Sensor Type Classification in Buildings Through Time-Series Data

Dezhi Hong[1], Jorge Ortiz[2], Kamin Whitehouse[1], Bong-Jun Ko[2]
[1]Department of Computer Science, University of Virginia, USA
[2]IBM Research, Yorktown Heights, NY USA
{hong, whitehouse}@virginia.edu, {jjortiz, bko}@us.ibm.com

ABSTRACT

TBD

Categories and Subject Descriptors

C.3 [**Special-Purpose And Application-Based Systems**]: Real-time and embedded systems

General Terms

Performance, Experimentation, Verification

Keywords

Building, Sensor Type, Random Forest, Classification

1. INTRODUCTION

Buildings contribute to a large portion of the energy bill in the US. Buildings are instrumented with sensors to monitor and improve their performance. The metadata of sensors are usually poorly maintained. We particularly delve into the type information which is useful for a lot of applications such as xx.

We conduct an empirical analysis on the data collected from 15 sensors in 5 rooms over a one-month period. Our study makes the following contributions:

- We propose a simple, general yet effective feature extraction scheme to achieve sensor type classification in the context of commercial buildings.
- We formulate an approach to identifying potential misclassified sensor streams (in terms of the type classes) when no ground truth labels are available.
- We evaluate our classification technique using data from over 2000 sensor series of 6 types in two buildings on two campuses, and our technique is able to achieve xx%-xx% accuracy in intra building scenario and yy%-yy% accuracy in inter building scenario.
- We also evaluate our solution to misclassification identification and the results demonstrate that we are able to identy zz% percent of the potential population.

The rest of the papers is organzied as follows: Section 2 motivates our work. Section 3 gives an overview of the feature extraction design, classification detail as well as misclassification identification. Section 4 presents the results of our evaluation on the proposed techinuqes. Section 5 discusses the limits of our work and some possible future directions. We conclude the paper with Section 6 & 7.

2. MOTIVATION

3. METHODOLOGY

We first describe how we design the feature extraction and explain why the feature set works. Then we discuss the classification technique we adopt as well as detail the training and testing process. In the end, we articulate a solution to identifying potential misclassification when no ground truth labels for sensor types are available.

3.1 Feature Extraction

Raw sensor time series usually contain millions of readings which are too general to be informative for classification tasks. We need to distill the information embedded in the sensor readings in the time domain. A signal in the time domain trends the amplitude of a sensor reading and intuitively, different types of sensor would in general occupy distinct amplitude bins as demonstrated in Figure []. As to characterize the amplitude distribution of a signal in the time domain, we can use percentiles in the readings, such as 25%, 50%, 75% and so forth. Considering that there also exist outlieres in sensor readings, we pick the 50% (also known as the median number) for use as a discriminator, which is more robust to outliers. However, on another side, sensor readings are subjective to the dynamics in the placed surroudings therefore sensors of different types can collide in a same amplitude bin. For example, during a rainy season, the humidity in an office can reach the range of 70 80 which is the same as typical temeprature sensor readings. Therefore, simply relying on reading amplitude might not be able to effetively differentiate different sensors. Figure [] show some of such examples. To capture these short term "events" as features, we also need to include the variance of the signals when formulating a feature vector for the time series.

When extracting features from a raw sensor reading, the original trace can span over days, weeks or even months,

BuildSys'13, November 14–15, 2013, Rome, Italy
Copyright 2013 ACM 978-1-4503-2431-1/13/11 ...$15.00.

and the trends can vary a lot even from day to day. Therefore extracting the certain features such as percentiles and variances from the entire sensor readings might make the features less discriminant, compared to doing so in shorter windowed time slices. But computing features over windowed slices might end up producing too many elements for a feature vector and also, having too many unnecessary feature variables might deteriorate the performance of classifier. To better summarize the dynamics of sensor traces, we apply feature extraction to every X-minute long window and compute the statistics of the accumulated features from windowed slices as the final features.

As a summary, the feature extraction procedure goes as follows. First, each single sensor signal is segmented into N non-overlapping X-minute long windows (we will discuss the decision of window length X in later section). Second, within each time window, we compute the median and variance of the signal, obtaining a vector of medians and a vector of variances after the window slides over the entire traces:

$$MED = \{median^1, median^2, ..., median^N\}$$

$$VAR = \{variance^1, variance^2, ..., variance^N\}$$

Where N is the number of time windows. The vector MED and VAR reflect short term changes but not all the intermidiate values are essentially helpful for classification. So as a statistical summary of the two vectors, as a last step, for each vector we compute the minimun, maximun, median and variance, resulting in a feature vector of eight elements:

$$F = \{min(MED), max(MED), median(MED), var(MED), \\ min(VAR), max(VAR), median(VAR), var(VAR)\}$$

And F is the feature vector for each sensor trace in our classification task.

3.2 Classification

After transforming all sensor time series into feature vectors, we leverage an ensemble classifier–random forest–as our solution to achieving the type classification task. In general, random forest [?] outperforms a single tree classifier by growing a bunch of classification trees. To classify a new coming object as a feature vector, feed the vector down each of the trees in the forest. Each tree gives a classification, in other words, the tree "votes" for that class. The forest chooses the class having the most votes over all the trees in the forest. As a quick overview of how each tree is grown in the forest, the process goes as follows:

1. Sample N instances at random with replacement, from the original data set. These samples will be the training set for growing this particular tree.
2. Specify M feature variables at random out of the total feature vector when growing each node of a tree. And the best split on these M is used to split the node. The value of M is constant during the forest growing.
3. Each tree is grown to the largest extent possible without pruning.

Specifically, we set N equal the number of instances in the original training set, M equal the square root of the number of original features and the number of the trees in the forest be 50. Usually these parameters are optimized through cross-validation and we refer interested readers to [?] for further deduction and proof related to random forest.

3.3 Identifying Misclassification

It is an easy job to identify misclassification when we have groud truth labels, but in many contexts motivating our solution the ground truth labels are not available, therefore, the identification of potential misclassification would suffer from the absence of ground truth. To identify the potential misclassified instances in our job, we leverage the ensemble of classifiers and make use of a probability-based approach.

4. EVALUATION

To demonstrate the effectiveness as well as usefulness of methodology, we evaluate the technique in two different scenarios: a) intra building, that is, the training and testing data for classification is taken within the same building, and b) inter buildings, where training and testing resources are from two distinct buildings. We also show an application as a case study built based on the generated type information in two buildings, which would be impossible to achieve in the absence of categorical metadata.

4.1 Taxonomy

In this paper, we consider 6 types of sensors, which are CO_2, humidity, room temperature, setpoint, air flow volume, other temperature. Specifically, room temperature includes only sensors that measure the air temperature of rooms and other temperatures covers all other temperature measurements such as supply air/return air/mixed air temperature and supply/return water temperature. We also put only one general type for setpoint which includes all types of setpoints instrumented in the buildings.

4.2 Experimental Setup

Table 1: Number of Each Sensor Type

Type	SDH	Rice
CO_2	1	2
humidity	2	4
room temp.	3	2
setpoint	4	7
air volume	5	5
other temp.	6	5

4.3 Baseline and Metrics

As a baseline, after we generate the two distributions described previously, we apply multidimensional scaling (MDS) to the corrcoeff matrix, in order to transform the original high-dimensional relative space to a 3-D space with an absolute origin, and run the k-means clustering algorithm. We choose the true-positive rate (TPR, also known as recall rate) and false-positive rate (FPR) as metrics to evaluate the performance of our method versus the naive approach, which correlates the raw traces. A true-positive (TP) is when a sensor pair in a room is classified as being co-located while a false-positive (FP) is when a sensor that is not in room is classified as being so.

4.4 Intra Building Performance

4.5 Inter Building Performance

4.6 Window Length Sensitivity and Training Bootstraping Analysis

4.7 Potential Misclassification

4.8 Case Study

5. DISCUSSION

5.1 Improvement on Classification Accuracy

We could further explore how using external or domain-specific knowledge would help improve the classification accuracy. For example, if we know the sun sets around 6 PM and observe a trough in the reading of some streams,

5.2 Extension of Taxonamy and Class Scope

We could extend the class scope to include more sensor types because there are more types than the most common ones included in this paper, e.g., alarm sensor, xxx. Meanwhile, we also want to build a more complicated taxonamy for some types, for instance, ”setpoints”, because there are setpoints for different parameters.

6. RELATED WORK

There has been much research work on sensor stream clustering and trace analysis. Chen and Tu [2] investigate how to cluster data streams in real-time using a density-based approach with a two-tiered framework. The first tier captures the dynamics of a data stream with a density decaying technique and then maps it to a grid. The second tier computes a grid density based on how it clusters the grid. Their approach differs from ours in that they focus on decreasing algorithm complexity for real-time sensor stream clustering. We run our analysis on historical traces and use correlation analysis in our clustering algorithm.

Kapitanova et al. [8] describe a technique to monitor sensor operations in the home and identify sensor failures. The classifier is trained on historical sensor data to obtain the relationship between sensors, assuming the number and location of sensors is known. When a failure or removal of a sensor occurs, the classifier's behavior deviates and the event is captured. Our method does not require any prior knowledge and instead tries to cluster feeds to discover their relative placement.

Lu and Whitehouse [10] formulate a new algorithm, particularly leveraging the semantic constraints interpreted from sensor data to determine sensor locations. The algorithm identifies how many rooms are present using motion sensors and determines room position based on physical constraints. Finally, it maps each sensor into the associated room. Our efforts focus on using intrinsic patterns typically pre-existing in building system sensor feeds to uncover physical relationships.

Fontugne et al. [4] propose a new method to decompose sensor signals with EMD. They extract the intrinsic usage pattern from the raw traces and show that sensors close to each other have higher intrinsic correlation. However, they do not explore the observation more deeply by answering whether there is a statistically discoverable boundary between sensor clusters in different rooms, or if there is a uniform threshold in the correlation coefficients able to be generalized to different rooms.

Fontugne et al. [5] carry on the work and propose an unsupervised method to monitor sensor behavior in buildings. They constructed a reference model out of the underlying pattens, obtained with EMD, and use it to compare future activity against it. They report an anomaly whenever a device deviates from the reference. This work exploits EMD as a method to detrend the signals and capture the inter-device relationships.

Much work utilizes EMD on medical data [1], speech analysis [6], image processing [12] and climate analysis [9]. Our method adopts EMD to determine whether a discoverable statistical boundary exists in sensors traces from sensors in different rooms and whether such a boundary can be generalized across rooms with various kinds of sensors.

7. CONCLUSION

8. REFERENCES

[1] A. Arafat and T. Hasan. Automatic detection of ECG wave boundaries using empirical mode decomposition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009.

[2] Y. Chen and L. Tu. Density-based clustering for real-time stream data. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, 2007.

[3] Department of Energy. 2011 Buildings Energy Data Book. http://buildingsdatabook.eren.doe.gov/.

[4] R. Fontugne, J. Ortiz, D. Culler, and H. Esaki. Empirical mode decomposition for intrinsic-relationship extraction in large sensor deployments. In *Workshop on Internet of Things Applications*, IoT-App'12, 2012.

[5] R. Fontugne, J. Ortiz, N. Tremblay, P. Borgnat, P. Flandrin, K. Fukuda, D. Culler, and H. Esaki. Strip, bind, and search: a method for identifying abnormal energy consumption in buildings. In *Proceedings of the 12th international conference on Information processing in sensor networks*, IPSN '13, 2013.

[6] H. Huang and J. Pan. Speech pitch determination based on Hilbert-Huang transform. *Signal Processing*, 86(4):792 – 803, 2006.

[7] N. E. Huang, Z. Shen, S. R. Long, M. C. Wu, H. H. Shih, Q. Zheng, N. C. Yen, C. C. Tung, and H. H. Liu. The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 1998.

[8] K. Kapitanova, E. Hoque, J. A. Stankovic, K. Whitehouse, and S. H. Son. Being SMART about failures: assessing repairs in SMART homes. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, 2012.

[9] T. Lee and T. B. M. J. Ouarda. Prediction of climate nonstationary oscillation processes with empirical mode decomposition. *Journal of Geophysical Research: Atmospheres*, 116(D6), 2011.

[10] J. Lu and K. Whitehouse. Smart blueprints: automatically generated maps of homes and the devices within them. In *Proceedings of the 10th*

international conference on Pervasive Computing, Pervasive'12, 2012.

[11] Y. Ma and F. Borrelli. Fast stochastic predictive control for building temperature regulation. In *American Control Conference (ACC), 2012*, 2012.

[12] H. Mohammadzade, F. Agrafioti, J. Gao, and D. Hatzinakos. BEMD for expression transformation in face recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011.