A Domain Adaptation Technique for Fine-Grained Occupancy Estimation in Commercial Buildings

Tianyu Zhang
University of Alberta
Edmonton, Alberta, Canada
tzhang6@ualberta.ca

Omid Ardakanian
University of Alberta
Edmonton, Alberta, Canada
ardakanian@ualberta.ca

ABSTRACT

Fine-grained occupancy information is essential to improve human experience and operational efficiency of buildings, yet it is quite challenging to obtain this information due to the lack of special-purpose sensors for occupancy monitoring, and insufficient training data for developing accurate data-driven models. This paper addresses this challenge by (a) utilizing recurrent neural network models to uncover latent occupancy patterns in individual rooms from trend data available through the building management system, and (b) applying a domain adaptation technique to transfer existing occupancy models trained in a controlled environment (i.e., the source domain) to another environment (i.e., the target domain) where labelled data is sparse or non-existent. We adjust the model parameters based on the apparent differences between the two environments and apply the adapted model to estimate the number of occupants in the target domain. Our results from two test commercial buildings in two continents indicate that the adapted model yields only slightly lower accuracy than a model that is originally built on the target domain given a large amount of labelled data. Furthermore, we study how much labelled data is required from the target domain for the semi-supervised domain adaptation technique to achieve promising results.

CCS CONCEPTS

• **Information systems** → *Data mining*; • **Computer systems organization** → *Sensors and actuators.*

KEYWORDS

Smart building, occupancy detection, domain adaptation, recurrent neural networks

1 INTRODUCTION

Buildings host complex Internet of Things (IoT) systems comprised of a multitude of networked sensors and actuators with a variety of different access methods and protocols, such as BACnet, Ethernet, WiFi, ZigBee, and Bluetooth. For example, a typical medium-sized commercial building contains hundreds of programmable thermostats, carbon dioxide sensors, power meters, and motion sensors that collect large volumes of data and serve different purposes in Heating Ventilation and Air Conditioning (HVAC), lighting, electrical, and security subsystems of the building. The sensor data is transferred to a Building Management System (BMS) that archives this data, monitors the building subsystems, controls indoor climate within a specified range, and alarms building managers in the event of a fault or a malfunction. In addition to the energy management and comfort functions of traditional BMS, several smart building applications rely on the trend data available through this system to improve human experience, operational efficiency, and sustainability of the built environment [2, 38].

While multiple systems have been developed in recent years to enable interoperability and portability of building applications [8, 15, 36], even well-established analytics and control applications do not get deployed at scale today. This is primarily because most applications rely on models (for occupancy, comfort, heat transfer, etc.) that are developed and validated in a small number of test buildings and do not work well in other buildings. Thus, to achieve scalable deployments of building applications across the building stock, it is necessary to automatically update or adapt these models to the new buildings.

This paper focuses on developing and adapting high-accuracy models for estimating the number of occupants in every room in a commercial building equipped with a BMS. Occupancy is one of the main factors determining the energy use in buildings. Recent studies suggest that incorporating fine-grained occupancy information (i.e., presence and count) in building controls can significantly improve human experience and operational efficiency of HVAC [1, 3], and lighting systems [33]. Nevertheless, it is quite challenging to develop accurate occupancy models in a building due to (a) the lack of special-purpose sensors, such as cameras, for occupancy monitoring, (b) the highly uncertain and complex nature of occupancy dynamics in the built environment, and (c) the lack of sufficient amounts of labelled occupancy data.

There are two main approaches to determining occupant presence and count in the built environment. The first approach builds high-dimensional thermal transfer models (such as the resistive-capacitative model [37]) and use them to infer the internal heat gain in individual rooms and attribute them subsequently to the occupants and appliances [35, 43]. These models must be customized for each room based on its size, layout, and the building envelop; hence, they cannot be simply used to distinguish the effect of occupancy from other confounding effects. The second approach builds data-driven models to determine occupant presence and count in the many rooms in a building [3, 5, 6, 23]. These models are easier to build and can substitute complex physics-based models with an insignificant loss of prediction accuracy [44]. However, developing such models requires an extended training phase and a significant amount of labelled occupancy data, which is not normally available for different rooms and buildings.

Our approach is to build data-driven occupancy models using the trend data, e.g., measurements of carbon dioxide and damper position sensors, which are readily available through the BMS in most commercial buildings. Since training these models requires an abundance of labelled occupancy data, we investigate the use of a semi-supervised domain adaptation technique to transfer occupancy models that are built in a controlled environment where sufficient labelled data is available to an environment where little or no labelled data is available. We study this problem when the source and target domains are in the same building. The contribution of this paper is threefold:

- We develop accurate data-driven models for occupancy estimation when sufficient ground truth data is available. Training these models requires only HVAC sensor data and weather data, and they are capable of describing the complex, nonlinear relationship between historical sensor data and the number of occupants, and the temporal dependency of the occupancy data.
- We adopt semi-supervised and unsupervised domain adaptation techniques to reduce the amount of ground truth data required for developing a well-suited model.
- We evaluate the efficacy of these models in different settings and study how the prediction accuracy would change with the amount of available ground truth data.

The rest of this paper is organized as follows. Section 2 describes related work on occupancy modelling and existing domain adaptation applications. Section 3 explains how occupancy information can improve building operations, shows the feasibility of estimating the room-level occupant count from the available trend data, and provides a high level overview of the two recurrent neural network models used in this work. Section 4 describes our data sets and defines the proposed methodology for training and adapting the occupancy models. Section 5 explains the evaluation results, and Section 6 presents discussion points and suggests directions for future work. The paper is concluded in Section 7.

2 RELATED WORK

2.1 Monitoring Occupant Presence and Count

Several attempts have been made to date to monitor building occupancy using wired and wireless sensor networks. For example, motion sensor data is fused with data from magnetic reed switches [1], thermal sensor arrays [9], carbon-dioxide sensors [24, 35], cameras [16], and other ambient sensors [29] to estimate the room-level occupancy state (viz. occupant presence or count). In recent work, a custom sensor tag has been designed which integrates and fuses a large number of sensing modalities [28]. This general-purpose sensor enables monitoring occupancy along with several other environmental facets. Nevertheless, all these systems require hardware retrofits, posing many challenges from sensor placement and calibration to ensuring that the sensors have a reliable network connection and power supply. Moreover, some of them utilize sensors that are recognized for being privacy invasive (e.g., cameras and microphones) and carry a heavy deployment stigma.

To address these limitations, a growing body of research focuses on the opportunistic use of existing building infrastructure to monitor occupancy, when possible. For example, the wireless networking infrastructure is leveraged in [7, 13, 34, 40, 46] to estimate the occupancy state of different spaces in a building. Similarly, building occupancy is inferred in [18] leveraging the wireless networking infrastructure, and security and access control systems. More recently, measurements of the room temperature, and damper and valve position, which are parts of the HVAC system, are used to infer the occupancy state of individual rooms [3, 19]. These techniques have two major shortcomings. First, they achieve an acceptable level of accuracy only if physical sensors are installed in suitable locations in the building. Second, modelling occupancy using these techniques requires an abundance of labelled occupancy data collected from the same environment. It is quite challenging to obtain labelled data as it requires substantial manual effort or reliable video-based systems which are costly and intrusive.

2.2 Domain Adaptation

Domain adaptation seeks to learn from one or multiple *source domains* a model that performs well on a related *target domain*. It is assumed that the source and target domains are associated with the same label space. Domain adaptation has been used extensively in computer vision and natural language processing [42]. The early applications of domain adaptation can be traced back to 1990s [11]. In recent years, domain adaptation has been applied to the image translation problem. Reference [45] develops two conditional Generative Adversarial Networks (GANs), one to translate an image from the source domain to the target domain ($Y \leftarrow f(X)$), and another one to translate it from the target domain to the source domain ($X \leftarrow g(Y)$). The two networks are trained to minimize the difference between X and $g(f(X))$, enabling translations such as a zebra to a horse or a photo to a Monet painting. Reference [25] transfers both texture and geometrical properties of an image, enabling them to successfully transform a chair to a car or a vehicle to a human face. Reference [17] employs a domain adaptation technique to build a Convolutional Neural Network (CNN) model for image recognition on a large set of car images drawn from e-commerce websites and Google Street View. In a different line of work, Reference [14] uses domain adaptation to find the common embedding between two languages to perform an accurate translation.

Despite the extensive literature on domain adaptation, little work has been done to investigate whether it can be applied to data collected by IoT devices which are possibly deployed in different environments. To our knowledge, Reference [5] is the only paper that utilizes domain adaptation to determine the number of occupants in a room by using carbon-dioxide measurements from this room and a large cinema. It proposes a human occupancy counter which employs an accurate occupancy model trained with minimum labelled data. This model is developed in a small room and then adapted to a larger room (a cinema with a seating capacity of 279 people). The authors develop a seasonal decomposition model which captures the nonlinear relationship between carbon dioxide and occupant count. This model has four components which are trained in the source domain and then adapted to the target domain. They show that it is possible to achieve a higher accuracy with the adapted model on the target domain. The accuracy they report is around 90% for binary occupancy detection and 60% for estimating the number of occupants.

Our work is similar to [5] in that we also leverage semi-supervised domain adaptation to estimate the number of occupants with minimum labelled data from the target domain. However, their approach has several shortcomings. First, they only consider one feature, which is the carbon dioxide concentration, and build a

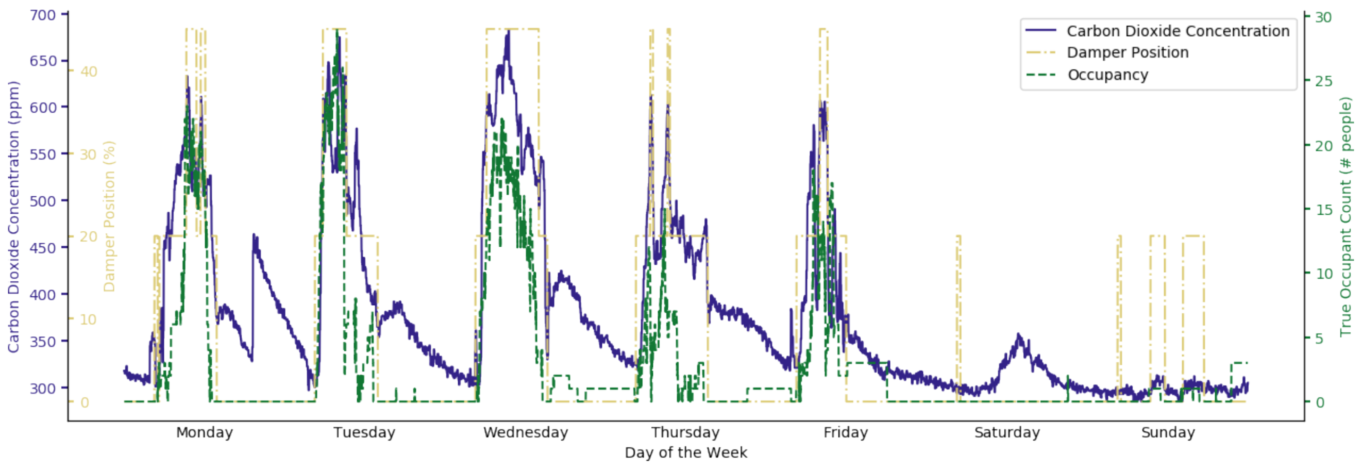

Figure 1: Ground truth occupancy data and measurements of carbon dioxide and damper position over one week in a room in Building A.

model that only works if this feature is available; hence, it cannot be extended to other occupancy-indicating features. The carbon dioxide sensor is not always available in the HVAC system, and it detects occupancy events after a certain delay since carbon dioxide builds up slowly. In our test, the carbon dioxide sensor takes about 15 minutes to sense any change in the carbon dioxide concentration level after an occupancy event. Second, they only study the case where the source and target domains have the same feature space. Third, the accuracy of their model is low when it comes to determining the number of occupants, especially given that they consider an occupancy estimation as accurate if the absolute differences between predicted and real occupant counts are less than 5 people. We address these shortcomings in this work, build recurrent neural network models that are general enough to be used with an arbitrary set of features, and evaluate our algorithm in two commercial buildings with different features located in two countries.

3 BACKGROUND

3.1 The Importance of Monitoring Occupancy in Buildings

Occupancy is one of the main factors determining the building energy consumption. Despite huge variation in occupancy over space and time, the whole building is often treated as a uniform environment controlled with a fixed ventilation rate and static schedules for air conditioning and lighting, thereby wasting a lot of energy in conditioning and lighting empty or partially occupied spaces. Thus, controlling HVAC and lighting systems based on occupancy can result in substantial energy savings and tangible improvements in occupant comfort.

While most existing building applications rely on the information about the binary occupancy state (i.e., occupied or vacant) of each room, discerning the number of occupants in each room enables even more applications, examples of which are smart lighting, demand-controlled filtration and ventilation [38], space utilization, safety and evacuation [27]. The more accurately the number of occupants is estimated, the better it would be for the application.

3.2 HVAC System and Trend Data

An HVAC system typically consists of one or more Air Handling Units (AHUs), which supply cool air through ducts to Variable Air Volume (VAV) systems, each controlling a thermal zone. If a zone requires cooling to balance the heat gained from occupants, appliances, and external sources, the VAV unit opens its dampers to the required extent to allow cooler air to flow into the zone. Conversely, if a zone requires heating to maintain its operating point, the VAV unit opens the radiator or reheat valve.

The heating or cooling action of a VAV unit is determined by a control system which keeps the zone temperature around its setpoint, while maintaining the required minimum airflow. The VAV control system monitors the zone temperature, and actuates the dampers and valves. The BMS typically logs the instantaneous values of the sensors and states of the actuators.

3.3 Feasibility of Estimating Occupancy using Trend Data

Figure 1 shows the ground truth occupancy (i.e. the number of people in the room), the carbon dioxide concentration level, and the damper position over one week in a room in our test building (as described in Section 4.1). It can be readily seen that the number of occupants is large when the carbon dioxide concentration peaks and the damper becomes half open. Furthermore, it can be seen that the carbon dioxide concentration is low and the damper is closed most of the time during the weekend when the room is generally unoccupied. This implies that these sensors are occupancy-indicating and it is possible to uncover the latent occupancy pattern of a room using only trend data available through the BMS and sophisticated machine learning algorithms.

3.4 Domain-Adaptive Occupancy Models

The basic idea of domain adaptation is to find a common representation between multiple domains to make them have similar distributions. The closer the two distributions are, the better the model would perform on the target domain [10]. We define a *domain* to be a certain space in the built environment that we are interested in determining its occupancy state. It may be a single room, a floor, or the whole building. We define *features* as sensing modalities that we use for occupancy estimation, and *labels* as ground truth occupancy data in the source domain or target domain.

There are three classes of domain adaptation: supervised, semi-supervised, and unsupervised. In supervised domain adaptation, it is assumed that all data points are labelled in the source and target domains. Semi-supervised domain adaptation is similar to supervised domain adaptation with a small difference, that is only a small fraction of data points are labelled in the target domain. In unsupervised domain adaptation, we assume that only a fraction of data points in the source domain are labelled, and the target domain does not have any labelled data. We compare the accuracy of models built using these three techniques in this paper.

3.5 Recurrent Neural Networks

Neural networks are computational models inspired by biological neural networks. They have proven to be useful in extracting patterns and detecting trends in high-dimensional data sets which cannot be analyzed with other computational models.

There are many variants of neural networks and recurrent neural networks (RNN) are one of the most popular models to deal with timeseries data (i.e., sequences of inputs). In particular, an RNN maintains a state to capture the history of the input sequence, enabling it to learn complex temporal dependencies. In the last few years, there has been an incredible success in applying RNNs to a variety of problems, such as speech recognition [20], language modeling [31], translation [12], etc.

Long short term memory networks and nonlinear autoregressive exogenous models are two types of RNNs we adopt in this paper. Both networks are designed to memorize historical data and identify trends. We explain them below.

3.5.1 Long Short Term Memory Networks.

Long Short Term Memory (LSTM) networks were first introduced in [22] and have been widely applied since then to learn long-term dependencies in data. As depicted in Figure 2, an LSTM cell is comprised of three gates: an input gate, an output gate, and a forget gate. Let us denote input and output vectors at time t by x_t and y_t, respectively, the cell state vector at time t by C_t, and weights and biases of the gates by W_f, W_i, W_o, W_C, b_f, b_i, b_o, and b_C, respectively.

The first (leftmost) gate determines how much the input would affect the state, or equivalently, how much of the history would be forgotten given the current input:

$$f_t = \sigma(W_f \cdot [y_{t-1}, x_t] + b_f)$$

Here σ is the logistic sigmoid function which returns a value between 0 and 1.

Next, the Sigmoid gate indicates which input (i_t) would update the state at time t, and the hyperbolic tangent function, denoted by *tanh*, creates the candidate values that could be added to the state.

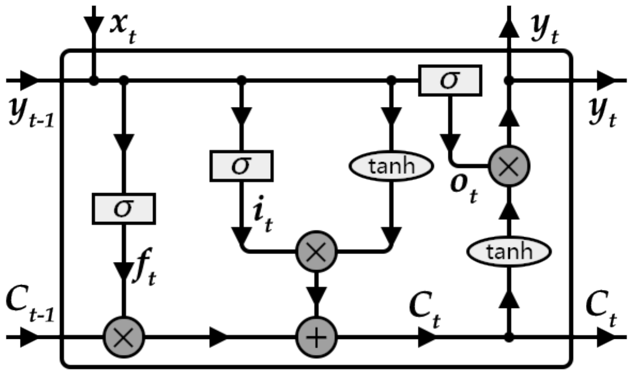

Figure 2: The internal structure of a single LSTM cell. Arrows indicate the direction of data flow.

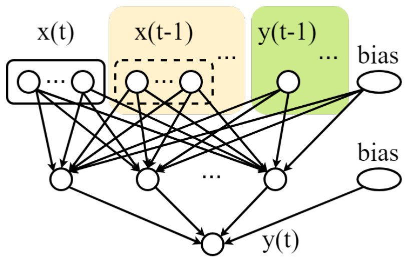

Figure 3: Schematic of an NARX network.

After these two gates the state is updated as follows:

$$i_t = \sigma(W_i \cdot [y_{t-1}, x_t] + b_i)$$
$$C_t = f_t \times C_{t-1} + i_t \times tanh(W_C \cdot [y_{t-1}, x_t] + b_C)$$

The last (rightmost) gate in the LSTM module computes the final output. The output is picked from the current state (passed through a *tanh*) using a Sigmoid gate to evaluate which value it should present in the output vector. The final output vector y_t can be computed as follows:

$$o_t = \sigma(W_o \cdot [y_{t-1}, x_t] + b_o)$$
$$y_t = o_t \times tanh(C_t)$$

The current output, y_t, is an input to the next LSTM module. We use the backpropagation-through-time algorithm to train this model.

Our LSTM network has one hidden layer which contains 64 cells. The output layer contains one output node representing the occupancy state and the input layer contains as many nodes as the number of features in our data set. The cost is computed using the softmax cross entropy between logits and labels. We partition data into multiple batches, where the batch size is equal to one day (from 12:00am to 11:59pm). For each batch, we minimize the cost using a first-order gradient-based optimization method [26].

3.5.2 Nonlinear Autoregressive Exogenous Network.

The nonlinear autoregressive exogenous network (NARX) is a nonlinear autoregressive timeseries model with one or more exogenous inputs influencing the output. It is a powerful model when it comes to discovering long-term dependencies. We use the

Table 1: Description of Building A

Room type	Study area		Lecture room	
Room number	**1**	**2**	**3**	**4**
Seating capacity	36	36	85	85
Var. occupancy	21.7	20.5	67.7	231.9
PAR occupancy	13.7	10.4	10.6	9.0
Min. CO_2 level (ppm)	256	268	370.88	304
Max. CO_2 level (ppm)	907.52	688	844.8	1384
Area (m^2)	125	125	139	139
Max. air flow (m^3/h)	3000	3000	4800	4800

Table 2: Description of Building B

Room type	Meeting room		
Room number	**A**	**B**	**C**
Seating capacity	10	8	10
Var. occupancy count	3.35	1.25	2.27
PAR occupancy count	107.1	89.3	95.1
Min. temperature (°C)	20.10	20.81	20.74
Max. temperature (°C)	27.76	25.97	28.39
Relative area	1.2	1	1.2
Number of windows	2	0	2
Max. air flow (L/s)	130	85	130
Min. air flow (L/s)	65	6	65

backpropagation-through-time algorithm to compute the gradient for calculating the weights.

The input to this model is the current and d past input values $(x_t, x_{t-1}, \cdots, x_{t-d})$, along with d past output values $(y_{t-1}, \cdots, y_{t-d})$, simulating the memory cell. Feeding previous outputs to the input layer allows for storing historical data and helps with learning the long-term dependencies. Figure 3 depicts the structure of an NARX network.

The state space representation of the NARX model can be written as [30]:

$$C_i(t+1) = \begin{cases} \Phi(u(t), C_i(t)), & i = 1 \\ C_i(t), & i = 2, \cdots, N \end{cases}$$

where $\Phi(\cdot)$ represents the nonlinear mapping of the neural network, $u(t)$ represents the past inputs at time t, N represents the total number of neurons in the model, and $C_i, i = 1, \cdots, N$ represent state variables of the recurrent neural network. The output at time t can be computed by $y(t) = C_1(t)$.

We use an NARX network with one hidden layer containing 40 nodes. We use the dynamic backpropagation algorithm to optimize the cost. The model is similar to a simple feed-forward neural network [32].

4 METHODOLOGY

Our goal is to transfer an existing model built for a room with sufficient training data to another room in the same building, for which training data is sparse or non-existent. Our hypothesis is that the adapted model is more accurate than a model that is originally built on the target domain using limited training data that is available. In the following, we describe our data set and then present our domain-adaptive occupancy models.

4.1 Data Set

Our data set is comprised of two commercial buildings located in two countries, which are referred to as Building A and Building B, respectively. Both buildings are equipped with a BMS system capable of logging, trending, and reporting. Building A is a $8,500m^2$ campus building at the University of Southern Denmark [4] with an average of 1,000 occupants on normal weekdays. The building contains graduate student and faculty offices, lecture rooms, and study areas. We consider four rooms in this building to develop and validate the data-driven occupancy models. Each rooms is equipped with high-precision people counting cameras mounted over the two entrances to record the number of occupants, which is treated as ground truth occupancy data. Two of those rooms, Room 1 and Room 2, are $125m^2$ study areas with the seating capacity of 36 people. The other two rooms, Room 3 and Room 4, are $139m^2$ lecture rooms with the seating capacity of 85 people. The HVAC system in this building supplies a maximum of $3000m^3/h$ and $4800m^3/h$ fresh air into the study areas and lecture rooms, respectively. Table 1 summarizes the information about these rooms, including the variance and peak-to-average ratio (PAR) of the number of occupants during the period that data was collected.

Each room constitutes a thermal zone and is controlled by a separate VAV system. The VAV system uses two sensors, i.e., carbon dioxide and damper position sensors, in each room to control the indoor climate. The carbon dioxide sensor samples the carbon dioxide concentration level in parts per million (ppm), and the damper position sensor measures the damper openness in percentage of fully opened. Both quantities are sampled at one minute intervals and measurements are archived by the BMS system. The measurements are obtained between March 21st, 2017 and April 6th, 2017 through the API of the BMS system. The ground truth data was also available for the whole period.

Building B is a four-story office building owned and operated by PCL Constructors in Edmonton, AB, Canada. It contains $44,000ft^2$ of office space, workstations, and meeting rooms. This building does not have a vision-based system for collecting ground truth occupancy data. Thus, we only consider three meeting rooms for which we could extract the number of attendees, time, and duration of meetings from their calendars. Admittedly, the obtained ground truth data is not reliable, because people may enter the room before the meeting starts and may leave before it ends. Moreover, the real number of attendees may be different from the number of people accepted the calendar invitation. In any case, this data shows the overall occupancy trend in each meeting room. Table 2 summarizes the information about these rooms. Each meeting room constitutes a thermal zone and is controlled by a separate VAV system. Each VAV system has several sensors in each room to control the indoor climate. We obtained temperature, airflow, pressure, and damper position data sampled by these sensors at 10-minute intervals. We obtained measurements between December 18th, 2016 and December 18th, 2017 through the BMS system.

Room A and Room C are corner meeting rooms on two consecutive floors of the building with the exact same size and layout; they have floor-to-ceiling glass on the north and east sides, and 4 diffusers. Room B is an interior room with no window and is on the same floor as Room A, but has a different layout. It has 2 diffusers. All meeting rooms have a rectangular conference table with chairs arranged around it on all four sides. The VAV systems of these three rooms are connected to the same AHU.

In addition to the trend data, we extend our feature space by acquiring cloud cover and outdoor temperature for the cities where Building A and Building B are located using the Dark Sky API [39]. In each case, the weather and climate data are available every one minute during the intervals that sensor data was collected. We consider these features when developing a model for perimeter rooms that have at least one window. This is because the outdoor temperature and cloud cover (as a proxy for solar irradiance) can affect the heat load in the room, thereby causing the HVAC system to respond, for example by opening or closing dampers. It is important to make sure that these effects are not confused with the heat gain due to occupants.

4.2 Preprocessing

Our data set contains noisy and missing values and must be cleaned before it can be used to build occupancy models. We specifically identify and remove redundant time value pairs, resample all features at the same frequency, and impute missing values using a simple linear interpolation of neighboring, non-missing values. Once the data are cleaned, we combine all features for each time slot and store them in an array data structure.

4.3 Data-Driven Occupancy Models

We develop four data-driven models to predict the number of occupants in the source domain where sufficient ground truth data (labelled occupancy data) is available. To this end, we train LSTM and NARX networks introduced in Section 3.5. These nonlinear models that have memory, are suitable for uncovering latent occupancy patterns. We also adopt Support Vector Regression (SVR) and Logistic Regression (LR) to estimate the number of occupants. These models are used to evaluate our proposed recurrent neural network models.

To understand how these models would generalize to an independent data set, we use 5-fold cross-validation. In particular, we split data from the source domain into five equal sized segments, and use four of them (80% of data) to train the model, and the rest (20% of data) to test it. This process is repeated five times with different segments selected for testing. We compute the Root-Mean-Square Error (RMSE) to score the models and set model parameters to the ones that had the smallest RMSE. In addition, we calculate the normalized RMSE (nRMSE) which is the ratio of the RMSE to the range of occupant counts in the corresponding room. These metrics are defined as follows:

$$RMSE = \sqrt{\sum_{t \in T} (\hat{y}_t - y_t)^2},$$

$$nRMSE = \frac{\sqrt{\sum_{t \in T} (\hat{y}_t - y_t)^2}}{\max y - \min y},$$

where y denotes the true number of occupants in a given room and $\hat{y}$ denotes the predicted number of occupant in that room. We note that we do not use Mean Absolute Percentage Error (MAPE) as an evaluation metric in this case since it not defined when the room is not occupied (it results in division by zero). Instead, we use nRMSE which also gives a relative sense of how accurate the occupancy estimations are.

4.4 Semi-Supervised Domain Adaptation Technique

We now present a domain adaptation technique to build a model for estimating the number of occupants in a given room. We assume that the true number of occupants is known in the source domain for the entire duration that trend data is available. Hence, we use all trend data and corresponding occupancy labels for model training. Domain adaptation transfers this model to the target domain where ground truth data is limited or non-existent. The adapted model is then used for classification or regression in the target domain.

We leverage both semi-supervised and unsupervised domain adaptation techniques. Both techniques update the model trained in the source domain based on the differences between the source and target domains. The difference is that the unsupervised domain adaptation technique applies the model trained in the source domain to estimate the number of occupants in the target domain without updating the model using labelled data from the target domain, whereas the semi-supervised domain adaptation technique utilizes the available labelled data from the target domain to retrain the model. This retraining phase is key especially when the source and target domains have different feature spaces.

Figure 4 shows the overall framework for developing occupancy models. Since sufficient ground truth data is available in the source domain, it is possible to develop a well-suited model using supervised learning. Turning our attention to the target domain, one can develop a model using supervised learning but since labelled data is sparse, if not non-existent, the accuracy of such a model will not be high. Alternatively, it is possible to apply the supervised learning model built on the source domain to estimate the number of occupants in the target domain. We refer to this approach as unsupervised domain adaptation. Lastly, it is also possible to use a semi-supervised domain adaptation technique to adapt the model trained in the source domain to the target domain. This is a two-step process which starts with re-weighting the model and then retraining it using a small amount of labelled data that is otherwise insufficient for training an accurate model. We explain these two steps below.

4.4.1 Re-weighting: The first step in our domain adaptation technique is called re-weighting. The basic idea is to compute transform matrices A_w and A_b, extract the weights w and biases b from the

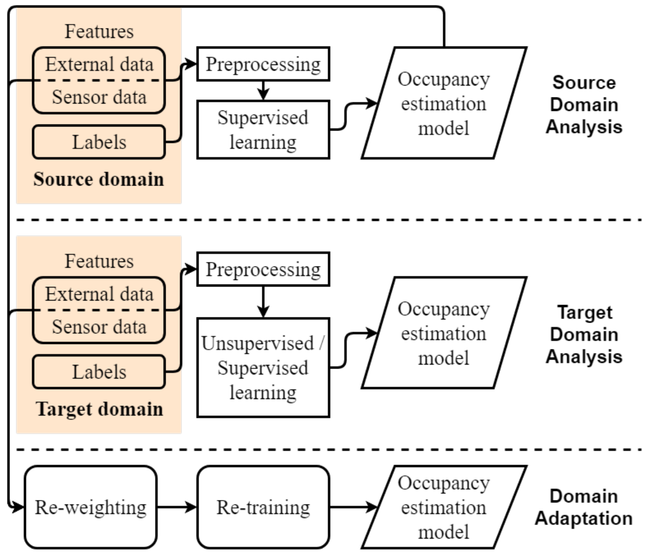

Figure 4: Data collection and analysis framework

model, and finally use $A_w w$ and $A_b b$ as the new weights and biases, respectively. When the target domain is identical to the source domain, the transform matrices, A_w and A_b, are identity matrices. Otherwise, we need to construct A_w and A_b based on the apparent differences between the two domains, including size, seating capacity, and maximum air flow (or ventilation power).

The damper position determines how much fresh air can enter the room. Hence, the higher the maximum air flow is, the larger the effect of the damper position is on the amount of supplied air. Thus, the weight corresponding to the damper position in the input node must be updated based on the ratio of the maximum air flow in the target domain to the maximum air flow in the source domain:

$$w_{damper\_in} \leftarrow w_{damper\_in} \cdot \frac{\text{Max. airflow}_{\text{target}}}{\text{Max. airflow}_{\text{source}}}$$

We only change the weight of the node corresponding to the damper position. The weights of all other nodes remain the same.

As the size of the room and the maximum air flow change, the weights of all gates that are related to the carbon dioxide concentration level need to be adjusted. The unit of the air flow is m^3/h which is the volume of fresh air that enters the room per hour. Suppose all rooms within a building have the same height. We have;

$$\text{Room's Volume} = \text{Height} \times \text{Floor Area},$$

$$\text{Time} = \frac{\text{Room Volume}}{\text{Air flow}},$$

where *Time* is approximately the amount of time that it takes to fill the room with fresh air. We use this value to estimate the carbon dioxide diffusion rate. When it is smaller, it implies that carbon dioxide diffuses faster, hence the carbon dioxide sensor would measure it faster. In this case, we must increase the weight of the carbon dioxide feature. But unlike the damper position, which varies between 0% and 100%, the carbon dioxide level does not have a fixed upper bound. This means that we should not change the input weight of the carbon dioxide sensor. Instead, we increase the weight in each

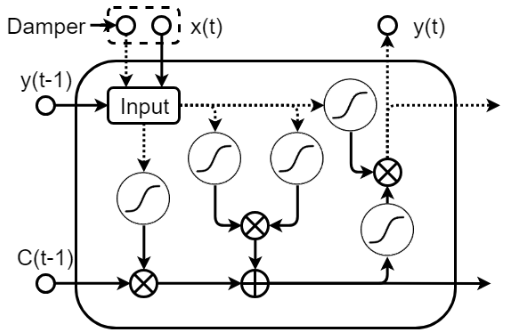

Figure 5: An illustration of the domain adaptation process. Dotted lines represent the weights that are updated.

gate that corresponds to the carbon dioxide feature. In particular, w_{co_2} must be updated as follows:

$$\begin{aligned} w_{co_2} &\leftarrow w_{co_2} \cdot \frac{\text{Time}_{\text{source}}}{\text{Time}_{\text{target}}} \\ &\leftarrow w_{co_2} \cdot \frac{\text{Height} \cdot \text{Area}_{\text{source}}/\text{Max. airflow}_{\text{source}}}{\text{Height} \cdot \text{Area}_{\text{target}}/\text{Max. airflow}_{\text{target}}} \\ &\leftarrow w_{co_2} \cdot \frac{\text{Area}_{\text{source}} \cdot \text{Max. airflow}_{\text{target}}}{\text{Area}_{\text{target}} \cdot \text{Max. airflow}_{\text{source}}} \end{aligned}$$

Moreover, we need to adjust the bias terms, b, when we adjust the weights. We do this based on the differences in the seating capacity of source and target domains because the HVAC system is designed to condition each room based on its maximum occupancy. We have:

$$b \leftarrow b \cdot \frac{Capacity_{target}}{Capacity_{source}}.$$

Figure 5 shows the weights of an RNN cell that we update in the re-weighting progress; these weights are represented with dotted lines.

4.4.2 Retraining: After re-weighting the model, we train it again using the available labelled data from the target domain to update the weights. We choose this training data from a contiguous time period so as to preserve the temporal dependency in the occupancy data. We use the same algorithm used to train the neural network model to retrain it, the only difference being that the weights are initialized to the weights that are calculated in the re-weighting step.

4.5 Post-processing

To prevent error propagation over time, we correct obvious erroneous predictions in the post-processing step. This includes a negative or an unreasonably large number of occupants, and drastic changes in the number of occupants. We say that the predicted number of occupants is 'unreasonably large' when it is greater than the seating capacity of the room plus Δ. We use a non-negative value for Δ to account for the cases that the number of occupants is temporarily above the seating capacity of the room (e.g., between two successive classes when some students are entering the room while others are leaving it). We refer to changes in the occupant

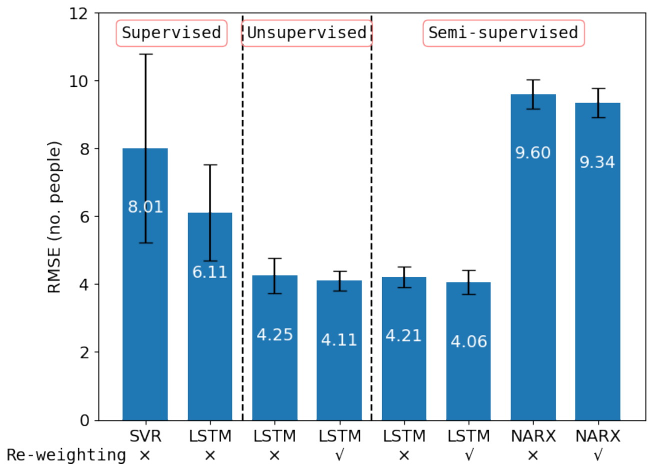

Figure 6: Comparing the accuracy of different supervised, semi-supervised, and unsupervised learning algorithms. Error bars represent the 97% confidence interval.

count as 'drastic' when the occupancy count increases or decreases by more than K occupants in an interval of length T. We set the values of Δ, K, and T of a given room based on its type, size, and function. In our experiments, we set Δ to 5. For study areas, we set K to 5 occupants and T to 4 minutes. For lecture rooms, we set K to 10 occupants and T to 1 minute.

To correct these errors, we replace negative occupant counts by zero, and unreasonably large occupant counts by the seating capacity of the room plus Δ. Moreover, drastic changes in the occupant count are replaced by K.

5 RESULTS

In this section we corroborate the efficacy of the proposed semi-supervised domain-adaptive models by comparing them with two supervised learning models developed for occupancy estimation. We run each model 10 times for a given parameter setting, compute the mean and standard derivation of its prediction accuracy, and report the 97% confidence interval. The results presented in this section belong to Building A unless otherwise stated.

5.1 Evaluating the Model Trained using Semi-Supervised Domain Adaptation

Figure 6 shows the accuracy of estimating the number of occupants using different techniques and models. This figure is divided into three parts to group supervised learning models, unsupervised domain adaptation models, and semi-supervised domain adaptation models. The labels below the x-axis show whether re-weighting is carried out for semi-supervised and unsupervised domain adaptation techniques. We choose the best parameters for each model. We assume that labelled data is available only for 1 hour from the target domain. The target domain is Room 1 and Room 2 and the source domain is Room 3 and Room 4 in Building A. The supervised learning models are trained using 1 hour of labelled data from the target domain only, ignoring labelled data from the source domain.

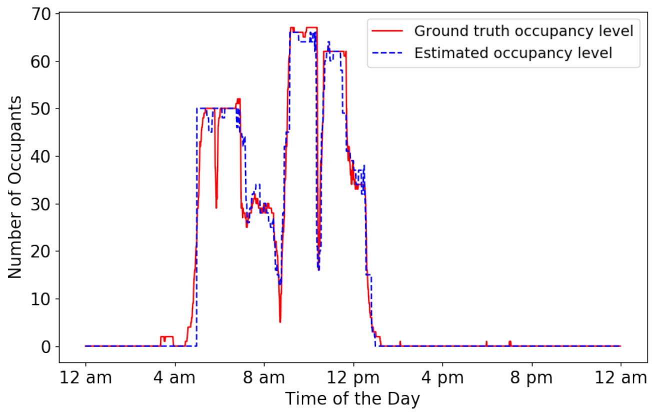

Figure 7: Estimated and true occupancy levels of a lecture room in Building A over one day.

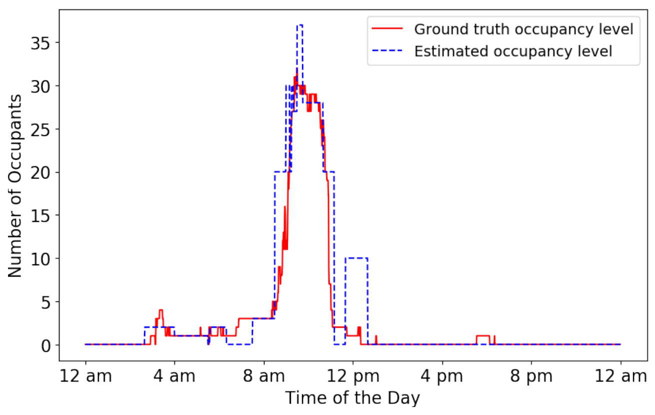

Figure 8: Estimated and true occupancy levels of a study area in Building A over one day.

It can be readily seen that the supervised learning models have the lowest accuracy. For example, the supervised LSTM model has an RMSE of 6.1094 (nRMSE of 17%), which is 43.76% higher than that of the LSTM model trained with the semi-supervised domain adaptation method. Moreover, the LSTM model has a significantly higher accuracy than the NARX model. The LSTM model trained with the unsupervised domain adaptation method on the target domain without re-weighting has the worst performance among all LSTM models built using domain adaptation. The LSTM model trained by the semi-supervised domain adaptation technique with re-weighting has the highest accuracy. In particular, it achieves an RMSE of 4.0599 (nRMSE of 11%) which is 1.23% lower than that of unsupervised domain adaptation with re-weighting and 3.69% better than that of semi-supervised domain adaptation without re-weighting.

Figure 7 and Figure 8 depict the true and estimated occupancy counts in a lecture room and a study area in Building A, respectively. The lecture room reached its maximum occupancy during the day shown in Figure 7. The prediction is obtained from the LSTM model trained using the semi-supervised domain adaptation technique

Table 3: Comparing the performance of supervised learning models trained using labelled data only from the target domain in terms of their average and standard deviation of RMSE, and nRMSE.

	SVR			LSTM			Logistic Regression		
Training data	RMSE	nRMSE	STD.DEV	RMSE	nRMSE	STD.DEV	RMSE	nRMSE	STD.DEV
1 hour	8.01	0.22	2.79	***6.11***	***0.17***	1.41	\	\	\
3 hours	***5.88***	***0.16***	1.33	5.99	0.17	1.23	\	\	\
1 day	***4.39***	***0.12***	0.68	4.67	0.13	0.41	4.89	0.14	0.36

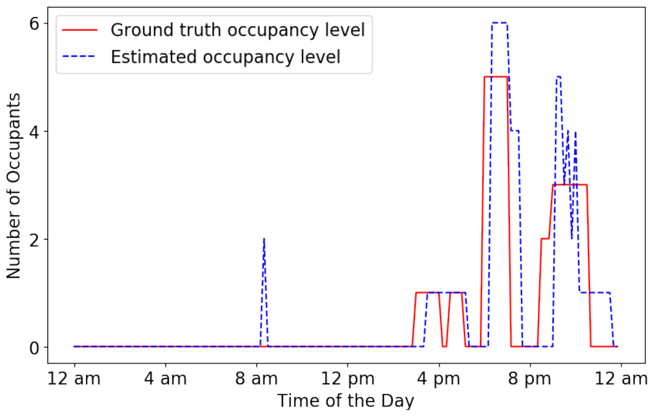

Figure 9: Estimated and true occupancy levels of a meeting room in Building B over one day.

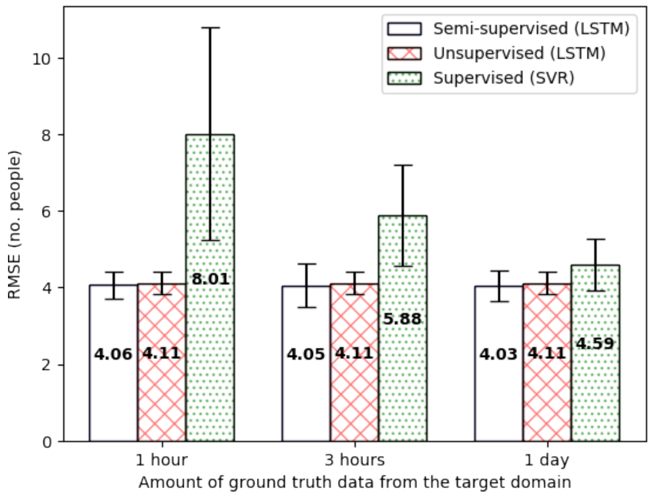

Figure 10: The impact of increasing the amount of available ground truth data on the accuracy of different approaches. Error bars represent the 97% confidence interval. The target domain includes Room 1 and Room 2 in Building A.

with re-weighting and retraining. We specifically use 3 hours of labelled data in the target domain to retrain the model. The data plotted in these figures is not part of the data used for retraining. Both figures suggest that the adapted model estimates the number of occupants accurately and successfully detects the overall occupancy pattern of the room. This confirms our main thesis that the number of occupants can be accurately estimated from trend data available through a rudimentary BMS.

Figure 9 shows the true and estimated occupancy of a meeting room in Building B during one day, where the semi-supervised domain-adaptive LSTM model is used for occupancy estimation. Specifically, we choose Room A as the source domain and Room B which has a different size and layout as the target domain, and assume that labelled data is available for one day in the target domain. The RMSE and nRMSE of this model are 1.87 and 18%, respectively, suggesting that the model successfully detects the overall occupancy pattern although its accuracy is lower in some intervals (for example, before and after meetings). The low accuracy in these intervals can be attributed to the fact that ground truth occupancy is extracted from the calendar, but occupants may arrive before the start of a meeting or leave before the meeting ends. We plan to investigate this in future work using a more reliable method of collecting ground truth data.

5.2 Changing the Amount of Labelled Data Available in the Target Domain

In this section, we investigate how much ground truth data is needed in the target domain to train an accurate model for occupancy estimation. Suppose the source domain is Room 3 and Room 4, and the target domain is Room 1 and Room 2 in Building A.

Table 3 shows the performance of supervised learning models trained using labelled data from the target domain only when it is provided for 1 hour, 3 hours, and 1 day. Although the LSTM model had the highest accuracy when only 1 hour labelled data is available, the SVR model outperforms the LSTM model when more labelled data becomes available. Therefore, we choose SVR as our supervised learning algorithm and compare the accuracy of this model with unsupervised and semi-supervised domain adaptation models. Note that we cannot train a logistic regression model when only 1 hour or 3 hour worth of labelled data is available.

Next, we build different models using supervised learning (i.e., the SVR algorithm), and unsupervised and semi-supervised domain adaptation (with re-weighting) when ground truth data is available for 1 hour, 3 hours, and 1 day in the target domain. We compare their accuracy in determining the number of occupants. As shown in Figure 10, the RMSE of the model developed using unsupervised domain adaptation is always the same. This is expected because the unsupervised model does not use the labelled data from the target domain. Another observation is that the accuracy of the semi-supervised domain adaptation model improves only slightly when more labelled data becomes available. This suggests that we do not need an abundance of labelled data when we use the semi-supervised domain adaptation technique and we get satisfactory performance when only 1 hour labelled data is available. Finally, it can be seen that the accuracy of the supervised learning model (SVR) improves markedly (by 40%) as more labelled data becomes available. But its accuracy is still less than that of the semi-supervised domain adaptation even when training data is available for 1 day.

To highlight the benefit of performing domain adaptation, we compare it with two cases where we train supervised learning models using all labelled data from the target domain, and from both source and target domains. Suppose all labelled data are available in the target domain and consider the supervised learning model (i.e., the SVR model) trained with 80% of this data from the target domain, using the other 20% for testing. The RMSE of this model is 3.72 (nRMSE of 10%), which is only slightly better than the semi-supervised domain-adaptive model trained using only 1 hour labelled data from the target domain. Furthermore, if we train a supervised learning model using all labelled data from both source and target domains without performing any adaptation, the RMSE increases to 7.87 (nRMSE of 21%). In this case training data come from two different distributions and cannot be simply combined to train a model. Using the same training data, the semi-supervised domain-adaptive LSTM yields an RMSE of 3.66 (nRMSE of 10%).

5.3 Different Choices for Source and Target Domains

We now change the source and target domains and compare the accuracy of different models in each case. Figure 11 shows the RMSE obtained for different combinations of source and target domains. In each case, the target domain is the two rooms identified below the figure, while the source domain is the other two rooms in Building A. In all cases we use 1 day labelled data from the target domain. The bar with a solid fill shows the RMSE of the semi-supervised domain adaptation method, and the bar with a hatch fill shows the RMSE of the best supervised learning method.

Note that we need to carry out re-weighting when the target domain is Room 1 and Room 2 and when it is Room 3 and Room 4 (the two leftmost pairs of bars). This is because Room 1 and Room 2 are both study areas, and Room 3 and Room 4 are both lecture rooms, causing the source and target domains to have completely different parameters: floor area, ventilation power, etc. Without re-weighting, it is expected that the adapted model performs poorly in the occupant count determination task because the two domains are completely different. In the other four cases, both source and target domains contain one study area and one lecture room. Hence, the re-weighting step is unnecessary since the parameters in the two domains are the same. Therefore, the domain adaptation methods will not include the re-weighting method.

It can be readily seen that the average RMSE of the semi-supervised domain adaptation technique with re-weighting (i.e., the first two target domain pairs) is 6.5575, the average RMSE of

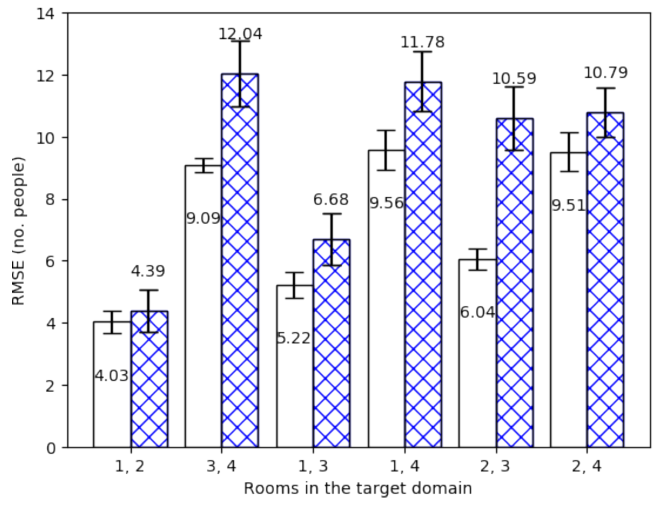

Figure 11: The RMSE of supervised and semi-supervised domain adaptation algorithm on different target domains given one day training data from the target domain. Error bars represent the 97% confidence interval.

the same technique without re-weighting (i.e., the last four target domain pairs) is 7.5854, and the average RMSE of the supervised learning method across all target domain pairs is 9.3785. This suggests that the domain adaptation method works better when the source and target domains have different settings and re-weighting is performed.

6 DISCUSSION

To achieve scalable deployments of building applications across the building stock, it is necessary to adapt existing occupancy models to new buildings. In this paper we showed that only 1 hour labelled occupancy data from the target domain is sufficient for the proposed semi-supervised domain adaptation technique to train a relatively accurate model for the target domain. This justifies the effort to collect a small amount of ground truth occupancy data so that building applications can adapt and reuse existing well-suited models.

Despite the novelty of our semi-supervised domain adaptation technique, it has several limitations. First, in our experiments, the source and target domains are different rooms within the same building (Building A or Building B). Thus, they are located in the same geography and share the same building envelop. This assumption makes it easier to adapt the model that is trained in the source domain. Second, the source and target domains share the same feature space, i.e., they have the same sensing modalities. Thus, it is not necessary to change the structure of our neural network model when we apply the domain adaptation technique.

In future work, we plan to investigate whether it is possible to train an occupancy detection model in one building and adapt to another building (possibly in a different climate) such that the performance of the adapted model is better than the performance of supervised learning models trained in that building using the available training data. Furthermore, we intend to study the domain adaptation problem when the two domains do not have the same feature space. Borrowing ideas from [41] and [21], which transfer the features into a latent space, we believe that this problem can be addressed in future work.

7 CONCLUSIONS

Despite the extensive body of research on determining occupant presence and count in the built environment by fusing a broad spectrum of sensing modalities, most related work does not discuss whether the proposed models can be applied to another room in that building. This paper attempted to address this gap in the literature by (a) training accurate data-driven models to discern the number of occupants of a room in a commercial building using the trend data acquired from the BMS, and (b) employing a domain adaptation technique to estimate the occupancy level of similar rooms in the same building with little labelled data. We showed that semi-supervised domain adaption is suitable when labeled occupancy data is sparse or non-existent. In particular, it can estimate the number of occupants in the target domain with an nRMSE of around 10% using only one hour labelled data, which is easy to obtain even manually. We showed that the semi-supervised domain adaptation model always outperforms the supervised learning model trained using the same amount of labeled occupancy data from the target domain. Furthermore, we found that the re-weighting step further improves the accuracy of the model (by about 4%) when the rooms in the source domain and target domain have different attributes.

ACKNOWLEDGEMENTS

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC), [funding reference number 530852 - 18]. We would like to thank PCL Constructors, Inc. for providing data from an office building in Edmonton, Canada.

REFERENCES

[1] Yuvraj Agarwal et al. 2011. Duty-cycling buildings aggressively: The next frontier in HVAC control. In *Proc. 10th International Conference on Information Processing in Sensor Networks (IPSN)*. ACM/IEEE, 246–257.

[2] Omid Ardakanian, Arka Bhattacharya, and David Culler. 2016. Non-Intrusive Techniques for Establishing Occupancy Related Energy Savings in Commercial Buildings. In *Proc. 3rd International Conference on Systems for Energy-Efficient Built Environments (BuildSys)*. ACM, 21–30.

[3] Omid Ardakanian, Arka Bhattacharya, and David Culler. 2018. Non-intrusive occupancy monitoring for energy conservation in commercial buildings. *Energy and Buildings* 179 (2018), 311–323.

[4] Krzysztof Arendt et al. 2018. Room-level Occupant Counts, Airflow and CO2 Data from an Office Building. In *Proc. 1st Workshop on Data Acquisition To Analysis*. 13–14.

[5] Irvan B Arief-Ang, Flora D Salim, and Margaret Hamilton. 2017. DA-HOC: Semi-Supervised Domain Adaptation for Room Occupancy Prediction using CO2 Sensor Data. In *Proc. 4th International Conference on Systems for Energy-Efficient Built Environments (BuildSys)*. ACM, 1:1–1:10.

[6] Anil Aswani et al. 2012. Reducing transient and steady state electricity consumption in HVAC using learning-based model-predictive control. *Proc. IEEE* 100, 1 (2012), 240–253.

[7] Bharathan Balaji et al. 2013. Sentinel: Occupancy Based HVAC Actuation Using Existing WiFi Infrastructure Within Commercial Buildings. In *Proc. 11th Conference on Embedded Networked Sensor Systems (SenSys)*. ACM, 17:1–17:14.

[8] Bharathan Balaji et al. 2016. Brick: Towards a Unified Metadata Schema For Buildings. In *Proc. 3rd International Conference on Systems for Energy-Efficient Built Environments (BuildSys)*. ACM, 41–50.

[9] Alex Beltran, Varick L. Erickson, and Alberto E. Cerpa. 2013. ThermoSense: Occupancy Thermal Based Sensing for HVAC Control. In *Proc. 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings (BuildSys)*. ACM, 11:1–11:8.

[10] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2007. Analysis of representations for domain adaptation. In *Advances in neural information processing systems*. 137–144.

[11] John S Bridle and Stephen J Cox. 1991. Recnorm: Simultaneous normalisation and classification applied to speech recognition. In *Advances in Neural Information Processing Systems*. 234–240.

[12] Kyunghyun Cho et al. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[13] Ken Christensen et al. 2014. Using existing network infrastructure to estimate building occupancy and control plugged-in devices in user workspaces. *International Journal of Communication Networks and Distributed Systems* 12, 1 (2014), 4–29.

[14] Alexis Conneau et al. 2017. Word translation without parallel data. *arXiv preprint arXiv:1710.04087* (2017).

[15] Stephen Dawson-Haggerty et al. 2013. BOSS: Building Operating System Services. In *Proc. 10th Symposium on Networked Systems Design and Implementation (NSDI)*. USENIX Association, 443–458.

[16] Varick L. Erickson, Stefan Achleitner, and Alberto E. Cerpa. 2013. POEM: Power-efficient Occupancy-based Energy Management System. In *Proc. 12th International Conference on Information Processing in Sensor Networks*. ACM, 203–216.

[17] Timnit Gebru, Judy Hoffman, and Li Fei-Fei. 2017. Fine-grained recognition in the wild: A multi-task domain adaptation approach. In *Proc. International Conference on Computer Vision (ICCV)*. IEEE, 1358–1367.

[18] Sunil K. Ghai, Lakshmi V. Thanayankizil, Deva P. Seetharam, and Dipanjan Chakraborty. 2012. Occupancy detection in commercial buildings using opportunistic context sources. In *Proc. International Conference on Pervasive Computing and Communications*. IEEE, 463–466.

[19] Shadan Golestan, Sepehr Kazemian, and Omid Ardakanian. 2018. Data-Driven Models for Building Occupancy Estimation. In *Proc. 9th International Conference on Future Energy Systems (e-Energy)*. ACM, 277–281.

[20] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 6645–6649.

[21] Hani Hagras, Victor Callaghan, Martin Colley, and Graham Clarke. 2003. A hierarchical fuzzy–genetic multi-agent architecture for intelligent buildings online learning, adaptation and control. *Information Sciences* 150, 1-2 (2003), 33–57.

[22] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[23] Ming Jin et al. 2016. Occupancy detection via environmental sensing. *Transactions on Automation Science and Engineering* (2016).

[24] Deokwoo Jung et al. 2013. EnergyTrack: Sensor-Driven Energy Use Analysis System. In *Proc. 5th Workshop on Embedded Systems For Energy-Efficient Buildings (BuildSys)*. ACM, 6:1–6:8.

[25] Taeksoo Kim et al. 2017. Learning to discover cross-domain relations with generative adversarial networks. *arXiv preprint arXiv:1703.05192* (2017).

[26] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[27] Mikkel Baun Kjærgaard, Martin Werner, Fisayo Caleb Sangogboye, and Krzysztof Arendt. 2018. DCount-A Probabilistic Algorithm for Accurately Disaggregating Building Occupant Counts into Room Counts. In *International Conference on Mobile Data Management*.

[28] Gierad Laput, Yang Zhang, and Chris Harrison. 2017. Synthetic Sensors: Towards General-Purpose Sensing. In *Proc. Conference on Human Factors in Computing Systems (CHI)*. ACM, 3986–3999.

[29] Sunil Mamidi, Yu-Han Chang, and Rajiv Maheswaran. 2012. Improving Building Energy Efficiency with a Network of Sensing, Learning and Prediction Agents. In *Proc. 11th International Conference on Autonomous Agents and Multiagent Systems*. IFAAMAS, 45–52.

[30] Danilo P Mandic and Jonathon Chambers. 2001. *Recurrent neural networks for prediction: learning algorithms, architectures and stability*. John Wiley & Sons, Inc.

[31] Tomáš Mikolov et al. 2010. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*.

[32] Kumpati S Narendra and Kannan Parthasarathy. 1990. Identification and control of dynamical systems using neural networks. *IEEE Transactions on neural networks* 1, 1 (1990), 4–27.

[33] Bill Von Neida, Dorene Manicria, and Allan Tweed. 2001. An Analysis of the Energy and Cost Savings Potential of Occupancy Sensors for Commercial Lighting Systems. *Illuminating Engineering Society* 30, 2 (2001), 111–125.

[34] Antonio Ruiz-Ruiz et al. 2014. Analysis methods for extracting knowledge from large-scale wifi monitoring to inform building facility planning. In *Proc. International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 130–138.

[35] Fisayo Caleb Sangogboye et al. 2017. Performance comparison of occupancy count estimation and prediction with common versus dedicated sensors for building model predictive control. In *Building Simulation*, Vol. 10. Springer, 829–843.

[36] Chenguang Shen et al. 2016. Beam: Ending Monolithic Applications for Connected Devices. In *USENIX Annual Technical Conference*. USENIX Association, 143–157.

[37] Hongsen Shi, Jia Liu, and Qian Chen. 2018. HVAC Precooling Optimization for Green Buildings: An RC-Network Approach. In *Proc. 9th International Conference on Future Energy Systems (e-Energy)*. ACM, 249–260.

[38] Jay Taneja, Andrew Krioukov, Stephen Dawson-Haggerty, and David Culler. 2013. Enabling advanced environmental conditioning with a building application stack. In *Proc. International Green Computing Conference Proceedings*. 1–10.

[39] The Dark Sky Company LLC. 2018, Retrieved. Dark Sky API. Online https://darksky.net/dev. (2018, Retrieved).

[40] Amee Trivedi et al. 2017. iSchedule: Campus-scale HVAC Scheduling via Mobile WiFi Monitoring. In *Proc. 8th International Conference on Future Energy Systems (e-Energy)*. ACM, 132–142.

[41] Chang Wang and Sridhar Mahadevan. 2011. Heterogeneous Domain Adaptation Using Manifold Alignment. In *Proc. 22nd International Joint Conference on Artificial Intelligence - Volume Two (IJCAI'11)*. AAAI Press, 1541–1546.

[42] Mei Wang and Weihong Deng. 2018. Deep Visual Domain Adaptation: A Survey. *Neurocomputing* (2018).

[43] Datong Zhou, Qie Hu, and Claire J. Tomlin. 2016. Model Comparison of a Data-Driven and a Physical Model for Simulating HVAC Systems. *CoRR* abs/1603.05951 (2016).

[44] Datong P Zhou, Qie Hu, and Claire J Tomlin. 2017. Quantitative comparison of data-driven and physics-based models for commercial building HVAC systems. In *American Control Conference (ACC), 2017*. IEEE, 2900–2906.

[45] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint* (2017).

[46] Han Zou et al. 2017. WinLight: A WiFi-based occupancy-driven lighting control system for smart building. *Energy and Buildings* (2017).