Notes paper - A data-driven and scalable approach for window operation detection in multi-family residential buildings

Juliet Nwagwu Ume-Ezeoke, Kopal Nihar, Catehrine Gorle, Rishee Jain

June 13, 2023

1 Abstract

Natural cooling, utilizing non-mechanical cooling, presents a low-carbon and low-cost way to provide thermal comfort in residential buildings. However, naturally cooled residential buildings are difficult to design without a clear understanding of the effect opening and closing windows will have on occupant thermal comfort. Predicting when and understanding why occupants open windows is a difficult task in itself. Although work has been done on creating predictive window opening and closing models, they traditionally rely on large datasets, specialized sensing, and/or require training data for each building. This makes them ineffective tools for scaling natural cooling to a large number of buildings. Here, we propose a novel unsupervised window opening and closing detection method that utilizes easy-to-deploy off-the-shelf temperature and humidity sensors. We assess our proposed models' effectiveness on an empirical dataset. We also compare the method's performance with a support vector machine (SVM) model that represents the state-of-the-art in window detection. We find that the proposed method outperformed the SVM on a number of key indicators, but had subpar performance on data in which the indoor and outdoor temperature had small differences. While the SVM's performance was sensitive to the characteristics of the time series features, our proposed method, which relies on indoor temperature alone, demonstrated robust performance in our pilot studies, making it a potential candidate for developing a highly scalable and generalizable model for window operation detection. Overall, this work aims to demonstrate the viability of using unsupervised data-driven methods for understanding window operations in residential buildings. In doing so, we aim to enable more accurate modeling of naturally cooled buildings and further catalyze widespread adoption of this low-cost and low-carbon technology.

2 Introduction

3 Literature Review

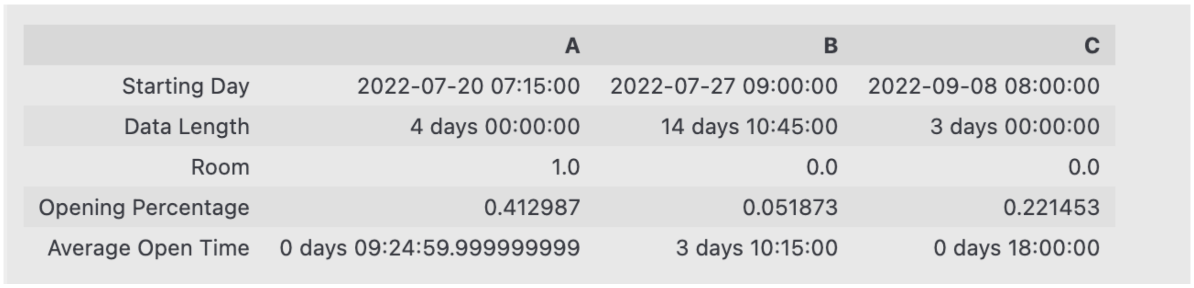
4 Methods

TODO brief overview here...

4.1 Data Collection

We collected data for short periods of time over three months in the summer. We placed HOBO sensors **(sensor name) temperature/ relative humidity sensors in two adjacent rooms in a multi-family residential building. The opening and closing schedules were not determined beforehand but were allowed to proceed naturally. This created three distinct scenarios for testing the efficacy of the model we developed.

Descriptions of the data are provided in **?@tbl-data-collected**. The experiments are labeled as experiments A, B, and C, corresponding to data recorded at the beginning on July 20, July 27, and September 8 respectively. Experiment B was much longer than the other experiments, about 14 days vs experiments A and C were collected over 3 and 4 days each. Experiment B also had a much shorter period of the window being open, with the window open around 5 percent of the time, vs experiments A and C which had the window open for at least 20 percent of the time. [1] [2]



	A	B	C
Starting Day	2022-07-20 07:15:00	2022-07-27 09:00:00	2022-09-08 08:00:00
Data Length	4 days 00:00:00	14 days 10:45:00	3 days 00:00:00
Room	1.0	0.0	0.0
Opening Percentage	0.412987	0.051873	0.221453
Average Open Time	0 days 09:24:59.999999999	3 days 10:15:00	0 days 18:00:00

Figure 1: Data Collected

To develop an understanding of how temperature and relative humidity, the quantities that were measured, changed across the different experiments, we examined the distributions of the data in Figure 2. We observe that experiment B had the widest distribution of internal temperatures. Experiment C had internal temperatures that were higher and closer to the

[1]TODO not sure if really need to discuss the average opening time...

[2]*TODO: check this => i think opening percentage for b should be much higher?? Or maybe include plots of the real data*

mean/median** of the external temperatures. This was due to observed higher temperatures during this time period, which makes for a particularly interesting edge case. The internal relative humidity was higher than the ambient relative humidity across experiments.

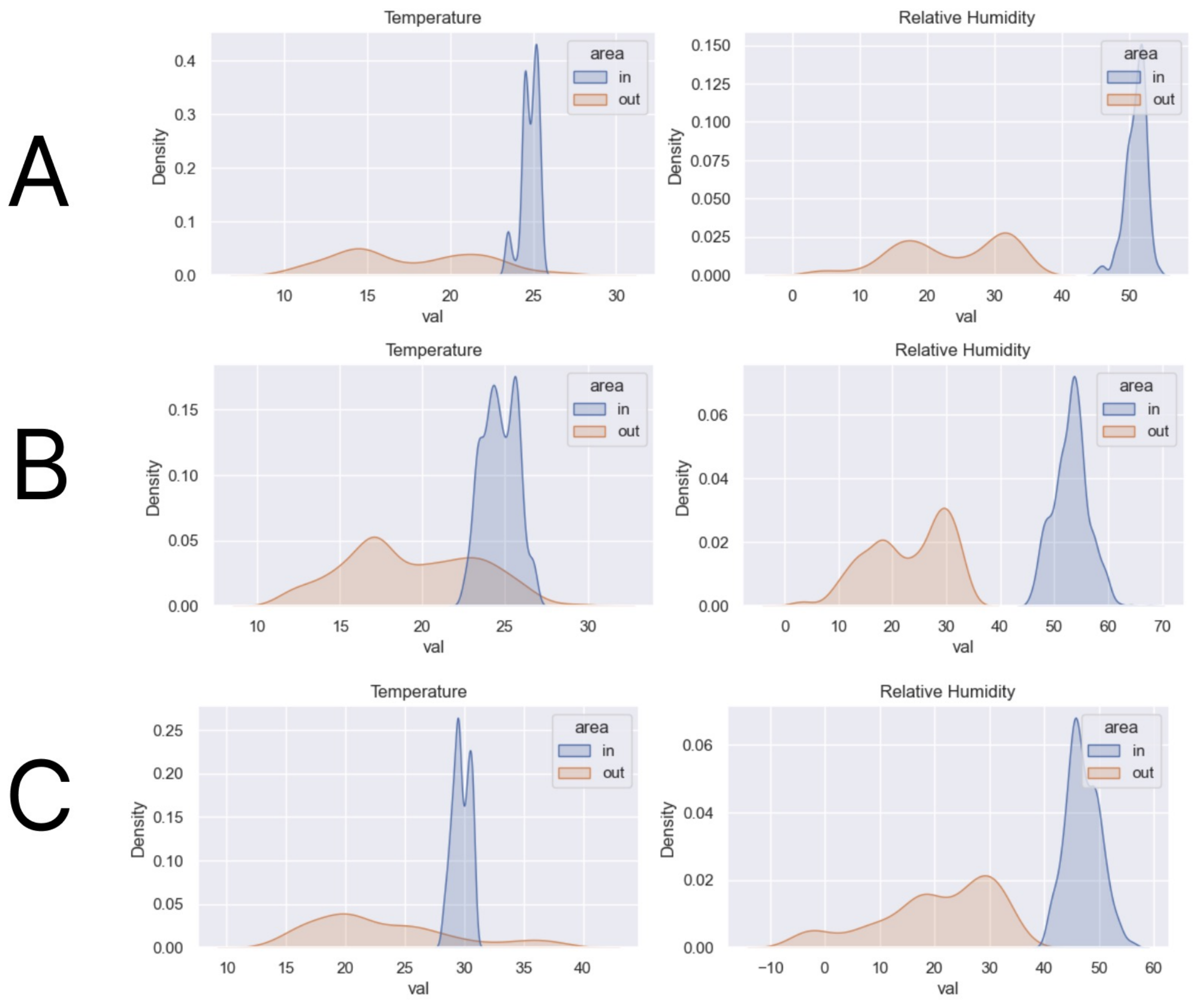


Figure 2: Distributions of Temperature and Relative Humididty Across Experiments

4.2 Detection Methods

some intro here [3]

[3]TODO: include this part: We anticipate that creating a physical model to more explicitly determine the response of the indoor temperature based on the ambient temperature and the building makeup could be an interesting approach to capture the "truth", or at least approximate it Also need some discussion about the emphasis identifying where the window state changes...

4.2.1 Smoothing Technique

Why we chose to develop this method

Our approach relies on an intuitive understanding of window operation detection. We expect a typical time series recording of a quantity of interest, such as temperature or relative humidity, to contain information that reflects the seasonality of ambient quantities, noise due to occurrences within the space where measurements are being taken, and the desired signal of due to a change in the window state. The way in which these three components of a measurement are combined is unknown, and, the noise component in particular cannot be known based on the measurements we have collected. Therefore, our task is to attempt to extricate the seasonality that is due to ambient processes from the measured data, which will ideally reveal the window detection and a bit of noise.

To remove the seasonality from the measured data, we examined three methods of deseasonalizing: using seasonal-trend-decomposition, using a sinusoidal fit, and using an exponentially weighted mean function. These all reflect a trade-off between capturing pure seasonality and capturing a bit of noise. The sinusoidal method, which is optimized for the indoor temperature time series, purely reflects seasonality whereas the exponentially weighted mean function is simply a moving average of the indoor temperature signal. Therefore, some noise is captured as well. The seasonal component of a seasonal trend decomposition is somewhere in the middle. We found that the exponentially weighted mean function performed the best in isolating window detection, which follows from our intuition that both the seasonal components and noise need to be isolated. [4]

How this method works

The smoothing technique proceeds as follows. The goal is to identify $W(t)$, window state as a function of time. This can take on two values: 0, representing window closed, or 1, representing window open. We have an observed variable $T(t)$, which represents the measurement of the indoor temperature. We apply an exponentially weighted mean function to the observed temperature, creating a smoothed time series, $\overline{T(t)}$, which in theory removes strong peaks that would reflect changes in window state, and isolates information concerning the seasonal response and additional noise [5]. Subtracting $\overline{T(t)}$ from $T(t)$ yields $T'(t)$, which is a time series that reflects changes in the window state and some additional noise. In order to more confidently identify the where the changes in window state occur, we examine the first and second derivatives of $T'(t)$, $\frac{dT'(t)}{dt}$ and $\frac{d^2T'(t)}{dt^2}$. Observing the second derivative in particular, is an effective way to identify change points. In order to predict where window changes occur, we apply a hypothesis test[6]. We assume that the time series $\frac{d^2T'(t)}{dt^2}$ is normally distributed.

[4] TODO: i think the previous 2 paragraphs can be condensed or removed entirely

[5] this assumes that the ewm function is perfectly able to capture seasonality, which in reality it isn't. . . => need to write down assumptions => that the instantaneous change in indoor temp due to window opening is greater than any other potential source of instantaneous temperature change.

[6] is hypothesis test the right word?, should i use vocabulary like the null hypothesis and the alternative hypothesis??

Therefore, any value in $\frac{d^2T'(t)}{dt^2}$ that is more than 2 standard deviations away from the mean is unlikely to occur, and could possibly indicate an instance of a change in window state. We will call these initial guesses $G(t)$. They take on positive or negative values depending on whether they are predicting a transition from window open to close, or window close to open. Therefore, we round the values of $G(t)$ to 0 or 1 to reflect this. Finally, we interpolate between the rounded values of $G(t)$, so that we only predict a change in window state when $G(t)$ transitions between 0 and 1. This prediction of the window state is called $I(t)$. We have included **?@fig-smoothing-tech** to show how the smoothing technique for window detection works.

[Smoothing Technique in Action]{#fig-smoothing-tech}

4.2.2 Machine Learning Method

Why we chose to use this method

We used a separate method in order to compare the efficacy of the smoothing technique that we developed. We chose to use an SVM, as it has been shown in literature (de Rautlin de Roy) to have robust performance across a wide range of features. SVMs are straightforward to implement with relatively few hyper-parameters, as compared to other methods that have been recently shown to have high performance on the window detection problem. [7]

How this method works

SVMs extend simple linear[8] regressions. In a similar way to a linear regression, SVMs approximate a line of best fit using the features and labels provided [9]. However, with the use of kernel functions that can map high dimensional feature spaces to ones of lower dimensions, SVMs also effectively create hyperplanes of best fit in order to classify datasets. The use of a kernel function also enables the introduction of mappings that may be a better fit for a given data set.

How we implemented it

Our focus was trying to get the best performance from the SVM using an optimal data set with optimal pre-processing functions applied[10]. We therefore came up with combinations of the various features we had access to, as well as their derivatives and differences from one another. The base features were: ambient temperature, measured temperature, the difference between ambient temperature and measured temperature, the difference between measured temperature and the derivative of measured temperature. We also had the same features for

[7]SVMs also have an unsupervised implementation, not sure if this exists for other ML models…

[8]could also say logistic regression to reinforce the SVM logistic regression similarity

[9]https://developers.google.com/machine-learning/glossary#f

[10]better way to say this?

relative humidity. After creating combinations of these base features, we had a test set of 113[11] combinations as shown in **?@fig-feature-combos**.

[Combinations of Features for SVM]{#fig-feature-combos}

For each data set we collected, we created an SVM for each of the combinations in the test set, which resulted in 3x113 [12]. We chose not to perform hyperparameter tuning for the SVMs, and used the default parameters defined by sklearn [13] which consists of a radial basis function kernel. As we are interested in the developing an unsupervised detection method, we used the One-Class SVM implementation also from sklearn, which makes decisions by essentially using clustering to create the hyperplanes for classification. [14]

How it compares to the previous method

The SVM approach differs from the smoothing technique that we have developed, and is similar to other window techniques in that it is a highly generalizable method that is not developed with window detection in mind. A wide array of input features can be tested in order to get an acceptable prediction accuracy. However, this might not be acceptable in practice, as different scenarios/ window operation behaviors, will demand different sets of input features. In the event the true window detection pattern is unknown, it will be difficult to know which input features are giving a trustworthy resresulttult.

4.3 Evaluation Metrics

We used 3 sets of metics to evaluate the methods we chose. The first two follow from a recent paper that compared the efficacy of different machine learning algorithms[15] , and the final set of metrics were developed to more closely examine specific window detection behavior.

We chose to use this many diverse metrics because…

4.3.1 Standard Metrics

Macro average F-1 score: The F-1 score is a classic metric for evaluating the performance of classification algorithms. It it condenses information about the precision of a model in predicting a certain class, as well as its ability to recall the the available data. The macro averaged F-1 score averages the F-1 scores for all individual classes, but does not introduce weights in the averaging to reflect that the classes may be unbalanced. As shown in **?@tbl-data-collected**, the data sets we have very from fairly balanced in experiment A, to highly unbalanced in experiment B. Therefore, using the macro average F-1 score provides a worst case

[11]program this…
[12]program this…
[13]need to cite this and include version.., and include the other default settings.
[14]need to double check this…
[15]dRdR

performance. Using this F1-score also provides a basis of comparison to [16] . The implementation for computing this metric comes from sci kit learns classification report. [17]

4.3.2 De Rautlin de Roy Metrics

The following metrics were introduced by [18], and represent novel window detection specific algorithms.

Opening Accuracy: The opening accuracy gives a score to each opening instance predicted by the window detection algorithm. An opening instance is an interval on t, such as $t_{nk} = t_n, t_{n+1}, \ldots, t_k$, where $I(t_{nk}) = 1$. An opening receives a score of 0 if it does not correspond to a a true opening: $I(t_{nk}) = 1, W(t_{nk} = 0)$, and a score of 1 it perfectly corresponds to a true opening: $I(t_{nk}) = 1, W(t_{nk} = 1)$. The score for a given predicted opening interval decreases by a penalty of 0.33 for each time step that does not align to a true opening. Therefore, a predicted opening interval $I(t_{nk}) = 1$ that aligns a true opening interval can have a score of not less than 0 if it extends for more than two times [19] steps away from a true value. The value for each opening interval are averaged to get the opening accuracy for a given instance of a model. An unbounded metric is also computed when the penalty applied to a given interval's score does not stop at 0 but is allowed to further decrement. This provides an unbounded opening accuracy that gives a clearer indication of how "off" models are, given that many models we examining actually had a bounded opening accuracy of 0.

True/False Opening Time: This true opening time reflects the total amount of time when $I(t) = W(t) = 1$. The false opening time is when $I(t) = 1, W(t) = 0$. The metric is somewhat similar to the F-1 score in that rather than looking at specifically at change points, it provides information the entire time period.

4.3.3 Custom Metrics

We developed the final set of metrics based on the intuition that if a model is perfectly able to capture the times when a window state changes, then it is performing extremely well. We classify a "guess", as any value of $I(t)$, and an "action" as any value of $W(t)$. A "hit" occurs when $I(t) = W(t)$ Like [20] we consider a prediction accurate if it is withing at most 2 timestamps of the true occurrence. Therefore, a "near hit" occurs when $I(t) = W(t \pm 1)$. The metrics we examine are given below.

Hits + Near Hits/ Guesses: This metric accounts for the variability in guesses. A ratio of 1 indicates that all guesses taken by the model were accurate within two timestamps. A ratio

[16]dRdR
[17]include equation for F1 score
[18]dRdR
[19]check drdr math/logic here
[20]dRdR

close to 0 indicates that a lot of guesses were taken, but relatively few were close to where change occurred in the true window state.

Guesses/Actions: This metric implicitly accounts for the ability of the model to capture the pattern of the changes in window state. If the true window state changed only 10 times, but the model predicts changes on the order of 100, then the model is not performing well. A perfect score is 1.

5 Results

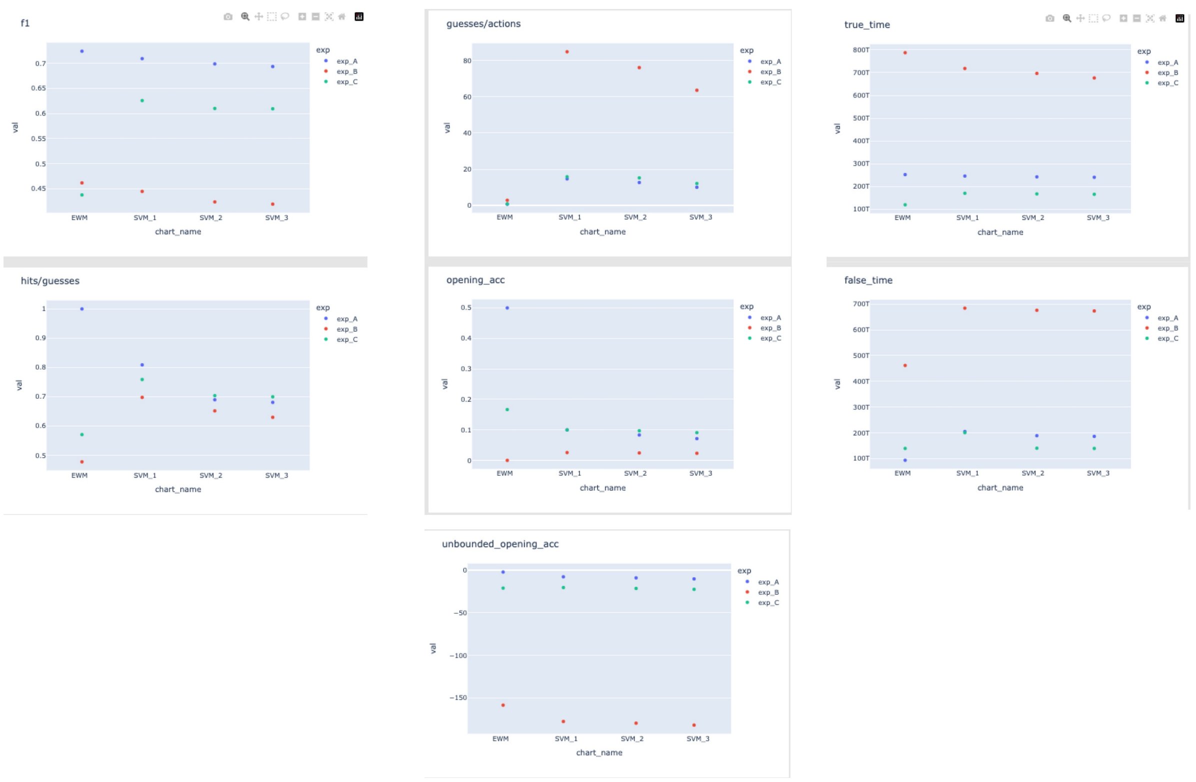


Figure 3: Results. . .

6 Conclusions

Windows presentation