensemble as a digital twin of the real building. Here we point out the different choices for the model hyperparameters and report the optimal results in the next chapter. Secondly, we cannot control the type of non-stationarity in the real building and hence we chose to optimize the architecture and the hyperparameters for the models assuming the presence of all non-stationarities.

[1] We want to clarify that this approach is different from deploying PPO directly online on the building due to previously mentioned safety issues. This issue is further demonstrated when we perform benchmarking experiments on the testbed.

7.5 Individual Component Architecture and Performance Evaluation

Here we provide the details of the tuned model architectures resulting from hyperparameter optimization process for the real building, benchmark the performance of the approach against a previously used rule-based controller. We performed this tuning process once, since we cannot control different exogenous variables in the real building.

7.5.1 Dynamic System Model

The network architecture for the different data-driven models in the *Dynamic System Model* are show in Table 7.2. All the layers used *Batch* normalization and *relu* activations except the last layer for the M_{vlv} model, which uses a *softmax* activation.

Table 7.2: Tuned network architecture of the Data-driven models. M_{heat} predicts heating energy and M_{cool} predicts the cooling energy, respectively. M_{vlv} predicts valve status.

Models	M_{heat}	M_{vlv}	M_{cool}
Network Inputs	oat_{t-1},orh_{t-1}, $E_{heating,t-1}$, $T_{disch,t-1}$	oat_{t-1},orh_{t-1}, $T_{disch,t-1}$, $T_{z,t-1}$ $T_{z,spt,t-1}$	oat_{t-1},orh_{t-1}, $E_{cooling,t-1}$, $T_{disch,t-1}$
Network Arch	lstm(4)→ lstm(4)→ fnn(16)→ fnn(16)→ fnn(16)→ fnn(16)→ fnn(16)→ fnn(1)	lstm(8)→ lstm(8)→ fnn(16)→ fnn(16)→ fnn(16)→ fnn(16)→ fnn(16)→ fnn(2)	lstm(8)→ lstm(8)→ fnn(16)→ fnn(16)→ fnn(16)→ fnn(16)→ fnn(16)→ fnn(1)
Network Outputs	$E_{heating,t}$	$logits(vlv_{z,\%,t})$	$E_{cooling,t}$

The energy models are trained using standard *MSE* loss, while the valve model is trained using *Binary Cross-Entropy* loss. The training and testing procedure is similar to the *Dynamic System Models* used in the testbed. The energy models were evaluated using CVRMSE error, while the valve predictors were evaluated using classification accuracy. The evaluations were performed by setting $N = 48$ hours. The results are show in Table 7.3. The valve model had an ROC-AUC of 0.857.

Table 7.3: Performance of the data-driven models for the *Dynamic System Models*.

Model Type	M_{heat} (CVRMSE)	M_{vlv} (Accuracy)	M_{cool} (CVRMSE)
Performance Metric	23.12% (5.03%)	88.62% (7.08%)	21.20% (6.11%)

7.5.2 Supervisory Controller

The *Supervisory Controller* consists of the Actor Critic framework, where policy is trained using the Proximal Policy Optimization [8]. The tuned network architecture is shown in Table 7.4

Table 7.4: Tuned network architecture of the Actor-Critic Network

Hyper params	Actor Network	Policy Network
Network Arch	fnn(128)→ fnn(128)→ fnn(64)→ fnn(2)	fnn(64)→ fnn(128)→ fnn(64)→ fnn(1)

The optimal values of the episode length l and discount factor γ were found to be $l = 1$ day and, $\gamma = 0.94$ respectively.

7.6 Benchmark Experiments

The results from the real building were influenced by non-stationary changes that were not under our control. We benchmark the performance of our approach against the previous Rule Based Controller for over a year. For our approach, we analyze performance from Feb 2021 to Feb 2022. For the rule-based controller, we analyze performance from Feb 2020 to Feb 2021. While we cannot absolutely ensure a fair comparison in terms of all the variables affecting the building, we ensure that the weather conditions on a day-to-day basis are the same for both our proposed approach and the previous rule-based controller. We perform dynamic time warping to find similar weather conditions with respect to outside air temperature and relative humidity to align and compare performance on similar days. The resulting day-to-day periods of similar weather variables aligned over a period of one year is shown in figure 7.2

Based on this, we compare the performance of the two controllers with respect to the metrics discussed before. We first compare the cooling, heating/steam and electrical energy consumption. Since, there are some differences with respect to the hours spent under dehumidification and non dehumidification modes by each type of controller given the best match between weather conditions, we present the energy consumption on a kBTUs/hour basis in Table 7.5. The distinction between the modes were of interest to the building energy managers, given the different costs of the energy sources.

Our approach saved cooling and electrical energy on a per-hour basis compared to the rule based controller by 13.1% and 14.30%, respectively. However, it resulted in a slight increase in steam energy consumption, but we noticed that the overall scale of steam energy consumption is two orders of magnitude lower than cooling and electrical, and it costs less compared to electrical energy. While it is difficult to exactly infer how our framework saves cooling and electrical energy, according to the building manager, the earlier controller used

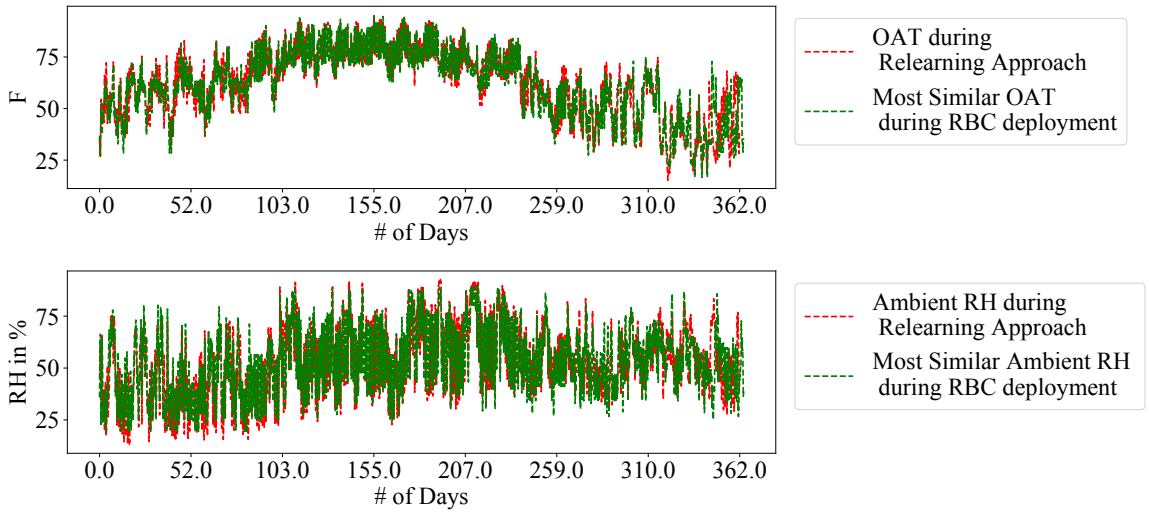

Figure 7.2: Similarity of weather during performance comparison of Relearning Approach and Rule Based Controller deployed on the real building

Table 7.5: Comparison of Energy Consumption in kBTUs/Hr for our proposed Relearning Controller vs Rule-based Controller on real building. Period of deployment under consideration: 20th Feb 2020 to 20th Feb 2021 for Rule-Based Controller and 21st Feb 2021 to 20th Feb 2022 for Relearning Controller

HVAC Mode	Metric	RBC (kBTU)	Relearning PPO (kBTU)
dehumidification	hours	4219	4355
	E_{cool}/hr	81.26	**73.80**
	E_{ee}/hr	51.20	**42.83**
	E_{stm}/hr	**0.25**	0.28
non-dehumidification	hours	4539	4403
	E_{cool}/hr	31.08	**23.86**
	E_{ee}/hr	51.80	**45.42**
	E_{stm}/hr	0.61	**0.53**

to reheat the air to ensure neutral relative humidity, but our approach avoids the extra heating as the higher relative humidity air mixes with the larger amount of lower relative humidity air already in the building and the overall increase in relative humidity is negligible. Our approach learned to exploit this by virtue of the reward function, where it emphasizes on keeping the setpoint as close as possible to the zone temperature requirements. Hence, the VRF and condenser energy consumption is minimized. Also, due to the different methods added to the vanilla DRL approach like multiagent training and forecasted performance generating more transitions, we are able to adapt faster to the humidification mode change requirements.

Finally, we show that the proposed controller maintained (1) lower deviation of zone temperature from set-point and (2) lower VRF fan-switching by virtue of our reward design. The reward function encourages the supervisory controller to discharge air at a temperature that is close to the zone set-point requirements for

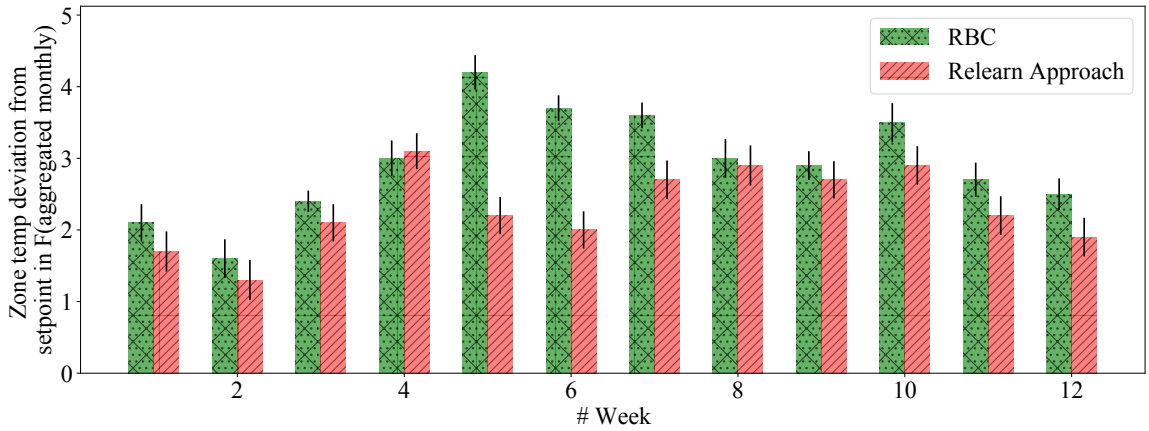

Figure 7.3: Comparison of hourly zone temperature deviation between relearning control and rule based Control

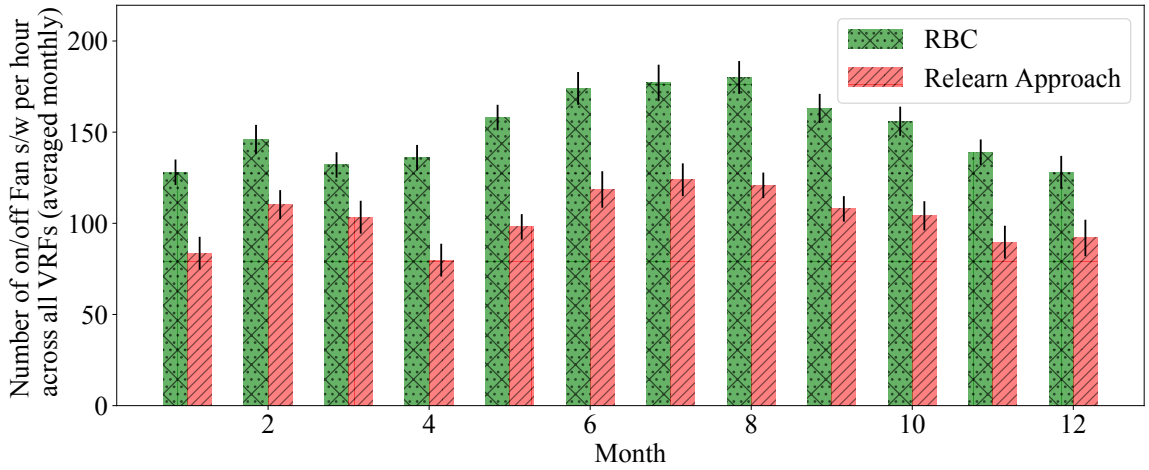

Figure 7.4: Comparison of hourly VRF fan On/Off switching between relearning control and rule based Control

the different zones (in our case, we used the average of the zone temperature). Hence, the zone deviations from set-points were on average lower and the zone VRF fans needed to switch less compared to the rule-based control as shown in Figures 7.3 and 7.4. Hence, we reduced zone temperature deviation by 21.31% and VRF fan switching by over 10.79%. According to the building manager, these numbers, over the long run, can contribute to significant energy savings and lesser actuation-related degradation of the components at the zone level.

7.7 Summary

In this chapter, we demonstrated the application of our proposed approach to a real building in our university campus. We described the approach, the implementation of the solution and the simplification of the hyperpa-

rameter optimization problem. Based on that, we found the optimal architecture of the different components of the approach.

We then ran benchmark experiments where we compared our approach to a previously used rule-based control based on reset-schedule strategy. Similar to the testbed, we observed that by virtue of the offline relearning technique, system model, forecasting and reward design, we were able to perform significantly better than the rule-based controller. Similar to the testbed, the multiple components used in the approach makes it complex and susceptible to suboptimal performance even if one of the simplest components were to underperform.

The validation of our approach on a real system with potential for energy savings and optimal comfort is an actively desired balance that building energy manager strive to attain. Also, it reduces actuation related wear and tear. Overall, it is able to achieve this without any human intervention and moves further towards the goal of true building automation for energy control.

CHAPTER 8

Conclusion and Future Work

In this work, we proposed a condition-triggered deep learning based modeling and reinforcement-learning control approach for optimizing performance in buildings modeled as non-stationary systems. The proposed approach augments vanilla deep reinforcement learning(DRL) approaches using an outer performance monitoring loop which triggers an inner relearning schedule whenever controller performance on the actual system deteriorates under non-stationarities. The inner loop adapts the system model based on new data and lets the controller policy network adjust to the new behavior of the system offline by training on this new model. A set of methods like exogenous variable forecasting, multi-actor training across parallel environments, elastic weighted consolidation based regularization procedures help in the agent training process facilitating faster convergence to a new optimum under new building behavior.

8.1 Motivation and Summary of our Approach

We identified, the shortcomings in existing rule-based, model-predictive control and deep reinforcement learning based approaches for solving the problem which assume the system to be stationary. We discussed the safety issues with optimizing the control strategy online, leading to the requirement of an offline planning or learning approach. Thereafter, we highlighted the challenges of using data-driven approaches for modeling large, complex buildings under non-stationarity and the application of reinforcement learning to these models for agent policy optimization under limitations of sampling rates and convergence issues with uncertain estimates of expected returns.

Hence, we augmented the existing approach with a conditional relearning framework based on learning the transition models and the agent policy incrementally under the assumption of Lipschitz Continuity. Inside this framework, an outer loop tracks the performance of the reinforcement learning controller deployed on the actual system. When performance degrades, it triggers the inner loop called the relearning phase, where the agent is adapted by training on an incrementally updated model of the system. We also recommended several empirical methods to improve the model training process under limited data, increase sample efficiency for policy gradient based reinforcement learning algorithms for faster convergence. Since our approach has numerous hyperparameters associated with it, we use concepts from Bayes Net and d-separation, to create independent hyperparameter subsets to simplify the hyperparameter optimization problem. Our approach is able to perform better than existing state-of-the-art approaches in modeling and control for buildings.

8.2 Contributions

We validated our approach through extensive experiments on both a testbed and a real building.

The performance monitor module helped identify non-stationarities under continuous mode changes and reduced the need for lifelong reinforcement learning on a system, optimizing resource utilization and removing the need for pretrained policies associated with different contexts.

The use of offline retraining for the DRL agent helped prevent unexpected behaviors on the real system, preventing actuation issues which might be safety critical in certain cases. This helped us attain better comfort with respect to PPO and DDPG on the testbed results.

The use of forecasts with the help of the exogenous predictor module helped address the issues of sample efficiency commonly encountered in policy gradient approaches. This was significantly highlighted in Chapter 6 where we compared training times for our approach versus an online approach based on PPO and DDPG.

The reward design emphasizing on a balance between energy, comfort and actuation is vital to our approach and led to the identification of optimal setpoints leading to better comfort compared to MPPI and Guideline-36. The ability to use feedback from multiple building sensors in the reward design also helped in avoiding artificial constraints in terms of delta temperature setpoint changes between subsequent timepoints. This is evident in the lesser actuation rate for our approach compared to Guideline-36. For the real building, the reset schedule based controller was not using these different feedbacks, which led to significant performance improvement when we deployed our approach.

8.3 Limitations

There are, however, certain limitations in our approach.

The assumption of Lipschitz Continuity is a strong one and limits our approach to non-stationarities where the effects of exogenous variables are not totally abrupt.

Since, buildings have large time-constants, the effects or responses for most of the exogenous variables are damped, allowing us to solve the problem with the assumption. However, faults have the potential to completely reconfigure building dynamics, and our approach fails to address this issue. In this regard, recent works on fault-tolerant control using meta-reinforcement learning is being actively researched.

We rely on a scalar reward signal to identify non-stationarity with the assumption that we are encompassing all possible effects into the reward formulation. Hence, for any problem, the reward has to be carefully designed to account for most of the effected variables and this can be difficult to formulate if there is limited domain knowledge about the system.

Another problem that we try to address partially in this work is the Out of Distribution(OOD) issue.

Since non-stationarities indicate a new mode of operational data for the system, existing data, though recent, can lead to errors in the state estimation leading to policy being suboptimal. We try to address these by adjusting the generalized advantage return, discount factor and by bootstrapping transitions across multiple actor workers. However, OOD can only be marginally reduced using these techniques and is an active area of research.

Lastly, as evident from the numerous components in our approach, it is susceptible to underperform if one of the components fail to perform optimally. Hence, consistent performance of all the components is essential in our approach. For this, we need to constantly evaluate the component data-driven model performances and if the accuracy or other relevant metrics indicates poor performance, steps have to be taken to ensure overall optimality for the approach.

8.4 Future Work

There are a few directions in which we wish to extend our approach for its applications to more buildings.

Our approach has the potential to be applied to multiple buildings. Since there are numerous components in the approach, it is beneficial to reuse pre-trained models from one building on the other. Hence, we wish to explore the idea of transfer learning, where we retrain parts of the models when we apply them to a different building.

In many buildings, there can be non-stationarities due to exogenous variables beyond the ones we have considered. Faults, maintenance or other transient behaviors may also trigger non-stationarities. Under these conditions, predicting exogenous variable behavior over a future horizon can be assisted by the contextual information. Nowadays, Temporal Fusion Transformers(TFTs) [158] which helps predicts exogenous variables better using contextual information like faults, discrete mode changes. Buildings often exhibit such mode changes, and such models can significantly improve estimate of exogenous variables, thereby improving estimated expected returns. This will result in a better agent training process.

With the increase in research to make buildings energy efficient, one might raise the question: How do these approaches scale to really large buildings or, in other words, whether there is a limitation with respect to how large we can make these buildings and still ensure optimal operating conditions. It becomes a problem in large buildings since multiple HVACs are used to cater to different parts of the building. Moreover, these buildings zones interact with each other, creating a coordinating-cooperating situation between the supervisory controllers. In such cases, we plan to investigate the use of multiagent reinforcement learning based coordinating and cooperative approaches. Such approaches have found success in multi-building energy optimization in a smart grid setting.

Finally, we want to conclude by stating an question. If buildings become energy efficient, would that

encourage people to build larger buildings so that they can utilize the same amount of energy to cater to a larger space? It would imply that the energy demand would stay the same and our approach is merely pivoted to help create larger buildings. What incentives can be given to limit energy consumption as a whole?

References

[1] U.S. energy facts explained - consumption and production - U.S. Energy Information Administration (EIA), May 2021. [Online; accessed 26. Jul. 2021].

[2] Department of Energy, Sep 2021. [Online; accessed 29. Sep. 2021].

[3] Buildings.Examples.VAVReheat, Jun 2021. [Online; accessed 15. Jul. 2021].

[4] Oar. Learn about Energy and its Impact on the Environment. *US EPA*, Jul 2021.

[5] Guideline 36: Best in Class HVAC Control Sequences, May 2021. [Online; accessed 21. May 2021].

[6] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566. IEEE, 2018.

[7] Anusha Nagabandi, Kurt Konolige, Sergey Levine, and Vikash Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, pages 1101–1112. PMLR, 2020.

[8] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[9] G. Wei, D. E. Claridge, et al. Optimize the Supply Air Temperature Reset Schedule for a Single-Duct VAV System. *Energy Systems Laboratory (http://esl.tamu.edu)*, 2000.

[10] Mingsheng Liu, Guanghua Wei, and DE Claridge. Calibrating ahu models using whole building cooling and heating energy consumption data. In *Proceedings of the ACEEE 1998 Summer Study on Energy Efficiency in Buildings*, volume 3, pages 3–229, 1998.

[11] Guanghua Wei, W Dan Turner, David E Claridge, and Mingsheng Liu. *Single-Duct Constant Air Volume System Supply Air Temperature Reset: Using Return Air Temperature or Outside Air Temperature?* Taylor & Francis, 2003.

[12] Jie Zhao, Khee Poh Lam, B Erik Ydstie, and Omer T Karaguzel. Energyplus model-based predictive control within design–build–operate energy information modelling infrastructure. *Journal of Building Performance Simulation*, 8(3):121–134, 2015.

[13] Jie Zhao, Khee Poh Lam, B Erik Ydstie, and Vivian Loftness. Occupant-oriented mixed-mode energyplus predictive control simulation. *Energy and Buildings*, 117:362–371, 2016.

[14] Huan Zhao, Junhua Zhao, et al. Hybrid-Model-Based Deep Reinforcement Learning for Heating, Ventilation, and Air-Conditioning Control. *Front. Energy Res.*, 0, 2021.

[15] Hussain Kazmi and Simona D'Oca. Demonstrating model-based reinforcement learning for energy efficiency and demand response using hot water vessels in net-zero energy buildings. In *2016 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, pages 1–6. IEEE, 2016.

[16] Tianshu Wei, Yanzhi Wang, and Qi Zhu. Deep reinforcement learning for building hvac control. In *Proceedings of the 54th Annual Design Automation Conference 2017*, pages 1–6, 2017.

[17] Yuanlong Li, Yonggang Wen, Dacheng Tao, and Kyle Guan. Transforming cooling optimization for green data center via deep reinforcement learning. *IEEE transactions on cybernetics*, 50(5):2002–2013, 2019.

[18] Contributors to Wikimedia projects. Data-driven control system - Wikipedia, Dec 2020. [Online; accessed 27. Jul. 2021].

[19] Fan Tang. Hvac system modeling and optimization: a data-mining approach. *Univ of Iowa MS Thesis*, 2010.

[20] Andrew Kusiak, Mingyang Li, and Zijun Zhang. A data-driven approach for steam load prediction in buildings. *Applied Energy*, 87(3):925–933, 2010.

[21] Yize Chen, Yuanyuan Shi, and Baosen Zhang. Modeling and optimization of complex building energy systems with deep neural networks. In *2017 51st Asilomar Conference on Signals, Systems, and Computers*, pages 1368–1373. IEEE, 2017.

[22] Jiahao Deng and Haoran Wang. Modeling and optimizing building hvac energy systems using deep neural networks. In *2018 International Conference on Smart Grid and Clean Energy Technologies (ICSGCE)*, pages 181–185. IEEE, 2018.

[23] Daniel L Marino, Kasun Amarasinghe, and Milos Manic. Building energy load forecasting using deep neural networks. In *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society*, pages 7046–7051. IEEE, 2016.

[24] Avisek Naug and Gautam Biswas. A Data Driven Method for Prediction of Energy Demand in Commercial Buildings. In *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, pages 335–340. IEEE, Aug 2018.

[25] Cheng Fan, Yongjun Sun, Yang Zhao, Mengjie Song, and Jiayuan Wang. Deep learning-based feature engineering methods for improved building energy prediction. *Applied Energy*, 240:35–45, 2019.

[26] Xianzhong Ding, Wan Du, and Alberto E. Cerpa. MB2C: Model-Based Deep Reinforcement Learning for Multi-zone Building Control. In *BuildSys '20: Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 50–59. Association for Computing Machinery, New York, NY, USA, Nov 2020.

[27] Chi Zhang, Sanmukh R. Kuppannagari, Rajgopal Kannan, and Viktor K. Prasanna. Building hvac scheduling using reinforcement learning via neural network based model approximation. In *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, BuildSys '19, page 287–296, New York, NY, USA, 2019. Association for Computing Machinery.

[28] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences of the United States of America*, 114(13):3521–3526, mar 2017.

[29] LAU SENG CHONG. *A MODELLING STUDY OF VARIABLE AIR-VOLUME FAN COIL UNIT (VAV-FCU)*. PhD thesis, National University of Singapore: NUS, 1999.

[30] A Eckhardt. Space hvac systems for the engine test facilities, wear and tear test facilities and transformer rooms of the ruesselsheim adam opel ag. description of the constructional and functional characteristics of the space hvac system housed in building n 25. *Tech. Bau;(Germany, Federal Republic of)*, 17(9), 1986.

[31] Abdul Afram and Farrokh Janabi-Sharifi. Review of modeling methods for hvac systems. *Applied Thermal Engineering*, 67(1-2):507–519, 2014.

[32] Archana Thosar, Amit Patra, and Souvik Bhattacharyya. Feedback linearization based control of a variable air volume air conditioning system for cooling applications. *ISA transactions*, 47(3):339–349, 2008.

[33] Xinhua Xu, Shengwei Wang, and Gongsheng Huang. Robust mpc for temperature control of air-conditioning systems concerning on constraints and multitype uncertainties. *Building Services Engineering Research and Technology*, 31(1):39–55, 2010.

[34] Fatemeh Tahersima, Jakob Stoustrup, Henrik Rasmussen, and Peter Gammeljord Nielsen. Thermal analysis of an hvac system with trv controlled hydronic radiator. In *2010 IEEE International Conference on Automation Science and Engineering*, pages 756–761. IEEE, 2010.

[35] Alessandro Beghi, Luca Cecchinato, Fabio Paggiaro, and Mirco Rampazzo. Vavac systems modeling and simulation for fdd applications. In *2011 9th IEEE International Conference on Control and Automation (ICCA)*, pages 800–805. IEEE, 2011.

[36] Bourhan Tashtoush, Mohammed Molhim, and Mohammed Al-Rousan. Dynamic model of an hvac system for control analysis. *Energy*, 30(10):1729–1745, 2005.

[37] ASHRAE Fundamentals Handbook. Energy estimating and modeling methods; si edn. *American Society of Heating, Refrigerating, and Air-conditioning Engineers, Atlanta, GA*, 2009.

[38] Sebastian Burhenne, Jan Radon, Matthias Pazold, Sebastian Herkel, and Florian Antretter. Integration of hvac models into a hygrothermal whole building simulation tool. In *Proceedings of Building Simulation 2011: 12th Conference of International Building Performance Simulation Association, Sydney, Australia*, 2011.

[39] Hui Jin. *Parameter estimation based models of water source heat pumps*. PhD thesis, Oklahoma State University, 2002.

[40] Drury B Crawley, Linda K Lawrie, Frederick C Winkelmann, Walter F Buhl, Y Joe Huang, Curtis O Pedersen, Richard K Strand, Richard J Liesen, Daniel E Fisher, Michael J Witte, et al. Energyplus: creating a new-generation building energy simulation program. *Energy and buildings*, 33(4):319–331, 2001.

[41] Matthew J Duffy, Marion Hiller, David E Bradley, Werner Keilholz, and Jeff W Thornton. Trnsys-features and functionalitity for building simulation 2009 conference. In *11th International IBPSA Conference-Building Simulation*, pages 1950–1954, 2009.

[42] Michael Wetter, Wangda Zuo, Thierry S Nouidui, and Xiufeng Pang. Modelica buildings library. *Journal of Building Performance Simulation*, 7(4):253–270, 2014.

[43] Lukáš Ferkl and Jan Širokỳ. Ceiling radiant cooling: Comparison of armax and subspace identification modelling methods. *Building and Environment*, 45(1):205–212, 2010.

[44] Jiankui Li and Zhidong Pan. Annihilator-preserving maps, multipliers, and derivations. *Linear Algebra Appl.*, 432(1):5–13, Jan 2010.

[45] Radu Bălan, Joshua Cooper, et al. Parameter identification and model based predictive control of temperature inside a house. *Energy Build.*, 43(2):748–758, Feb 2011.

[46] Liyan Jia, Zhe Yu, et al. Multi-scale stochastic optimization for Home Energy Management. In *2011 4th IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 13–16. IEEE, 2011.

[47] Jonathan Leclere, Frédéric Wurtz, and Etienne Wurtz. A low order envelope model for optimised predictive control of indoor temperature: development methodology and calibration with a numerical model. *Proceedings of BS2013*, 2013.

[48] I. Zajic, T. Larkowski, et al. Modelling of an Air Handling Unit: A Hammerstein-bilinear Model Identification Approach. In *2011 21st International Conference on Systems Engineering*, pages 59–63. IEEE, Aug 2011.

[49] Natarajkumar Hariharan and Bryan P. Rasmussen. Parameter estimation for dynamic HVAC models with limited sensor information. In *Proceedings of the 2010 American Control Conference*, pages 5886–5891. IEEE, Jun 2010.

[50] Daniel Seita. Data-Driven Deep Reinforcement Learning, May 2021. [Online; accessed 29. May 2021].

[51] Chi-man Jacob Yiu et al. Statistical modelling and forecasting schemes for air-conditioning system. *Hong Kong Polytechnic University*, 2008.

[52] Jingran Ma, S. Joe Qin, et al. Economic model predictive control for building energy systems. In *ISGT 2011*, pages 1–6. IEEE, Jan 2011.

[53] Gurvinder S Virk and Dennis L Loveday. Model-based control for hvac applications. *IEEE, PISCATAWAY, NJ,(USA).*, 3:1861–1866, 1994.

[54] Amir Alizadeh Safa. Performance analysis of a two-stage variable capacity air source heat pump and a horizontal loop coupled ground source heat pump system. *Toronto: Ryerson University*, 2012.

[55] Jung-Ho Huh and Michael J. Brandemuehl. Optimization of air-conditioning system operating strategies for hot and humid climates. *Energy Build.*, 40(7):1202–1213, Jan 2008.

[56] Rishee K Jain, Kevin M Smith, Patricia J Culligan, and John E Taylor. Forecasting energy consumption of multi-family residential buildings using support vector regression: Investigating the impact of temporal and spatial monitoring granularity on performance accuracy. *Applied Energy*, 123:168–178, 2014.

[57] Fan Zhang, Chirag Deb, Siew Eang Lee, Junjing Yang, and Kwok Wei Shah. Time series forecasting for building energy consumption using weighted support vector regression with differential evolution optimization technique. *Energy and Buildings*, 126:94–103, 2016.

[58] Bing Dong, Cheng Cao, and Siew Eang Lee. Applying support vector machines to predict building energy consumption in tropical region. *Energy and Buildings*, 37(5):545–553, 2005.

[59] Zeyu Wang, Yueren Wang, Ruochen Zeng, Ravi S Srinivasan, and Sherry Ahrentzen. Random forest based hourly building energy prediction. *Energy and Buildings*, 171:11–25, 2018.

[60] Zhun Yu, Fariborz Haghighat, Benjamin CM Fung, and Hiroshi Yoshino. A decision tree method for building energy demand modeling. *Energy and Buildings*, 42(10):1637–1646, 2010.

[61] Jin Yang, Hugues Rivard, and Radu Zmeureanu. On-line building energy prediction using adaptive artificial neural networks. *Energy and buildings*, 37(12):1250–1259, 2005.

[62] Robert H. Dodier and Gregor P. Henze. Statistical Analysis of Neural Networks as Applied to Building Energy Prediction. *J. Sol. Energy Eng.*, 126(1):592–600, Feb 2004.

[63] JF Kreider, DE Claridge, P Curtiss, R Dodier, JS Haberl, and M Krarti. Building energy use prediction and system identification using recurrent neural networks. *Transactions of ASME : Journal of Solar Energy Engineering*, 1995.

[64] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, Nov 1997.

[65] Nivethitha Somu, Gauthama Raman MR, and Krithi Ramamritham. A deep learning framework for building energy consumption forecast. *Renewable and Sustainable Energy Reviews*, 137:110591, 2021.

[66] X. J. Luo and Lukumon O. Oyedele. Forecasting building energy consumption: Adaptive long-short term memory neural networks driven by genetic algorithm. *Adv. Eng. Inf.*, 50:101357, Oct 2021.

[67] Anjukan Kathirgamanathan, Mattia De Rosa, Eleni Mangina, and Donal P Finn. Data-driven predictive control for unlocking building energy flexibility: A review. *Renewable and Sustainable Energy Reviews*, 135:110120, 2021.

[68] Irwin S Dunietz, John LC Hsu, Michael T McEachern, James H Stocking, Mark A Swartz, and Rodney M Trombly. Mpcs—the manufacturing process control system. *AT&T technical journal*, 65(4):35–45, 1986.

[69] Hailei Jiang, Sirish L Shah, Biao Huang, Bruce Wilson, Rohit Patwardhan, and Foon Szeto. Model analysis and performance analysis of two industrial mpcs. *Control engineering practice*, 20(3):219–235, 2012.

[70] Samuel Prívara, Jiří Cigler, et al. Building modeling as a crucial part for building predictive control. *Energy Build.*, 56:8–22, Jan 2013.

[71] Yaohui Zeng, Zijun Zhang, et al. Predictive modeling and optimization of a multi-zone HVAC system with data mining and firefly algorithms. *Energy*, 86:393–402, Jun 2015.

[72] HVAC system study: A data-driven approach - ProQuest, Sep 2021. [Online; accessed 22. Sep. 2021].

[73] Maryam Gholamzadehmir, Claudio Del Pero, et al. Adaptive-predictive control strategy for HVAC systems in smart buildings – A review. *Sustainable Cities and Society*, 63:102480, Dec 2020.

[74] Farinaz Behrooz, Norman Mariun, et al. Review of Control Techniques for HVAC Systems—Nonlinearity Approaches Based on Fuzzy Cognitive Maps. *Energies*, 11(3):495, Feb 2018.

[75] A. T. D. Perera and Ashen Wijesiri. Designing smart hybrid renewable energy systems with V2G. In *7th International Conference on Information and Automation for Sustainability*, pages 1–5. IEEE, Dec 2014.

[76] Farinaz Behrooz, Norman Mariun, Mohammad Hamiruce Marhaban, Mohd Amran Mohd Radzi, and Abdul Rahman Ramli. Review of control techniques for hvac systems—nonlinearity approaches based on fuzzy cognitive maps. *Energies*, 11(3):495, 2018.

[77] Young M. Lee, Raya Horesh, et al. Optimal HVAC Control as Demand Response with On-site Energy Storage and Generation System. *Energy Procedia*, 78:2106–2111, Nov 2015.

[78] Abdul Afram and Farrokh Janabi-Sharifi. Theory and applications of hvac control systems–a review of model predictive control (mpc). *Building and Environment*, 72:343–355, 2014.

[79] Richard S. Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, 2(4):160–163, jul 1991.

[80] Simon Haykin. Neural networks: A guided tour. *Nonlinear Biomedical Signal Processing*, 1:53–68, 2000.

[81] Chang-Soon Kang, Chang-Ho Hyun, and Mignon Park. Fuzzy logic-based advanced on–off control for thermal comfort in residential buildings. *Applied energy*, 155:270–283, 2015.

[82] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer science & business media, 2013.

[83] Yuan Lin, John McPhee, et al. Comparison of Deep Reinforcement Learning and Model Predictive Control for Adaptive Cruise Control. *arXiv*, Oct 2019.

[84] Joseph Lubars, Harsh Gupta, et al. Combining Reinforcement Learning with Model Predictive Control for On-Ramp Merging. *arXiv*, Nov 2020.

[85] Gianluca Serale, Massimo Fiorentini, et al. Model Predictive Control (MPC) for Enhancing Building and HVAC System Energy Efficiency: Problem Formulation, Applications and Opportunities. *Energies*, 11(3):631, Mar 2018.

[86] Ion Hazyuk, Christian Ghiaus, et al. Model Predictive Control of thermal comfort as a benchmark for controller performance. *Autom. Constr.*, 43:98–109, Jul 2014.

[87] Francesco D'Ettorre, Paolo Conti, et al. Model predictive control of a hybrid heat pump system and impact of the prediction horizon on cost-saving potential and optimal storage capacity. *Appl. Therm. Eng.*, 148:524–535, Feb 2019.

[88] M. Luzi, M. Vaccarini, et al. A tuning methodology of Model Predictive Control design for energy efficient building thermal control. *Journal of Building Engineering*, 21:28–36, Jan 2019.

[89] Yudong Ma, Francesco Borrelli, et al. Model predictive control for the operation of building cooling systems. In *Proceedings of the 2010 American Control Conference*, pages 5106–5111. IEEE, Jun 2010.

[90] A. I. Dounis and C. Caraiscos. Advanced control systems engineering for energy and comfort management in a building environment—A review. *Renewable Sustainable Energy Rev.*, 13(6):1246–1261, Aug 2009.

[91] J. Le Dréau and P. Heiselberg. Energy flexibility of residential buildings using short term heat storage in the thermal mass. *Energy*, 111:991–1002, Sep 2016.

[92] Kody T. Ponds, Ali Arefi, et al. Aggregator of Demand Response for Renewable Integration and Customer Engagement: Strengths, Weaknesses, Opportunities, and Threats. *Energies*, 11(9):2391, Sep 2018.

[93] Simin Ahmadi-Karvigh, Burcin Becerik-Gerber, et al. Intelligent adaptive automation: A framework for an activity-driven and user-centered building automation. *Energy Build.*, 188-189:184–199, Apr 2019.

[94] Jiří Cígler, Dimitrios Gyalistras, Jan Široky, V Tiet, and Lukaš Ferkl. Beyond theory: the challenge of implementing model predictive control in buildings. In *Proceedings of 11th Rehva world congress, Clima*, volume 250, 2013.

[95] Abdul Afram, Farrokh Janabi-Sharifi, et al. Artificial neural network (ANN) based model predictive control (MPC) and optimization of HVAC systems: A state of the art review and case study of a residential HVAC system. *Energy Build.*, 141:96–113, Apr 2017.

[96] Jonathan Reynolds, Yacine Rezgui, et al. A zone-level, building energy optimisation combining an artificial neural network, a genetic algorithm, and model predictive control. *Energy*, 151:729–739, May 2018.

[97] Antonio E. Ruano, Shabnam Pesteh, et al. The IMBPC HVAC system: A complete MBPC solution for existing HVAC systems. *Energy Build.*, 120:145–158, May 2016.

[98] Ján Drgoňa, Damien Picard, et al. Approximate model predictive building control via machine learning. *Appl. Energy*, 218:199–216, May 2018.

[99] Stripping off the implementation complexity of physics-based model predictive control for buildings via deep learning, Dec 2019. [Online; accessed 22. Sep. 2021].

[100] Trent Hilliard, Lukas Swan, et al. Experimental implementation of whole building MPC with zone based thermal comfort adjustments. *Build. Environ.*, 125:326–338, Nov 2017.

[101] Achin Jain, Madhur Behl, et al. Data Predictive Control for Building Energy Management: Poster Abstract. In *BuildSys '16: Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, pages 245–246. Association for Computing Machinery, New York, NY, USA, Nov 2016.

[102] Achin Jain, Derek Nong, Truong X Nghiem, and Rahul Mangharam. Digital twins for efficient modeling and control of buildings: An integrated solution with scada systems. In *2018 Building Performance Analysis Conference and SimBuild*, 2018.

[103] author:M. Wetter - Google Scholar, May 2021. [Online; accessed 3. May 2021].

[104] Michael Wetter and Jonathan Wright. A comparison of deterministic and probabilistic optimization algorithms for nonsmooth simulation-based optimization. *Build. Environ.*, 39(8):989–999, Aug 2004.

[105] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[106] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[107] Enda Barrett and Stephen Linder. Autonomous hvac control, a reinforcement learning approach. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 3–19. Springer, 2015.

[108] Volodymyr Mnih, Koray Kavukcuoglu, et al. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, Feb 2015.

[109] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019.

[110] Heider Berlink and Anna HR Costa. Batch reinforcement learning for smart home energy management. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[111] Zhiang Zhang, Adrian Chong, Yuqi Pan, Chenlu Zhang, and Khee Poh Lam. Whole building energy model for hvac optimal control: A practical framework based on deep reinforcement learning. *Energy and Buildings*, 199:472–490, 2019.

[112] Donald Azuatalam, Wee-Lih Lee, Frits de Nijs, and Ariel Liebman. Reinforcement learning for whole-building hvac control and demand response. *Energy and AI*, 2:100020, 2020.

[113] Georgios D Kontes, Georgios I Giannakis, Víctor Sánchez, De Agustin-Camacho, Ander Romero-Amorrortu, Natalia Panagiotidou, Dimitrios V Rovas, Simone Steiger, Christopher Mutschler, Gunnar Gruen, et al. Simulation-based evaluation and optimization of control strategies in buildings. *Energies*, 11(12):3376, 2018.

[114] G. T. Costanzo, S. Iacovella, et al. Experimental analysis of data-driven control for a building heating system. *Sustainable Energy Grids Networks*, 6:81–90, Jun 2016.

[115] Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms*. Chapman and Hall/CRC, 2019.

[116] Ján Drgoňa, Javier Arroyo, Iago Cupeiro Figueroa, David Blum, Krzysztof Arendt, Donghun Kim, Enric Perarnau Ollé, Juraj Oravec, Michael Wetter, Draguna L Vrabie, et al. All you need to know about model predictive control for buildings. *Annual Reviews in Control*, 2020.

[117] Damien Ernst, Mevludin Glavic, Florin Capitanescu, and Louis Wehenkel. Reinforcement learning versus model predictive control: a comparison on a power system problem. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):517–529, 2008.

[118] Richard S Sutton, Andrew G Barto, et al. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

[119] Sebastian Thrun. Lifelong Learning Algorithms. In *Learning to Learn*, pages 181–209. Springer, Boston, MA, Boston, MA, USA, 1998.

[120] Mark Bishop Ring. *Continual learning in reinforcement environments*. PhD thesis, University of Texas at Austin Austin, Texas 78712, 1994.

[121] Bruno C. da Silva, Eduardo W. Basso, et al. *Dealing with non-stationary environments using context detection*. Association for Computing Machinery, New York, NY, USA, Jun 2006.

[122] Sindhu Padakandla. A survey of reinforcement learning algorithms for dynamically varying environments, 2020.

[123] Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(4), 2010.

[124] Assaf Hallak, Dotan Di Castro, and Shie Mannor. Contextual markov decision processes, 2015.

[125] Yingying Li and Na Li. Online Learning for Markov Decision Processes in Nonstationary Environments: A Dynamic Regret Analysis. *2019 American Control Conference (ACC)*, pages 1232–1237, Oct 2012.

[126] Maruan Al-Shedivat, Trapit Bansal, Yuri Burda, Ilya Sutskever, Igor Mordatch, and Pieter Abbeel. Continuous adaptation via meta-learning in nonstationary and competitive environments, 2018.

[127] John Schulman, Filip Wolski, et al. Proximal Policy Optimization Algorithms. *arXiv*, Jul 2017.

[128] Rahaf Aljundi, Francesca Babiloni, et al. Memory Aware Synapses: Learning what (not) to forget, 2018. [Online; accessed 15. Jan. 2021].

[129] James Kirkpatrick, Razvan Pascanu, et al. Overcoming catastrophic forgetting in neural networks. *Proc. Natl. Acad. Sci. U.S.A.*, 114(13):3521–3526, Mar 2017.

[130] Rahaf Aljundi, Klaas Kelchtermans, et al. Task-Free Continual Learning, 2019. [Online; accessed 15. Jan. 2021].

[131] Vincenzo Lomonaco, Karan Desai, Eugenio Culurciello, and Davide Maltoni. Continual reinforcement learning in 3d non-stationary environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 248–249, 2020.

[132] Christos Kaplanis, Murray Shanahan, and Claudia Clopath. Policy consolidation for continual reinforcement learning. *arXiv preprint arXiv:1902.00255*, 2019.

[133] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, sep 2016.

[134] Erwan Lecarpentier and Emmanuel Rachelson. Non-Stationary Markov Decision Processes a Worst-Case Approach using Model-Based Reinforcement Learning. In *Advances in Neural Information Processing Systems*, pages 7214–7223, 2019.

[135] Andreas C Bryhn and Peter H Dimberg. An operational definition of a statistically meaningful trend. *PLoS One*, 6(4):e19241, 2011.

[136] Xiangtian Deng, Yi Zhang, and He Qi. Towards optimal hvac control in non-stationary building environments combining active change detection and deep reinforcement learning. *Building and Environment*, page 108680, 2022.

[137] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[138] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, 1992.

[139] Ivo Grondman, Lucian Busoniu, Gabriel A. D. Lopes, and Robert Babuska. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1291–1307, 2012.

[140] Aviral Kumar, Justin Fu, et al. Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction. *arXiv*, Jun 2019.

[141] Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

[142] Stuart Russell and Peter Norvig. Artificial intelligence: a modern approach, 4^{th} ed. In *Artificial intelligence: a modern approach*, 2021.

[143] Robert Ian Bowers and Catherine Salmon. Causal Reasoning. In *Encyclopedia of Evolutionary Psychological Science*, pages 1–17. Springer, Cham, Switzerland, Jul 2017.

[144] Dan Geiger, Thomas Verma, and Judea Pearl. d-separation: From theorems to algorithms. In *Machine Intelligence and Pattern Recognition*, volume 10, pages 139–148. Elsevier, 1990.

[145] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyperparameter optimization. *Advances in neural information processing systems*, 24, 2011.

[146] Bernard W Silverman. *Density estimation for statistics and data analysis*. Routledge, 2018.

[147] Thitiya Trithipkaiwanpon and Unchalisa Taetragool. Sensitivity Analysis of Random Forest Hyperparameters. In *2021 18th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pages 1163–1167. IEEE, May 2021.

[148] I. M. Sobol'. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Math. Comput. Simul.*, 55(1):271–280, Feb 2001.

[149] X.-Y. Zhang, M. N. Trame, et al. Sobol Sensitivity Analysis: A Tool to Guide the Development and Evaluation of Systems Pharmacology Models. *CPT: Pharmacometrics Syst. Pharmacol.*, 4(2):69, Feb 2015.

[150] Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E Gonzalez, and Ion Stoica. Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*, 2018.

[151] lbl srg. modelica-buildings, Jul 2021. [Online; accessed 15. Jul. 2021].

[152] AvisekNaug. 5ZoneTestBedControl, Sep 2021. [Online; accessed 26. Sep. 2021].

[153] Vijay R. Konda and John N. Tsitsiklis. Actor-citic agorithms. In *NIPS'99: Proceedings of the 12th International Conference on Neural Information Processing Systems*, pages 1008–1014. MIT Press, Cambridge, MA, USA, Nov 1999.

[154] Akram Zaytar and Chaker El Amrani. Sequence to Sequence Weather Forecasting with Long Short-Term Memory Recurrent Neural Networks. *undefined*, 2016.

[155] Zahra Karevan and Johan A. K. Suykens. Transductive LSTM for time-series prediction: An application to weather forecasting. *Neural Networks*, 125:1–9, May 2020.

[156] A ten-minute introduction to sequence-to-sequence learning in Keras, Sep 2020. [Online; accessed 27. Feb. 2022].

[157] Anusha Nagabandi, Chelsea Finn, and Sergey Levine. Deep online learning via meta-learning: Continual adaptation for model-based rl, 2018.

[158] Bryan Lim, Sercan O Arik, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *arXiv preprint arXiv:1912.09363*, 2019.