An IoT-Based Thermal Model Learning Framework for Smart Buildings

Xiangyu Zhang, Manisa Pipattanasomporn, *Senior Member, IEEE*, Tao Chen, and Saifur Rahman, *Life Fellow, IEEE*

***Abstract*—With the development of the Internet of Things (IoT) technologies, implementing intelligent controls in buildings to reduce energy consumption is becoming increasingly popular. Building climate control is of strong research interest due to high energy savings potentials associated with heating, ventilation, and air-conditioning (HVAC) controls. Because the operation of HVAC systems directly influences occupant thermal comfort, building-specific thermal models are needed for proper control. Such models describe how indoor temperature changes under different environmental conditions and HVAC status. Previous studies mainly developed thermal models for accurate building thermal simulation which are not practical for use in real-world applications as they require domain knowledge and lengthy building-by-building configuration. In this article, taking advantage of IoT technologies, a plug-and-play learning framework is proposed to automatically identify the thermal model of each thermal zone in a building without manual configuration. In contrast to the existing methods, a thermal model is learned using low-resolution temperature readings from IoT-based smart thermostats. The learning framework has been validated using the data collected from an IoT platform installed in a building over a summer. The performance of the learned model and the error of indoor temperature prediction based on this model have been evaluated and quantified. Findings demonstrate the learning process can be automated with either edge-computing or cloud-computing. This article shows the validity of the proposed IoT-based thermal model learning framework and offers a pragmatic solution for providing reliable thermal models to future smart building climate controls.**

***Index Terms*—Building energy management (BEM), building thermal model, Internet of Things (IoT), smart buildings.**

I. Introduction

With an increasing global concern for energy conservation and carbon footprint reduction, energy efficiency improvement has become the center of many research efforts. Residential and commercial buildings in particular, accounting for around 39% of the total energy consumption in the U.S. in 2017 [1] and over 70% of the total U.S. electricity usage [2], have great untapped energy-saving potential. Thanks to the Internet of Things (IoT) technologies [3], [4] and recent development on modern learning and control algorithms [5]–[7], the concept of smart homes and smart buildings has become reality. Previously, only large commercial buildings could afford to install the building energy management (BEM) systems due to their high costs. Such systems, with the most basic functionality, have an estimated initial cost on average of $2.50/square foot [8] making them prohibitive for the majority of home/building owners. Fortunately, many cost-effective solutions have now become available with the advent of IoT-based smart devices. Examples of open source IoT-based smart home/building software platforms are OpenHAB [9], HomeAssistant [10], and BEM open source software (BEMOSS) [11]. Given these affordable IoT platforms, a variety of smart building solutions have been developed for a range of purposes, such as occupant number and activity detection [12], [13], health care in smart homes [14], and many energy related applications and services [15]–[19]. From the algorithm perspective, deep learning is also widely used in different aspects of smart building, such as load forecasting [20], device control [21], and comfort modeling [22]. Among many smart home/building applications, effective control of heating, ventilation, and air-conditioning (HVAC) systems have gained increasing popularity [23]–[26]. To optimally control an HVAC system, especially during extreme temperature days, occupant thermal comfort should be taken into consideration. To this end, a reliable building thermal model is indispensable for building HVAC control. Such model describes the thermal behavior of a building, showing how thermal condition changes with respect to the indoor/outdoor environment and the HVAC status.

Existing literature shows there are two commonly used building thermal models: the first one is called first principle physics model, which uses thermal dynamics equations to describe the thermal equilibrium [27], [28]. This type of model is commonly used in building simulations (e.g., EnergyPlus [29]) and has shown good performance. The second type of thermal model is lumped-parameter reduced order model (e.g., Resistance-Capacitance models [30]–[33]), which simplifies the system's representation but still captures the

Manuscript received June 14, 2019; revised August 27, 2019 and October 16, 2019; accepted October 21, 2019. Date of publication November 4, 2019; date of current version January 10, 2020. *(Corresponding author: Xiangyu Zhang.)*

X. Zhang was with the Bradley Department of Electrical and Computer Engineering, Virginia Tech, Arlington, VA 22203 USA, and also with the Advanced Research Institute, Virginia Tech, Arlington, VA 22203 USA. He is now with the Computational Science Center, National Renewable Energy Laboratory, Golden, CO, USA (e-mail: zxymark@vt.edu).

M. Pipattanasomporn is with the Bradley Department of Electrical and Computer Engineering, Virginia Tech, Arlington, VA 22203 USA, with the Advanced Research Institute, Virginia Tech, Arlington, VA 22203 USA, and with the Smart Grid Research Unit, Chulalongkorn University, Bangkok 10330, Thailand (e-mail: mpipatta@vt.edu).

T. Chen and S. Rahman are with the Bradley Department of Electrical and Computer Engineering, Virginia Tech, Arlington, VA 22203 USA, and also with the Advanced Research Institute, Virginia Tech, Arlington, VA 22203 USA (e-mail: taoc@vt.edu; srahman@vt.edu).

Digital Object Identifier 10.1109/JIOT.2019.2951106

relevant physics of a first principle model. However, both of these models are not practical to be widely adopted in real-life application, for a few reasons. First, domain knowledge is needed for model selection and parameter identification. Second, thermal models are zone-specific, meaning the model for each thermal zone in a building needs to be individually and manually configured through calibration. Third, these models are not capable of coping with model drift or the change in building operation. For these reasons, data-driven approaches have become increasingly popular. Mustafaraj *et al.* [34], Aliberti *et al.* [35], and Mba *et al.* [36] proposed methods to combine time-series analysis and neural networks to conduct room temperature prediction using the collected temperature data. Similarly, Serra *et al.* [25] built a linear model to predict temperature and used least square estimator to identify parameters. However, these existing data-driven models either are tested using simulated data or require relatively accurate time-series temperature data, and therefore in real-world applications, a sophisticated sensor network is needed, which inevitably incurring extra commission effort and expenditures.

In this article, to overcome the drawbacks of the physics-driven modeling methods and avoid the additional hardware investment required by existing data-driven approaches, we propose an alternative pragmatic building thermal model learning framework based on low-cost IoT devices. It is designed to automatically identify and learn a building thermal model that describes building thermal behaviors under different environments and controls. To this end, our previous work in [37] and [38] proposed a method using low-resolution temperature data collected from smart thermostats to perform demand response in buildings. Built on the application presented in [37], this article proposes a plug-and-play building thermal model learning framework that can be easily integrated with any IoT-based BEM system. The proposed framework only relies on data collected from low-cost IoT-based smart thermostats, which are affordable for most building owners. To the best of our knowledge, no other previous work has proposed such a cost-effective and practical building thermal model learning framework for use in real-world applications. Furthermore, we developed and deployed the framework in a real building and quantified the indoor temperature prediction errors based on the learnt model. Therefore, this article is expected to give an insight into the development of a building thermal model for the IoT-based BEM industry, and help to enable and popularize advanced HVAC control in IoT-based smart building applications. We also demonstrate the computation efficiency of the learning framework on both edge device and cloud infrastructure.

The rest of this article is arranged as follows. Section II introduces the building and the IoT BEM platform used in this article. Section III presents a thermal model adapted to this data-driven solution. Section IV describes the learning framework, including the learning pipeline and two types of implementations. In Section V, a case study based on a real building is conducted for validating the framework and evaluating the resulting thermal model. This article is concluded with highlights and future work in Section VI.

(a) (b)

Fig. 1. (a) Exterior view of the office building. (b) Edge devices deployed in the building inside an NEMA box.

II. Testbed Building and Installed IoT Platform

An office suite in a building on Virginia Tech campus in Blacksburg, VA, USA was used as a testbed for the proposed thermal model learning framework. This office suite shares the exterior walls (south and west) with the building. In this suite, there are ten small offices, a small kitchen, and a public area, with a total area of approximately 3000 square feet.

A BEM system is a software system that provides operation monitoring, device control, and intelligent applications to buildings, aiming at optimally managing the building energy usage and indoor comfort. The proposed learning framework was implemented as an application on a BEM system. Similar to a cellphone app on a mobile operating system, while a BEM system provides lower level services (e.g., interfacing with IoT devices in the building and historical data storage), the applications (e.g., the proposed framework) focus on solving some specific problems (e.g., learning building thermal model). In this article, BEMOSS [11], an IoT-based multiagent BEM system, was deployed in this test building, operating on an edge device (an octa-core single board computer (SBC) with 2-GB RAM). Though relying on BEMOSS in this article, the proposed learning framework can be implemented on any other BEM systems.

To collect the building thermal-related data, an off-the-shelf smart thermostat was deployed to control the HVAC rooftop unit of this office suite and monitor indoor temperature. Communicating via Wi-Fi, BEMOSS's device agent collected the operating data from the thermostats and saved them in a NoSQL database on the edge device. As mentioned earlier, choosing smart thermostats over a dedicated temperature sensor network is more affordable and smart thermostats provide multiple functionalities. Thus, they have higher return on investment for massive deployment in smart buildings. It is also assumed that thermostats are properly commissioned and can correctly reflect the overall thermal condition of a room. However, due to the cost constraint and sensor sensitivity, most smart thermostats in the market have coarse-grained temperature readings with typical measurement granularities of 0.5 °F or 1.0 °F. Because of this, it is difficult to model the building thermal property using traditional time-series approaches (e.g., ARIMAX). As a result, to learn building thermal model from these low-resolution data, a special form of building thermal model was derived as discussed in Section III.

III. Building Thermal Model

In this section, the underlying concept is to incorporate previously validated empirical building thermal models into

a data-driven approach. The overall objective is to develop accurate thermal models based on low-resolution data from smart thermostats. Note that although experiments in this article were conducted in summer, the proposed framework can also be used in winter with little modifications.

A. Building Thermal Dynamics

Shao [39], [40] proposed an empirical building thermal model to calculate indoor temperature at the next time step (T_{i+1}), given current indoor temperature (T_i), air-conditioner (AC) OFF/ON status ($S_i \in \{0, 1\}$) and other parameters, as shown in

$$T_{i+1} = T_i + \Delta t \cdot \frac{G_i}{\overline{C_{\text{air}}} \cdot \overline{V_{\text{zone}}}} + \Delta t \cdot \frac{\overline{C_{HVAC}} \cdot S_i}{\overline{C_{\text{air}}} \cdot \overline{V_{\text{zone}}}}. \quad (1)$$

G_i above is the heat gain rate, which is calculated using

$$G_i = \left(\frac{\overline{A_{\text{ceiling}}}}{\overline{R_{\text{ceiling}}}} + \frac{\overline{A_{\text{window}}}}{\overline{R_{\text{window}}}} + \frac{\overline{A_{\text{wall}}}}{\overline{R_{\text{wall}}}} \right) \cdot \left(T_{\text{out},i} - T_i \right) + \overline{SHGC} \cdot \overline{A_{\text{south\_window}}} \cdot I_{\text{solar}} \cdot \overline{\lambda} + G_{\text{occ}}. \quad (2)$$

In (1) and (2), all parameters with an overline are the time invariant parameters (constants) for the thermal zone, e.g., $\overline{V_{\text{zone}}}$ represents the volume of the thermal zone. $\overline{C_{HVAC}}$ is the cooling capacity, a negative number. $\overline{C_{\text{air}}}$ is heat capacity of air. In (2), the room's ceiling, window, and wall have the areas of $\overline{A_{\text{ceiling}}}$, $\overline{A_{\text{window}}}$, and $\overline{A_{\text{wall}}}$, respectively. Additionally, the window facing south has the area of $\overline{A_{\text{south\_window}}}$. Their heat resistances are $\overline{R_{\text{ceiling}}}$, $\overline{R_{\text{window}}}$, and $\overline{R_{\text{wall}}}$. Δt is the time interval. $T_{\text{out},i}$ is the outdoor temperature at time step i. $\overline{SHGC}$ is the window's solar heat gain coefficient. I_{solar} is the solar irradiance, $\overline{\lambda}$ is a constant coefficient and G_{occ} describes the heat gain from occupants.

As previously mentioned, these model parameters usually are not readily accessible, require domain knowledge to understand, or require additional special sensors to capture such information (e.g., for I_{solar}). Thus, in practice, (1) is not suitable to be directly integrated into an indoor temperature prediction model. Instead, we adapt the model to an alternative data-driven approach.

Based on (1), temperature gradient (dT_i/dt) is defined with approximation in

$$\frac{dT_i}{dt} \approx \frac{T_{i+1} - T_i}{\Delta t} = \frac{G_i + \overline{C_{HVAC}} \cdot S_i}{\overline{C_{\text{air}}} \cdot \overline{V_{\text{zone}}}} = \begin{cases} \frac{G_i}{\overline{C_{\text{air}}} \cdot \overline{V_{\text{zone}}}} = \left(\frac{dT_i}{dt}\right)^+, & \text{if } S_i = 0 \\ \frac{G_i - |\overline{C_{HVAC}}|}{\overline{C_{\text{air}}} \cdot \overline{V_{\text{zone}}}} = \left(\frac{dT_i}{dt}\right)^-, & \text{if } S_i = 1 \end{cases} \quad (\forall i > 0). \quad (3)$$

According to (3), indoor temperature gradient (dT_i/dt) includes rate of temperature increase $(dT_i/dt)^+$, and rate of temperature decrease $(dT_i/dt)^-$, both of them depend on G_i. When Δt is small, variables that determine G_i ($T_{\text{out},i}$, T_i, I_{solar}, and G_{occ}) can be assumed constant. Based on this assumption, dT_i/dt is also taken as constant during a short period. As a result, by learning from building operation data, a data-driven thermal model has been proposed in [37] to predict dT_i/dt accurately under different environments. The problem of learning a building thermal model is therefore transformed into a supervised learning problem: mapping some influencing factors to dT_i/dt. In fact, in a building, most of the parameters, such as the architectural and material properties (e.g., $\overline{V_{\text{zone}}}$), remain constant. So, for a specific building, dT_i/dt can be modeled as it is only affected by the time-variant influencing factors (i.e., $T_{\text{out},i}$, T_i, I_{solar} and G_{occ}).

Finally, with the predicted $(dT_i/dt)^+$ and $(dT_i/dt)^-$, the next step temperature can be calculated using

$$T_{i+1} = \begin{cases} T_i + \left(\frac{dT_i}{dt}\right)^+ \cdot \Delta t, & \text{if } S_i = 0 \\ T_i - \left(\frac{dT_i}{dt}\right)^- \cdot \Delta t, & \text{if } S_i = 1 \end{cases} \quad (\forall i > 0). \quad (4)$$

It is worth noting that one major advantage of this approach is that it allows the use of low-resolution temperature data, on which the traditional time-series analysis methods do not have good performance.

B. Feature Selection and Feature Engineering

Because not all features are directly observable (i.e., I_{solar} and G_{occ}), some easily accessible surrogate features that reflect both internal heat gain and outdoor heat gain are used to predict dT_i/dt. These features are indoor and outdoor temperatures in °F (T_i and T_{out}), outdoor relative humidity in percentage (ϕ_{out}), hour of day in the 24 h (t), day of week with day=1∼7 meaning Monday to Sunday (d) and four classes of weather types (w). Among them, the day of week and time of day features are used to approximate the occupancy level (G_{occ}), and the hour of day and weather class are used to approximate the solar irradiance (I_{solar}). These two approximations are suitable for buildings with comparatively fixed schedules and without the budget necessary to install sensors for occupancy and solar irradiance. More details about the validity of these assumptions can be found in [23]. Although the measurements of occupancy level and solar irradiance are assumed unavailable in this study, it is worth noting that the model is flexible enough to utilize these values should they be available.

In all, the goal is to train a learning model that reveals the relationship shown in

$$\left| \left(\frac{dT_i}{dt} \right)^* \right| = f^*(T_i, T_{\text{out}}, \phi_{\text{out}}, t, d, w). \quad (5)$$

Considering that under the same environment, the values of $(dT_i/dt)^+$ and $(dT_i/dt)^-$ are most likely to be different, two separate models are used for the $(dT_i/dt)^+$ and $(dT_i/dt)^-$ prediction. Therefore, in (5), these models predict absolute value of temperature gradient for simplicity (* can be either + or −).

Before using these features directly, feature engineering is needed for different types of features, as follows.

1) Continuous Features: Continuous features (T_i, T_{out}, and ϕ_{out}) remain the same.

2) Cyclic Ordinal Features: Time and day are cyclic ordinal features (e.g., 23 o'clock is close to 1 o'clock, though their

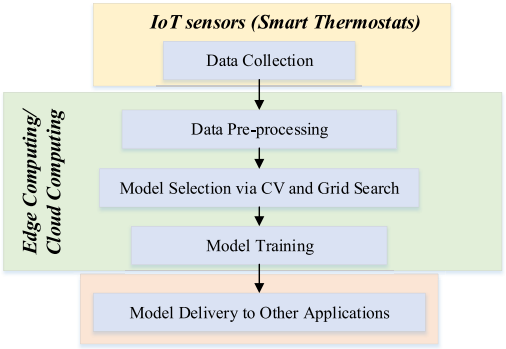

Fig. 2. Thermal model learning pipeline.

ordinal value separate apart). Trigonometric transformation is used to project the feature into a vector while keeping the proximity of two similar features, as shown in

$$t_1, t_2 = -\cos\left(\frac{t}{24}\cdot 2\pi\right), \sin\left(\frac{t}{24}\cdot 2\pi\right) \tag{6}$$

$$d_1, d_2 = -\cos\left(\frac{d}{7}\cdot 2\pi\right), \sin\left(\frac{d}{7}\cdot 2\pi\right). \tag{7}$$

3) Categorical Features: The weather class is a four-value categorical feature, dummy variable encoding is used to preprocess the feature before putting them into the learning model.

In terms of the above feature preprocessing, the supervised learning model (5) becomes the following equation with $X = [T_i, T_{\text{out}}, \phi_{\text{out}}, t_1, t_2, d_1, d_2, w_1, w_2, w_3]$:

$$\left|\left(\frac{dT_i}{dt}\right)^*\right| = f^*(X). \tag{8}$$

Preprocessed data are standardized so that each feature has zero mean and unit variance.

For the rest of this article, the building thermal model refers to the temperature gradient prediction model (8). The proposed framework that automatically learns the thermal model based on IoT devices and platform is discussed next.

IV. IoT-Based Thermal Model Learning Framework

This section describes the thermal model learning framework architecture and the hardware used to implement it.

A. Thermal Model Learning Pipeline

The thermal model learning pipeline is shown in Fig. 2.

It consists of three major steps: 1) data collection from thermostats; 2) thermal model learning on an edge device or a cloud infrastructure; and 3) final learned model delivery to other smart control applications. The overall pipeline is fully automated, without any human intervention. The learning process starts with data preprocessing. That is, historical temperature data for the past 30 days are retrieved and prepared as training dataset. In this article, the choice of a 30-day period assumes that the approximations on occupancy and solar radiance (Section III-B) are valid through this short term but not for any longer term. The ground truth values for temperature gradient is calculated according to

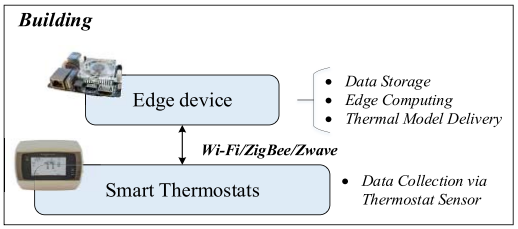

Fig. 3. Learning pipeline implemented on an edge device.

$$\left(\frac{dT}{dt}\right)^- = \frac{\Delta T}{\Delta t_{\text{dec}}} \tag{9}$$

$$\left(\frac{dT}{dt}\right)^+ = \frac{\Delta T}{\Delta t_{\text{inc}}}. \tag{10}$$

Here ΔT is the thermostat control dead band and $\Delta t_{\text{dec}}/\Delta t_{\text{inc}}$ are the duration for the temperature decrease and increase.

Input features are processed as explained in Section III-B. With both target values and input features in the desired format, they are sent to the second step. K-fold cross validation (CV) [41] and grid search for optimal model hyperparameters are performed on these training data. The reason for conducting model selection is that different models might be suitable for different buildings. The best models are selected according to the least root mean squared error (RMSE). Five commonly used supervised learning models are tested as candidates [41].

1) Polynomial regression (PR).
2) Support vector regression (SVR).
3) Random forest (RF).
4) Extreme gradient boost (XGB).
5) Neural network (NN).

Lastly, the best performing model and hyper-parameters are used for training the temperature gradient prediction model. The number of folds in CV is typically 5 or 10. In this article, if data sample size is less than 200, 5-fold CV is used; otherwise, we choose 10-fold CV. All data used for model learning are easily accessible and the acquisition process is fully automated, which guarantees the self-learning feature of the framework. The thermal model is retrained every few days with retrain frequency chosen depending on model performance.

B. Implemented Architectures

The proposed learning framework can be implemented on either an edge device or a cloud infrastructure.

The first scenario relies on an edge device or an edge device cluster located inside a building to collect data from smart thermostats over a local area network. The edge device is responsible for storing temperatures and AC status in a database, performing thermal model learning, and delivering the learned thermal model delivery to other applications (see Fig. 3).

In the second scenario, the entire learning pipeline resides on cloud infrastructure. It is responsible for collecting building operation data from smart thermostats either via a smart

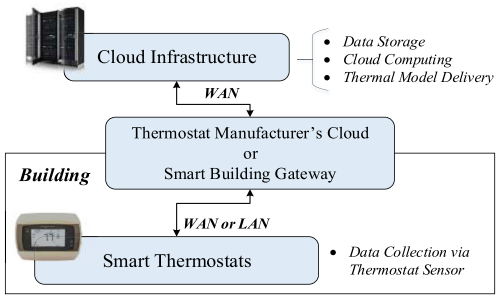

Fig. 4. Learning pipeline implemented on a cloud infrastructure.

TABLE I
SAMPLE THERMOSTAT DATA COLLECTED BY BEMOSS

Date	Time	T_{set}	T	AC Mode	AC Status
2016-06-10	20:57:20	73.0	73.5	COOL	OFF
2016-06-10	21:19:40	73.0	74.0	COOL	COOL
2016-06-10	21:27:40	73.0	73.5	COOL	COOL
2016-06-10	21:30:40	73.0	73.0	COOL	OFF

building gateway for a secure data relay or another cloud infrastructure from the thermostat manufacturer (see Fig. 4).

V. CASE STUDY

This section discusses the data collection and test process of the learning model framework in the testbed building, as well as its performance and computational efficiency implemented on an edge device and a cloud infrastructure.

A. Data Collected

The thermostat readings used in this article were collected in the months of May, June, July, August, and September in 2016. The temperature data and AC operation data were recorded in the Apache Cassandra database, as exemplified in Table I.

In Table I, T_{set} is the temperature set point in °F; T is the indoor temperature in °F. This smart thermostat has the granularity of 0.5 °F and the control dead-band of 1.0 °F. Note that the thermostat used in this article is a U.S. device, hence it has Fahrenheit as temperature unit. However, the proposed method is valid regardless of the temperature unit used by the device. AC mode (COOL/HEAT) indicates whether the AC is running on cooling or heating mode; and AC status (ON/OFF) shows whether the compressor in the AC unit is actively cooling or not operating. Weather information were logged in a separate table.

B. Data Preparation

Data preparation consists of two steps: 1) Data fusion –aggregating data from different tables and 2) Feature engineering, explained in Section III-B. Example results from the data fusion step is shown in Table II. Using the same dataset, Table III shows the results from both the data fusion and the feature engineering steps. The target and features are denoted as y and $\mathbf{X}$.

TABLE II
SAMPLE THERMOSTAT DATA AFTER PREPROCESSING

$\frac{dT_i}{dt}$ (°F /second)	T_i (°F)	T_{out} (°F)	ϕ_{out} (%)	t	d	w
9.009*10-4	73.0	72.1	73.0	11	1	2
7.936*10-4	73.0	72.5	69.0	12	1	3

TABLE III
EXAMPLE OF FINAL LEARNING MODEL INPUTS (TARGET AND FEATURES)

Target	*Features*
y (10^{-4})	$\mathbf{X}=[T_i, T_{out}, \phi_{out}, t_1, t_2, d_1, d_2, w_1, w_2, w_3]$
9.009	(73.0, 72.1, 73.0, 0.97, 0.26, −0.62, 0.78, 0.0, 1.0, 0.0)
7.936	(73.0, 72.5, 69.0, 1.0, 0.0, −0.62, 0.78, 1.0, 0.0, 0.0)

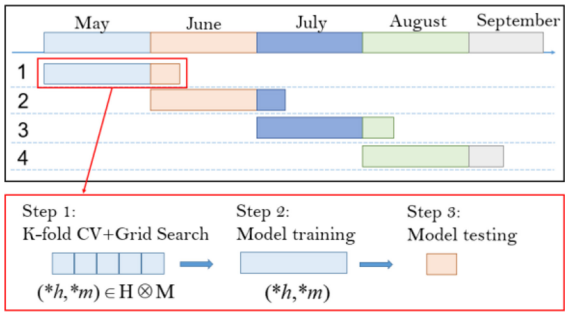

Fig. 5. Data sets and procedure for model selection, training and testing. H is the hyper-parameters grid, M are all candidate models. $*h$ and $*m$ are the selected hyper-parameter and model pair.

C. Cross-Validation and Grid-Search Using Training Data

In this article, data collected from May to early September were divided into four training/testing datasets, as shown in Fig. 5. Data from the 1st to the 30th/31st of each month were used for training while the first 100 data points of the next month were used for testing. This is following the idea of using the previous 30-day data as the training dataset mentioned earlier.

For each training/testing dataset, the following steps were followed to evaluate different machine learning models.

Step 1: Conduct K-fold CV using the training data on each model with every possible hyper-parameter set ($H \otimes M$).

Step 2: Train the best model, hyper-parameters pair ($*h$, $*m$) selected from step 1 with the training data.

Step 3: Evaluate the trained model using the testing data.

D. Thermal Model Evaluation Results

1) Cross-Validation Results: Among all models in $H \otimes M$, the top five estimators for each training dataset are listed for each month in Table IV. In addition, corresponding sample size of the training dataset is also listed. As shown, the top performers in the CV using the training data in the month of May are five PR models with different hyper-parameters. According to Table IV, SVR appears the most times as the best estimators to forecast the temperature gradient [i.e., $(dT_i/dt)^-$ and

TABLE IV
TOP FIVE ESTIMATORS FOR EACH MONTH IN BOTH TEMPERATURE INCREASING/DECREASING CASES

Training Month	$(dT_i/dt)^-$ Forecast		$(dT_i/dt)^+$ Forecast	
	Number of Samples	Top Performers*	Number of Samples	Top Performers*
May	166	**P**/P/P/P/P	154	**P**/P/P/P/P
June	417	**S**/X/X/S/X	396	**X**/X/S/X/X
July	280	**S**/S/P/P/P	260	**S**/S/S/P/P
August	392	**S**/S/S/S/S	361	**R**/R/P/S/R

* P-Polynomial Regression, S-SVR, X-XGBoost, R-Random Forest

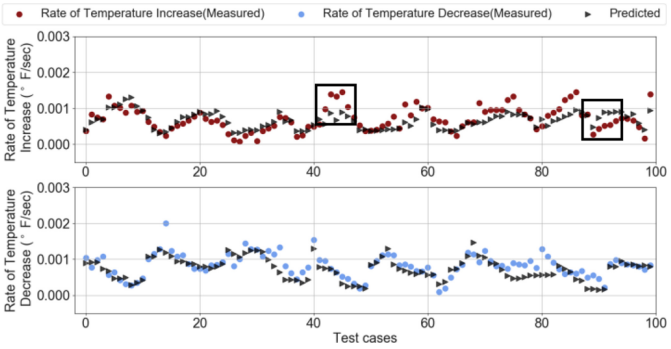

Fig. 6. Predicted and measured temperature gradient in July test cases.

TABLE V
VARIANCE EXPLAINED AND SCALED MAE OF (dT_i/dt) PREDICTION

Month of Test Data	Variance explained		scaled MAE (%)	
	$(dT_i/dt)^+$	$(dT_i/dt)^-$	$(dT_i/dt)^+$	$(dT_i/dt)^-$
June	0.6111	0.3418	26.2427	20.3892
July	0.5024	0.5802	26.0111	20.0527
Aug	0.6442	0.3534	26.7047	20.5943
Sept	0.7743	0.7522	19.5635	15.0893

$(dT_i/dt)^+$] in the above eight CVs. All of these top performing SVR models used the RBF kernel. Hence, the RBF kernel SVR can better capture the nonlinear relationship between the features and the temperature gradient for this specific office suite. However, with less training data as shown in the May test cases, PR model appears to perform better than other models. According to Table IV, the neural network does not appear as one of the top performing models. This could be because: 1) while NN requires more training data than other models to have good performance, the amount of training data is relatively small in this application and 2) searching for the proper combination of hyper-parameters is not as easy as other models.

2) Temperature Gradient Prediction: After being trained with the best performing hyper-parameters and models selected in cross-validation, the thermal models were used to predict the temperature gradient in the test data sets. The comparison between predicted and ground-truth values are illustrated in Fig. 6, exemplified using the July test cases. For the July test cases, the $(dT_i/dt)^+$ and $(dT_i/dt)^-$ prediction models were implemented using the XGBoost and SVR model, respectively, according to the CV results. According to Fig. 6, the predicted values of temperature gradient closely match the ground-truth values in most of the cases. However, in some cases, the deviations between the predicted and real values appear larger, as shown in the black squares. This, in general, was caused by abnormal daytime office activities, such as higher than usual occupancy level or late office open.

To quantify the error, two metrics were used: the variance explained and the scaled mean absolute error (scaled MAE)

$$\text{explained\_variance}(y, y*) = 1 - \frac{\text{Var}\{y - y*\}}{\text{Var}\{y\}} \tag{11}$$

$$\text{scaled\_MAE} = \frac{\text{Mean Absolute Error}(y, y*)}{\text{mean}(y)}. \tag{12}$$

The variance explained and scaled MAE of all four test cases are shown in Table V. The results indicate the variance explained of 34.18%–77.43% and the scaled MAE is around 20%. Month-to-month variations exist due to varying building operating conditions, like changes in occupancy level or window open/close status. This makes it harder for the proposed learning framework to learn the precise thermal property of a thermal zone/building.

Since this method is proposed as a practical and cost-efficient solution, it is not expected to be extremely accurate and cannot explain all the variations in building operation due to the absence of sophisticated sensors. In contrast, this article aims to quantify the extent of the model's prediction accuracy. In addition to determining the model prediction accuracy on temperature gradient, the model performance to predict indoor temperature is discussed next.

3) Indoor Temperature Prediction: Given the indoor temperature gradient prediction model, the indoor temperature can be forecasted using Algorithm 1. One of the inputs is the operating schedule of AC units. The output is the indoor temperature profile during the prediction horizon under such operating schedule.

Because the indoor temperature and status of the HVAC unit were logged, the historical data were used to evaluate the performance of this prediction algorithm in a posterior manner. That is, the indoor temperature profile was "predicted" during a selected past period and compared with the recorded HVAC operation profile. To better describe the experiment procedure, a few terms used in the following discussion are presented.

1) *Status Change:* The event when the status of the AC unit changes, either from ON to OFF or from OFF to ON.
2) *Session:* A session is between any AC status changes. Hence, there are AC operation sessions (when room temperature decreases) and AC nonoperation sessions (when room temperature increases).
3) *Period:* A duration consists of several consecutive sessions.

Based on these definitions, the procedure below was followed to evaluate the temperature prediction errors.

1) Train the $(dT_i/dt)^+$ and $(dT_i/dt)^-$ prediction models with the CV selected estimators using 30-day data.

Algorithm 1 Indoor Temperature Prediction Algorithm

1: Divide the prediction horizon into N steps, each step represents a 5-min interval ($\Delta t = 300$).
2: Initializing: obtain T(0) and AC unit operating schedule for the prediction horizon.
3: **for** $i = 1 \rightarrow N$:
4: a. Prepare the feature data based on forecasted weather data.
5: b. data preprocessing (transforming the data to the 10-dimension standard feature input vector).
6: c. check AC unit operating schedule at Step i
if $status(i) = ON$:
predict rate of temperature decrease using the learned prediction model: $\frac{dT}{dt} = -\left|(\frac{dT}{dt})^{-}\right|$
elif $status(i) = OFF$:
predict rate of temperature increase using the learned prediction model: $\frac{dT}{dt} = \left|(\frac{dT}{dt})^{+}\right|$
end if
7: d. predict temperature at Step i according to the following equation: $T(i) = T(i-1) + \Delta T = T(i-1) + \frac{dT}{dt} \times \Delta t$
8: **end for**
9: **return** T(i) for $i = 1 \rightarrow N$

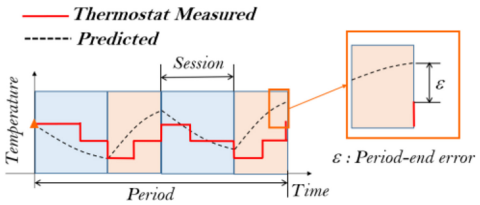

Fig. 7. Error evaluation on time-series prediction of indoor temperature.

2) Choose the first nine days in the next month, and use data collected between 6:00 and 21:00 for prediction tests.
3) Specify the length of the prediction period as number of sessions (e.g., length equals 4 sessions).
4) For each period, predict indoor temperature in 5-min prediction intervals.
5) Calculate error between the predicted and actual temperature at the end of each prediction period for all periods in the experiment days.
6) Repeat the above steps for other months and period lengths.

Fig. 7 illustrates the error at the end of a four-session period.

In this article, periods with session number of 1–8 were investigated, showing the propagation of indoor temperature prediction error through time. Table VI shows the number of prediction cases in each testing month, with different session numbers. The large number of testing cases is expected to unbiasedly reflect the true distribution of prediction errors.

The distribution of the period-end error for all prediction cases in four testing months is visualized in Fig. 8. Different colors represent different session numbers in the prediction period. Because in these experiments, the prediction periods were defined as session length instead of time, therefore, the average duration of prediction periods in hours was calculated to better describe the time-related error. This is shown in Fig. 9. On average, the duration of one session in this article

TABLE VI
NUMBER OF PREDICTION CASES (PERIODS WITH 1–8 SESSIONS)

Session Numbers	Number of Prediction Cases in Each Testing Month			
	June	July	August	September
1	1450	1031	1120	835
2	1405	986	1075	800
3	1360	941	1030	765
4	1315	896	985	730
5	1270	851	940	695
6	1225	806	895	660
7	1180	761	850	625
8	1135	716	805	590

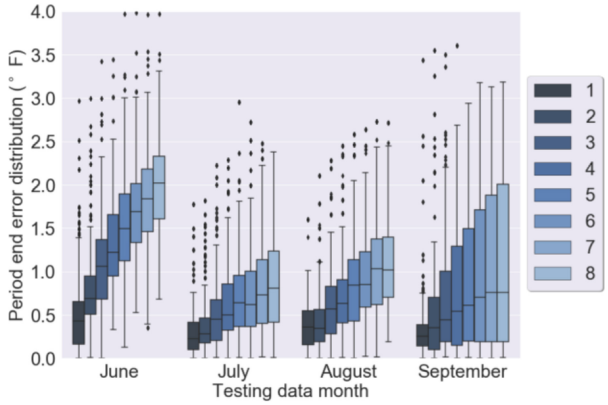

Fig. 8. Period-end error distributions of the four-month testing data. Legend shows the number of sessions in the period.

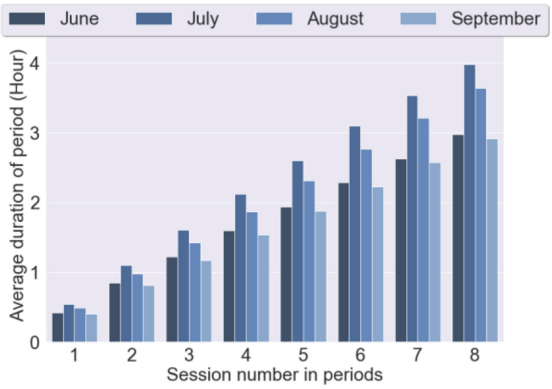

Fig. 9. Average duration of periods.

was around 30 min, and the duration of an eight-session period ranged from 3 to 4 h.

According to Fig. 8, due to the error propagation, the longer prediction period resulted in the larger average error at the end of the prediction period. Another observation was that the error was much more significant when testing using the data in June. This was because June is the transition month from spring to summer, considering the location of the building under test. As shown in Fig. 10, there was a great difference in the daily maximum temperature distribution between May and June. Such a difference resulted in different occupants' behaviors in May as compared to those in early June. One good example is that in comfortable and cool weather days, occupants are more

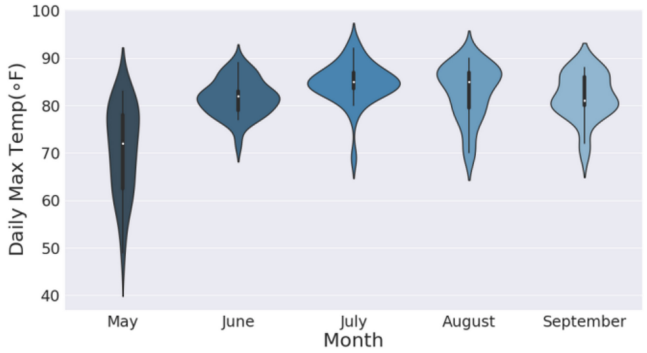

Fig. 10. Monthly distribution of daily maximum temperatures.

TABLE VII
NUMBER OF HYPER-PARAMETER COMBINATIONS IN GRID SEARCH

Models	PR	SVR	XGB	RF	NN
Number	12	72	72	18	60

likely to open windows and thus changing the building thermal envelop. This inevitably influenced the accuracy of the indoor temperature prediction in June.

For other mid-summer months, when the daily weather condition was more stable and similar to each other, the average period-end errors were much smaller. For a one-session prediction, lasting about 20 min to half an hour, the average error was around 0.3 °F. For an eight-session prediction, lasting about 3–4 h, the error median was around or below 1.0 °F. Because of such error scale and considering that human beings are insensitive to 1.0 °F–2.0 °F temperature difference [42], it is appropriate to say that the proposed practical algorithm can produce comparatively reliable forecast results for smart building HVAC control. Considering that most smart building AC control requires a prediction horizon of less than 4 h; the temperature prediction error could be around 1.0 °F or less.

Note that the error quantification discussed in this section was from the experiment conducted in an office, where the day-to-day occupancy and activity levels were comparatively stable. In future work, other types of buildings with more erratic occupancy patterns should be investigated as well.

E. Computational Efficiency

To investigate the computational efficiency of the proposed learning framework, an edge device used was Odroid XU4—a SBC with an octa-core processor and 2-GB RAM; and the cloud infrastructure used was based on Amazon Web Services EC2 instance (c5.large). Ubuntu operating system was used in both platforms.

Table VII shows the number of hyper-parameter combinations that were scanned in the grid search during model selection. Testing results, i.e., computation times for the model selection and training described in Section IV, are summarized in Table VIII. In each case, they are the average values over three separate experiments.

According to Table VIII, on both platforms, the majority of tasks was completed within 360 s (6 min), for this single thermal zone. The EC2 instance computed faster due to its higher computational power. In addition, the computation time also increased with the amount of training data.

TABLE VIII
COMPUTATION TIME ON EDGE DEVICE AND CLOUD INFRASTRUCTURE

Platform	Computation Time (Seconds)			
	May	June	July	August
Odroid	109.52	421.26	287.26	351.65
c5.large	72.96	272.68	199.72	243.15

Based on these results, in a building with 10 thermal zones, the overall process for training thermal models for all zones can take less than or around an hour (10 zones, 6 min each). This can be scheduled accordingly, for example, the EC2 instance can be started when it is cheaper, utilizing spot pricing [43].

It is worth noting that the results above were based on the search space specified in Table VII, with five potential machine learning models. As a result, another way to improve the efficiency is to reduce the search space for hyper-parameters. An algorithm can be designed to shrink the search space based on previous trained results.

VI. CONCLUSION

In this article, an IoT-based plug-and-play building thermal model learning framework was developed, deployed and investigated. This framework was designed to learn building thermal properties directly from building operation historical data, and the learning process was designed to be fully automated without human intervention. Though flexible in integrating data from multiple sensors, the developed framework was demonstrated using the thriftiest case with only smart thermostats as hardware, and the performance of the framework was demonstrated using the most affordable setup.

The case study based on the real-world operational data of an office building revealed that under different environments and AC control strategies, the average error of indoor temperature prediction during the period of 3–4 h was around 1.0 °F. This implied that the learned model could be used to provide reliable thermal comfort evaluation when implementing intelligent control. The developed framework has a demonstrated potential to be integrated with an IoT-based BEM and to provide an accurate thermal model to other intelligent AC control algorithms at no additional hardware cost. Computational efficiency of the developed framework was also investigated. Findings indicated that the learning process on both the edge device and the cloud infrastructure could deliver the learned model in minutes, for a single thermal zone.

Overall, this fully automated learning framework enables avoidance of building-by-building configuration, thus largely accelerating the technology-to-market process and making it possible for more buildings to adopt advanced control. Future work on this topic includes: 1) testing the framework on other types of buildings with more volatile occupancy and operation patterns and 2) expanding the framework to consider multistage cooling/heating systems.

ACKNOWLEDGMENT

Building operation data used in this study were collected in a project supported in part by the U.S. Department of Energy under Contract DE-EE 0006352. The authors would like to thank two researchers from the U.S. National Renewable Energy Laboratory: Dr. Y. Li from the Commercial Buildings Research Group and Dr. D. Biagioni from the Computational Science Center, for their comments. Last but not least, they would like to thank the journal editors and all reviewers for their valuable comments and suggestions.

REFERENCES

[1] *U.S. Energy Information Administration How Much Energy is Consumed in U.S. Residential and Commercial Buildings*. Accessed: Jan. 15, 2019. [Online]. Available: https://www.eia.gov/tools/faqs/faq.php?id=86&t=1

[2] S. Somasundaram *et al.*, "Reference guide for a transaction-based building controls framework," Pac. Northwest Nat. Lab., Richland, WA, USA, Rep. PNNL-23302, 2014.

[3] G. Fortino, W. Russo, C. Savaglio, W. Shen, and M. Zhou, "Agent-oriented cooperative smart objects: From IoT system design to implementation," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 11, pp. 1939–1956, Nov. 2018.

[4] D. Minoli, K. Sohraby, and B. Occhiogrosso, "IoT considerations, requirements, and architectures for smart buildings—Energy optimization and next-generation building management systems," *IEEE Internet Things J.*, vol. 4, no. 1, pp. 269–283, Feb. 2017.

[5] S. Gao, M. Zhou, Y. Wang, J. Cheng, H. Yachi, and J. Wang, "Dendritic neuron model with effective learning algorithms for classification, approximation, and prediction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 2, pp. 601–614, Feb. 2019.

[6] T. Liu, B. Tian, Y. Ai, L. Li, D. Cao, and F.-Y. Wang, "Parallel reinforcement learning: A framework and case study," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 4, pp. 827–835, Jul. 2018.

[7] D. P. Bertsekas, "Feature-based aggregation and deep reinforcement learning: A survey and some new implementations," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 1, pp. 1–31, Jan. 2019.

[8] G. Rawal and S. Raschke, "Costs, savings, and ROI for smart building implementation," Intel Blog, Santa Clara, CA, USA, 2016. Accessed: Mar. 1, 2019. [Online]. Available: http://blogs.intel.com/iot/2016/06/20/costs-savings-roi-smart-building-implementation/

[9] K. Kreuzer. *The Open Home Automation Bus*. Accessed: Jan. 21, 2019. [Online]. Available: www.openhab.org

[10] P. Schoutsen. *Home Assistant*. Accessed: Jan. 21, 2019. [Online]. Available: https://www.home-assistant.io/

[11] X. Zhang, R. Adhikari, M. Pipattanasomporn, M. Kuzlu, and S. Rahman, "Deploying IoT devices to make buildings smart: Performance evaluation and deployment experience," in *Proc. IEEE 3rd World Forum Internet Things (WF-IoT)*, 2016, pp. 530–535.

[12] J. Yang, H. Zou, H. Jiang, and L. Xie, "Device-free occupant activity sensing using WiFi-enabled IoT devices for smart homes," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3991–4002, Oct. 2018.

[13] L. Zimmermann, R. Weigel, and G. Fischer, "Fusion of nonintrusive environmental sensors for occupancy detection in smart homes," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2343–2352, Aug. 2018.

[14] P. Verma and S. K. Sood, "Fog assisted-IoT enabled patient health monitoring in smart homes," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1789–1796, Jun. 2018.

[15] X. Zhang, M. Cai, M. Pipattanasomporn, and S. Rahman, "A power disaggregation approach to identify power-temperature models of HVAC units," in *Proc. IEEE Int. Smart Cities Conf.*, 2018, pp. 1–6.

[16] M. H. Y. Moghaddam and A. Leon-Garcia, "A fog-based Internet of energy architecture for transactive energy management systems," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 1055–1069, Apr. 2018.

[17] W. Li, T. Logenthiran, V.-T. Phan, and W. L. Woo, "Implemented IoT-based self-learning home management system (SHMS) for Singapore," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2212–2219, Jun. 2018.

[18] J. Pan, R. Jain, S. Paul, T. Vu, A. Saifullah, and M. Sha, "An Internet of Things framework for smart energy in buildings: Designs, prototype, and experiments," *IEEE Internet Things J.*, vol. 2, no. 6, pp. 527–537, Dec. 2015.

[19] G. Fortino, C. Savaglio, and M. Zhou, "Toward opportunistic services for the industrial Internet of Things," in *Proc. 13th IEEE Conf. Autom. Sci. Eng. (CASE)*, 2017, pp. 825–830.

[20] M. Cai, M. Pipattanasomporn, and S. Rahman, "Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques," *Appl. Energy*, vol. 236, pp. 1078–1088, Feb. 2019.

[21] E. Mocanu *et al.*, "On-line building energy optimization using deep reinforcement learning," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 3698–3708, Jul. 2019.

[22] W. Zhang, W. Hu, and Y. Wen, "Thermal comfort modeling for smart buildings: A fine-grained deep learning approach," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2540–2549, Apr. 2019.

[23] A. Rajith, S. Soki, and M. Hiroshi, "Real-time optimized HVAC control system on top of an IoT framework," in *Proc. 3rd Int. Conf. Fog Mobile Edge Comput.*, 2018, pp. 181–186.

[24] H. Park and S.-B. Rhee, "IoT-based smart building environment service for occupants' thermal comfort," *J. Sensors*, vol. 2018, p. 10, May 2018.

[25] J. Serra, D. Pubill, A. Antonopoulos, and C. Verikoukis, "Smart HVAC control in IoT: Energy consumption minimization with user comfort constraints," *Sci. World J.*, vol. 2014, p. 11, Jun. 2014.

[26] H. Hui, Y. Ding, W. Liu, Y. Lin, and Y. Song, "Operating reserve evaluation of aggregated air conditioners," *Appl. Energy*, vol. 196, pp. 218–228, Jun. 2017.

[27] X. Xu and S. Wang, "A simplified dynamic model for existing buildings using CTF and thermal network models," *Int. J. Therm. Sci.*, vol. 47, no. 9, pp. 1249–1262, 2008.

[28] X. Q. Li, Y. Chen, J. D. Spitler, and D. Fisher, "Applicability of calculation methods for conduction transfer function of building constructions," *Int. J. Therm. Sci.*, vol. 48, no. 7, pp. 1441–1451, 2009.

[29] D. B. Crawley *et al.*, "EnergyPlus: Creating a new-generation building energy simulation program," *Energy Build.*, vol. 33, no. 4, pp. 319–331, 2001.

[30] L. Laret, "Use of general models with a small number of parameters, Part 1: Theoretical analysis," in *Proc. Conf. Clima*, 1980, pp. 263–276.

[31] M. M. Gouda, S. Danaher, and C. P. Underwood, "Building thermal model reduction using nonlinear constrained optimization," *Build. Environ.*, vol. 37, no. 12, pp. 1255–1265, 2002.

[32] G. Fraisse, C. Viardot, O. Lafabrie, and G. Achard, "Development of a simplified and accurate building model based on electrical analogy," *Energy Build.*, vol. 34, no. 10, pp. 1017–1031, 2002.

[33] E. Á. R. Jara *et al.*, "A new analytical approach for simplified thermal modelling of buildings: Self-adjusting RC-network model," *Energy Build.*, vol. 130, pp. 85–97, Oct. 2016.

[34] G. Mustafaraj, G. Lowry, and J. Chen, "Prediction of room temperature and relative humidity by autoregressive linear and nonlinear neural network models for an open office," *Energy Build.*, vol. 43, no. 6, pp. 1452–1460, 2011.

[35] A. Aliberti *et al.*, "Indoor air-temperature forecast for energy-efficient management in smart buildings," in *Proc. IEEE Int. Conf. Environ. Elect. Eng. IEEE Ind. Commercial Power Syst. Europe (EEEIC/I&CPS Europe)*, 2018, pp. 1–6.

[36] L. Mba, P. Meukam, and A. Kemajou, "Application of artificial neural network for predicting hourly indoor air temperature and relative humidity in modern building in humid region," *Energy Build.*, vol. 121, pp. 32–42, Jun. 2016.

[37] X. Zhang, M. Pipattanasomporn, and S. Rahman, "A self-learning algorithm for coordinated control of rooftop units in small-and medium-sized commercial buildings," *Appl. Energy*, vol. 205, pp. 1034–1049, Nov. 2017.

[38] X. Zhang, *A Data-Driven Approach for Coordinating Air Conditioning Units in Buildings During Demand Response Events*, Virginia Polytech. Inst., State Univ., Blacksburg, VA, USA, 2018.

[39] S. Shao, *An Approach to Demand Response for Alleviating Power System Stress Conditions due to Electric Vehicle Penetration*, Virginia Polytech. Inst., State Univ., Blacksburg, VA, USA, 2011.

[40] S. Shao, M. Pipattanasomporn, and S. Rahman, "Development of physical-based demand response-enabled residential load models," *IEEE Trans. Power Syst.*, vol. 28, no. 2, pp. 607–614, May 2013.

[41] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, NY, USA: Springer, 2009. [Online]. Available: https://link.springer.com/book/10.1007/978-0-387-84858-7#about

[42] *Thermal Environmental Conditions for Human Occupancy, American Society of Heating, Refrigerating and Air-Conditioning Engineers*, ANSI/ASHRAE Standard 55-2013, 2017.

[43] *Amazon EC2 Spot Instances*. Accessed: Feb. 2, 2019. [Online]. Available: https://aws.amazon.com/ec2/spot/

Xiangyu Zhang received the B.S. degree in electrical engineering from Wuhan University, Wuhan, China, in 2012, the M.S. degree in electrical engineering from Tsinghua University, Beijing, China, in 2014, and the Ph.D. degree from the Electrical and Computer Engineering Department, Virginia Tech, Arlington, VA, USA, in 2018, where this research is conducted.

He is currently working with the Computational Science Center, National Renewable Energy Laboratory, Golden, CO, USA.

Manisa Pipattanasomporn (S'01–M'06–SM'11) received the B.S. degree in electrical engineering from Chulalongkorn University, Bangkok, Thailand, in 1999, the M.S. degree in energy economics and planning from the Asian Institute of Technology, Bangkok, in 2001, and the Ph.D. degree in electrical engineering from Virginia Tech, Arlington, VA, USA, in 2004.

She is an Associate Professor of Smart Grid Research Unit, Chulalongkorn University, and an Adjunct Faculty with Advanced Research Institute, Virginia Tech. Her research interests include smart grid, smart building, demand response, machine learning with particular applications on energy savings, building-level load forecasting, and blockchain for peer-to-peer trading of solar electricity.

Tao Chen received the B.S. degree in electrical engineering from Anhui University, Hefei, China, in 2012, the M.S. degree in electrical engineering from the Tampere University of Technology, Tampere, Finland, in 2014, and the Ph.D. degree in electrical engineering from University of Michigan–Dearborn, Dearborn, MI, USA, in 2018.

He is currently works with the Advanced Research Institute, Virginia Tech, Arlington, VA, USA.

Saifur Rahman (S'75–M'78–SM'83–F'98–LF'16) received the Ph.D. degree in electrical engineering from Virginia Tech, Blacksburg, VA, USA.

He is the Founding Director of the Advanced Research Institute, Virginia Tech, Arlington, VA, USA, where he is the Joseph Loring Professor of electrical and computer engineering.

Prof. Rahman is an IEEE Millennium Medal Winner. He was the Founding Editor-in-Chief of *IEEE Electrification Magazine* and the IEEE TRANSACTIONS ON SUSTAINABLE ENERGY. He serves as the President for the IEEE Power and Energy Society from 2018 to 2019 and the Vice President of the IEEE Publications Board in 2006.