Advanced Engineering Informatics 37 (2018) 14–30

Contents lists available at ScienceDirect

Advanced Engineering Informatics

journal homepage: www.elsevier.com/locate/aei

Full length article

A large-scale evaluation of automated metadata inference approaches on sensors from air handling units

Jingkun Gao*, Mario Bergés

Dept. of Civil and Environmental Engineering, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, United States

ARTICLE INFO

Keywords:
Sensor metadata
Building automation system
FDD application

ABSTRACT

Building automation systems provide abundant sensor data to enable the potential of using data analytics to, among other things, improve the energy efficiency of the building. However, deployment of these applications for buildings, such as, fault detection and diagnosis (FDD) on multiple buildings remains a challenge due to the non-trivial efforts of organizing, managing and extracting metadata associated with sensors (e.g., information about their location, function, etc.), which is required by applications. One of the reasons leading to the problem is that varying conventions, acronyms, and standards are used to define this metadata. To better understand the nature of the problem, as well as the performance and scalability of existing solutions, we implement and test 6 different time-series based metadata inference approaches on sensors from 614 air handling units (AHU) instrumented in 35 building sites accounting for more than 400 buildings distributed across United States of America. We infer 12 types of sensors and actuators in AHUs required by a rule-based FDD application: AHU performance and assessment rules (APAR). Our results show that: (1) the average performance of these approaches in terms of accuracy is similar across building sites, though there is significant variance; (2) the expected accuracy of classifying the type of points required by APAR for a new unseen building is, on average, 75%; (3) the performance of the model does not decrease as long as training data and testing data are extracted from adjacent months.

1. Introduction

Many software applications that leverage building automation systems (BASs), such as energy management, fault detection and diagnosis (FDD), fire detection, building operation monitoring and performance improvement [1–5], require time-series records from a variety of sensors in buildings. These time-series records are typically stored in a building automation system (BAS), along with a unique identifier and additional metadata that describes, explains, locates, or contextualizes the sensors and actuators that generate them. However, this metadata does not generally follow a consistent convention across buildings and, as a consequence, significant effort is spent by facility managers and other building stakeholders in order to interpret and understand it [6–8]. In this paper, we investigate the possibility of leveraging statistical information contained in these time-series records in order to directly and automatically identify the type of sensor that generated them.

To illustrate the problem, Table 1 shows the metadata for seven sensors measuring the same physical phenomena in two different buildings. The metadata for these sensors is represented by a text field (or tag, as can be seen in column one and two of Table 1) which encodes the type of the sensor, the equipment this sensor is associated with, and the location of this sensor including the floor and the building name. As different acronyms and conventions are being used in defining this metadata, it is easy to see how there could be challenges for building managers to retrieve the needed time-series sensor values. For example, when building managers need to retrieve the supply air temperature set point from an air handling unit (AHU) in Building 2, they need to know that “SAS” represents the sensor they are looking for. However, building managers might not be able to figure out which sensor is the one they want by simply looking at this metadata, or even having some pre-existing knowledge of the convention. They might need to reach out to the contractor who set up these sensors and named them in the system, or make a reasonable guess based on their past experience combined with information from design prints and software interface, which could still be incorrect. Hence, the cost of gathering and preparing the required inputs for building applications is inevitably increased due to the inconsistent names of sensing and actuation points in buildings.

Using standard metadata schemas could help to reduce this cost, as

* Corresponding author.
E-mail addresses: jingkun@cmu.edu (J. Gao), marioberges@cmu.edu (M. Bergés).

https://doi.org/10.1016/j.aei.2018.04.010
Received 29 August 2017; Received in revised form 1 February 2018; Accepted 18 April 2018
Available online 01 May 2018

Table 1
Defined tag names of sensors and set points in buildings. The last column shows the measured object or phenomena.

Building 1	Building 2	measured object or phenomenon
N2-1.EN2.AHU-2.PH-VLV	MI.AHU.3FL.011.HCO	hot water pre-heating valve
N2-1.EN2.AHU-2.CLG-VLV	MI.AHU.3FL.011.CCO	chilled water cooling valve
N2-1.EN2.AHU-2.MA-T	MI.AHU.3FL.011.MAT	mixed air temperature
N2-1.EN2.AHU-2.OA-T	MI.AHU.3FL.011.OAC	outside air temperature
N2-1.EN2.AHU-2.RA-T	MI.AHU.3FL.011.RAT	return air temperature
N2-1.EN2.AHU-2.DA-T	MI.AHU.3FL.011.SAT	supply air temperature
N2-1.EN2.AHU-2.DAT-SP	MI.AHU.3FL.011.SAS	supply air temperature set point

such inconsistency is mainly due to the lack of a commonly agreed-on schema to standardize how metadata should be defined when equipment and devices are initially being set up in buildings. Therefore, previous researchers have proposed different metadata schemas that provide a formal way to define the metadata in building systems (e.g., [9,10,8,11,12]). These schemas can benefit new buildings if different vendors agree on the same convention when setting up a new BAS. With all new buildings using a single consistent metadata schema, building mangers would spend less time and efforts to deploy applications that can improve building performance and, according to some studies, reduce energy consumption by an estimated 10% to 30% [9,13].

However, even if these standard schemas become widely adopted, older buildings with inconsistent naming tags would still need to be mapped to it. As a result, metadata inference approaches have been developed to convert inconsistent metadata information (e.g., what we see in the first two columns of Table 1) from existing buildings to a common schema [14–17]. These approaches leverage time-series samples [18,14,15] and/or tag descriptors [19–22] to learn a mapping between BAS points (which are sensors and actuators inside the building system) with inconsistent naming tags and the consistent metadata defined in the common schema.

Depending on what kind of data are being used during the inference procedure, metadata inference approaches can be categorized into time-series based approaches (the focus of this paper), tag-based approaches or combined approaches. For such purpose, the choice of the schema to use is irrelevant as long as it can represent the information required by the application. These metadata inference approaches have shown the potential of standardizing metadata and further facilitating deployment of portable building applications[1] [23,15,16]. Nevertheless, studies to date have been preliminary and most of the approaches have been evaluated only on a small scale (typically on two or three buildings).

Moreover, each application for which we use metadata inference approaches would have different sets of required BAS points, and each approach would obtain those points with different performance. For example, if we were interested in deploying an occupancy-based supervisory control algorithm that required access to zone-level temperature setpoints and temperature measurements, it is not clear which of the existing metadata inference approach would be best suited to support this application. The performance of each inference approach is also affected by a variety of other factors, which include but are not limited to: the type of the points, the length of the available historical data, the evaluation strategies, the performance metrics, etc. Additionally, the amount of human work required to configure and run each inference approach in order to achieve its best performance in a given application also varies. Hence, there is a need to better understand the trade-offs of these choices.

In this paper, we intend to shed light on these issues to improve our understanding of the limitations and provide answers to questions such as: is there one metadata inference approach that generally works well on different building sites? Is the information available from a subset of buildings rich enough to represent the distribution of another group of buildings? How will the performance of inference approaches be affected when we vary the data used to train the models?

To answer these questions, we evaluate 6 metadata inference approaches on more than 400 buildings on which a large BAS vendor has installed their systems. Due to the fact that there was considerable consistency in the tags used in this dataset (given that they come from a single vendor), we limited our scope to time-series based approaches. Furthermore, to ensure that the results are driven by application-level considerations, we focus on FDD applications and evaluate the ability of these inference approaches to map the points required by the AHU performance and analysis rules (APAR) proposed in [24,25] to detect and diagnose faults. This reduction in the scope of the evaluation was drive by practical considerations, though further experiments should be conducted to evaluate the general performance of these methods. AHUs are very prevalent within the US commercial building stock and account for a large portion of the energy usage of HVAC systems. Similarly, APAR is a simple rule-based FDD approach that has been widely cited in the literature (e.g., [2,3,26]). Specifically, we evaluate 6 time-series based metadata inference approaches on 12 types of BAS points collected from 614 AHUs serving 421 buildings located on 35 different sites.[2] across the continental United States (US). The source code of the metadata inference approach evaluation used in this paper is also released publicly[3] to facilitate further research in this domain.

2. Background and related work

To better illustrate the problem to be solved, Fig. 1 shows the information associated with one BAS point, including the observed information at the top and the consistent metadata at the bottom. All the associated information can facilitate the deployment of building applications through improving people's understanding and interpretations of this BAS point. The observed information from BAS typically includes data such as time-series values and metadata such as string descriptors (i.e., tags), measurement units, data types, etc. As stated earlier, very often this metadata is difficult to interpret and requires experts to decode it. Additionally, different building sites tend to use distinct conventions as different vendors are involved in setting up each system. It is worth noting that the problem of inconsistency also exists in other domains, such as data integration from heterogeneous sources [27] or data management from multi-dimensional building information [28].

The challenges to integrate heterogeneous information from different building systems to enable FDD applications have also been explored by others (e.g., [29]). Such integration challenges hinder the deployment of FDD applications in real world despite the large number

[1] Here portable applications refer to those that can be run on multiple buildings with minimal customized configurations once being deployed.

[2] Each site contains a group of buildings from one organization in a city.

[3] We released the source code on GitHub at https://github.com/INFERLab/metadata_inference.

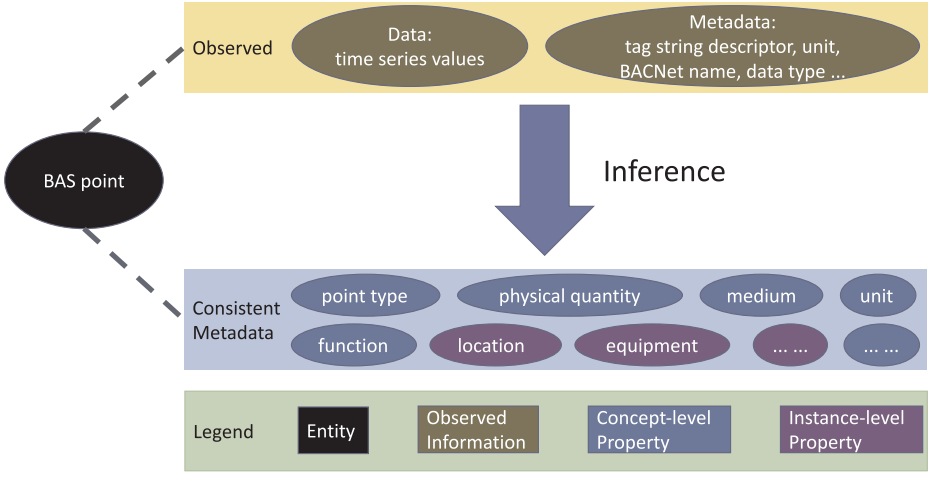


Fig. 1. An illustration of the metadata standardization problem for buildings.

Table 2
A concrete example of two points in BAS where we have observed information including time-series data and tag string descriptors, as well as the consistent metadata including concept-level and instance-level properties.

		point A	point B

Observed information	Time-series data	{
		2015-01-03 9:45:20 AM: 4.75;
		2015-01-03 9:46:19 AM: 4.58;
		…
		}
			{
			2015-12-17 11:53:23 AM: 60.23;
			2015-12-17 11:54:23 AM: 60.61;
			…
			}

	Tag string descriptor	MI.AHU.3FL.011.HCO	PC-NAE-1/N2-1.EN2.AHU-2.DAT-SP
Consistent metadata	Point type	Sensor	Set point
	Physical quantity	Valve status	Temperature
	Medium	Water	Air
	Unit	Percentage	Fahrenheit
	Function	Heating output of the coil	Supply
	Location	Mellon Institute	Purnell Center
	Equipment	Air handler unit – 011 on the third floor	Air handler unit – 2 in N2-1.EN2 zone

of academic publications in this area. Moving forward, the industry has also taken a lead to tackle the inconsistency issues to deploy FDD applications effectively by supporting standardization efforts and their surrounding products, such as Project HayStack,[4] SkySpark[5] and others. Interestingly, in the experience of the authors most of the deployed FDD systems are using rule-based FDD approaches similar to APAR. All of this points to a greater need for approaches that can simplify the integration challenges faced by the industry.

The consistent metadata, as is shown at the bottom of Fig. 1, is based on a schema that describes or annotates the BAS point entity in a consistent way. In the figure, we divide this consistent metadata into concept-level properties and instance-level properties. Concept-level properties associated with distinct entities from different buildings can have common values as they describe the same concept associated with the points. The distinct values of concept-level properties are finite. These properties include but are not limited to (1) the point types (e.g., sensors, set points or commands), (2) the physical phenomena the point is measuring or changing (e.g., temperature, humidity, pressure), (3) the medium the point is interacting with (e.g., water, air), (4) the unit representing the magnitude of the data values (e.g., pascal, Fahrenheit), (5) the function the point is serving inside the equipment (e.g., return, supply) and others. Instance-level properties usually have their own specific values for entities across buildings, and the distinct values of instance-level properties could be infinite such as the physical location the point resides in (site, building, floor, room, zone layout, etc.), or the equipment the point is associated with such as the specific AHU, or fan coil unit (FCU).

These definitions of the data and meta-data fields have also been similarly proposed by others [30,12,22]. Table 2 shows a concrete example of two BAS points from different systems described using them. Fig. 2 also shows typical patterns for six types of sensors found in an AHU from our dataset. For the particular set of sensors shown there, there are statistical characteristics (e.g., mean, maximum, minimum) that can be used to distinguish between them. Armed with this background knowledge, we now proceed to review the relevant literature on the problem.

2.1. Schemas and standards

In the past years, many conventions, systems, and schemas have been proposed and developed to address the problem of inconsistent metadata [31,9,32,33,10,8,34,11]. These works attempt to either define a model to organize the metadata using a schema (focusing on the relationships between different point entities and their properties [20,11,35]), or suggesting conventions for naming each point individually in a consistent manner (i.e., assuming that the name alone

[4] http://project-haystack.org/.
[5] https://skyfoundry.com/skyspark/.

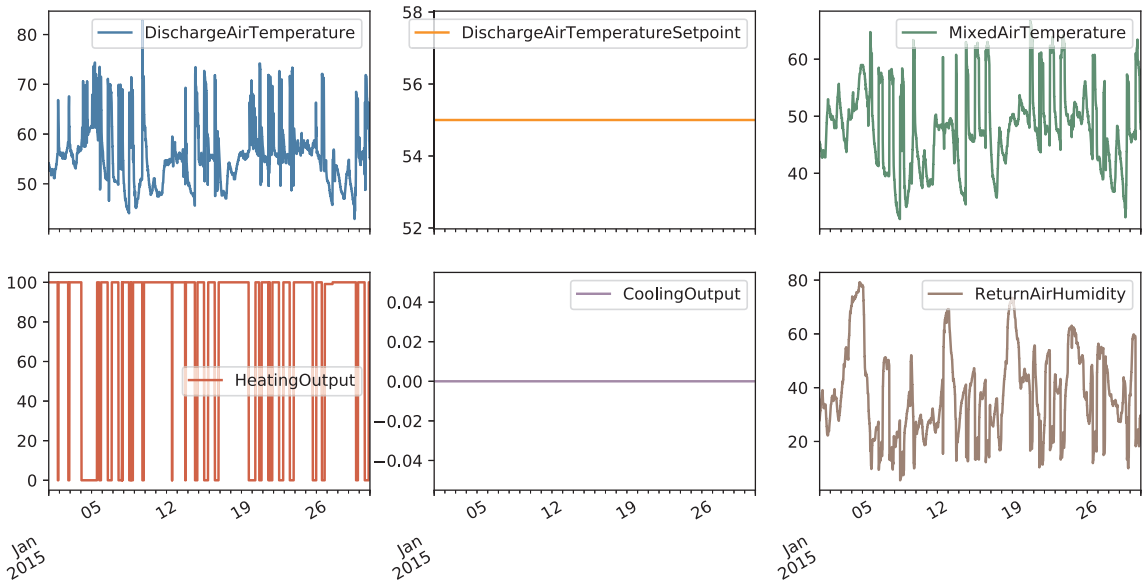


Fig. 2. Six types of sensors in an AHU with different statistical characterizations, such as maximum, minimum, mean, range, variance, etc.

contains enough metadata information) [9,10,8]. However, naming conventions are often insufficient to encode complicated relationships among points and devices (e.g., the location or functional relationships), and many of these schemas are oriented towards the information from the design and construction phase of the building, and cannot capture relationships and concepts needed for many applications in buildings [36–38].

To partially address limitations of existing approaches, recently the Brick [12] schema has been designed and proposed as a potential solution to manage metadata associated with entities, subsystems, and relationships among them to support portable building applications. By devising a normalized vocabulary (based on domain terms) and relationships interpretable to programs, Brick defines a concrete ontology for sensors, subsystems and relationships among them. This ontology allows both building managers to represent their BAS metadata consistently as well as software application developers to write portable applications that can be deployed in different buildings.

All these efforts have shown significant value to address the problem of inconsistent metadata for new buildings where building stakeholders can adopt the standard schema when setting up the system. However, as mentioned earlier, it is still expensive to convert older building systems to this standard manually.

2.2. Metadata inference approaches

Metadata inference seeks to map BAS points to a common schema using the available information for both older and newer buildings. As expressed earlier, these approaches can be divided into time-series based, tag-based, or a combination of both. Time-series based approaches, which will be the focus of our investigation, utilize time-series values from BAS points to learn the mapping [39,18,15,40,41]. They require the availability of historical sensor data collected from buildings. Tag-based approaches, on the other hand, rely on the tag names associated with BAS points [19,21], which is determined by how vendors from different BAS companies name the points in the first place. Some researchers have also used the combination of both time-series data and tag strings to infer the consistent metadata [14,23,16]. An extended version of the above approaches is to utilize a portion of metadata (certain associated properties) to infer the other metadata. For example, researchers have used time-series values and "type" of the sensors to infer the spatial information [42]. Additionally, in [43,17], authors adopted active approaches to perturb control points to infer location and functionality relationships. These active approaches show the potential of identifying metadata related to location and equipment. However, unlike the other passive approaches, they require control of the system, which may only be feasible for some buildings and during the specific time period.

In terms of the information being inferred, most of these approaches focus on the "type" property [39,19,14,44,16,15,40] of the metadata. Some others infer the location [45–47,18,48], functions [4,43], relationships [22], and other contextual information associated with sensors [42]. Experiments have been conducted on a small scale to infer "type" property while for inferring other metadata, most experiments are conducted at the scale of several rooms only. It is also worth pointing out that the "type" property of the metadata can be interpreted at different levels. For example, researchers can either distinguish "Return Air Humidity Sensor" and "Outside Air Humidity Sensor" for AHUs as two different types, or treat them as the same type ("Humidity Sensor") depending on what levels of details researchers are concerned. Such a different definition of the "type" property can lead to varying reported performance of the metadata inference approaches as well.

In summary, metadata inference approaches show the potential of constructing metadata semi-automatically. However, there are still questions we need to answer before we can deploy them effectively in real-world buildings at scale to enable portable applications. One of them is to understand whether these approaches can generalize well on a large number of buildings. Additionally, the applicability of these approaches when we vary the data (including the amount, duration, types of points, etc.) to train the inference models, also needs to be studied in order to understand the implementation feasibility.

3. Methodology

As stated previously, the goal of this paper is to address the generalizability and applicability of metadata inference approaches based on historical sensor measurements. To achieve that goal, we evaluate six time-series based approaches [49,15,16,23,14,17] on more than 400 buildings. Five of these approaches were selected based on a literature review that we conducted to find time-series based metadata inference approaches applied to building automation data to infer sensor types, and they represent the totality of the publications we found meeting that criteria. However, we realize that there may be other approaches that exist, and many more will be developed later, so we leave it to other researchers to extend the evaluation work. The sixth approach

Table 3
A summary of six time-series based inference approaches.

Year	Paper	Feature	Model	Metadata	Evaluation strategy	Testbed
1994	Li and Clifton [49]	Statistical quantities (mean, variance, coefficient of variation), then using SOM to reduce dimension and generate clusters	Trained back propagation network on one database with the cluster number as the label, then use the model to predict the cluster label for other databases	Similar attributes from heterogeneous databases	Test two databases with no common information and shows a low similarity; calculate the similarity between two groups of attributes and show similarities inside one database	3 pairs of existing databases
2015	Gao, Ploennigs and Bergés [15]	Statistical quantities (mean, median, quantiles, max, min)	7 classifiers including kNN, Decision Trees, GNB, RBF SVM, Logistic Regression, AdaBoost, LDA	Types of points in BAS (VAV, AHU, FCU, electrical/light systems)	Stratified 20% train, 80% test, show TP/TN/FP/FN and F_1 score	2 buildings
2015	Hong et al. [16]	Statistical quantities (min, max, median, rms, quantiles, var, skew, kurt, slope) on 60-min sliding windowed (30-min overlapping), and then apply summary statistics (min, max, median, variance) on top of features	Combine the prediction from locally weighted classifiers, which are logistic regression, SVM and random forest	Types of points in BAS	Use one building to train and another to test	3 buildings
2015	Bhattacharya et al. [23]	Summary statistics (min, max, median, variance) on median and variance values of the 45-min sliding window (no overlapping)	Random forest	Properties of points in BAS (in addition to type)	Increase the examples to label and track the qualified sensors to be identified by the algorithm	3 buildings
2015	Balaji et al. [14]	(1) min, max, mean, quartiles, range; (2) 3 Haar wavelets and 3 Fourier coefficients; (3) location and magnitude of top 2 components from piece-wise constant model, error variance; (4) first and second variance of difference between consecutive samples, max variance, number of up and down changes, edge entropy measure	Random forest	Types of points in BAS	Increase the number of labeled points of each type and check the accuracy	1 building[a]
2016	Koh et al. [17]	Mean, variance, dominant frequency, noise (error variance), skew, kurt	8 classifiers including GNB, LSVM, RF, RBF SVM, NN, DT, Adaboost, Bernoulli Naive Bayes (BNB)	Type, location and dependency	Use the co-location information of all VAV point types and the ground truth point types for one zone to infer point types from other zones	1 building

[a] Four buildings are included in the testbed while time-series based approach is tested on one building.

was taken from the database community where it is applied to the problem of schema matching [50], which shares many similarities with the metadata inference problem in the building community. Mapping inconsistent metadata to a common schema is similar to mapping and integrating schemas from different databases. As a result, we can borrow some instance-level based schema matching approaches, as is presented in [49], to help our mapping task using time-series data. A detailed summary of these six time-series based approaches can be seen in Table 3 where features, models, types of metadata, evaluation strategies and testbeds are listed.

To evaluate them in a consistent manner, we need to make sure they are compared within the same context. We can see the major differences among these approaches are: the features being extracted and the models being constructed. Some approaches utilize active learning to pre-cluster the data first to reduce the amount of required training data [14,23]. We do not consider these steps in our evaluation as we treat the clustering based active learning approach as a technique to select and reduce training samples. The evaluation incorporating active approaches is left as future work. For all six approaches using different feature extraction methods, we use seven widely used linear and non-linear classifiers from column 4 in Table 3. Eventually, we end up evaluating six types of features from six time-series based metadata inference approaches using seven classifiers on each feature.

In addition to features and classifiers, we select same sets of BAS points and evaluation strategies to be applied on the same dataset. Specifically, we choose points driven by one application (APAR) to detect faults in AHU systems [25]. The effective implementation of ARAR has shown the ability to detect faults, reduce energy waste and bring many other benefits. As we analyze the BAS points in AHUs from different buildings, we are only concerned with one specific metadata property: the type of BAS points. It is worth pointing by type, we refer to more than just the different phenomena measured by the sensing points required by APAR. For example, if different temperature sensors are installed in different positions serving distinct functions (e.g., return air temperature, discharge air temperature), then we treat each of them as being of a different type.

To start the evaluation, the first step is the data preparation where we generate the required datasets from observed BAS information. During this step, we extract all AHU points from BAS. Also, we provide labels ("type" of points in our case) for the data and conduct pre-processing to remove outliers.

Then as a second step, depending on each inference approach, we conduct feature extraction on the prepared data. In this paper, since we focus on time-series based inference, the features are derived from observed time-series values using descriptive statistics for example, though in general the features can also be extracted from the tag strings and other metadata using natural language processing techniques, etc.

The third step is to train and evaluate the model on the features derived from the data. Before training the model, the evaluation strategies need to be decided. This can vary depending on the use case for specific people. For example, for a building manager with hundreds of buildings where BAS points need to be labeled, she or he may prefer being able to train a model on some labeled buildings and use the model to produce the consistent metadata for the rest buildings, instead of labeling some data from a new building every time such unified metadata needs to be produced. Once the predicted labels are generated, a valid metric can be used to evaluate the performance depending on what people value, which could be precision, recall, F_1 score or others.

In the last step, we analyze results to better understand how the approach is obtaining the result and how it performs under different scenarios. This is achieved by examining the confusion matrix of the prediction, observing the performance change when varying the amount of the training data, the duration of the data, the temporal and spatial effects of the data, etc.

3.1. Experiments

Having introduced six inference approaches and the evaluation process, we now describe three sets of experiments using distinct evaluation strategies to answer the questions about the generalizability and applicability of metadata inference approaches.

3.1.1. Generalizability on single site (S1)

To understand whether there is one inference approach that generally works well on each site, in this experiment, we train the model using a certain ratio of data on each site and test the model on the same site using the remaining data, and then we iterate over all sites. The ratio of data to be trained is selected as 10% at first. We vary this ratio later to explore how the performance is affected. For training on each site using stratified random 10% of data, we repeat the process 20 times to ensure coverage of the samples. We refer this experiment as Strategy 1 (S1).

This strategy envisions that, for any new unlabeled buildings, we can just label 10% of the BAS points and use this approach to infer the metadata for rest of the points. In this scenario, some sites may not have enough samples to use as 10% of training data, which means none of the classes have more than 10 points, and when this is the case we ignore these sites. Additionally, for some sites, there are less than 10 points in certain classes, we ignore those classes and evaluate the approaches on data from the remaining classes.

3.1.2. Generalizability on multiple sites (S2)

To explore the generalizability on multiple sites, we conduct another experiment using leave-one-site-out cross validation. That is, we use data from all but one sites to train and use the data from the remaining site to test, and we iterate over sites. We refer to this experiment as Strategy 2 (S2).

This strategy makes sure that no data from the same site will appear in both training and testing samples. The reason for splitting by sites instead of buildings is to make sure we have enough test instances to evaluate. Such an evaluation can help us understand how the model performs on the unseen dataset. By using each of the sites as the testing site and observing the performance, we can reason whether the distribution drawn from a subset of buildings is generalizable. The vision is that we can use the trained model to predict the needed metadata for a new, unseen site.

3.1.3. Effects of data (S3)

To study the effects of the amount of training data on the approaches, instead of using the whole-year-long data directly as we did in previous two strategies, we conduct a group of experiments varying the data being used to train the model. We refer to this as Strategy 3 (S3). Specifically, we consider the following four scenarios:

(1) Varying the amount of data: we extend S1 by increasing the training ratio from 10% to 90% to study how the performance changes. Similarly, we extend S2 by changing the number of sites being used for training from 10 to 25 instead of 35;
(2) Varying data duration: we use both weekly and monthly data to conduct the same analysis for S2 instead of using one-year-long

Table 4
Climate zone definitions according to CBECS.[a]

Climate Zone	Cooling Degree Days	Heating Degree Days
cold	Fewer than 2000	More than 7000
cool	Fewer than 2000	5500 to 7000
normal	Fewer than 2000	4000 to 5499
warm	Fewer than 2000	Fewer than 4000
hot	2000 or More	Fewer than 4000

[a] https://www.eia.gov/consumption/commercial/maps.php.

samples;

(3) Temporal effects: we further study how the model performs when training the model on one month and testing on another. We vary our data by site and month. For each month, we use any combination of 34 out of the 35 sites to train. We test on the remaining site for prediction performance over each of 12 months. We always train with one month of data and on 34 sites, and test on the remaining one site over all months;

(4) Spatial effects: we also study the spatial effects of the data when we consider splitting data into different climate zones based on cooling degree days and heating degree days in the past 30 years as is seen in Table 4, which is defined by CBECS.[6] Since each zone contains different sets of points, to have a fair comparison across zones we synthetically generate a balanced data from the raw data where each zone has the same number of points for each point label.

Specifically, we first pick the point labels (i.e., classes) which have at least shown up 15 times within each climate zone (the number is selected so that we have a balance of the number of classes and the counts of samples). Once the labels are picked, we randomly draw 15 samples from each class without replacement for each zone. We end up having 105 points from 7 classes (15 per class) for each climate zone. Due to the limited number of points found in buildings that are in the hot zones, we only have data from four zones (cold, cool, normal, and warm).

To evaluate the spatial effects to the performance, for each zone we randomly use 50% of data from each class to train and test on the remaining 50% of data from this zone as well as all the data from rest zones.

3.2. Metrics

Notice all the experiments above are dealing with the multi-class classification problems. Evaluating the performance of the multi-class classifier model is not a trivial task as there are many different metrics to choose with each depicting certain aspects of the model performance and there is no single best metric measure for the model comparison. Common choices of metrics include *single-class focus threshold metrics* such as sensitivity/specificity, precision/recall, and F-measure, *multiple-class focus threshold metrics* such as accuracy, error rate, and kappa measures, and *ranking methods and metrics* such as receiver operation curve (ROC) analysis, precision-recall curves, and area under curve (AUC) [51]. Multiple-class focus metrics consider the overall performance and are less suited for the class-imbalanced situation as they are biased towards to the class with more samples [51]. Meanwhile, F-measure, a typical single-class focus metrics, is a popular metric in the information retrieval community and has been widely used for text classification due to the multiple classes and high class imbalance nature of text datasets [52]. Typical ranking methods like ROC and AUC-based comparisons depict the trade-off between true positive rates and false alarm rates, and are independent of the choice of classification threshold [53]. They also demonstrate advantages on datasets with skewed class distribution and unequal classification error costs [54].

When dealing with an unbalanced dataset, F_1 score and AUC are preferred. Nevertheless, both F_1 score and AUC are originally defined for binary classifiers. Extending these metrics for multiple classes requires averaging over the metric for each class.[7] Considering different averaging methods and assuming the distribution of each class (point type) in the real world is close to what we see in the data, we decide to use *micro* F_1 score to report the overall performance of the model described below.

For predictions of class i out of C classes, for each fold/iteration j out of K folds/iterations, we calculate number of true positives ($TP_i^{(j)}$), number of false positives ($FP_i^{(j)}$), and number of false negatives ($FN_i^{(j)}$), by treating class i as positive and rest all negative. Then we calculate aggregated TP,FP,FN over each class i and each fold/iteration j, and define *micro* F_1 score as follows:

$$TP := \sum_{j=1}^{K}\sum_{i=1}^{C} TP_i^{(j)}$$

$$FP := \sum_{j=1}^{K}\sum_{i=1}^{C} FP_i^{(j)}$$

$$FN := \sum_{j=1}^{K}\sum_{i=1}^{C} FN_i^{(j)}$$

$$F_1 := \frac{2\cdot TP}{2\cdot TP + FP + FN}.$$

This *micro* F_1 mathematically happens to be the same as the accuracy,[8] which is defined as the total number of true positives divided by the total number of predictions. Since for each class i, counts of false positives from another class $\hat{i}$ will be counted towards false negatives of this class i and vice versa, all aggregated FP and FN in the definition above are counted twice when we are using them to define the total number of predictions:

$$TP + \frac{1}{2}FP + \frac{1}{2}FN.$$

As a result, the accuracy is defined as:

$$Acc := \frac{TP}{TP + \frac{1}{2}FP + \frac{1}{2}FN} = = F_1.$$

Thus, we will use accuracy as the metric in this paper. We do understand that using a single metric to describe a model could be limited and lose certain perspectives of the model, hence, we also provide the detailed performance of more other metrics including *macro* F_1 score and AUC score, as well as the single-class metrics including F_1 score, precision, recall, and AUC for each class in Appendix B, and we just use the accuracy simply as a way to compare the performance. In addition to all these different metrics, we also analyze the confusion matrix after the prediction to understand the nature of the misclassifications.

4. Testbed and data

The data used for this study was collected using a platform developed by Johnson Controls.[9] We have access to sensor data collected in 614 AHUs from 421 buildings across 35 different sites. One site can be regarded as a group of buildings from one organization in a city. These sites encompass a wide variety of building types including educational institutes, office buildings, hospitals, libraries and others constructed in different years all in the US. Fig. 3 shows the site distribution of the buildings we have data from by state, covering different climate zones

[6] A spreadsheet file providing the climate zone for each US county can be found at https://www.eia.gov/consumption/commercial/data/archive/cbecs/CBECS%20climate%20zones%20by%20county.xls.

[7] Averaging can be done using *macro*, *micro* or *weighted* strategies. The choice of the averaging depends on how each class is valued. The *macro* strategy calculates the unweighted mean, while *micro* uses the global quantities (e.g., precision, recall, true positives) to calculate the score and does not give advantages to small classes; and the *weighted* strategy calculates the weighted average, where weights correspond to the number of instances for each class. A *macro* average is more biased towards small classes and indicates the expected performance on a dataset with balanced classes. On the other hand, the *weighted* average is more biased to classes with more samples as it gives more weights to them.

[8] An illustration example of different multi-class metrics showing *micro* F_1 and accuracy are equivalent can be seen at https://github.com/INFERLab/metadata_inference/blob/master/multiclass_metric_test.ipynb.

[9] http://www.johnsoncontrols.com/.

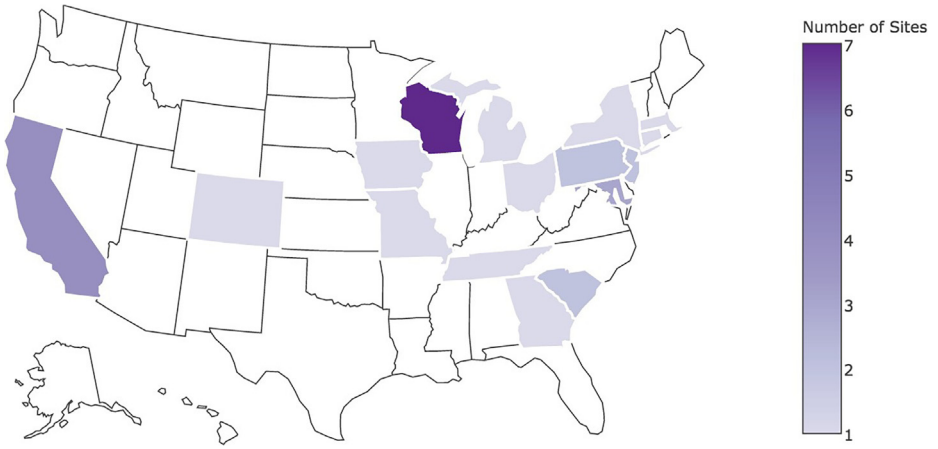


Fig. 3. State-wise site distribution of AHU data in the United States.

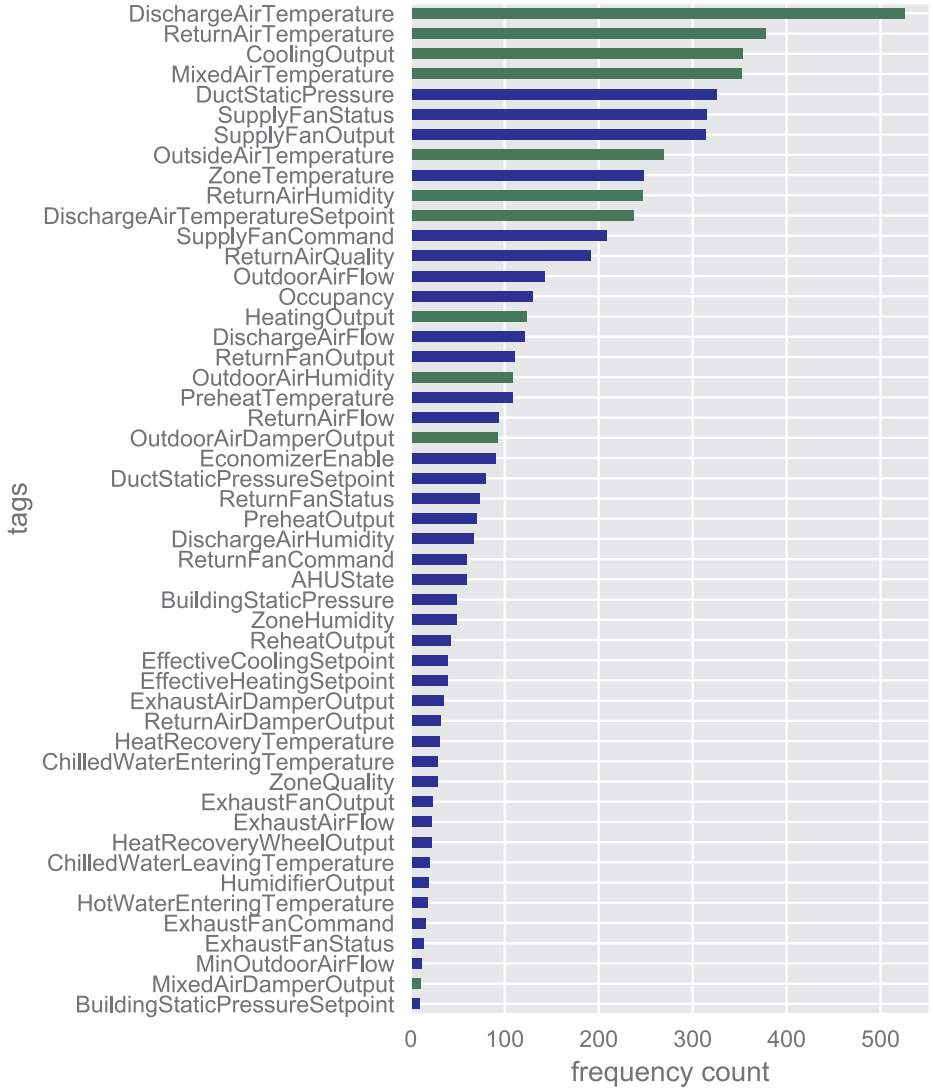


Fig. 4. Counts of top 50 frequent tags in the dataset. Those selected by APAR are marked in green.

and a total of 16 states. The data were collected for one year (from Jan. 1st, 2015 to Dec. 31st, 2015), and contain the historical values of measurements reported by different types of points located inside the AHUs. We ignore points that do not have data for one year or longer. We choose the one year limit to make sure the data collected show the possible seasonal effects.

As different sensing points have distinct sampling intervals ranging from one second to one hour, we re-sampled all the points to 15 min intervals using padding by filling values forward. Additionally, we removed outlier points if they either had unclear descriptions or exhibited

Table 5
Point name mappings between the vendor convention and Brick.

Vendor tag names	Brick names
HeatingOutput	AHU_Heating_Valve_Command
CoolingOutput	AHU_Cooling_Valve_Command
MixedAirTemperature	AHU_Mixed_Air_Temperature_Sensor
OutsideAirTemperature	AHU_Outside_Air_Temperature_Sensor
ReturnAirTemperature	AHU_Return_Air_Temperature_Sensor
DischargeAirTemperature	AHU_Discharge_Air_Temperature_Sensor
DischargeAirTemperatureSetpoint	AHU_Discharge_Air_Temperature_Setpoint
OutdoorAirHumidity	AHU_Outside_Air_Humidity_Sensor
ReturnAirHumidity	AHU_Return_Air_Humidity_Sensor
OutdoorAirDamperOutput	AHU_Outside_Air_Damper_Position_Command
MixedAirDamperOutput	AHU_Mixed_Air_Damper_Position_Command
Other	Other

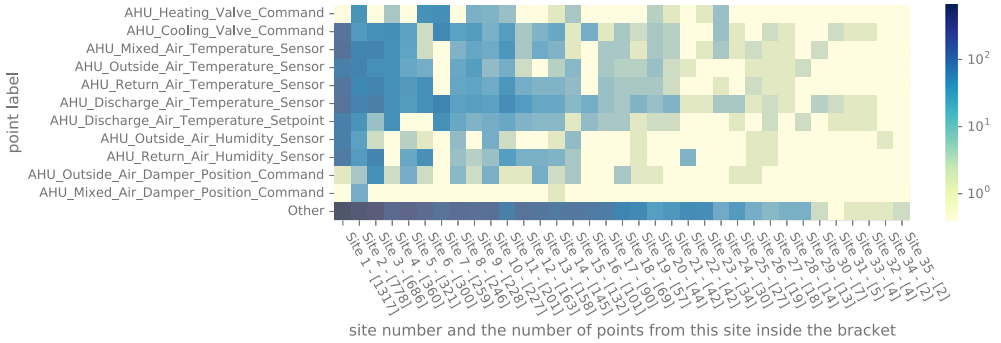
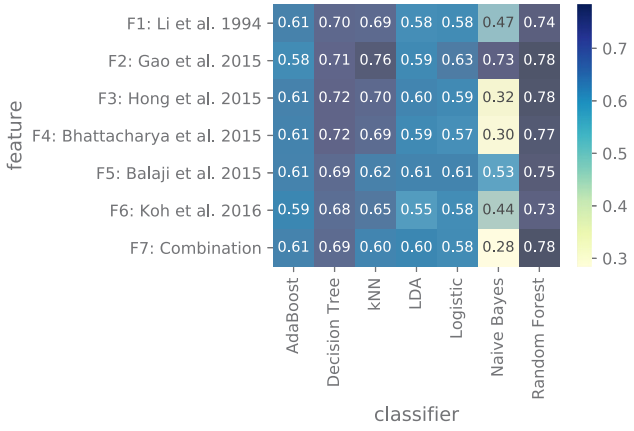


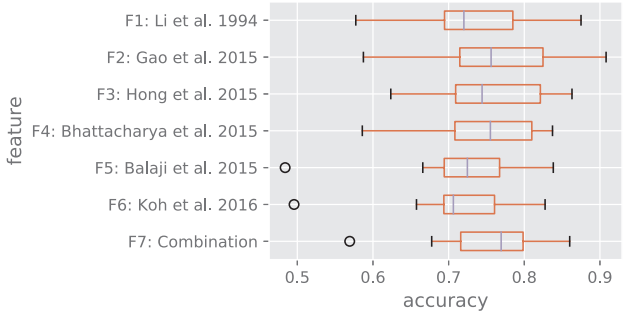
Fig. 5. Frequency counts of each point label across 35 different sites, the number inside the square bracket on the x-axis represents the total number of points at this site.

abnormal values (e.g., temperature values less than −50 Fahrenheit, or negative humidity values).[10] This eventually gives us a raw data matrix X of size 6145 × 35,040, representing 6145 BAS points with each having 35,040 samples for the whole year (i.e., 1 sample every 15 min). For each BAS point, a tag is attached to it following an internal convention inside the company that is considerably consistent. As shown by [37], the frequency with which these tags are used in buildings typically follows an almost power law distribution. The top 50 frequent tags are shown in Fig. 4. It is worth noting that these tags actually encode the metadata information including point types, physical quantities, medium, and functions. For example, "DischargeAirTemperature-Setpoint" represents a set point controlling the temperature of the air to be discharged out of an AHU.

As mentioned earlier, we focus on points required by APAR, which are marked with green colors in Fig. 4. It can be seen that about half of the frequent tags (counts greater than 100) are selected by APAR. In order to map these points into a common schema, we choose to use Brick [12] here to map the required points. For the unselected points, we label them as "Other" as APAR does not require the metadata associated with those points. Another option could be to use all the labels during the training but focus on the points we are concerned with during the evaluation. However, the performance of models would drop by including more classes as it results in a more complicated decision boundary. As a result, we end up having 12 different types of point labels, as is seen in Table 5 where we have both the original vendor specific tag names and the Brick names. To better understand how these 12 different types of point labels spread over building sites, Fig. 5 shows the number of counts of each label across sites, sorted from the site with most numbers to the least. We can see the distribution is quite unbalanced where some sites could have up to 1317 points and some only have 2 points. Such a small number is likely due to the fact that old



(a) accuracy score matrix



(b) box plot over 15 sites for features using Random Forest

Fig. 6. Box plot of accuracy score and score matrix for different features and classifiers (S1).

[10] The details of the pre-processing can be found in an example on GitHub at https://github.com/INFERLab/metadata_inference.

buildings still rely on the pneumatic systems and only a limited number of digital sensors are integrated into the BAS.

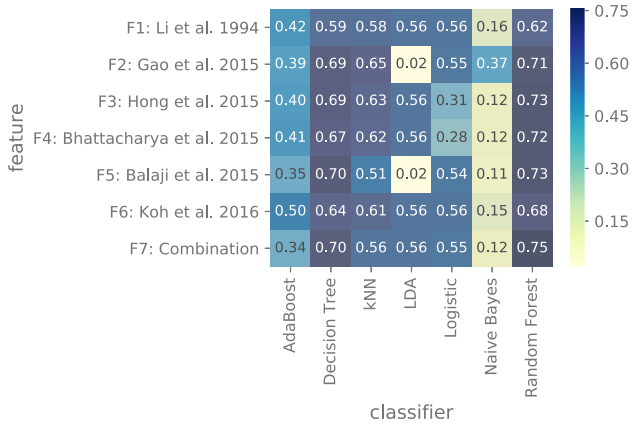
5. Results and discussions

In this section, we present results and discussions. The implementation details can be seen in Appendix A. It is worth pointing out that in addition to six different features from six metadata inference approaches, we also concatenate all features to produce the seventh feature "F7: Combination" as a comparison with others.

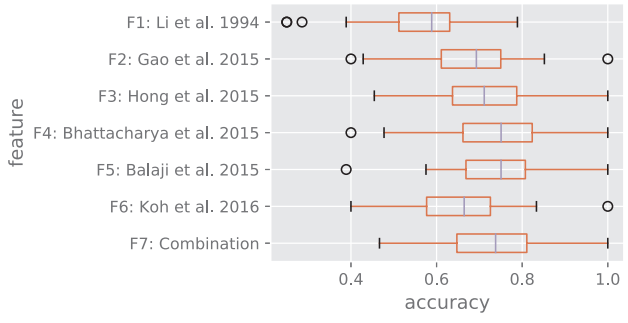
5.1. Generalizability on single site (S1)

Fig. 6(a) shows the accuracy over 15 sites for each feature and classifier. As we can see, Random Forest outperforms the rest of the classifiers all the time, yielding the highest accuracy for each feature. To understand how each feature performs over sites, Fig. 6(b) shows the box plot of accuracy score over 15 sites for different features using Random Forest. The score does vary drastically across sites for the same feature, with the difference between the maximum and the minimum (excluding outliers) being 20% to 30%. The tiny circles in the plot represent outliers, and these show that the model performs very poorly on certain sites.

The result in Fig. 6(b) indicates that the same metadata inference approach can perform quite differently on different sites with a standard deviation from 0.07 to 0.09. This variance is due to the distinct behaviors of points on each site. Additionally, all features show close performance as they are all similar in the sense that they are based on descriptive statistics (e.g., maximum, mean, median, etc.). We conduct the Kruskal-Wallis H test [55] to test whether accuracy scores over sites from each approach are drawn from the same distribution. The resulting p-value is $p \ll 0.001$, indicating that there is not enough



(a) accuracy score matrix



(b) box plot over 35 sites for features using Random Forest

Fig. 7. Box plot of accuracy score and score matrix for different features and classifiers (S2).

evidence to reject the null hypothesis that scores generated from different approaches are from the same distribution. When we examine the feature for each site yielding the highest accuracy, we find that almost all features achieve their highest site-specific performance using Random Forest. Moreover, for any fixed feature, Random Forest outperforms the rest of the classifiers all the time, as shown in the last column of Fig. 6(a). This signals that Random Forest is well suited for classifying point types in buildings due to its capabilities in dealing with flexible and overlapping decision boundaries and noisy data, which is also aligned with our prior research results [15]. The implication of this experimental result is that it is feasible to select a building site, label 10% of metadata for each point type, train a model using inference approaches, and we are expected to label 78% of the rest points with consistent metadata correctly. However, the actual performance can vary depending on which specific building site is being used.

5.2. Generalizability on multiple sites (S2)

To summarize the experimental results of S2, where the goal is to evaluate the inference performance of the model on unseen buildings based on training data from well-labeled buildings, we compute the accuracy matrix of different features across different classifiers. The results are shown in Fig. 7(a). We also show the box plots of the accuracy scores over 35 iterations of test sites in Fig. 7(b). As is expected, all statistical-based features achieve similar results with Random Forest being the best classifier.

On average, the scores from S2 are slightly lower than those from S1. Part of the reason is that S2 is using a stricter condition where the test building sites do not overlap with the training sites. The standard deviation of the accuracy score across sites is also larger for S2 (standard deviation value: 0.18) as compared with S1 (standard deviation value: 0.09). This makes sense, given that the variation in S2 is stronger due to the disjoint training and testing samples, as well as the increased number of sites. Similarly, we conduct the Kruska-Wallis H test on the 35 accuracy scores from each approach and obtain a p-value of $p < 0.001$, again failing to reject the null hypothesis that the distributions are the same. We also notice the performance difference between these two strategies is not remarkable, which might imply that the information from a subset of buildings is capable of representing the distribution of the statistical features being derived from each point type using the historical time-series of another group of buildings. This indicates that time-series values associated with points from multiple buildings could have similar distributions, which is of special interest as it shows the possibility of training a model on some buildings and using the model for other unseen buildings. However, it is worth noting that this initial finding is based on points in AHUs from buildings within one vendor's portfolio. The validity of the conclusion remains to be evaluated on more diverse building portfolios.

To further understand how the approaches perform under S2, we look at the confusion matrix using "F7: Combination" and Random Forest to see which predictions are incorrect. Due to the unbalanced number of samples for each class, we show a normalized confusion matrix (i.e., each element is divided by the sum of all the elements in the corresponding row). The values in each row represent the average probability vector of this type being predicted to each of 12 types in Fig. 8. The number inside the parentheses beside the label name on the vertical axis represents the number of testing instances for this class. We can see that "Outside Air Damper Position Command" and "Mixed Air Damper Position Command" are most easily confused with "Cooling Valve Command". This is a reasonable mistake, as they are all generating values within the range [0, 100] and the damper output values are strongly correlated with the cooling status, which can impact each other and show similar behaviors. The same result can be seen in Table B.9 where we compute the precision, recall and F_1 score for each class. We also notice that many point labels are misclassified as "Other", which is due to the diverse behavior of excluded points in AHUs. If we

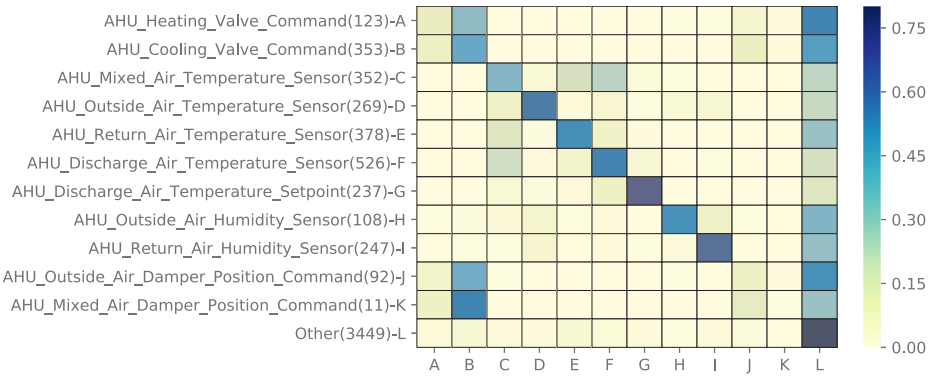


Fig. 8. Normalized confusion matrix by row using F7: Combination and Random Forest (S2). The number inside the bracket beside the label name on the vertical axis represents the number of testing instances for this class.

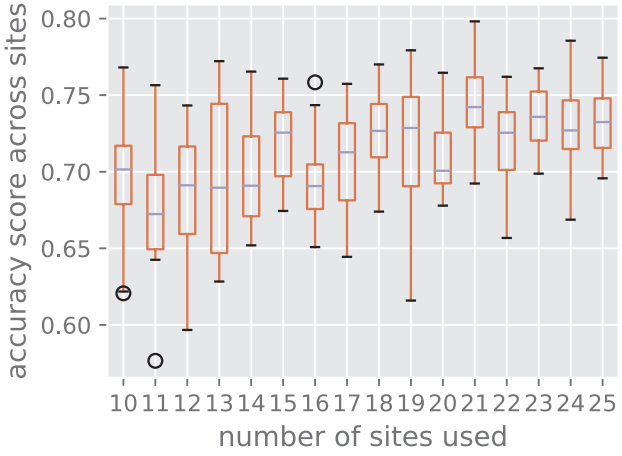


Fig. 9. Boxplot of accuracy change when we vary the number of sites (S2).

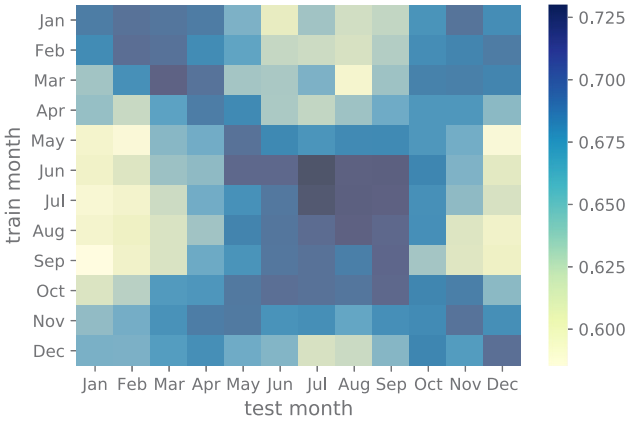


Fig. 10. Accuracy score when training on one month and testing on another.

Table 6
Statistics of accuracy when using data from different durations.

accuracy	year	month	week
mean	0.735	0.701	0.671
median	0.739	0.687	0.662
standard deviation	0.158	0.155	0.158

can somehow exclude all "Other" points from the analysis and only focus on the selected 11 types of point labels, the accuracy score increases to 80% using S2.

5.3. Effects of data (S3)

For this subsection, we explore how the model performs under S3 where we consider varying the amount of data used for training the model as well as the duration represented in the data (e.g., a full year of measurements), the seasons that are represented, as well as other temporal and spatial effects.

5.3.1. Amount of data

We first explore how the accuracy changes when we vary the amount of training data for S1 and S2 while keeping the temporal duration (1 year) of each sample fixed and not paying attention to the spatial location of the buildings in the training sample. For S1, we increase the training ratio from 10% to 90% using "F4: Hong et al. 2015" and Random Forest.[11] As is expected, the accuracy increases from 78% to 90%.

Similarly, for S2, we vary the number of sites being used. We start with only 10 sites and we use the "leave-one-site-out" strategy to evaluate the performance. By adding more sites into the model, we want to find out how the performance changes. Each time, a number of sites are randomly chosen out of 35 sites and the process is iterated 20 times. We pick the number of sites to vary from 10 to 25 due to the fact that the number of possible combinations for choosing 10 out of 35 is the same as choosing 25 out of 35. We then calculate accuracy score over 20 iterations as the performance metric. Fig. 9 shows how it changes when we vary the number of sites. As we see, the general trend of the accuracy score is slightly increasing and the standard deviation is decreasing, indicating that the model becomes more accurate and stable when we have data from more building sites.

5.3.2. Duration of data

To study the effects of the duration of the data used for training, we divide the year-long data into week-long and month-long segments and evaluate the model performance for each segment using S2 where we train the model using data from 34 sites and test on the data from the remaining site, and we iterate until each site has been used as the test site once. The evaluation gives us 52 values of the accuracy score on each testing site for weekly data and 12 values for monthly data. Table 6 summarizes the result of the accuracy score from data of different durations. For the yearly case, we report the statistics for accuracy score given 35 iterations on each test site. As is seen, the yearly data provides slightly better performance compared with others, which

[11] If not specified, the following explorations of data effects are all using this feature and classifier as shown in earlier results. The choice of feature does not significantly affect the result as long as it is one of the statistically based features, which summarize the descriptive statistics of the time-series.

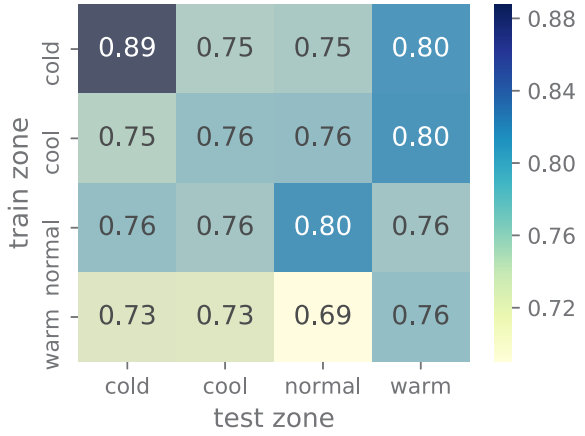


Fig. 11. Accuracy score when training from one climate zone and testing on another.

makes sense as a longer duration is able to capture more temporal characteristics of point behaviors. Given the performance drop for the weekly data is not significant, we may still be able to use metadata inference approaches with a short duration of data when one-year-long data are not available.

5.3.3. Temporal effects

To study the temporal effects, we report the average accuracy score across all testing sites for all possible pairs of training and testing months.

Fig. 10 shows the average performance of training on one month and testing on another month. If we sum the values in the diagonal and take the mean, the value should be close to the mean of the monthly result 0.701 in Table 6. The results from Fig. 10 indicate that training and testing on the adjacent months is likely to produce slightly better performance. This implies that when training and testing models on different building sites, it is not necessary to make sure the data are from the same temporal period. The model will generally perform well as long as the data are temporally adjacent.

5.3.4. Spatial effects

We also wanted to explore how the model performs when we consider spatial differences and split the data into different climate zones. Specifically, we iterate the experiment process (as is described in Section 3.1.3) 20 times for each zone on the synthetic dataset and report the average accuracy when training on one climate zone and testing on another.

Fig. 11 shows the performance of training on one climate zone and testing on another zone. We can see the performance is slightly better within each zone compared with training and testing across zones. Training on data from cold zones tends to provide better results. Furthermore, if we check the variations of each experiment, the standard deviation is between 0.02 and 0.06, which means the difference of training and testing on different zones is not that significant. This is also aligned with the conclusion we drew previously in S2 that the time-series values associated with points from different buildings have similar distributions, regardless of the location of the building.

5.4. Probability perspective

So far, we have been interpreting prediction results deterministically. However, another interesting perspective is to look at the predicted probability mass vector. In other words, for each time-series, the predicted output is not a simple label, instead, we have a vector indicating the probability that this time-series belongs to each class.

Fig. 12 explains the idea using 12 instances (one from each class). Each row in Fig. 12(a) is a probability mass vector indicating the likelihood of this point belonging to each class. The ideal prediction occurs when the most likely predictions for each vector fall on the diagonal. This is similar to Fig. 8 where we show the average probability for each class, while Fig. 12(a) represents the probability for each specific point.

Given a probability threshold $p\,(0 < p < 1)$, we have N time-series data $\mathbf{X} = \{\mathbf{x}_1,...,\mathbf{x}_N\}$ with each time-series $\mathbf{x}_i \in \mathbb{R}^T$ representing T time ticks. These N time-series are predicted to N probability vectors represented by $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}_1,\cdots,\hat{\mathbf{y}}_N\}$. Each vector $\hat{\mathbf{y}}_i \in \mathbb{R}^m$ represents the probability of predicting the time-series i to each of m classes and this vector always sums to one ($\sum \hat{\mathbf{y}}_i = 1$ for any i). The final prediction $\hat{y}_i$ is generated through $\text{argmax}\hat{\mathbf{y}}_i$. For each probability vector $\hat{\mathbf{y}}_i$, we count time-series i as part of the covered prediction set $\mathscr{Y}_p^{cover}$ if $\max(\hat{\mathbf{y}}_i) > p$. Fig. 12(b) shows the case when we set 0.4 as a threshold. Basically, the covered prediction set includes predictions with more confidence. Denote $Y = \{y_1,...,y_N\}$ as the true labels, we can define the following two metrics given probability threshold p:

coverage: percentage of predictions with confidence higher than p

$$\frac{|\mathscr{Y}_p^{cover}|}{N}$$

prediction accuracy: percentage of correct predictions among covered set

$$\frac{\sum_{i\in\mathscr{Y}_p^{cover}} \mathbf{1}(\hat{y}_i = y_i)}{|\mathscr{Y}_p^{cover}|}$$

Additionally, if we tolerate mistakes generated by the probability prediction and assume that the predictions are correct as long as the actual label is within the top k predictions ranked by probability vector, we can define another metric given the tolerance number k:

tolerance metric: denote $\hat{\mathbf{y}}_i^k$ as the top k predictions ranked by probability vector $\hat{\mathbf{y}}_i$, the accuracy when we tolerate k mistakes is

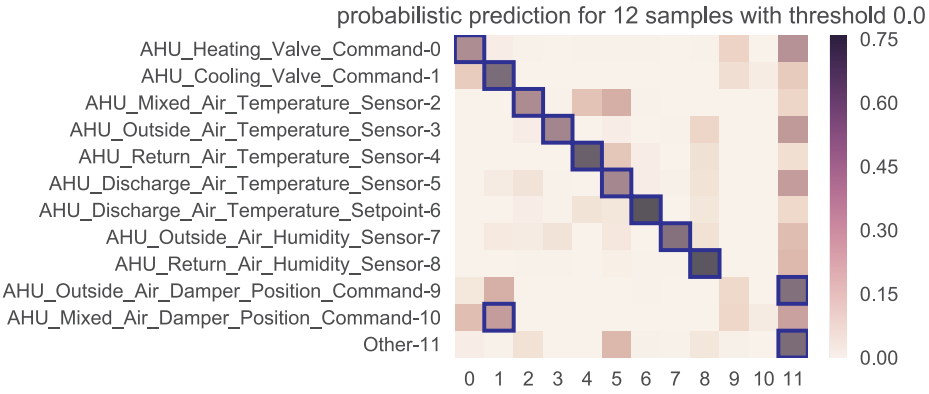
$$\frac{\sum_i \mathbf{1}(y_i \in \hat{\mathbf{y}}_i^k)}{N}$$

In the example shown in Fig. 12(a), the original accuracy is 83% (10/12). However, we can have an accuracy of 90.9% (10/11) with 92% (11/12) coverage by setting up 40% as the probability threshold; and the tolerance metric being 92% (11/12) by setting the tolerance number to 3 ($\hat{\mathbf{y}}_9^3$ contains the true label and $\hat{\mathbf{y}}_{10}^3$ does not).

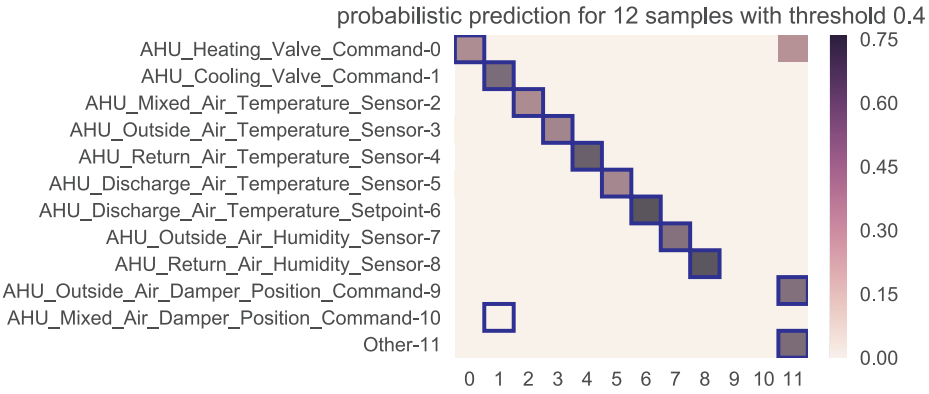
Using the definitions above, we calculate these metrics by varying the tolerance number and the probability threshold for both S1 and S2 in Fig. 13. As we can see the tolerance metric can go up to 95% if we tolerate 3 guesses. The use case for this is to reduce the labeling efforts from identifying 1 label out of 12 different labels to identifying only 1 out of 3. Another perspective is to set up the probability threshold. By setting it to.6 for example, we can cover 60% of the points with an accuracy up to 80–90%. If we want to be more aggressive, we can choose to only cover 40% of the points with an accuracy up to 95% in the case fo S2. This indicates we can trust the algorithm with high probability (up to 95%) to label 40% of the total data and we only need to manually label the rest 60%. By incorporating probabilistic perspectives into the predictions, it can be more efficient for building operators and managers to produce the consistent metadata for buildings in practice.

6. Conclusions

This paper investigates the generalizability and applicability of six time-series based metadata inference approaches by evaluating them on sensors from 614 AHUs and studying how the data used to train the models affect their performance. We find that when evaluating the approaches on such a dataset, we can achieve the best performance with an accuracy of 75%, regardless of training and testing on the same site (S1: 10% to train, 90% to test) or training and testing on different

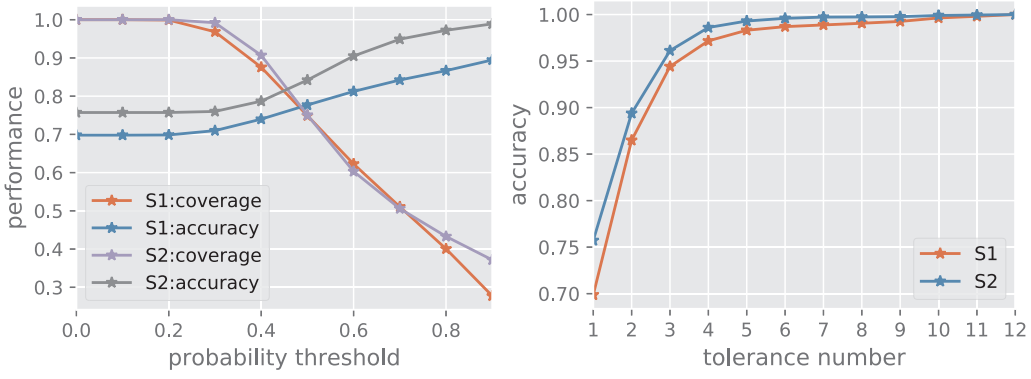


(a) raw prediction, the blue square box highlights the most likely prediction



(b) setting .4 cutting threshold, the blue square box highlights the most likely prediction

Fig. 12. An example showing the probability prediction metric.



(a) coverage and accuracy metric (b) tolerance metric

Fig. 13. Probability metrics from two perspectives.

sites (S2: leave-one-site-out cross validation). Moreover, these different testing approaches do not exhibit a significant difference in terms of performance.

Another way to interpret these results is as follows. Consider 10 building sites containing a total of 1000 distinct BAS points, where each site has 100 points. If we are able to obtain trust-worthy labels for at least 10 representative points in each site (i.e., where each sensor type should be covered) and use them to train the model, existing metadata inference approaches could impute the rest of the labels (i.e., the remaining 90 points on each site) with 78% accuracy on that same site. When we are training and testing on different sites, the result indicates that we can randomly pick 9 sites to use 900 points to train the model, and we are expected to predict 75% (75 points) of 100 points from an unseen site correctly. It may seem as if training and testing on different sites requires more training data at the first glance, but it does not require any training data for a new unseen site, which would reduce the amount of training efforts significantly as the number of testing sites increases.

To study the applicability of these approaches in more realistic conditions, we explore proxies for the amount of human effort required

to train the models, including varying the amount, duration and temporal/spatial factors of the training data. We find that by increasing the training ratio from 10% to 90%, we can improve the accuracy score from 78% to 90% when training and testing on the same site. Increasing the amount of data being used also helps to reduce the variance of the performance of the model in the case of training and testing on different sites. We also observe that yearly data show strongest patterns to differentiate distinct point labels. By using training and testing data from different time periods, we find the model can generally perform well as long as the data are adjacent temporally. However, when we pick data from different climate zones, we haven't found training and testing on the similar climate zones can provide significantly better results other than using data from cold zones, indicating the spatial effects to the model are less significant compared with the temporal effects.

Additionally, from a probabilistic perspective, we define metrics including prediction accuracy based on the coverage and tolerance metric based on the tolerance number. These metrics can make the predictions of the metadata from the model more useful for building operators and managers who need to label BAS points in buildings, as they can reduce the amount of time to focus on the points to be of special interested selected by the model. Specifically, for instance, direct predictions can only label 75% of points correctly, however, with probability perspectives, we can cover 40% of points and predict almost all of them correctly with very high probability guarantee, and for the rest 60% of points, we can make sure the predictions are correct if we can accept 3 most likely candidates, reducing the searching efforts from the original large space to a much smaller space.

Several future working directions are suggested in this research field. First, more advanced feature extraction techniques considering temporal evolution and multivariate relationships of BAS points should be studied to differentiate inseparable points by simple statistical features. These could include autoregressive-moving-average models, graph and network analysis of sensor nodes, etc. Secondly, a more comprehensive representation of metadata needs to be reasoned from existing BAS on a large scale in addition to the types of BAS points, such as, the location of the points, the equipment the point belongs to, the functions and interactions between sensors and building components. All these research directions will lead us towards an automated metadata standardization in BAS to facilitate the ultimate vision of portable building applications.

Acknowledgments

We would like to first acknowledge Siebel Foundation and Scholarship Council for the funding that supported the research presented in this paper. This research was also partially supported by the U.S. Department of Energy (DOE) under award DE-EE0006353. We would also like to sincerely thank Dr. Youngchong Park, Erik Paulson, and Andrew Boettcher from Johnson Controls for providing the data used in the research. The opinions expressed here are those of the authors and do not necessarily reflect the views of the sponsors.

Appendix A. Implementation details

In this appendix, we specifically talk about the implementation details for feature extractions and the parameters for the classifiers. We use scikit-learn [56] package for all implementation.

A.1. Features

We implemented 6 different types of features as is seen in Table 3. Additionally, we combine all 6 features to generate the 7-th feature. The details of each feature are described as follows:

- For "F1: Li et al. 1994" [49], we extract mean, variance and coefficient of variation;
- For "F2: Gao et al. 2015" [15], in additional to what is described in the table, we include 2-nd to 4-th order of central moments of the data, as well as the entropy. The entropy is calculated by digitizing the data to 100 bins evenly if it contains more than 100 discrete values.
- For "F3: Hong et al. 2015" [16], we use the exact features described in the table.
- For "F4: Bhattacharya et al. 2015" [23], we use the exact features described in the table.
- For "F5: Balaji et al. 2015" [14], we also use 100 bins to digitize the data when calculating the entropy.
- For "F6: Koh et al. 2016" [17], we use the amplitude of the first 3 frequency components.
- For "F7: Combination", we simply combine all the previous features.

A.2. Classifiers

Seven classifiers are used, namely k-nearest neighbor (kNN), naive Bayes, logistic regression, linear discriminant analysis (LDA), decision tree, random forest, and AdaBoost. Both random forest and Adaboost use decision trees as the base classifiers to build the ensemble classifier. We vary some parameters of those classifiers but we notice the performance is not significantly affected. We did also try SVM with RBF kernel. Due to the long running time and low performance, we did not include it in the results. For reference purpose, the following parameters are used for the classifiers (see Table A.7):

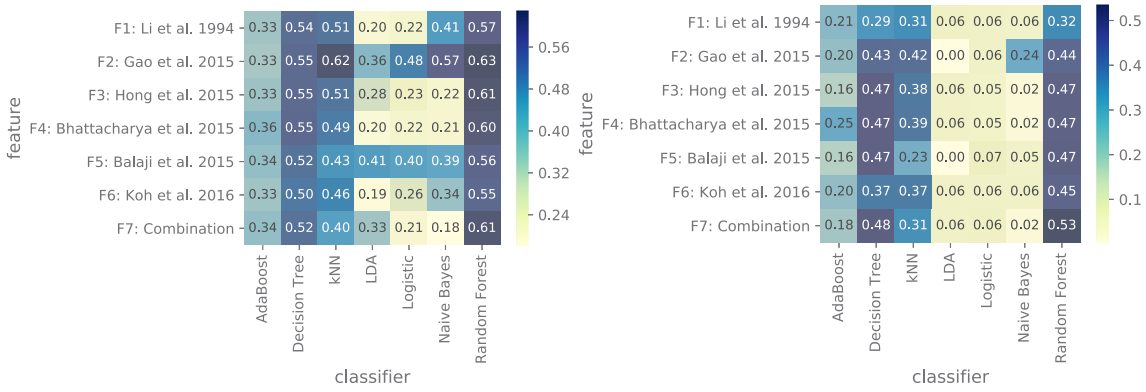
Table A.7
The parameters used for different classifiers.

classifier	parameters
kNN	k = 3
Logistic	C = 1e5
Decision Tree	max depth = 10
Random Forest	max depth = 10, number of estimators = 20
AdaBoost	max depth = 10, number of estimators = 100

Appendix B. Performance of other metrics

B.1. Macro F_1 score matrix for features and classifiers

We show *macro* F_1 score matrices for both strategies in Fig. B.14, which have the similar trend compared with accuracy score. However, the overall values are smaller compared with *micro* F_1 score (accuracy) due to a few classes with low performance decreases the overall *macro* F_1 score.

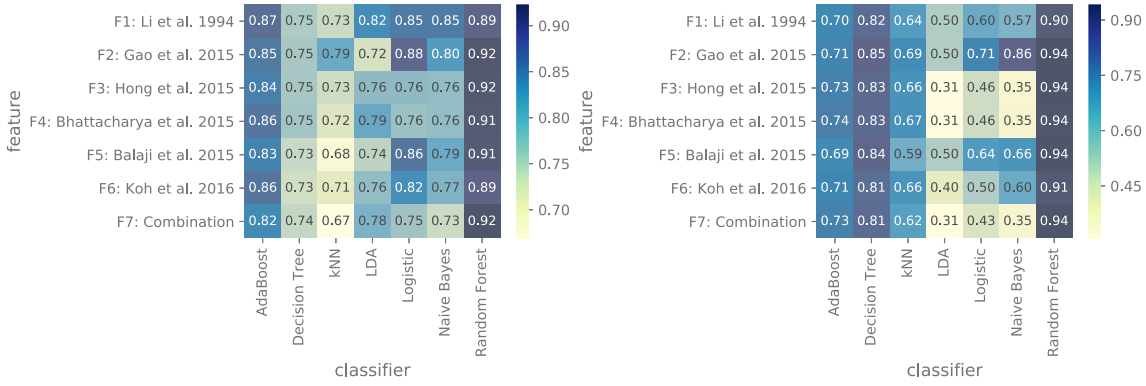


(a) macro F_1 score matrix (S1) (b) macro F_1 score matrix (S2)

Fig. B.14. Macro F_1 score matrix from two strategies.

B.2. macro AUC score matrix for features and classifiers

We show macro AUC score matrices for both strategies in Fig. B.15, which have the similar trend compared with accuracy score. Macro AUC is generated by "averaging" over individual AUC calculated based on a "one versus all" binary classifier is built for each class. It shows a very high value, which is largely due to the number of true negatives is pretty high.

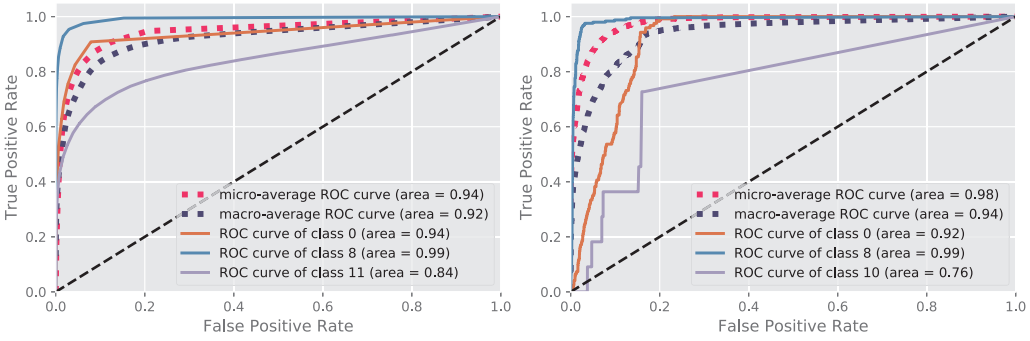


(a) macro AUC score matrix (S1) (b) macro AUC score matrix (S2)

Fig. B.15. Macro AUC score matrix from two strategies.

B.3. ROC examples

We show the ROC examples using"F7: Combination" and "Random Forest" for both strategies in Fig. B.16.



(a) ROC examples (S1) (b) ROC examples (S2)

Fig. B.16. ROC examples from two strategies.

B.4. Single class metrics for each class

We show the precision, recall, f1-score, auc and support using“F7: Combination” and “Random Forest” for both S1 and S2 in Tables B.8,B.9. The column “support” represent the ratio of samples for the corresponding class.

Table B.8
Precision, recall, f1-score, AUC and support for each class (S1).

	precision	recall	f1-score	auc	support
AHU_Heating_Valve_Command	0.63	0.43	0.51	0.939	0.016
AHU_Cooling_Valve_Command	0.66	0.63	0.65	0.962	0.057
AHU_Mixed_Air_Temperature_Sensor	0.49	0.44	0.46	0.911	0.058
AHU_Outside_Air_Temperature_Sensor	0.89	0.90	0.90	0.987	0.043
AHU_Return_Air_Temperature_Sensor	0.57	0.53	0.55	0.938	0.063
AHU_Discharge_Air_Temperature_Sensor	0.63	0.68	0.65	0.878	0.089
AHU_Discharge_Air_Temperature_Setpoint	0.78	0.62	0.69	0.942	0.033
AHU_Outside_Air_Humidity_Sensor	0.96	0.87	0.91	0.991	0.016
AHU_Return_Air_Humidity_Sensor	0.88	0.81	0.84	0.992	0.037
AHU_Outside_Air_Damper_Position_Command	0.35	0.24	0.29	0.888	0.009
AHU_Mixed_Air_Damper_Position_Command	0.11	0.01	0.01	0.804	0.002
Other	0.85	0.89	0.87	0.843	0.578

Table B.9
Precision, recall, f1-score, AUC and support for each class (S2).

	precision	recall	f1-score	auc	support
AHU_Heating_Valve_Command	0.04	0.01	0.01	0.916	0.020
AHU_Cooling_Valve_Command	0.51	0.52	0.51	0.956	0.057
AHU_Mixed_Air_Temperature_Sensor	0.48	0.49	0.48	0.936	0.057
AHU_Outside_Air_Temperature_Sensor	0.86	0.68	0.76	0.976	0.044
AHU_Return_Air_Temperature_Sensor	0.51	0.67	0.58	0.960	0.062
AHU_Discharge_Air_Temperature_Sensor	0.68	0.70	0.69	0.968	0.086
AHU_Discharge_Air_Temperature_Setpoint	0.89	0.86	0.87	0.980	0.039
AHU_Outside_Air_Humidity_Sensor	0.98	0.81	0.89	0.994	0.018
AHU_Return_Air_Humidity_Sensor	0.76	0.74	0.75	0.989	0.040
AHU_Outside_Air_Damper_Position_Command	0.00	0.00	0.00	0.925	0.015
AHU_Mixed_Air_Damper_Position_Command	0.00	0.00	0.00	0.763	0.002
Other	0.84	0.87	0.85	0.916	0.561

References

[1] W.W. Jones, R.W. Bukowski, Critical information for first responders, whenever and wherever it is needed, in: Proceedings of the 9th International Interflam Conference, vol. 2, 2001, pp. 1073–1082.
[2] S. Katipamula, M.R. Brambley, Review article: methods for fault detection, diagnostics, and prognostics for building systems-a review, part i, HVAC&R Res. 11 (1) (2005) 3–25, http://dx.doi.org/10.1080/10789669.2005.10391123.
[3] S. Katipamula, M.R. Brambley, Review article: methods for fault detection, diagnostics, and prognostics for building systems-a review, part ii, HVAC&R Res. 11 (2) (2005) 169–187, http://dx.doi.org/10.1080/10789669.2005.10391133.
[4] F. Xiao, C. Fan, Data mining in building automation system for improving building operational performance, Energy Build. 75 (2014) 109–118, http://dx.doi.org/10.1016/j.enbuild.2014.02.005.
[5] C. Regnier, A. Robinson, Achieving a Net Zero Energy Retrofit – In a Humid, Temperate Climate: Lessons from the University of Hawaii at Manoa, 2013. http://dx.doi.org/10.2172/1170598.
[6] S. Kaldorf, P. Gruber, Practical experiences from developing and implementing an expert system diagnostic tool/discussion, ASHRAE Trans. 108 (2002) 826.
[7] R. Jagpal, Computer Aided Evaluation of HVAC System Performance: Technical Synthesis Report, Tech. Rep., International Energy Agency, 2006.
[8] W. Livingood, J. Stein, T. Considine, C. Sloup, Review of Current Data Exchange Practices: Providing Descriptive Data to Assist with Building Operations Decisions, Tech. Rep., National Renewable Energy Laboratory, 2011.
[9] A. Dexter, J. Pakanen, Demonstrating Automated Fault Detection and Diagnosis Methods in Real Buildings, Technical Research Centre of Finland (VTT), 2001.
[10] J.F. Butler, Point naming standards, ASHRAE J. 52 (2010) B16.
[11] Haystack, Project Haystack, 2014. < http://project-haystack.org/ > .
[12] B. Balaji, A. Bhattacharya, G. Fierro, J. Gao, J. Gluck, D. Hong, A. Johansen, J. Koh, J. Ploennigs, Y. Agarwal, M. Berges, D. Culler, R. Gupta, M.B. Kjærgaard, M. Srivastava, K. Whitehouse, Brick: Towards a unified metadata schema for buildings,

in: Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments, BuildSys '16, ACM, New York, NY, USA, 2016, pp. 41–50. http://dx.doi.org/10.1145/2993422.2993577.
[13] K.W. Roth, D. Westphalen, M.Y. Feng, P. Llana, L. Quartararo, Energy Impact of Commercial Building Controls and Performance Diagnostics: Market Characterization, Energy Impact of Building Faults and Energy Savings Potential, Tech. Rep., Building Technologies Program, 2005.
[14] B. Balaji, C. Verma, B. Narayanaswamy, Y. Agarwal, Zodiac: organizing large deployment of sensors to create reusable applications for buildings, Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments – BuildSys '15, ACM Press, New York, New York, USA, 2015, pp. 13–22, , http://dx.doi.org/10.1145/2821650.2821674.
[15] J. Gao, J. Ploennigs, M. Bergés, A data-driven meta-data inference framework for building automation systems, in: Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments – BuildSys '15, 2015. http://dx.doi.org/10.1145/2821650.2821670.
[16] D. Hong, H. Wang, J. Ortiz, K. Whitehouse, The building adapter, Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments – BuildSys '15, ACM Press, New York, New York, USA, 2015, pp. 123–132, , http://dx.doi.org/10.1145/2821650.2821657.
[17] J. Koh, B. Balaji, V. Akhlaghi, Y. Agarwal, R. Gupta, Quiver: using control perturbations to increase the observability of sensor data in smart buildings, CoRR abs/1601.07260.
[18] M. Koc, B. Akinci, M. Bergés, Comparison of linear correlation and a statistical dependency measure for inferring spatial relation of temperature sensors in buildings, Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings – BuildSys '14, ACM Press, New York, New York, USA, 2014, pp. 152–155, , http://dx.doi.org/10.1145/2674061.2674075.
[19] Y. Park, Point naming standards: a necessary evil for building information integration, in: ISA Automation Week 2012: Control Performance, 2012.
[20] J. Ploennigs, B. Hensel, H. Dibowski, K. Kabitzsch, Basont – a modular, adaptive building automation system ontology, in: IECON 2012 – 38th Annual Conference on IEEE Industrial Electronics Society, 2012, pp. 4827–4833. http://dx.doi.org/10.1109/IECON.2012.6389583.
[21] A. Schumann, J. Ploennigs, B. Gorman, Towards automating the deployment of energy saving approaches in buildings, Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings – BuildSys '14, ACM Press, New York, New York, USA, 2014, pp. 164–167, , http://dx.doi.org/10.1145/2674061.2674081.
[22] F. Leonardi, H.M. Reeve, T.C. Wagner, Z. Xiong, W. June, Assisted point mapping to enable cost-effective deployment of intelligent building applications, in: International Compressor Engineering, Refrigeration and Air Conditioning, and High Performance Buildings Conferences, 2016, pp. 1–8.
[23] A.A. Bhattacharya, D. Hong, D. Culler, J. Ortiz, K. Whitehouse, E. Wu, Automated metadata construction to support portable building applications, Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments – BuildSys '15, ACM Press, New York, New York, USA, 2015, pp. 3–12, , http://dx.doi.org/10.1145/2821650.2821667.
[24] J.M. House, H. Vaezi-Nejad, J.M. Whitcomb, An expert rule set for fault detection in air-handling units/discussion, ASHRAE Trans. 107 (2001) 858.
[25] J. Schein, S.T. Bushby, N.S. Castro, J.M. House, A rule-based fault detection method for air handling units, Energy Build. 38 (12) (2006) 1485–1492, http://dx.doi.org/10.1016/j.enbuild.2006.04.014.
[26] Y. Yu, D. Woradechjumroen, D. Yu, A review of fault detection and diagnosis methodologies on air-handling units, Energy Build. 82 (2014) 550–562, http://dx.doi.org/10.1016/j.enbuild.2014.06.042.
[27] W.J. O'Brien, J. Hammer, M. Siddiqui, O. Topsakal, Challenges, approaches and architecture for distributed process integration in heterogeneous environments, Adv. Eng. Inform. 22 (1) (2008) 28–44, http://dx.doi.org/10.1016/j.aei.2007.08.008.
[28] A. Ahmed, J. Ploennigs, K. Menzel, B. Cahill, Multi-dimensional building performance data management for continuous commissioning, Adv. Eng. Inform. 24 (4) (2010) 466–475, http://dx.doi.org/10.1016/j.aei.2010.06.007.
[29] X. Liu, B. Akinci, M. Bergés, J. Garrett Jr., Exploration and comparison of approaches for integrating heterogeneous information sources to support performance analysis of HVAC systems, 2012, pp. 25–32.
[30] X. Liu, B. Akinci, J.H. Garrett Jr., M. Bergés, Requirements for an integrated framework of self-managing HVAC systems, in: Computing in Civil Engineering, 2011, pp. 802–809. http://dx.doi.org/10.1061/41182(416)99.
[31] V. Bazjanac, D. Crawley, Industry foundation classes and interoperable commercial software in support of design of energy-efficient buildings, in: Proceedings of Building Simulation99, vol. 2, 1999, pp. 661–667.
[32] M. Botts, A. Robin, OpenGIS sensor model language (SensorML) implementation specification, OpenGIS Implementation Specification OGC 7(000), 2007.
[33] N. Dawes, K.A. Kumar, S. Michel, K. Aberer, M. Lehning, Sensor metadata management and its application in collaborative environmental research, 2008 IEEE Fourth International Conference on eScience, IEEE, 2008, pp. 143–150, , http://dx.doi.org/10.1109/eScience.2008.27.
[34] S. Roth, Open green building xml schema: a building information modeling solution for our green world, gbxml schema, 2014.
[35] V. Charpenay, S. Kabisch, D. Anicic, H. Kosch, An ontology design pattern for iot device tagging systems, in: 2015 5th International Conference on the Internet of Things (IOT), 2015, pp. 138–145.
[36] X. Liu, B. Akinci, Requirements and evaluation of standards for integration of sensor data with building information models, in: Computing in Civil Engineering (2009), ASCE, 2009, pp. 95–104.
[37] A. Bhattacharya, J. Ploennigs, D. Culler, Short paper: analyzing metadata schemas for buildings, Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments – BuildSys '15, ACM Press, New York, New York, USA, 2015, pp. 33–34, , http://dx.doi.org/10.1145/2821650.2821669.
[38] E. Holmegaard, A. Johansen, M.B. Kjargaard, Towards a metadata discovery, maintenance and validation process to support applications that improve the energy performance of buildings, 2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops), IEEE, 2016, pp. 1–6.
[39] J.P. Calbimonte, Z. Yan, H. Jeung, O. Corcho, K. Aberer, Deriving semantic sensor metadata from raw measurements, in: Proceedings of the 5th International Conference on Semantic Sensor Networks, vol. 904, SSN'12, CEUR-WS.org, Aachen, Germany, Germany, 2012, pp. 33–48.
[40] E. Holmegaard, M.B. Kjærgaard, Mining building metadata by data stream comparison, in: Proceeding of the 2016 IEEE Conference on Technologies for Sustainability, 2016, pp. 28–33. http://dx.doi.org/10.1109/PERCOMW.2016.7457145.
[41] D. Hong, Q. Gu, K. Whitehouse, High-dimensional time series clustering via cross-predictability, in: A. Singh, J. Zhu (Eds.), Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research, PMLR, Fort Lauderdale, FL, USA, vol. 54, 2017, pp. 642–651.
[42] R. Fontugne, J. Ortiz, D. Culler, H. Esaki, Empirical mode decomposition for intrinsic-relationship extraction in large sensor deployments, in: Workshop on Internet of Things Applications, IoT-App, vol. 12, 2012.
[43] M. Pritoni, A.A. Bhattacharya, D. Culler, M. Modera, Short paper: a method for discovering functional relationships between air handling units and variable-air-volume boxes from sensor data, Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments – BuildSys '15, ACM Press, New York, New York, USA, 2015, pp. 133–136, , http://dx.doi.org/10.1145/2821650.2821677.
[44] D. Hong, H. Wang, K. Whitehouse, S. Art, Clustering-based active learning on sensor type classification in buildings, The 24th ACM International Conference on Information and Knowledge Management, ACM Press, New York, New York, USA, 2015, pp. 363–372.
[45] C. Ellis, J. Scott, I. Constandache, M. Hazas, Creating a room connectivity graph of a building from per-room sensor units, Proceedings of the Fourth ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings – BuildSys '12, ACM Press, New York, New York, USA, 2012, p. 177, , http://dx.doi.org/10.1145/2422531.2422563.
[46] J. Lu, K. Whitehouse, Smart blueprints: automatically generated maps of homes and the devices within them, International Conference on Pervasive Computing, Springer, 2012, pp. 125–142.
[47] D. Hong, J. Ortiz, K. Whitehouse, D. Culler, Towards automatic spatial verification of sensor placement in buildings, in: Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings, ACM, 2013, pp. 1–8. http://dx.doi.org/10.1145/2528282.2528302.
[48] B. Akinci, M. Berges, A.G. Rivera, Exploratory study towards streamlining the identification of sensor locations within a facility, in: Computing in Civil and Building Engineering (2014), ASCE, 2014, pp. 1820–1827.
[49] W.S. Li, C. Clifton, Semantic integration in heterogeneous databases using neural networks, in: VLDB, vol. 94, 1994, pp. 12–15.
[50] E. Rahm, P.A. Bernstein, A survey of approaches to automatic schema matching, VLDB J. 10 (4) (2001) 334–350.
[51] N. Japkowicz, Assessment Metrics for Imbalanced Learning, John Wiley & Sons, Inc., 2013.
[52] G. Forman, M. Scholz, Apples-to-apples in cross-validation studies, ACM SIGKDD Explor. Newsletter 12 (1) (2010) 49, http://dx.doi.org/10.1145/1882471.1882479.
[53] D.J. Hand, R.J. Till, A simple generalisation of the area under the ROC curve for multiple class classification problems, Mach. Learn. 45 (2) (2001) 171–186, http://dx.doi.org/10.1023/A:1010920819831.
[54] T. Fawcett, An introduction to ROC analysis, Pattern Recogn. Lett. 27 (8) (2006) 861–874, http://dx.doi.org/10.1016/j.patrec.2005.10.010.
[55] W.H. Kruskal, W.A. Wallis, Use of ranks in one-criterion variance analysis, J. Am. Stat. Assoc. 47 (260) (1952) 583–621.
[56] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.