Clustering-based Active Learning on Sensor Type Classification in Buildings

Dezhi Hong, Hongning Wang, Kamin Whitehouse
Department of Computer Science, University of Virginia, VA USA
{hong, hw5x, whitehouse}@virginia.edu

ABSTRACT

Commercial and industrial buildings account for a considerable portion of all energy consumed in the U.S., and thus reducing this energy consumption is a national grand challenge. Based on the large deployment of sensors in modern commercial buildings, many organizations are applying data analytic solutions to the thousands of sensing and control points to detect wasteful and incorrect operations for energy savings. Scaling this approach is challenging, however, because the metadata about these sensing and control points is inconsistent between buildings, or even missing altogether. Moreover, normalizing the metadata requires significant integration effort.

In this work, we demonstrate a first step towards an automatic metadata normalization solution that requires minimal human intervention. We propose a clustering-based active learning algorithm to differentiate sensors in buildings by type, e.g., temperature v.s. humidity. Our algorithm exploits data clustering structure and propagates labels to their nearby unlabeled neighbors to accelerate the learning process. We perform a comprehensive study on metadata collected from over 20 different sensor types and 2,500 sensor streams in three commercial buildings. Our approach is able to achieve more than 92% accuracy for type classification with much less labeled examples than baselines. As a proof-of-concept, we also demonstrate a typical analytic application enabled by the normalized metadata.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Algorithms, Performance, Experimentation

Keywords

Active learning, Clustering, Metadata normalization

CIKM'15, October 19–23, 2015, Melbourne, Australia.
© 2015 ACM. ISBN 978-1-4503-3794-6/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2806416.2806574.

1. INTRODUCTION

According to recent reports from U.S. Department of Energy [27, 17], commercial and industrial buildings in the U.S. account for almost 20 percent of the country's total energy use and a good 30 percent of that energy is used "inefficiently or unnecessarily." Reducing this energy usage is a national grand challenge: in 2011, the U.S. government launched the Better Buildings Challenge to make these buildings at least 20 percent more efficient by 2020 [10]. To achieve this goal, many organizations are applying data analytics to the thousands of sensing and control points in a typical commercial building to detect wasteful and incorrect operations.

The analytic-based approach is highly effective but very difficult to scale because the metadata about the type, location, and relationships between the sensing and control units are inconsistent between buildings, or even missing altogether. Generally, the sensing and control points in a building are intended to be used for manual inspection or by handcrafted control loops in a building automation system (BAS); they are not designed for automatic extraction and consumption by external software. As a result, mapping the sensor streams to the inputs of a data analytic engine requires significant integration effort and anecdotally takes about a week or longer for each commercial building. For large organizations that are applying this approach to hundreds or more buildings, such as Microsoft's 88 Acres project [28], this process can take years. Even if this highly manual process is performed once, the need for additional mapping is not necessarily eliminated. New types of metadata will be required as the building is modified or renovated, as the equipment is upgraded, or as new conditions and algorithms are added to the analytics engine.

Building	Point Name
A	`Zone Temp 2 RMI204`
	`spaceTemperature 1st Floor Area1`
B	`SDH_SF1_R282_RMT`
	`SDH_S1-01_ROOM_TEMP`
C	`SODA1R300__ART`
	`SODA1R410B_ART`

Table 1: Example point names for temperature sensors from three different buildings.

In modern commercial buildings, a sensing or control "point" is a sensor measurement, a controller, or a software value, e.g., a temperature sensor installed in an office room. The metadata about the point indicates the physical location, the type of sensor or controller, how the sensor or controller relates to the mechanical systems, and other important contextual information. Most of the time, the metadata is encoded as a "point name", which is usually a short text string with several concatenated abbreviations. Table 1 lists a few point names of sensors in three different building management systems contracted by Trane[1], Siemens[2] and Barrington Controls[3]. For example, the point name `SDH_SF1_R282_RMT` is constructed as a concatenation of the name of the building (`SDH`), the supply fan unit identifier (`SF1`), the room number (`R282`) and the sensor type (`RMT`, room temperature). As the name indicates, this sensor stream measures the temperature in a specific room; and it also reveals the control unit that can affect the temperature in this room. Unfortunately, different naming conventions are used in most buildings due to different equipment, vendors, manufacturers, and contractors being used. As shown in the table, the notion of *room temperature* is encoded with different abbreviations in these three buildings: `Temp`, `RMT` and `ART`. Similar variation exists between buildings on a single commercial campus and even between buildings with the same contractor.

We envision a system that will allow an advanced analytic engine to quickly connect to and analyze the data from a commercial building. It would extract or infer metadata values about sensing and control points to a normalized metadata standard. Such a tool would not only save time but also allow building managers to experiment with many different kinds of analytic engines. Nevertheless, there is limited work on this topic. Bhattarcharya et al. [4] exploit a programming language based solution, where they derive a set of regular expressions from a handful of labeled examples to normalize the point name of sensors. This approach assumes a consistent format for all point names and one single pattern for each type, which is not the case in practice, as shown in Table 1. Schumann et al. [22] develop a probabilistic framework to classify sensor types based on the similarity of a raw point name to the entries in a manually constructed dictionary. However, the performance of this method is limited by the coverage and diversity of entries in the dictionary, and the dictionary size becomes intractable when there exist a lot of variations of the same type, or conflicting definitions of a dictionary entry in different buildings.

In this work, we demonstrate a first step towards an automatic metadata normalization solution that requires minimal human intervention. We focus on a key category of metadata: the type of sensor associated with a sensing point. In contrast to the aforementioned methods, where the annotation process is isolated from model training, we develop a novel active learning based solution to integrate these two processes so as to minimize the manual labeling effort throughout the metadata normalization process. Beyond the traditional active learning solutions [24, 6, 5], we further accelerate the learning process by exploiting the clustering structure of point names. We should note that although the naming schemata vary significantly across buildings, there will be definite variants within the same building, given there are only finite types of sensors deployed in a building. As a result, during active learning, unlabeled examples can be clustered; those in the same cluster are more likely to share the same label and hence do not need to be queried repeatedly. Moreover, the acquired labels can be further propagated to the unlabeled neighbors in the same cluster to expedite classifier training. In our solution, the examples selected for labeling are chosen based on both their representativeness in the cluster and the informativeness of the cluster itself. A Gaussian Mixture Model with Dirichlet Process Prior [20] is used to cluster the instances on the fly to accommodate the dynamic nature of active learning. Label propagation is performed with respect to the connectivity of labeled examples' neighborhood in an adaptive manner.

To investigate the effectiveness of the proposed solution for sensor type classification, we performed extensive experimental comparisons against the state-of-the-art active learning algorithms on a large collection of real sensor stream data, which includes over 20 different sensor types and 2,500 sensors in three different commercial buildings. Our method achieved increased classification performance with reduced amount of manual labels.

Our main contributions in this paper can be summarized as follows:

- We propose a novel, effective yet general active learning approach by exploiting the clustering structure of instances and label propagation to unlabeled neighbors, in order to achieve better convergence rate.
- We evaluate our proposed solution in sensor type classification based on real metadata containing 20 different sensor types and 2,500 sensors in three different commercial buildings; and our solution achieved more than 92% accuracy with at least 16% less manual labels than the state-of-the-art active learning algorithms.
- We also illustrate how data analytics can be performed on top of the automatically normalized building metadata with an example of keyword-based search.

2. RELATED WORK

To the best of our knowledge, there is few existing work addressing the sensor metadata normalization problem studied in this paper. Besides [4, 22] discussed before, there are two major bodies of related work to ours, i.e., active learning in machine learning and schema matching in database.

The main idea behind active learning is that a machine learning algorithm can achieve greater accuracy with fewer labeled training instances if it is allowed to choose the data from which it learns [24]. Therefore, the key research question in active learning is evaluating the informativeness of unlabeled instances for querying. Various solutions have been proposed from different perspectives, including uncertainty reduction [6], query by committee [12], and expected error reduction [2]. In particular, Nguyen and Smeulders also incorporated clustering idea into active learning to select the most informative examples [16]. Their proposed query strategy gives priority to instances that are close to both classification boundary and cluster centroid. Dasgupta and Hsu utilized a hierarchical clustering structure to alleviate sampling bias and improve learning efficiency [7]. They present an algorithm that is statistically consistent and guarantees to have better label complexity than supervised learning. However, in those two methods clustering is performed in an ad-hoc manner, e.g., with predefined cluster size; and neither of them considers the label proximity between adja-

[1] `http://www.trane.com/`

[2] `http://www.siemens.com/`

[3] The company is no longer in business.

cent examples or propagates labels to reduce the amount of labels required for model training.

Automatic schema matching [19] is a classical problem in the database community where correspondences between elements of two schemata are identified as part of the data integration process. Many techniques have been proposed to achieve a partial automation of the match operation for specific application domains. Doan et al. ask a user to provide semantic mappings for a small set of data sources and then train a set of learners with existing machine learning approaches to find the mappings for new data sources [9]. Dhamankar et al. extend [9] to a semi-supervised setting, where domain-specific knowledge is introduced for complex expressions learning [8]. Madhavan et al. exploit a large collection of schemata with known mappings to learn a prior distribution of the elements and their properties [14]. The learned prior distribution is then used as constraints to help a suite of base learners to complete the matching. Though adapting learning techniques, these works mostly focus on offline supervised settings and do not emphasize the efficiency of learning methods, i.e., to reduce manual efforts throughout the learning procedure.

3. METHODOLOGY

In this work, we develop a novel active learning based solution to perform automated sensor type classification in commercial buildings. Because in practice one can get direct access to all point names in a target building, our approach adopts the *pool-based* sampling setting, which selects the most informative instances from the entire collection of unlabeled data points. The data clustering structure is exploited to aid manual labeling of instances and to avoid repeated selection of similar instances. In addition, the acquired labels are propagated to their unlabeled neighbors to accelerate the training process. To accommodate the dynamic nature of active learning, we appeal to a non-parametric Bayesian approach to identify the clustering structure, and perform label propagation in an adaptive manner.

3.1 Clustering-based Active Learning

Formally, the problem of using active learning for automated sensor type classification can be described as follows. For a given collection of unlabeled point names $\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$, in which the text string of point name x_i is represented as a d dimensional feature vector, we aim at learning a classifier $f : f(x) \rightarrow y$, with respect to a set of labeled instances $\mathcal{D}_l = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ acquired during classifier training. In particular, y_i is the true sensor type for the point name x_i and takes value from a set of predefined labels. Our goal of learning is to maximize $f(x)$'s classification accuracy on the future testing data while minimizing the size of $\mathcal{D}_l$. We should note that our proposed solution has no assumption about the classifier $f(x)$; any supervised multi-class classifier can be used in our method.

Conventional active learning methods mostly focus on efficient search through the hypothesis space. Each time a new label is added to $\mathcal{D}_l$, the set of hypothesis space shrinks, e.g., filters the classifiers that are inconsistent with the labels seen so far. Great research attention has been devoted onto designing effective query strategies for label acquisition. Typical solutions include uncertainty-based sampling [6], query by committee [12], and expected model change [2].

However, such solutions implicitly assume unlabeled instances are independent, and thus fail to exploit any information conveyed in the density of data, i.e., the marginal distribution of $p(x)$ is assumed to be uniform. However, if the unlabeled instances in $\mathcal{D}$ form clusters, i.e., their clustering membership is consistent with the underlying class labels, one will only need one label from each cluster to estimate a perfect classifier. Although this example is overly optimistic, the clustering structure can still unveil the representativeness of instances with respect to their neighborhood. Hence special emphasis should be given to those most representative instances while labeling them. In addition, because instances in the same cluster are more likely to share the same label [31], one can accelerate active learning by avoiding labeling instances from the same cluster and propagating labels to the nearby unlabeled instances.

In our solution, the density of unlabeled instances, i.e., $p(x)$, is exploited via its clustering structure. In particular, we use a probabilistic mixture model to identify the latent clusters, e.g., $p(x) = \sum_c p(x|c)p(c)$, where c denotes a cluster label. The detailed instantiation of this mixture model, e.g., how to decide the cluster size and the specification of cluster conditional likelihood, will be discussed in Section 3.3. With the identified clustering structure at hand, we devise a divide-and-conquer strategy for selecting the query instances: 1) it should come from the most *informative* cluster; and 2) it should be *representative* in that cluster. This query strategy focuses on the entire input space; comparing to those individual instance based selection methods, it can thus help to avoid querying outliers or repeatedly querying similar instances.

To quantify the "informativeness" of a cluster, we exploit an information theoretic metric - class entropy $H(c)$. For every cluster c, we apply $f(x)$ to predict the labels for all the unlabeled instances in it. Then based on these predicted labels, we compute the class entropy for this cluster as,

$$H(c_i) = -\sum_{y \in Y_{c_i}} p(y) \log p(y) \tag{1}$$

where Y_{c_i} is the set of unique labels in cluster c_i predicted by classifier $f(x)$.

A cluster with larger class entropy indicates increased discrepancy between the current classifier's prediction and clustering structure inferred from the density of unlabeled instances. Therefore instances from such a cluster are considered potentially more helpful in introducing new information to classifier training. In addition, the size of a cluster is also an important criterion for measuring its informativeness. When multiple clusters have similar class entropy, a larger cluster will indicate more uncertainty of class labels in it. Hence acquiring a label for such a cluster can mostly reduce the classifier's uncertainty on a larger portion of instances. Combining these two aspects, we locate the cluster of choice by the product of cluster proportion $p(c)$ and class entropy $H(c)$ as follows,

$$\hat{c} = \arg\max_c p(c)H(c) \tag{2}$$

Once we locate the candidate cluster $\hat{c}$, we need to choose the most "representative" instance from it for labeling. We use the conditional likelihood $p(x|\hat{c})$ over all the unlabeled instances to select such an instance. The intuition behind this choice is that the instance with the maximum conditional likelihood best captures the homogeneity of instances in the selected cluster, such that knowing its label will provide a substantial boost for the classifier to predict the class labels within this cluster.

In addition, we also view high class entropy in the selected cluster as an indication of low resolution of clustering results in that local region: classification boundary goes inside the cluster. To reduce this divergence between predicted classes and clustering results, we need to further separate this cluster into finer clusters. The benefit of this sub-clustering is that the class distribution estimated by the classifier is introduced into clustering; this helps generate more homogeneous clusters for later instance selection.

3.2 Label Propagation

The basic assumption in our clustering-based active learning solution is that instances in the same cluster tend to share the same class label. The query strategy described in the previous section exploits this assumption to avoid repeated selection of similar instances. In this section, we will further capitalize on this assumption to propagate the acquired labels to their nearby unlabeled neighbors to reinforce our current knowledge about the underlying class distribution.

The idea of label propagation is popularized in transductive learning [31, 30, 3], where class labels are propagated from the labeled instances to their unlabeled neighbors based on the structure of data manifold. In practice, the data manifold is approximated by the nearest neighbors of each instance derived from data features. To empirically validate the feasibility of applying label propagation in our problem, we randomly selected a small portion of labeled sensor point names in our evaluation corpus, computed the Euclidean distance between all pairs of instances based on their feature vectors, and grouped them into pairs from the same class and different classes. We plot the cumulative distributions of those three types of distances in Figure 1.

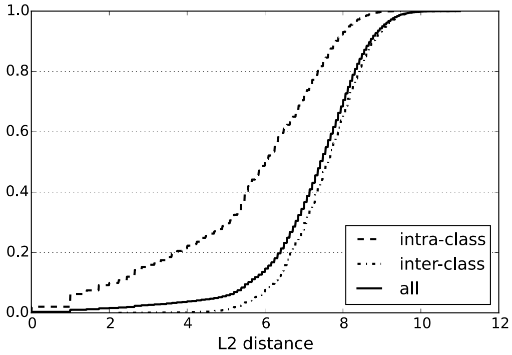


Figure 1: Distribution of pairwise distance between instances from the same class and different classes.

As we can clearly notice from the results, there is a clear gap between the cumulative distribution of distances between instances from the same class and those from different classes. This result also confirms our assumption in the proposed clustering-based active learning that nearby instances tend to share the same class label. Therefore, with proper choice in constructing the data manifold, we can confidently propagate the labels to their closest unlabeled neighbors and use them to update the classifier. This label propagation will amplify the importance of acquired labels and better avoid repeatedly querying of similar instances.

A typical approach to construct the data manifold for label propagation in transductive learning is to look for the k nearest neighbors of each instance. However, in our solution it is challenging to find a universal setting of k. To accommodate the clustering structure, label propagation should only be performed within the same cluster. However, since the resulting clusters vary and change across iterations, a fixed setting of k might lead to inconsistency between the data manifold and clustering results. To address this issue, we introduce a distance threshold r to define the data manifold for label propagation in our active learning process. When the label y is acquired for instance x, all the unlabeled instances located within the distance r to x will be assigned the same label y. Those instances will be then removed from $\mathcal{D}$ and added to a collection of propagated label set $\mathcal{D}_p$ for later classifier training.

The optimal threshold r should be the minimum inter-class distance between any pair of instances. However, it is impossible to calculate without knowing the true class labels. In our solution, we keep an estimation of r from all the labeled instances in $\mathcal{D}_l$ during active learning,

$$r = \underset{(x_i,y_i),(x_j,y_j)\in\mathcal{D}_l}{\arg\min} \frac{d(x_i,x_j)}{2}, \;\; with \;\; y_i \neq y_j \qquad (3)$$

$d(x_i, x_j)$ is the Euclidean distance between x_i and x_j.

We divide the minimum distance by 2 to avoid possible overlapping of label propagation between two labeled instances. The adaptive estimation of distance threshold defined in Eq (3) will shrink during active learning and therefore refines the propagated labels. Accordingly, we need to correct previously propagated labels where the estimation of r was less accurate. In particular, when we have a new estimation of r, we will remove instances from $\mathcal{D}_p$ that fall outside the region defined by the latest r to any closest labeled neighbors with the same label, and put them back into the unlabeled set $\mathcal{D}$. This correction happens before the new round of classifier training.

3.3 Clustering with Non-parametric Bayesian

In our proposed clustering-based active learning, data density is exploited via its latent clustering structure. Since we assume instances in the same cluster tend to share the same class label, we choose Gaussian Mixture Model (GMM) [32], a partitional clustering algorithm, to perform the clustering.

In GMM, the cluster label for every instance is treated as a latent variable, which is drawn from a multinomial distribution $p(c)$, i.e., $p(c) \propto \alpha_c$, where $\forall c, \alpha_c \geq 0$ and $\sum_c \alpha_c = 1$. In any given cluster c, the conditional data likelihood of an instance x is specified by a multivariate Gaussian distribution. To reduce the number of parameters to be estimated, we choose the isotropic Gaussian in our solution,

$$p(x|c) = (2\pi\sigma^2)^{-d/2} \exp - \frac{(x-\mu_c)^{\mathsf{T}}(x-\mu_c)}{2\sigma^2} \qquad (4)$$

where the variance σ^2 is shared by all the clusters. $\{\alpha_c, \mu_c\}_{c=1}^{k}$ and σ are considered as model parameters in GMM.

However, in GMM, we need to manually specify the number of clusters for a given input data set; and the clustering result of GMM is very sensitive to such setting. More importantly, in our solution, we also need to perform subclustering whenever we encounter a cluster contradicting with the current classifier's prediction. It is impossible for us to predefine the size of those sub-clusters during active learning. To make clustering feasible in our solution, we appeal to a non-parametric Bayesian solution: we assume the model parameters (α, μ) in each cluster are random variables, which are drawn from a Dirichlet Process prior [11].

A Dirichlet Process $DP(G_0, \eta)$ with a base distribution G_0 and a scaling parameter η is a distribution over distributions [11]. The base distribution G_0 specifies the prior distribution of model parameters, e.g., mean parameter μ in each cluster, and the scaling parameter η specifies the concentration of samples drawn from DP, e.g., cluster proportion $p(c)$. An important property of the DP is that though the draws from a DP have countably infinite size, they are discrete with probability one, which leads to a probability distribution on partitions of data. The number of unique draws, i.e.,the number of clusters, varies with respect to the data and therefore is random, instead of being pre-specified.

As a result, with the introduced $DP(G_0, \eta)$ prior, data density in a given collection of instances can be expressed using a stick-breaking representation [23]:

$$p(x) = \sum_{c=1}^{\infty} \alpha_c \mathcal{N}(x|\mu_c, \sigma) p(\mu_c|G_0) \tag{5}$$

where $\alpha = \alpha_{c=1}^{\infty} \sim Stick(\eta)$ represents the proportion of clusters in the whole collection. The stick-breaking process $Stick(\eta)$ for the cluster proportion parameter α is defined as: $\alpha'_c \sim Beta(1, \eta), \alpha_c = \alpha'_c \prod_{i=1}^{c-1}(1 - \alpha'_i)$. Since the variance σ^2 is fixed in all clusters, we use a conjugate prior for μ in G_0, i.e., for $\forall c, \mu_{ci} \sim \mathcal{N}(a, b)$, with the assumption that each dimension in μ_c is independently drawn from a univariate Gaussian. This will greatly simplify the later on inference procedure.

Because the data density distribution defined in Eq (5) only has finite support at the points of $\{\alpha_c, \mu_c\}_{c=1}^{k}$, we can calculate the posterior distribution of latent cluster labels in each unlabeled instance to discover the clustering structure for active learning. Following the sampling scheme proposed in [15], we appeal to a Gibbs sampling method to infer the posterior of cluster membership. Detailed specifications of this sampling algorithm can be found in [15]. In particular, we use the same hyper-parameter setting of (a, b) in G_0 and η for initial clustering and subsequent sub-clustering during our cluster-based active learning process.

Putting it all together: Algorithm 1 summarizes our clustering-based active learning solution for the sensor type classification problem. In each iteration, we select the most "informative" cluster measured by the class entropy of predicted labels by the classifier. We acquire a label for the instance centered in the selected cluster, and propagate the newly obtained label to its unlabeled neighbors within an estimated distance. At the end of this iteration, we perform sub-clustering on the selected cluster to refine its local clustering structure for later instance selection.

3.4 Discussion

Comparing to the existing solutions for the sensor type classification, our proposed algorithm addresses the problem from a totally different perspective. As we discussed before, because existing solutions [4, 22] isolated the annotation process from model training, it likely leads to wasted effort in data annotation. With active learning, we can restrict our effort to a smaller set of instances that are the most helpful for classifier training. A practical benefit of this learning approach is that we can rapidly bootstrap the building analytic software across different sites. Instead of going back and forth for several rounds of blind annotation, a building manager only needs to interact with the system for several iterations on the fly to achieve immediate satisfactory results.

Algorithm 1: Clustering-based Active Learning

```
Input: point names $\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$, and label budget $B$
Output: predicted labels of the point names $Y$
Initialize: Generate clusters with $DP(G_0, \eta)$ on $\mathcal{D}$, reset the labeled instance set $\mathcal{D}_l$ and propagated label set $\mathcal{D}_p$ to empty
while $B > 0$ do
    Train the classifier $f(x) \to y$ based on $\mathcal{D}_l$ and $\mathcal{D}_p$;
    Apply $f(x)$ to all instances in $\mathcal{D}$;
    Compute class entropy $H(c)$ defined in Eq (2) for each cluster $c$;
    Retrieve a cluster by $\hat{c} = \arg\max_c p(c)H(c)$;
    Acquire label $\hat{y}$ for example $\hat{x}$ given by $\arg\max_{x \in \mathcal{D}} Pr(x|\hat{c})$, and move $(\hat{x}, \hat{y})$ from $\mathcal{D}$ to $\mathcal{D}_l$;
    Update the distance threshold $r$ by Eq (3);
    Update all previous propagated label in $\mathcal{D}_p$ with new threshold $r$;
    Assign $\hat{y}$ to all unlabeled examples $x_u$, where $x_u \in \hat{c}$ and $d(x_u, \hat{x}) < r$;
    Move $(x_u, y)$ from $\mathcal{D}$ to $\mathcal{D}_p$;
    Perform sub-clustering with $DP(G_0, \eta)$ in $\hat{c}$;
end
```

Comparing to the other active learning algorithms, our proposed method focuses on exploiting information conveyed in the data clustering structure. Based on the assumption that nearby instances are more likely to share the same class label, we select the instances for labeling based on the identified latent clustering structure of instances and propagate the labels to their adjacent unlabeled neighbors. This helps us avoid repeatedly querying similar instances and enhance the importance of labeled instances. In addition, the clustering structure and label propagation strategy are adapted in an online fashion. In Nguyen and Smeulders's work [16], clustering is also exploited for active learning; but their clustering structure is static and no label propagation is performed. In Dasgupta and Hsu's work [7], hierarchical clustering is used to provide a more flexible and dynamic clustering structure, but no label propagation is performed as well. We should note this proposed active learning algorithm is general, it only assumes the proximity of label distribution in nearby instances. Therefore, this algorithm can also be applied in a broader context, e.g., document categorization [26] and image retrieval [25].

4. EVALUATION

To demonstrate the effectiveness of our proposed solution in addressing sensor type classification, we evaluate our clustering-based active learning algorithm on point names collected from three distinct buildings. Extensive experimental comparisons confirm that our method achieved the

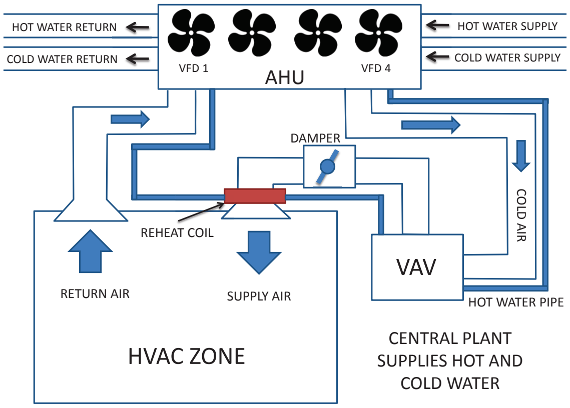


Figure 2: A typical HVAC system consisting of an air handler unit (AHU), several variable air volume boxes (VAV), water-based heating/cooling pipes and air circulation ducts. (Figure used with permission from the authors of [1].)

same classification accuracy with much fewer labeled examples than all the baseline methods. We also demonstrate how metadata normalization can enable meaningful analytic applications with the raw sensor streams under the context of commercial building energy control and comfort assessment.

4.1 Background & Building Sensor Taxonomy

Figure 2 illustrates a typical heating, ventilation, and air conditioning (HVAC) system deployed in modern commercial buildings. An HVAC system usually uses a combination of hot and cold water pipes in conjunction with air handler units (AHU) to maintain the appropriate thermal environment within the building. An HVAC system usually consists of several AHUs and each AHU is responsible for a physical zone in the building. An AHU consists of variable speed drives that supply cold air (cooled by the supplied cold water) using ducts to VAV boxes distributed throughout the building. The hot water loop is also connected to these VAV boxes using separate pipes. Each VAV box controls the amount of air to be let into an HVAC zone using dampers, whose opening angle can be programmed. A reheat coil, which uses supplied hot water, is used to heat the air to meet the appropriate HVAC settings for each zone.

Our evaluation data set is collected from sensor streams from over 2,500 sensors of more than 20 different types deployed in three commercial buildings. Specially, Building A uses a building management system contracted with Trane and building B comprises deployment by KETI[4] and Siemens, while building C uses an archaic system by Barrington Controls.

Table 2 summarizes all the types of sensors evaluated in these three buildings and the number of sensors of each type. For example, "room temperature" measures the temperature in room and for a better understanding, all the other temperature measurements on water circulation and air ventilation are illustrated in Figure 2. For setpoints, we assign only one general type which includes all set points for every actuator configured in the building.

	Building		
Type	A	B	C
CO_2	16	52	0
Humidity	54	52	0
Air Pressure	142	216	215
Room Temp	159	231	208
Facility Operation Status	59	72	41
Facility Control	0	138	403
Setpoint	140	486	229
Air Flow Volume	14	172	9
Damper Position	0	290	10
Fan Speed	0	25	15
HW Supply Temp	27	1	0
HW Return Temp	15	1	0
CW Supply Temp	18	2	11
CW Return Temp	15	3	10
Supply Air Temp	20	17	3
Return Air Temp	6	2	4
Mixed Air Temp	5	2	3
Ice Tank Entering Temp	1	2	0
Ice Tank Leaving Temp	1	4	0
Occupancy	25	52	0
Timer	0	0	15
Sum	575	1124	1166

Table 2: Number of points by type for the 3 test buildings. "Temp" stands for "temperature", "HW" for "hot water" and "CW" for "cold water".

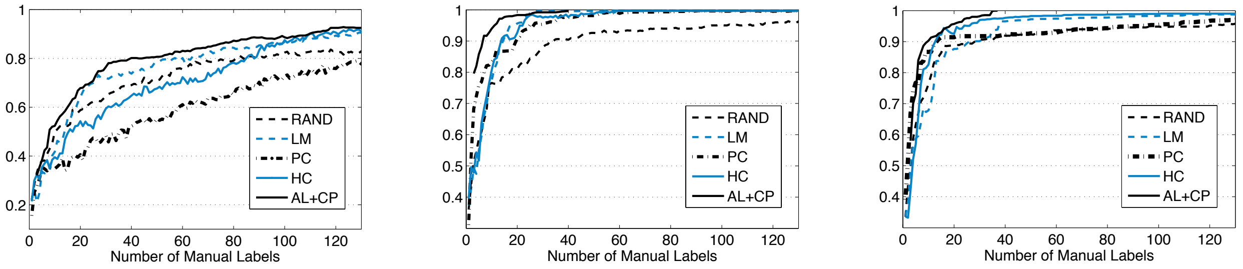
4.2 Feature Construction

The sensor point names are the input of our active learning algorithm. As shown in our motivating example in Table 1, the point names are short text strings with several concatenated abbreviations. To represent the primitive point names as feature vectors for classifier training, we first convert all point names to lower cases and trim out the numerical characters, resulting in a series of words, e.g., `Zone Temp 2 RMI204` becomes `{zone, temp, rmi}`. To capture possible variants of abbreviations in point names, e.g., "tmp" and "temp" for temperature, we adopt k-mers [13] as our features. The term k-mer refers to all the possible substrings of length k, which are contained in a string. This feature is popularly used in protein and gene sequence analysis in bioinformatics. And it helps measure sequence similarity without alignment. In our case, we limit the k-mers computation only within a word boundary. In general, having too small a k will increase the chance of overlapping k-mers, making the points less differentiable. Therefore, we compute k-mers of length 3 and 4 for all point names. For example, `{zone, temp, rmi}` will yield a set `{zon, one, tem, emp, rmi}` with k=3. A dictionary of k-mers is constructed with all the k-mers generated from each point name. Each point name is then converted into a feature vector based on the frequency of k-mers in it. For example, a set of k-mers `{zon, tem, emp, zon}` will be transformed into a vector of `(2,0,1,1,0)` with the dictionary `{zon, one, tem, emp, rmi}`. This feature representation will be used in our later evaluations.

4.3 Baselines

To evaluate the performance of our proposed algorithm, we adopt four active learning algorithms as baselines.

[4] `http://www.keti.re.kr/e-keti/`



(a) Building A

(b) Building B

(c) Building C

Figure 3: Classification accuracy on three different buildings: comparing to the baselines, our method (AL+CP) is able to achieve better accuracy on all buildings with less labeled examples.

	Building A			Building B			Building C		
Labeled Percentage	1%	5%	10%	1%	5%	10%	1%	5%	10%
RAND	0.204	0.582	0.696	0.711	0.933	0.970	0.795	0.915	0.937
LM	0.222	0.572	0.769	0.736	0.995	0.995	0.835	0.980	0.988
PC	0.232	0.515	0.581	0.752	0.958	0.968	0.762	0.862	0.987
HC	0.326	0.516	0.669	0.829	0.993	0.996	0.867	0.977	0.987
AL+CP	**0.416**[*]	**0.765**[*]	**0.819**[*]	**0.919**[*]	**1.000**[*]	**1.000**[*]	**0.890**[*]	**1.000**[*]	**1.000**[*]

[*] p-value<0.01

Table 3: Weighted macro F1 score of classification on three buildings with different labeling budgets. Our method (AL+CP) converges with less than 5% examples being labeled in Building B and C. It consistently outperforms the baselines in all cases.

Random (RAND): this method selects an example at random uniformly from the unlabeled set in each iteration.

Least Margin (LM) [21]: this method adopts the a simple yet effective sampling strategy, which queries the instances for labeling with least confidence measured by the difference of posterior probability of the first and second most probable class labels predicted by the classifier:

$$x^*_M = \arg\min_x \; p(\hat{y}_1|x) - p(\hat{y}_2|x),$$

where $\hat{y}_1$ and $\hat{y}_2$ are the first and second most probable predicted class labels, respectively. We compute the class prediction probability by SVM based on the method proposed in [18].

Pre-clustering (PC) [16]: this method pre-clusters the instances and selects the example for querying satisfying two criteria: 1) locate at the classification boundary and 2) representative of dense clusters. The clusters are constructed by the same Gaussian Mixture Model with Dirichlet Process prior as used in our method.

Hierarchical Clustering (HC) [7]: this method leverages a hierarchical clustering structure, which is represented as a binary tree, and estimates the purity of labels for each cluster node. The algorithm iteratively selects examples from a subtree whose size is significantly large or the impurity of labels is high.

In all baselines, and our method, a linear SVM model is used as the classifier.

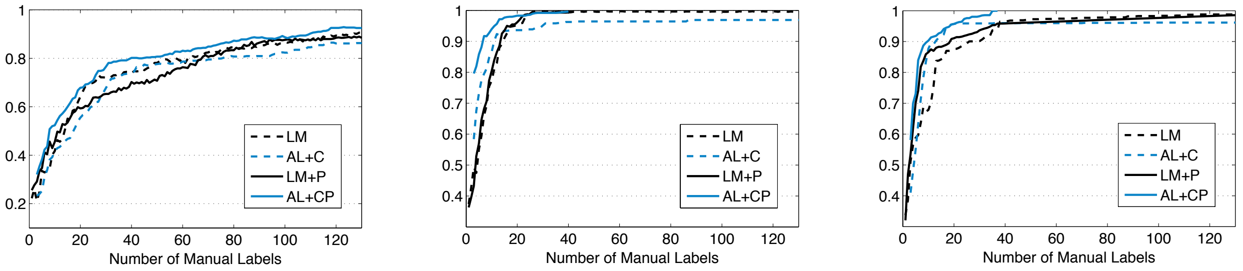
We measure the overall classification accuracy of each active learning algorithm with different amount of manual labels, and examine how many examples are needed by each method to reach a required accuracy level. Besides classification accuracy, we also compute the weighted macro F1 score of each method given different labeling budgets (e.g., 5%, 10%, etc of the entire unlabeled set) in each building.

In our experiments, to reduce possible bias introduced by training/testing split, we perform 10-fold cross validation for each active learning method, and repeat it 10 times with different random seeds. The average performance of 10 runs from each method is reported.

4.4 Classification Results

In Figure 3, we illustrate the comparison results of sensor type classification accuracy from all three buildings over all methods. In all three buildings, our method performed the best against all the baselines. In particular, our method consistently requires the least amount of manual labels to achieve a satisfactory and converged accuracy. In building A, to achieve the accuracy of 90%, our clustering based active learning method (denoted as AL+CP) requires 105 labeled examples while HC needs 110, which is the best among all baselines. LM takes 127 labeled examples and the other two baselines take a hundred more labeled examples to obtain the same accuracy. In building B, both our method and LM reach the accuracy of 99% with 26 examples, while it takes 51, 61 and more than 200 examples, respectively, for the HC, PC and RAND baselines. In building C, our method achieves 99% accuracy with 35 labels while HC and LM requires 101 and 135 labels, respectively, for the same accuracy. Again, PC and RAND require much more labeled examples to get the same performance.

On all three buildings, RAND performs reasonably on initial iterations (e.g., before 20) but becomes the worst later on. This is because of imbalanced class distribution in our three data sets. RAND selects more examples from larger classes at the beginning and quickly gets a fraction of the major classes correctly classified. However, in later iterations, RAND has a lower chance in picking examples from smaller classes and converges to a local optimal quickly.



(a) Building A

(b) Building B

(c) Building C

Figure 4: Comparison of margin-based active learning (LM), clustering-based active learning without label propagation (AL+C), margin-based active learning with label propagation (LM+P), and clustering with label propagation based active learning (AL+CP). The label propagation significantly improves the performance of learning with clustering only.

The PC baseline, which also exploits data clustering structure, performs significantly worse than other baselines for Building A. PC gives priority to the examples that are: 1) close to the decision boundary, and 2) representative in a cluster. Such criteria are very similar to those in our method. Further inspection reveals that examples selected by the PC baseline were neither informative nor representative enough to distinguish different types of instances. This stems from the fact that for building A, the initial clusters from GMM fairly overlap with each other. This biases the PC baseline to select more examples with relatively high uncertainty but less representative in a cluster.

The HC baseline is quite effective in avoiding querying similar examples from dense areas. However, because HC does not perform label propagation, after all clusters become roughly equally pure with sufficient sampling, it will start repeatedly querying previously sampled areas, which results in a long plateau for building B and C. For building A, the larger classes contain a few variations in their point names, therefore when HC avoids sampling from the same dense area at first, it misses important informative examples and is only able to pick them up in later iterations.

Moreover, we notice that, for all methods, the convergence rate on building B and building C is much better than building A, despite the fact that there are more points in those two buildings. The reason is that the point names in these two buildings share stronger regularities - the last segment always indicates the sensor type. Recall the examples shown in Table 1: `SDH_SF1_R282_RMT` from building B and `SODA1R410B_ART` from building C. Both of them adhere to the same order of segment categories - `building name-air handler/supply fan identifier-room number-sensor type`, only with a slightly different set of delimiters. Our string feature captures such regularity and helps the learning algorithm converge faster. However, the same rule doesn't apply to building A because there the type information can be encoded in any segment in a point name. For instance, a temperature measurement can be named in many conventions, such as `averageSpaceTemperature 1st Floor Area1`, `East Space Temperature scc2UIP33` or `RM511A Zone Temp`. With such variety in the naming conventions, the learning algorithms need to explore all possible variants of point names for convergence in building A.

On such imbalanced data sets, investigation of accuracy only is not enough. We also measure the weighted macro F1 score of classification for each method with different labeling budgets, i.e., different percentage of total instances that can be labeled. We examine the F1 score at three different labeling budgets, 1%, 5% and 10%. The weighted macro F1 score is an altered version of macro F1 score [29], which calculates the F1 score for each class, where "one-versus-all" binary classification is performed, and weight the resulting F1 of each class by support (the number of true instances for each label). Paired two sample t-test is performed to validate the statistical significance of improvement from our method over the best-performing baseline. The results are shown in Table 3. In general, more labeled examples leads to better classification performance over all classes; and our method performed the best in all cases. As we discussed earlier, all methods converge much faster in building B and C than in building A. In those two buildings, our method can achieve an 100% classification accuracy with slightly more than 1% of total examples being labeled.

In addition, to investigate the contributions of data clustering and label propagation in our proposed active learning algorithm, we also conduct experiments by disabling data clustering and label propagation in our method. From the results in Figure 4, we can observe that with clustering only, the performance of our method is no better than margin-based active learning; and beyond a certain point, the performance plateaus because the classification boundary converges to the clustering structure, and no more new examples could be selected for labeling. However, we do observe some improvement from clustering only over margin-based active learning for building B and building C in early iterations. This is because the same type of points in these two buildings contain few variations and these initial examples at the center of relatively pure clusters are more representative for a dense region. Interestingly, after adding label propagation, we clearly see a boost in the performance than performing clustering only. We attribute the improvement of our method to label propagation which considerably amplifies the amount of available labeled examples for classifier training, and therefore helps to estimate a better classifier.

Besides, we also examine how the label propagation alone can help active learning in general. We adopt the same distance threshold estimation method as defined in Eq (3) to introduce propagated labels in the margin-based baseline and denote it as LM+P in Figure 4. On building A, the inclusion of label propagation makes LM even worse, because in early stage the estimated threshold r in LM+P is not accurate enough, and it mistakenly propagates the labels. Only when more labels are acquired in later stages, the estimation of distance threshold r improves and therefore LM receives propagated labels with improved quality. In our method, because of data clustering, the labeled examples in the early stage are more representative; it in turns helps better estimate the distance threshold r. As a conclusion, data clustering and label propagation complement with each other in our proposed method, and help active learning acquire more informative examples quickly.

4.5 Proof-of-Concept Study

As a first step towards automated metadata normalization, our algorithm is able to classify and transform the type information in the primitive metadata into a normalized name space. It provides a better opportunity for running uniform analysis on heterogeneous metadata across buildings with different management systems. As a proof-of-concept, we demonstrate an illustrative example, in which we search over the normalized metadata for different types of sensors that are in the same room under the application context of building comfort assessment and energy control.

Ideally, with the occupancy information of spaces considered, a properly configured building should automatically stop conditioning unoccupied rooms and areas. However, one typical problem plaguing buildings that incurs energy waste is unoccupied room being conditioned. For a building manager to inspect the building and locate such spots for better scheduling, he should be able to run simple keyword based searches over the metadata to look for different types of streams, such as room temperature and occupancy streams. With the desired types of streams returned, he can match different types of streams by room location to perform further examination, e.g., which room is still comfortable during unoccupied periods.

To accomplish the task, we first classify the sensor streams by type with both our method and the best baseline (LM). Then based on the predicted sensor type for each stream, we normalize each type of sensors into a common name space, e.g., all the streams predicted as room temperature are named as “room temperature”. With the type information normalized, we can simply retrieve all the streams of a certain type with one keyword, e.g, using “temperature” to search for temperature streams. The next step is to match different types of sensors by room location. In particular, we derive a few regular expressions to find the segment indicating room location in the point names of each sensor.

In this experiment, we search for four types of sensors - occupancy, temperature, humidity and CO_2 - by simply using these type names as the keywords. Then we match the room location of these four groups with regular expressions on their original primitive point names. Specifically, we examine the accuracy of three searches: 1) occupancy and temperature in the same room; 2) occupancy, temperature and humidity in the same room; and 3) occupancy, temperature, humidity and CO_2 in the same room, given these combinations are usually used to assess the comfort of a room or identify potential waste in unoccupied rooms.

Table 4 illustrates the performance of basic searches on building A for occupancy (O), temperature (T), humidity (H) and CO_2 (C). Our method can return better results with both higher recall and precision because of better predictions on type class for the streams. Given the four returned groups of streams, we further run regular expression based matching on the primitive metadata to find the pairs in the same room. For instance, we first identify the room location of each returned occupancy stream with regular expressions, then we search for the temperature stream with the same room number as each of these occupancy streams. Similarly, we search for humidity and CO_2 streams in the same room. With these different types of streams grouped into the same room, a building manager can compare the actual data against some standards to decide if a room is problematic or not. We summarize the search results in Table 5. In general, the normalized metadata by our method produces search results with higher recall and slightly lower precision. Such results are expected considering the practical demand from building managers - it is acceptable to return some unexpected results (false positives) while not missing the expected ones (false negatives), where he can manually examine a much smaller group of candidates and filter out the incorrect ones. We conclude that with normalized metadata for a building, people such as a building manager can more easily retrieve expected streams and conduct meaningful analysis to identify problems in a building.

	AL+CP		LM	
	Precision	Recall	Precision	Recall
O	0.930	1.000	1.000	0.688
T	0.892	0.935	0.635	0.975
H	0.962	1.000	0.778	0.519
C	0.556	0.833	0.222	0.286

Table 4: The performance of searches over the normalized metadata with our method (AL+CP) and the best baseline (margin based active learning, LM). We search for four specific types of streams: occupancy (O), temperature (T), humidity (H) and CO_2 (C).

	AL+CP		LM	
	Precision	Recall	Precision	Recall
O+T	0.832	1.000	1.000	0.900
O+T+H	0.926	1.000	1.000	0.400
O+T+H+C	0.375	0.833	1.000	0.125

Table 5: The performance of searches for different types of streams that are in the same room. We consider the pair of occupancy and temperature (O+T), the combination of occupancy, temperature and humidity (O+T+H), and also the combination of occupancy, temperature, humidity along with CO_2 (O+T+H+C).

5. CONCLUSION & FUTURE WORK

In this paper, as a first step towards automated sensor metadata normalization, we investigate the problem of building sensor type classification and introduce a novel, effective yet general active learning method to address the problem. Following the assumption that similar instances are more likely to share the same class label, our solution exploits the data clustering structure and propagates the labels to their nearby unlabeled examples to accelerate the learning process. Extensive experimental comparisons are performed against several state-of-the-art active learning algorithms with over 20 different sensor types and 2,500 sensor streams collected from three buildings. Our proposed solution is able to achieve satisfactory classification accuracy with much less labeled examples than the baseline algorithms. In addition, we also demonstrate that the normalized metadata can potentially enable meaningful analytic applications with the raw sensor streams under the context of commercial building comfort assessment and energy control.

Our proposed active learning algorithm is general, and therefore it is applicable in a broader context, e.g., document categorization [26] and image retrieval [25]. In our current metadata normalization process, only the text features from point names are utilized. However, another important aspect of the sensor streams is the actual data from the sensor readings. Features constructed from such raw signals can also be introduced to characterize the stream. Such feature becomes vital when the point names are missing or corrupted. In addition, our current problem setting is limited to one building; it is necessary for us to solve the normalization problem across buildings, e.g., classification model learned in one building can be used to bootstrap the learning in another building.

6. REFERENCES

[1] B. Balaji, J. Xu, A. Nwokafor, R. Gupta, and Y. Agarwal. Sentinel: Occupancy based hvac actuation using existing wifi infrastructure within commercial buildings. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, SenSys '13, pages 17:1–17:14, 2013.

[2] M.-F. Balcan, A. Broder, and T. Zhang. Margin based active learning. In *Proceedings of the 20th Annual Conference on Learning Theory*, COLT'07, pages 35–50, Berlin, Heidelberg, 2007. Springer-Verlag.

[3] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *The Journal of Machine Learning Research*, 7, 2006.

[4] A. Bhattacharya, D. E. Culler, J. Ortiz, D. Hong, and K. Whitehouse. Enabling portable building applications through automated metadata transformation. Technical Report UCB/EECS-2014-159, EECS Department, University of California, Berkeley, Aug 2014.

[5] D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine learning*, 15(2), 1994.

[6] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *J. Artif. Int. Res.*, 4(1):129–145, Mar. 1996.

[7] S. Dasgupta and D. Hsu. Hierarchical sampling for active learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 208–215, New York, NY, USA, 2008. ACM.

[8] R. Dhamankar, Y. Lee, A. Doan, A. Halevy, and P. Domingos. imap: Discovering complex semantic matches between database schemas. In *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*, SIGMOD '04, pages 383–394, New York, NY, USA, 2004. ACM.

[9] A. Doan, P. Domingos, and A. Y. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, SIGMOD '01, pages 509–520, New York, NY, USA, 2001. ACM.

[10] U. DOE. Better buildings challenge. *http://www4.eere.energy.gov/challenge/sites/default /files/uploaded-files/may-recognition-fs-052013.pdf (Feb. 26, 2014)*, 2013.

[11] T. S. Ferguson. A Bayesian Analysis of Some Nonparametric Problems. *The Annals of Statistics*, 1, 1973.

[12] Y. Freund, H. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2-3):133–168, 1997.

[13] C. S. Leslie, E. Eskin, A. Cohen, J. Weston, and W. S. Noble. Mismatch string kernels for discriminative protein classification. *Bioinformatics*, 20(4), 2004.

[14] J. Madhavan, P. A. Bernstein, A. Doan, and A. Halevy. Corpus-based schema matching. In *Proceedings of the 21st International Conference on Data Engineering*, ICDE '05, pages 57–68, Washington, DC, USA, 2005. IEEE Computer Society.

[15] R. Neal. Markov chain sampling methods for dirichlet process mixture models. *Journal of computational and graphical statistics*, 9(2), 2000.

[16] H. T. Nguyen and A. Smeulders. Active learning using pre-clustering. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 79–86, New York, NY, USA, 2004. ACM.

[17] U. D. of Energy Better Buildings program. Total annual cost of energy in the commercial and industrial sector. Mar. 2015.

[18] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *ADVANCES IN LARGE MARGIN CLASSIFIERS*. MIT Press, 1999.

[19] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *the VLDB Journal*, 10(4), 2001.

[20] C. E. Rasmussen. The infinite gaussian mixture model. In *NIPS*, volume 12, 1999.

[21] T. Scheffer, C. Decomain, and S. Wrobel. Active hidden markov models for information extraction. In *Advances in Intelligent Data Analysis*. Springer, 2001.

[22] A. Schumann, J. Ploennigs, and B. Gorman. Towards automating the deployment of energy saving approaches in buildings. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, BuildSys '14, pages 164–167, New York, NY, USA, 2014. ACM.

[23] J. Sethuraman. A constructive definition of dirichlet priors. *Statistica Sinica*, 4, 1994.

[24] B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[25] S. Tong and E. Chang. Support vector machine active learning for image retrieval. In *Proceedings of the ninth ACM international conference on Multimedia*. ACM, 2001.

[26] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, 2, 2002.

[27] E. S. p. U.S. Environmental Protection Agency. Useful facts and figures. June 2007.

[28] J. Warnick. 88 acres: How microsoft quietly built the city of the future. *http://www.microsoft.com/en-us/stories/88acres/88-acres-how-microsoft-quietly-built-the-city-of-the-future-chapter-1.aspx (May 8, 2015)*, 2012.

[29] Y. Yang. An evaluation of statistical approaches to text categorization. *Inf. Retr.*, 1, 1999.

[30] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. *Advances in neural information processing systems*, 16(16), 2004.

[31] X. Zhu, Z. Ghahramani, J. Lafferty, et al. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, volume 3, 2003.

[32] Z. Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 2. IEEE, 2004.