AGILE: GIScience Series, 2, 9, 2021. https://doi.org/10.5194/agile-giss-2-9-2021
Proceedings of the 24th AGILE Conference on Geographic Information Science, 2021.
Editors: Panagiotis Partsinevelos, Phaedon Kyriakidis, and Marinos Kavouras.
This contribution underwent peer review based on a full paper submission.

Prophet model for forecasting occupancy presence in indoor spaces using non-intrusive sensors

Alec Parise[a], Miguel A. Manso-Callejo[b], Hung Cao[a] (corresponding author) and Monica Wachowicz[a,c]

alec.parise@unb.ca, m.manso@upm.es, hcao3@unb.ca, monicaw@unb.ca

[a]People in Motion Lab, University of New Brunswick, Canada
[b]Universidad Politécnica de Madrid, Spain
[c]RMIT, Australia

Abstract. The Internet of Things is a multi-sensor technology with the unique advantage of supporting non-intrusive and non-device occupancy detection, while also allowing us to explore new forecasting occupancy models. However, forecasting occupancy presence is not a trivial task, since it is still unknown the main criteria in selecting a forecasting modelling approach according to a non-intrusive sensing strategy. Towards this challenge, this paper proposes an analytical workflow developed to support the Prophet model for forecasting occupancy presence in indoor spaces throughout the tasks of sensing, processing, and analysing event triggered data generated from ten non-intrusive sensors, including motion, temperature, luminosity, CO_2, TVOC, sound, pressure, accelerometer, gyroscope, and humidity sensors. The usefulness of this analytical workflow is demonstrated with the implementation of an IoT platform for an experiment operating non-intrusive sensing in a classroom. The assessment is made at different time intervals and the results confirm that there is a relationship between the event-count and occupancy presence in such a way that the larger the number of events triggered in an indoor space, the higher the probability of an indoor space being occupied.

Keywords. Internet of Things, occupant behavior, non-intrusive sensing, Prophet forecasting model

1 Introduction

Forecasting occupant behavior in indoor spaces provides relevant information for planning building automation, evaluating energy efficiency scenarios, and simulating emergency protocols (Jia et al., 2017; Trivedi and Badarla, 2020). Occupant behavior refers to the presence and numbers of occupants in indoor spaces, and their various interactions that can take place over time such as opening/closing windows and doors, or turning on/off lighting in a room. These interactions can also be associated with activities, including meeting someone, giving a lecture, or working on a desk. Due to the stochastic nature of occupant behavior, previous occupancy forecasting models considerably diverge in terms of the types of sensors being used to gather occupancy data; the complexity level of single versus multi-occupant forecasting; and the arbitrary selection of short versus long-term forecast horizons (Hutchins et al., 2007; Chen et al., 2018; Alawadi et al., 2020).

With the advent of the Internet of Things (IoT), a wider spectrum of non-intrusive sensors and networking communication technologies are available for improving the collection, transportation, and analysis of time-series (i.e. recorded timestamped readings at successive and equal time intervals) and event triggered data (i.e. recorded timestamped readings when a sensor is triggered due to an activity or interaction happening in an indoor space). These non-intrusive sensors are generally easier to install, and they can also be cheaper than intrusive sensors (Laput et al., 2017).

Overall, we can distinguish three levels of non-intrusive sensing strategies in indoor spaces (Zou et al., 2017; Saha et al., 2019). At the most basic level of non-intrusive sensing, a selection of sensors such as Pyroelectric InfraRed Sensors (PIR) motion, CO_2, temperature, luminosity, acoustic, and humidity sensors can be used for occupancy detection. A variety of data mining approaches have been proposed for analysing the time-series data generated by these sensors, including random forest, Hidden Markov Models (HMM), Support Vector Machine (SVM), Convolution Networks (CNN), and Long Short Term Memory (LSTM) (Rueda et al., 2020).

The second strategic level aims at occupancy counting where occupancy-count estimations in a predefined zone are obtained through sensor fusion using opportunistic sensor data from Wi-Fi access points, CO_2 sensors placed in a room, PIR motion detectors at doors, and plug and light electricity load meters. Multiple linear regression models and deep learning models have been explored to merge individual readings from different sensors in an exhaustive number of permutations (Hobson et al., 2019).

Finally, the highest strategic level consists of estimating the location of each occupant using RFID, Bluetooth, and WiFi communication technologies that are usually available in indoor spaces, but requiring occupants to carry a device or an additional tag (Bai et al., 2020; Rohei et al., 2020). In the case of WiFi positioning, the fingerprinting and trilateration methods based on WiFi signal strength measurements are usually employed (Mok and Retscher, 2007; Zhang et al., 2020).

Independently of the strategy being used in sensing occupant presence an indoor space, analysing time-series with high sampling rates containing a large volume of redundant readings can cause unnecessary noise and increase the computational costs in forecasting models. Moreover, different sampling intervals might be used depending on the type of non-intrusive sensors being deployed, the feature selection, and the indoor space in which they are sensed. Therefore, selecting an appropriate non-intrusive sensing strategy is still of great concern and remains as a main challenge for adopting new forecasting occupancy models.

Towards this challenge, we propose a synergy of levels one and two of non-intrusive sensing in indoor spaces. Our overall strategy relies on accurately inferring occupancy presence by deploying complementary non-intrusive sensors. But instead of processing their actual time-series data or performing sensor fusion, we propose to create a new time-series data based on counting the events triggered by these sensors due to any activity or interaction that have occurred in an indoor space (i.e. occupancy counting level). The rule of thumb is that there is a ratio between the event-count and occupancy presence in such a way that the larger the number of events triggered in an indoor space, the higher the probability of an indoor space being occupied.

This paper describes an analytical workflow developed to forecast occupancy presence in indoor spaces by sensing, processing, and counting the total number of triggered events generated from non-intrusive sensors. The Prophet model previously proposed by Taylor and Letham (2018) is selected for forecasting occupancy presence using three time intervals (i.e. 30 minutes, one hour, and two hours). Our aim is to evaluate whether the Prophet model would equally perform independently from the duration of a time interval.

The remainder of this paper is organised as follows: related work is described in Section 2, and Section 3 introduces our proposed analytical workflow. Section 4 will outline the proposed IoT Architecture and Section 5 describes the experiment that took place at a computer lab. Section 6 discusses the results, and lastly in section 7 the conclusions and future work are considered.

2 Related Work

Significant research work can be found in the literature on detecting occupancy in indoor spaces using intrusive and device-based approaches (Chen et al., 2018; Rueda et al., 2020). Fewer attempts have been found on detecting occupancy using non-intrusive sensors, despite the availability of cheap off-the-shelf sensors that are key to collecting time-series data needed for forecasting models.

Saha et al. (2019) provide an extensive review on the main models developed for occupancy detection, counting and tracking in indoor spaces using CO_2 sensors, PIR motion detectors, and optical counting sensors. Examples of models developed for occupancy detection include random forest, Hidden Markov Models (HMM), Support Vector Machine (SVM), Convolution Networks (CNN), Long Short Term Memory (LSTM), clustering analysis and statistical learning methods. One major finding in this review is that when using non-intrusive sensors, feature selection plays an important role in reducing the vast amount of time-series data when no valuable information is actually added to a forecasting model as well as handling sensor readings conflicting each other and occurrence of rare readings.

Therefore, it should not come as a surprise that previous models used for occupancy detection have also been explored for occupancy forecasting in indoor spaces. Alawadi et al. (2020) provide a comprehensive comparison of 36 offline machine learning models applied for forecasting the indoor temperature by combining temperature sensor data from an indoor space and the meteorological conditions from a nearby weather station. Using the Friedman rank and the R-coefficient to evaluate the accuracy of three forecast horizons, they conclude that the neural network models (e.g. avNNET) were more sensitive to outliers and the inherited noise in temperature sensor data. The regressors with less sensitivity were ExtraTRees, cubist and random forest. The Prophet model was not considered in this comparison study.

Previous research has also pointed out the importance of using non-intrusive sensing due to privacy concerns and inaccurate measurements from cameras. For example, the HMM model was proposed to predict occupancy presence using indoor and outdoor CO_2 concentration ratios, indoor CO_2 concentration 15-min moving average, and the total energy consumption of the lighting system and appliances (Ryu and Moon, 2016). The results demonstrate the suitability of the HMM model to forecast daily high occupancy rather than low daily occupancy.

Towards a non-intrusive sensing strategy for building occupancy forecasting models, Hutchins et al. (2007) explore the Markov Chain Monte Carlo (MCMC) probabilistic model using a sample of occupant-count data gathered from optical sensors positioned at an entrance/exit door of a building that were capable of registering a count when an optical beam was interrupted. The model includes the occupancy-counts measured by the optical sensors, non-measured variables representing the true occupancy at a specific timestamp, and parameters such as Poisson rates. In the experiments, the MCMC model and a baseline model are compared daily, revealing the imbalanced larger number of entering predictions in comparison to the exiting predictions. This problem has raised due to the expected noise in sensor readings corresponding to both over and under-counting.

In contrast, alternative forecasting models, such as ARIMA, SARIMA, and Prophet models, are potential new methods for forecasting indoor occupancy using non-intrusive sensing, but they have been neglected so far. They have been successfully applied for forecasting hourly traffic volume and pollutant values (Chikkakrishna et al., 2019). Using the mean square error and mean absolute square error (MAPE), the predictions show that the SARIMA model generated the most accurate predicted traffic volume. However, the Prophet model produced more accurate trends in the predicted traffic, allowing the non-smooth data to fit into the model the best.

The SARIMA and Prophet models have also been used for finding annual forecasts using historical data from 2005 to 2015 containing information about hazardous pollutants such as RSPM (Respirable Suspended Particulate Matter), NO_2, SO_2, and SPM (Suspended Particulate Matter) (Samal et al., 2019). In this research work, the Prophet model has generated the most accurate predictions based on their RMSE and MSE values.

Alternatively, the Croston method has been proposed for forecasting irregular demand patterns when they are not representing a normal distribution (Syntetos et al., 2011). This method suggests that the demand of Stock Keeping Units (SKUs) is sporadic and occurs at random, meaning there can be no demand at all or a constant demand, and likewise there might be instances that demand is not a single unit during some periods. Occupancy behaviour exhibits the same sporadic demand, where some indoor spaces will be unoccupied most of the time, but in other periods, they may vary in the number of people occupying a space. Assuming class and work schedules introduce seasonality to the sporadic nature of occupancy; individual occupancy demand may be random but group occupancy should occur regularly. Therefore, seasonality should be included when building a forecasting occupancy model.

Our research premise is that our proposed non-intrusive strategy requires generalized additive models such as the Prophet model, where linear trends are fit with hourly, daily, and weekly seasonality. The forecasting of event-counts are considered a powerful proxy for forecasting occupancy presence, avoiding gathering large volumes of time-series data from multi-sensors having and sensor readings conflicting each other.

3 Analytical Workflow

Developing the tasks of an analytical workflow is a crucial process for sensing, preprocessing, and analysing non-intrusive sensor data. Figure 1 provides an overview of the main tasks of our proposed analytical workflow, described as follows:

- *Sensing*: The aim is to select the type of sensors (e.g. CO_2 sensor) and their respective thresholds (e.g. any recording changing in 50ppm for a CO_2 sensor will be registered as an event). The outcome of this task is a data set containing all the events triggered by any sensor at a specific timestamp.

- *Preprocessing*: This task consists of generating a series of the total number of events y that have been triggered over a period of time in a particular indoor space. The outcome of this task is a time-series of event-counts according to a fixed time interval.
- *Forecasting*: This task consists of applying the Prophet model for predicting future event-counts.The outcomes of this task are forecast $\hat{y}$ values and their respective MAPE error.
- *Labelling*: The aim is to compute Z-score values for labelling the forecast $\hat{y}$ values as occupied or unoccupied indoor spaces.

These tasks are further explained in in the sections below.

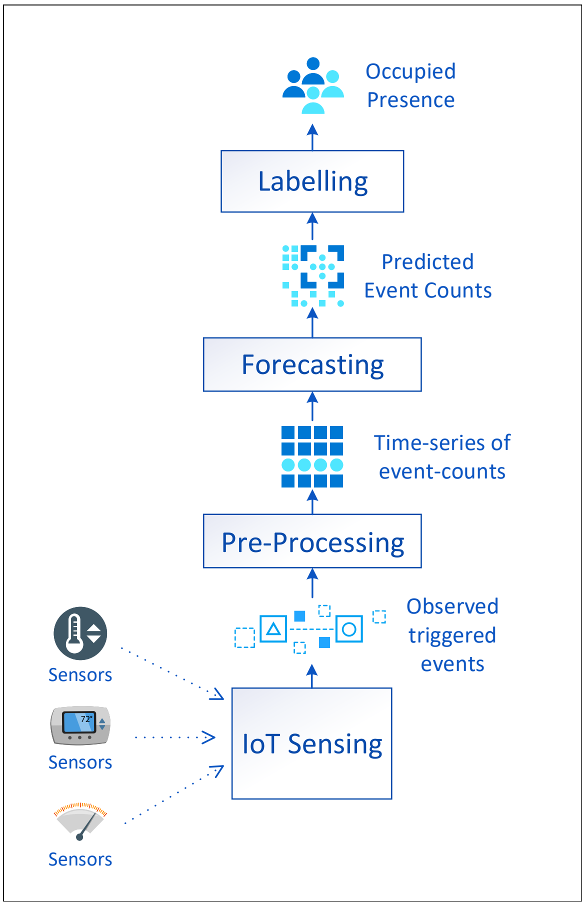


Figure 1. Overview of the proposed analytical workflow

3.1 Sensing

This task aims to collect event-triggered data from a sensor node deployed in an indoor space. A sensor node may include many types of sensors, such as temperature, humidity, motion, sound, luminosity, pressure, accelerometer, gyroscope, CO_2, smoke, and TVOC sensors.

Independently of the sensor being used, events are triggered by an individual sensor when its recording reaches a value higher than an a-priori defined threshold, which can be associated to interactions and activities happening in an indoor space. Some examples include an occupant entering a classroom and switching on lights, starting a meeting, or giving a lecture.

However, each individual sensor requires different approaches to set up their thresholds. With numerical recordings from sensors, such as temperature, humidity, and luminosity, events are triggered using a delta threshold: if temperature, humidity or luminosity recordings increase or decrease by a factor of X, the recordings will be registered as an event. In contrast, for CO_2 and TVOC recordings, a fixed threshold X is used because CO_2 and TVOC recordings have a regulated base value of 400 ppm and 0 ppm. Recordings from motion and sound sensors are binary recordings (i.e. motion detected or not detected). If motion or sound is detected within an indoor space, an event is registered.

The outcome of this task is a set of timely ordered tuples representing an event-triggered by a sensor,which consists of a timestamp, the sensor recordings from all sensors at a particular timestamp, and the type of a triggered event as illustrated in Table 1. Types of triggered events include motion was detected; temperature has changed above the threshold; or noise was detected.

Table 1. Data tuples of recorded triggered events

Tuple	Sensor Recordings	Event	Timestamp
t_1	$S_1, S_2 ... S_n$	$Event_S1$	ts_1
t_2	$S_1, S_2 ... S_n$	$Event_S2$	ts_1
t_3	$S_1, S_2 ... S_n$	$Event_S2$	ts_2
t_4	$S_1, S_2 ... S_n$	$Event_S3$	ts_2
....	...., ...	.....	....
t_n	$S_1, S_2 ... S_n$	$Event_Sn$	ts_n

3.2 Preprocessing

The event-triggered data is usually noisy and requires either removing tuples with missing categorical values or replacing missing numerical values using the mean (Raschka, 2014). Another important step is the creation of a time-series containing the total number of triggered events within a time interval. Depending on the required data rate, and the expected latency and bottlenecks of the communication network to transport the sensor data, we partition the data by generating timeframes of ordered event-counts for an indoor space, as shown in Table 2.

Table 2. Time-series of event-counts

Timeframe	Event-Count
T_1	E_1
T_2	E_2
T_3	E_3
....	...
T_i	E_i

3.3 Forecasting

This task aims to apply the Prophet model for forecasting future event-counts using different time intervals for the time-series. The Prophet model was first introduced by Taylor and Letham (2018) and consists of three main model components: trend, seasonality, and holidays, which are combined in the following equation:

$$y(t) = g(t) + s(t) + h(t) + \xi(t) \tag{1}$$

where, $g(t)$ is the trend function used for modelling non-periodic changes, $s(t)$ represents the periodic changes (e.g., hourly or daily seasonality), $h(t)$ represents the effects of holidays which occur on potentially irregular schedules, and ξ(t) is the error term which accounts for any idiosyncratic changes which are not accommodated by the model.

The time-series of event-counts generated in the previous task of the proposed analytical workflow is expected to display seasonality with multiple periods, and have strong linear trend changes, outliers, and holiday effects. Therefore, the Prophet model allows us to define the forecasting of occupancy presence in indoor spaces as a curve-fitting exercise, which is inherently different from previous forecasting models that explicitly rely on the temporal dependence structure in time-series data such as the SARIMA model.

The linear trends changes are represented as a piecewise constant rate of growth, which is defined in the following equation:

$$g(t) = (k + a(t)^T \delta)t + (m + a(t)^T \gamma) \tag{2}$$

where k is the growth rate, δ serves as the rate at which $a(t)^T$ is adjusted, m is the offset parameter, and to make the function continuous γ_j is set to $-s_j\delta_j$.

The changepoints (s_j) are usually related to growth-altering events such as the beginning of a class. To calculate the changepoints, the Laplace transformation is applied on the prior δ_j such that $\delta_j \sim$ Laplace(0,τ). The parameter τ directly determines the flexibility of the model. However, linear growth occurs when τ approaches 0 standard. This happens when the adjustments made on δ results in having no impact on the growth rate k.

The future changepoints are randomly sampled so that the average changepoints over time match those that in the historical data, defined as:

$$\forall_j > T, \begin{cases} \delta_j = 0(w.p.)\frac{T-S}{T}, \\ \delta_j \text{ Laplace}(0, \lambda)(w.p.)\frac{S}{T} \end{cases} \tag{3}$$

The uncertainty for the forecast trend is based on the assumption that the future will see the same average frequency and magnitude of rate changes that were seen in the historical data. It is estimated with a constant trend rate, creating a generative model. The generative model is based on S changepoints over a history of T event-counts. In essence, the future rate changes at each $\delta_j \sim$ Laplace(0,τ) is done by replacing τ with a variance inferred from data. This is done by using the maximum likelihood estimate (MLE) of δ_j:

$$\lambda = \frac{1}{S}\sum_{j=1}^{S}\left|\delta_j\right| \tag{4}$$

The Prophet model relies on standard Fourier series to provide a flexible generative model for representing seasonality. The arbitrary smooth seasonal effects can be computed using the Equation 5.

$$s(t) = \sum_{i=1}^{N}\left(a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right)\right) \tag{5}$$

where P is the regular period we expect the time series to have, n is the coefficient for seasonal smoothing, and t is time. The Akaike Information Criterion (AIC) can be applied to automatically determine N for reducing the effects of over/under-fitting. However, the default values such as N = 10 or N = 3 have proven to be accurate to represent yearly and weekly seasonality, having a P = 354.25 for yearly seasonality or P=7 for weekly seasonality respectively.

Furthermore, holidays can have a negative impact on a forecasting model because their effects are not well modelled by a smooth cycle. But a list of holidays can straightforward be incorporated to the Prophet model by assuming the impacts of holidays are independent.

This is carried out in a similar way as seasonality by generating a matrix of regressors

$$Z(t) = [1(t\epsilon D_1)..., 1(t\epsilon D_L)] \tag{6}$$

where D_i is a set of past and future dates for holidays i, and later assigning a parameter κ_i for each holiday by taking the indicator function

$$h(t) = Z(t)k \tag{7}$$

with a prior $\kappa \sim$ Normal(0, ν^2).

Finally, forecasts in the Prophet model are made over a certain forecast horizon H, which represents the period of future event-counts that will each be associated with some error. Let $\hat{y}(t|T)$ represent the forecasts made at time t based on the historical data up to time T. The $d(y, y\prime)$ serves as a distance metric, which can be used to calculate MAPE, which is defined as:

$$M = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right| \tag{8}$$

where A_t is the actual value and F_t is the forecast value.

However, to apply this metric for assessing the empirical accuracy of a forecast of $h\epsilon(0, H]$ periods ahead of time T, the following formula is derived:

$$\theta(T, h) = d(\hat{y}(T + \frac{h}{T}), y(T + h)) \tag{9}$$

A non-parametric approach is used to estimate out-of-sample error that is comparable to the well-known cross-validation approach. The model is then fit with the expected errors at the different horizons h shown in Equation 10:

$$\xi(h) = E[\phi(T, h)] \tag{10}$$

The procedure known as Simulated Historical Forecasts (SHF) is used to generate historical forecast errors to fit the model. This was achieved by producing k forecasts at various cutoff points in the historical data, which allows for the total error to be evaluated based on a rolling origin evaluation that uses a small sequence of cutoff points, rather than using the entire historical data set. The advantage of using a rolling origin is that it generates fewer correlated errors and higher performance computation (Taylor and Letham, 2018). In essence, SHF simulates the errors at each cutoff point as if the horizon lied within the historical data. The simulated errors were then fit to the forecast model and the MAPE score is calculated for the predicted event-counts. The output is the $\hat{y}$ forecasts at each rolling cutoff, which were used to create the future event-count forecasts.

3.4 Labelling

The objective of this task is to add an occupancy presence label for each forecast event-counts. The Z-score is selected to determine how close a forecast event-count value $\hat{y}$ is to the population mean μ, as defined in Equation 11.

$$z = \frac{x - \mu}{\sigma} \tag{11}$$

where x is the forecast $\hat{y}$ value, μ is the population mean, and σ is the population variance.

It is an exceptional statistical measure for determining outliers based on how many standard deviations a forecast $\hat{y}_i$ value is away from the mean of its data set. For example, a Z-score of 0 indicates that an $\hat{y}_i$ value is identical to the mean. Alternatively, a Z-score equal to 2.0 indicates that an $\hat{y}_i$ value is two standard deviations from the mean. The Z-scores can be positive or negative, depending on whether their $\hat{y}$ values are below or above the mean. Therefore, if Z-score values are greater than 0, their respective $\hat{y}$ values are labelled occupied (O_n); otherwise they are labelled unoccupied (U_n).

4 IoT Platform

This section describes the implemented three-layer architecture for supporting the proposed analytical workflow. They are: Sensing, Network Communication, and Cloud Access and Processing layers. The overall IoT platform is illustrated in Figure 2.

4.1 Sensing Layer

The sensor nodes were designed to collect event-triggered tuples from nine sensors: motion (HC-SR501), temperature (BMP280), luminosity (SI1145), humidity (HC1080), CO_2/TVOC (SGP30), sound (KY-038), accelerometer, gyroscope, and infrared. The sensors were connected to a NodeMCU microcontroller using Arduino. The Arduino script was designed to gather tuples of triggered events containing

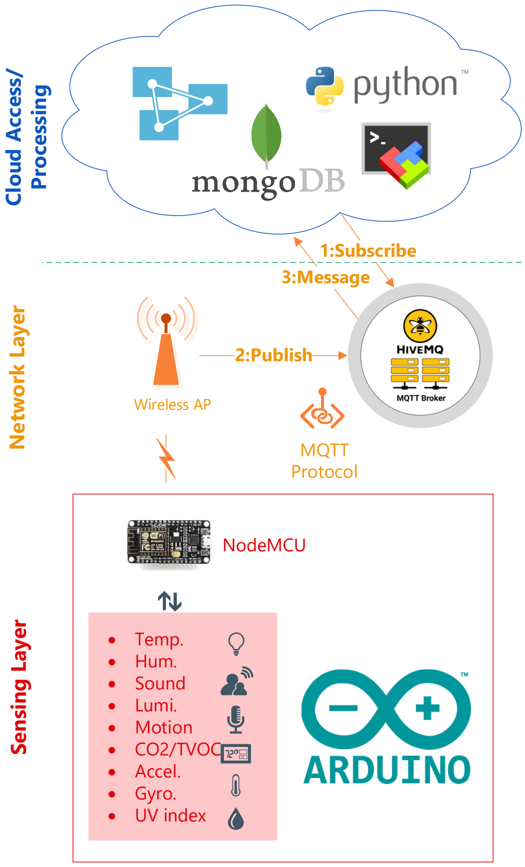


Figure 2. Overview of the proposed IoT platform

the multi-sensor recordings, the type of event which has occurred, and the timestamp as shown in Table 1.

4.2 Network Communication Layer

Wi-Fi operating in 802.11 protocol with Wi-Fi frequency band of $2,4$ GHz was selected due to its wide availability and ability to transfer large amounts of data. For data exchange, a public MQTT broker, Hive MQ was used for IoT communication. This protocol relies on a publish/subscribe method: the NodeMCU client publishes the event-triggered data to the MQTT broker, then subscribes to that topic by another device, such as the cloud, where the data is stored. MQTT offers flexibility because it supports three levels of quality enforcement, which are message sending without an acknowledgement request; message sending once with an acknowledgment request; and message sending through a handshake mechanism.

4.3 Cloud Access and Processing Layer

As the raw event-triggered streams were passed on by the broker, they were stored in the MongoDB, a document-oriented database because it was cost-effective and easier to set up, configure, and run in comparison to a commercial GIS. It was also suitable for storing event-triggered data as documents using JavaScript frameworks (i.e. JSON). After the data was acquired and stored it was then extracted from MongoDB using MobaXterm, for serving as input into the analytical workflow.

5 Experiment

5.1 Sensing

The indoor space was a lab classroom with unique features, including 32 computers, a projector, lecture podium, two entrances, two heat pumps, and multiple windows. Environmental sensors were selected to collect the ambient room temperature, humidity, luminosity, CO_2, and TVOC. Meanwhile, PIR motion, sound, accelerometer, and gyroscope sensors were selected to record physical activity in the in the lab classroom. The experiment aimed at collecting events triggered by all the sensors connected to a sensor node, which was placed in the middle of the room and tested for maximum coverage as shown in Figure 3.

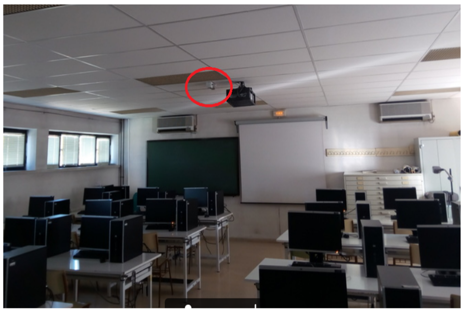


Figure 3. The location of the sensor node unit in the lab classroom (red circle)

Events were triggered by a delta threshold (i.e. temperature, humidity and luminosity), fixed thresholds (i.e. CO_2 and TVOC) and binary thresholds (i.e. motion and sound). The delta, fixed, and binary thresholds were manually programmed in the Arduino script. When a threshold was exceeded, the type of event would be registered and the data tuples, containing all sensor measurements and timestamps, were sent to the cloud. Each threshold value was determined during a preliminary experiment to evaluate the measured sensor values for when the room was free or occupied. The adopted threshold values are shown in Table 3.

Table 3. Threshold values used for the deployed sensors

Sensor Type	**Threshold**
Temperature	+/- 0.2°C
Humidity	+/- 2 RH
Luminosity	+/- 20 lux
CO_2	50 ppm
TVOC	20 ppm
Motion	motion detected
Sound	sound detected
Accelerometer	0.05 m/s^2
Gyroscope	10 RPS

5.2 Preprocessing

The following steps were implemented for the preprocessing task:

- Cleaning the raw triggered-event data: The tuples containing missing values were removed. The data set did not contain any duplicate tuples; however, there were delays in fetching the recordings from the PIR motion sensor.
- Converting the timestamps: This step was included to convert a timestamp to the datestamp Pandas format YYYY-MM-DD HH:MM:SS, which is required by the Prophet algorithm.
- Creating a time-series: The groupby function in Python was used to group the events per hour. Once the hourly timeframes were created, a count of events was executed. The result consists of a dataframe with two columns: ds (datestamp) and y (event-counts). The y is the variable (i.e. number of events) that we aim to forecast $\hat{y}$. Table 4 illustrates a dataframe generated for the time-series. It is important to point out that event-counts do not represent a real-world occupancy event.

5.3 Forecasting

The Prophet model is available as open source software in Python at https://github.com/facebook/prophet). This algorithm had a a single dataframe consisting of two columns: the timeframe (ds) and the event-count (y). Figure 4 illustrates the observed trend and seasonality found in the time series.

Table 4. Dataframe used as an input to the Prophet model

$T_i(ds)$	$E_i(y)$
1	400
2	480
3	501
4	650
5	760
6	810

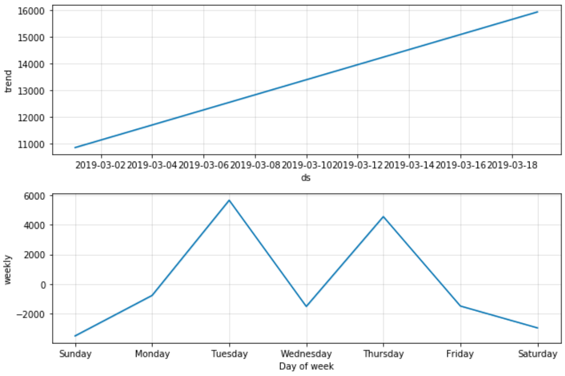


Figure 4. Observed trend and seasonality

It was reassuring to observe that the actual trend was actually linear, and due to the regular class schedules on Tuesday and Thursdays, this strong weekly seasonality was also observed in the time series. For representing this seasonality, a Fourier order of $N = 3$ was selected for modelling the smooth seasonal effects.

Only 80 percent of the time series data was used for computing the changepoints in order to have plenty of runway for projecting the trend forward and to avoid over-fitting fluctuations at the end of the time series. Figure 5 shows the 13 changepoints that were selected for our Prophet model and their respective change rates.

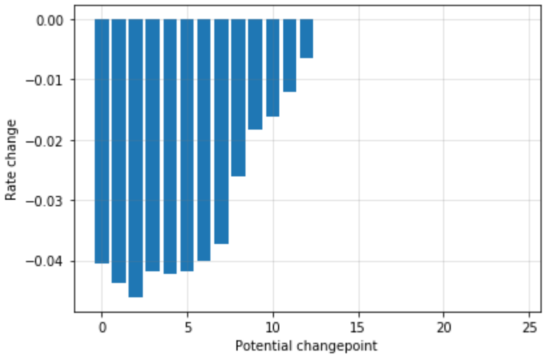


Figure 5. The rate of change per checkpoint

The training period was carried out for 15 days, and the forecast horizon period was 6 days. Three time intervals were used (i.e. 30 minutes, 1 hour, and 2 hours) for the training, and the cutoff was made after 14 days. The output of the forecasting was a dataframe containing the forecast values $\hat{y}$, at each time interval. The MAPE was computed for each timeframe to evaluate the accuracy of the forecasting.

5.4 Labelling

The labels were generated using a conditional statement that calculated Z-score values for every $\hat{y}$ value. Depending of its Z-score value, an $\hat{y}_i$ value was labelled as being either unoccupied (U_n) or occupied (O_n), as shown in the algorithm below. The rule is that if $Z_i > 0$ for an $\hat{y}_i$ value, the lab classroom was considered to be occupied during that timeframe; otherwise the lab classroom was unoccupied.

```
Algorithm 1: Event-count labelling using Z-scores.
Data: Forecast $\hat{y}$ and $T(_{i,...n})$ generated from the Prophet model
Result: Labels unoccupied $(U_n)$ or occupied $(O_n)$
Function Zscore_Time_frame($z$):
    for event-count in $(\hat{y}_1, y_2, , \dots y_n)$ do
        the calculation of $z$ scores
        Defined by $z = ((\hat{y}_1, y2, , \dots y_n) - \mu)/\sigma$
    end
Initialize: Set of occupancy $U_n$ labels generated from Z-scores per
  timeframe T
Function Occupancy_Time_frame($O_n$):
    forall $Z_{i,...n}$ in $T'(_{i,...n})$ do
        if $Z$ is $> 0$ then Return $O_n$;
        else Return $U_n$;
    end
```

In total, there were 87 hours out of the 20 days that the lab classroom was scheduled for a class (Figure 6). A total of 12 hours were wrongly annotated as occupied hours, meaning that the overall accuracy of the O_n class was 84%. When evaluating the times when the lab classroom was not scheduled for a class, the labelling task achieved 68% accuracy for the U_n class. In other words, the lab classroom was labelled as being occupied even when there was no class in session. Assuming that occupancy only occurs during scheduled class times, the model has achieved an acceptable accuracy.

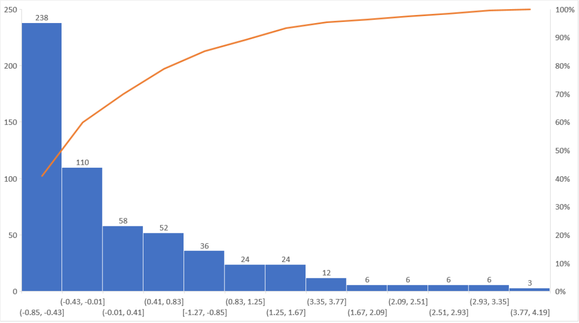


Figure 6. Distribution of Z-score values

6 Discussion of the Results

6.1 Low and High Occupancy Presence

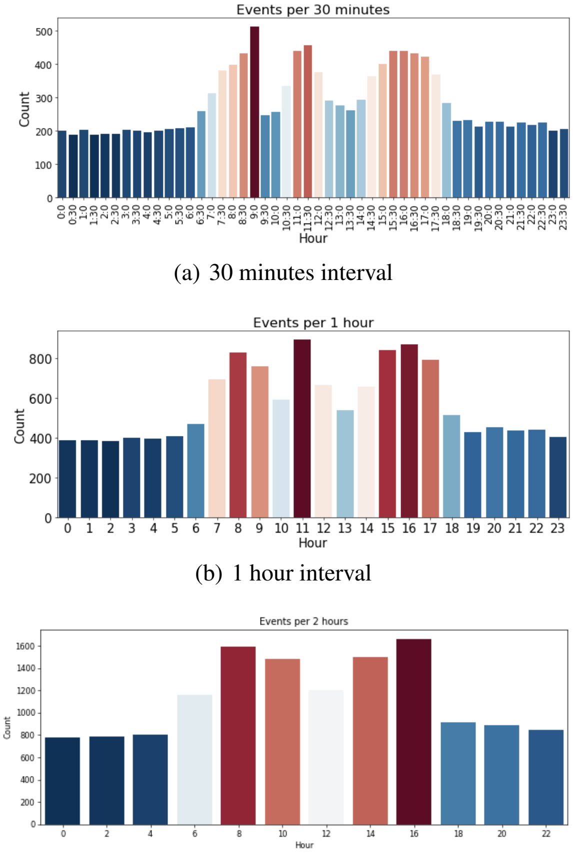
The raw event triggered data contained over 400,000 tuples during a 20 day period. The time-series generated 480 timeframes containing hourly total number of event-counts. They provided empirical evidence on the low/high occupancy presence in the lab classroom.

Figure 7(b) illustrates the changes in the frequency of hourly event-counts during one day when three classes have taken place at 8h, 11h, and 16h. For example, low frequency number of event-counts (dark blue) were found from 1h to 5h in the morning, which have increased between 6h to 7h, until a high-frequency of event-counts has occurred at 8h (dark/light red).

In between classes, a medium frequency of event-counts (light blue) was observed until 11h when the second class started. After the end of the third class at 16h, the same low frequency pattern was observed in the next couple of hours. The same pattern was also observed with the 30-minute (Figure 7(a)) and 2-hour intervals (Figure 7(c)).

Overall, it is important to point out that similar data distributions have been found throughout the duration of the experiment, confirming the suitability of using event-counts as a proxy measure to represent high and medium levels of occupancy presence in the lab classroom. However, there were always a lower frequency of number of event-counts occurring during the timeframes when the lab classroom was known to be unoccupied.

Looking at the types of triggered events that have generated such a low frequency of event-counts, we were able to identify three sensors that have predominantly triggered the events. They were the PIR motion, sound, and luminosity sensors. One hypothesis is that the projector and the computers might have contributed to triggering these events. Moreover, sensors can behave erratically at times. We have also observed that high temperature recordings were found in the classroom when there were no classes taking place. The daily evolution of the luminosity (daily cycle) or the entry of the cleaning staff in the classroom have also triggered events. Therefore, more research is needed to study the impact of the location of a sensor node has on false/true triggered-events.



(a) 30 minutes interval

(b) 1 hour interval

(c) 2 hours interval

Figure 7. Event-counts distribution during different time intervals

6.2 Model Performance

The Prophet model has a number of hyperparameters that were consider for tuning. They were:

- changepoint prior scale: This is probably the most influential parameter because it defines the trend flexibility. In other words, it determines how much a trend changes at the changepoints. The value of 0.05 has been used for the three time intervals.

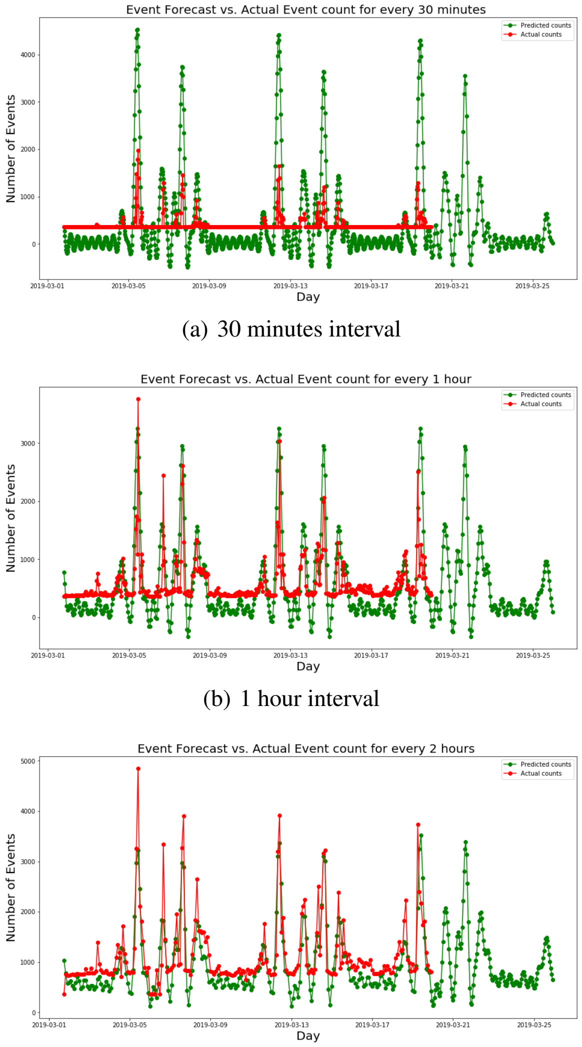
- seasonality prior scale: This parameter determines the seasonality trend, and it was set as 10 to avoid over-fitting.

- holidays prior scale: This controls flexibility to fit holiday effects. It was tuned to 10 for applying no regularization.

- seasonality mode: The option multiplicative was chosen because of the seasonal fluctuations (i.e. Tuesdays and Thursdays)

The hyperparameters that have not been tuned were growth (linear); changepoints (13 change points were used), yearly seasonality (off); weekly seasonality (on); holidays (specified holidays); interval width (30min, 1h, and 2h).

In Figures 8(a) and 8(b), the green line represents the observed y values, meanwhile the red line represents the forecast $\hat{y}$ values from the Prophet model. During the training using only 80% of the historical data, it is clear that the Prophet model had a superior fitting to high and medium observed y values rather than lower y values, specially when evaluating the 1h time interval with the 30 min interval. In addition, the forecasting of peaks was relatively more robust for the 1h interval time series.



(a) 30 minutes interval

(b) 1 hour interval

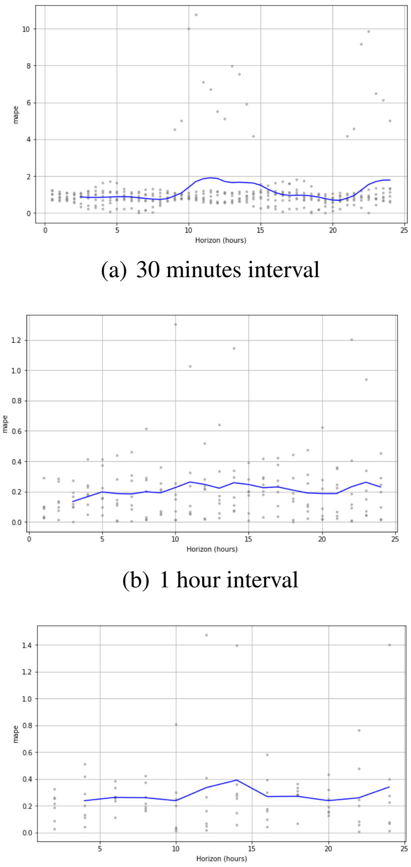
(c) 2 hours interval

Figure 8. Comparison between the observed y and forecast $\hat{y}$ values for a 30 min, 1h, and 2h time interval during training

In contrast, Figure 8(c) illustrates the over-fitting in the forecasts using the 2h interval due to the constant low frequency of events count, which have been probably triggered by the PIR motion, luminosity, and sound sensors. Our aim in generating a time series of event-counts and avoiding future selection when using non-intrusive sensing was analytically achieved. However, the results suggest that more historical data is needed for training the Prophet model in order to evaluate if the same patterns will persist when using the 2h interval.

6.3 Forecasting Accuracy versus Time Interval

The accuracy of the forecast $\hat{y}$ values using different time intervals was measured by the mean absolute percentage error (MAPE) for one day prediction (24 hours) in the future. In Figure 9, the grey dots show the absolute percent error that was computed for each forecast, and the blue lines represent the mean fitting curve that specifies the proportion of forecasts used in each rolling window.



(a) 30 minutes interval

(b) 1 hour interval

(c) 2 hours interval

Figure 9. The MAPE scores obtained for different time intervals

We can observe that for the 30min time interval, the absolute percent errors were around 0% to 2% as shown in Figure 9(a), which are significant lower than 5%, which is typical for predictions of one month into the future. In contrast, the errors fluctuated around 0.2 when using the 1h time interval, generating a smoother average curve as shown in Figure 9(b).

Furthermore, Figures 9(b) and 9(c) reveal how these time intervals have generated similar patterns. However, the fitting curve with 2 hour interval has show an increasing trend between 10h and 15h, revealing the less accurate predictions.

7 Conclusions and Future Work

The analytical workflow presented in this paper was implemented using an IoT architecture based on a sensor node that was continuously collecting raw triggered events from sensors including PIR motion, temperature, luminosity, CO_2, TVOC, sound, pressure, accelerometer, gyroscope, and humidity. This workflow is unique in devising a non-intrusive sensing strategy for occupancy forecasting, and its IoT architecture was designed to be a low-cost and scalable solution.

The four workflow tasks were developed to perform a variety of steps. The IoT sensing task was designed to run in an online mode, as it needs to handle the data flow from the sensors to the cloud. The ensuing tasks, such as data pre-processing, forecast modelling, and labelling, were executed in an offline mode in the cloud. The labelling task was the only task that was automated in the analytical workflow, but we expect that all tasks should be automated in the near future. More research is needed to develop stream-based data pre-processing methods.

The Prophet model was evaluated in depth to ensure over- or under-fitting did not occur. Overall, the forecasting achieved 80% accuracy when compared to the class schedule. In the future, more ground truth data is needed to validate the occupant behavior outside of scheduled class time. Moving forward, more data will be collected to cover the entire scholastic year.

The results have shown how event-triggered data can be used to understand occupancy presence in indoor spaces, simply by computing the seasonality of occupant behavior at different time intervals. The event-counts time series has demonstrated how event-counts are a powerful proxy for inferring occupancy using a learning model such as the Prophet forecasting model.

The applicability of this research is not contained only to a lab classroom, but is expected to accurately forecast occupancy of offices, halls, and corridors of a building. Different indoor spaces will require that different thresholds be set for triggering the events, as they depend on occupant behavior. Future research will focus on developing a Prophet forecasting model for different spatial granularities of indoor spaces.

8 Software and Data Availability

The Prophet library supporting this publication is published in R and Python package at https://facebook.github.io/prophet/. The used version is archived at https://github.com/facebook/prophet.

Research data supporting this publication is not available due to privacy concerns. Sample of synthetic data can be provided upon request.

The platform code for leveraging the Prophet forecasting model in this publication cannot be publicly shared due to current IP Agreement between UNB and Cisco System Canada.

Acknowledgements. This work was supported by the NSERC/Cisco Industrial Research Chair, Grant IRCPJ 488403-1. We would like to thank Compute Canada for providing us with the cloud resources.

Author contributions. The authors contributed equally to this work. All authors have read and agreed to the published version of this manuscript.

References

Alawadi, S., Mera, D., Fernandez-Delgado, M., Alkhabbas, F., Olsson, C. M., and Davidsson, P.: A comparison of machine learning algorithms for forecasting indoor temperature in smart buildings, Energy Systems, Springer Verlag, 2020.

Bai, L., Ciravegna, F., Bond, R., and Mulvenna, M.: A Low Cost Indoor Positioning System Using Bluetooth Low Energy, IEEE Access, 8, 136 858–136 871, 2020.

Chen, Z., Jiang, C., and Xie, L.: Building occupancy estimation and detection: A review, Energy and Buildings, 169, 260–270, 2018.

Chikkakrishna, N. K., Hardik, C., Deepika, K., and Sparsha, N.: Short-Term Traffic Prediction Using Sarima and FbPROPHET, in: 2019 IEEE 16th India Council International Conference (INDICON), pp. 1–4, IEEE, 2019.

Hobson, B. W., Lowcay, D., Gunay, H. B., Ashouri, A., and Newsham, G. R.: Opportunistic occupancy-count estimation using sensor fusion: A case study, Building and environment, 159, 106 154, 2019.

Hutchins, J., Ihler, A., and Smyth, P.: Modeling count data from multiple sensors: a building occupancy model, in: 2007 2nd IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing, pp. 241–244, IEEE, 2007.

Jia, M., Srinivasan, R. S., and Raheem, A. A.: From occupancy to occupant behavior: An analytical survey of data acquisition technologies, modeling methodologies and simulation coupling mechanisms for building energy efficiency, Renewable and Sustainable Energy Reviews, 68, 525–540, 2017.

Laput, G., Zhang, Y., and Harrison, C.: Synthetic sensors: Towards general-purpose sensing, in: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, pp. 3986–3999, ACM, 2017.

Mok, E. and Retscher, G.: Location determination using WiFi fingerprinting versus WiFi trilateration, Journal of Location Based Services, 1, 145–159, 2007.

Raschka, S.: Predictive modeling, supervised machine learning, and pattern classification, https://sebastianraschka.com/Articles/2014_intro_supervised_learning.html#/-the-big-picture, [Online; Accessed on 2020-04-28], 2014.

Rohei, M. S., Ahmad, N. B., Salwana, E., and Kakar, A. S.: Design and Testing of an Epidermal RFID Mechanism in A Smart Indoor Human Tracking System, IEEE Sensors Journal, 2020.

Rueda, L., Agbossou, K., Cardenas, A., Henao, N., and Kelouwani, S.: A comprehensive review of approaches to building occupancy detection, Building and Environment, p. 106966, 2020.

Ryu, S. H. and Moon, H. J.: Development of an occupancy prediction model using indoor environmental data based on machine learning techniques, Building and Environment, 107, 1–9, 2016.

Saha, H., Florita, A. R., Henze, G. P., and Sarkar, S.: Occupancy sensing in buildings: A review of data analytics approaches, Energy and Buildings, 188, 278–285, 2019.

Samal, K. K. R., Babu, K. S., Das, S. K., and Acharaya, A.: Time Series based Air Pollution Forecasting using SARIMA and Prophet Model, in: Proceedings of the 2019 International Conference on Information Technology and Computer Communications, pp. 80–85, 2019.

Syntetos, A. A., Babai, M. Z., Lengu, D., and Altay, N.: Distributional assumptions for parametric forecasting of intermittent demand, pp. 31–52, 2011.

Taylor, S. J. and Letham, B.: Forecasting at scale, The American Statistician, 72, 37–45, 2018.

Trivedi, D. and Badarla, V.: Occupancy detection systems for indoor environments: A survey of approaches and methods, Indoor and Built Environment, 29, 1053–1069, 2020.

Zhang, W., Yu, K., Wang, W., and Li, X.: A Self-Adaptive AP Selection Algorithm Based on Multi-Objective Optimization for Indoor WiFi Positioning, IEEE Internet of Things Journal, 2020.

Zou, H., Jiang, H., Yang, J., Xie, L., and Spanos, C.: Non-intrusive occupancy sensing in commercial buildings, Energy and Buildings, 154, 633–643, 2017.