Learning a Taxonomy of Predefined and Discovered Activity Patterns

Narayanan Krishnan, Diane J. Cook and Zachary Wemlinger
School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA 99164, USA

Abstract. Many intelligent systems that focus on the needs of a human require information about the activities that are being performed by the human. At the core of this capability is activity recognition. Activity recognition techniques have become robust but rarely scale to handle more than a few activities. They also rarely learn from more than one smart home data set because of inherent differences between labeling techniques. In this paper we investigate a data-driven approach to creating an activity taxonomy from sensor data found in disparate smart home datasets. We investigate how the resulting taxonomy can help analyze the relationship between classes of activities. We also analyze how the taxonomy can be used to scale activity recognition to a large number of activity classes and training datasets. We describe our approach and evaluate it on 34 smart home datasets. The results of the evaluation indicate that the hierarchical modeling can reduce training time while maintaining accuracy of the learned model.

Keywords: activity discovery, activity recognition, hierarchical clustering

1. Introduction

Many artificial intelligence applications that focus on the needs of a human require information about the activities being performed by the human. At the core of such technologies, then, is activity recognition, which is a challenging and well researched problem. Approaches to activity recognition differ in terms of the underlying sensing technology, the machine learning models and the realism of the environment. Irrespective of these variables, the literature is abundant with evidence that existing techniques work well, particularly in controlled settings.

Activity recognition techniques have improved over the past few years in terms of accuracy and reliability. In order to move to the next level, the techniques need to be improved for generalizability and scalability. The long-term goal of our research project is to design machine learning approaches that recognize a large number of activities for a broad segment of the population. Because activity models need to generalize over variations in environmental and resident characteristics, and because robust activity models require plenty of training data, we want to use as many data sources as are available to train activity models. When activity recognition data is collected from multiple sources, challenges begin to arise not only because of differences in physical settings and human subjects, but also because the activity labels and annotation styles vary from one data source to another. We hypothesize that the similarities between activity patterns can be detected automatically. These similarities can be used to analyze the relationship between activity labels and annotation styles and to find relationships between automatically-discovered activities and predefined activity classes.

In this paper we validate our hypothesis that clustering activities into a hierarchical activity taxonomy can facilitate analysis of activity patterns from multiple sources. In addition, we also postulate that such a taxonomy can also be used to scale activity recognition. Instead of using a single classifier to distinguish between large numbers of activities, we design a hierarchy of classifiers, each of which distinguishes between child nodes at a particular location in the hierarchy. We describe our proposed technique, evaluate alternative clustering algorithms, and demonstrate how the method improves model training time and accuracy. We also introduce a recognition precision measure, called HC-Index. We validate our method for 55 pre-defined activities as well as 55

automatically-discovered activities found in data collected from 34 smart home datasets. In addition, we demonstrate how the technique can be used to represent the relationship between predefined activities and those that are automatically discovered from unlabeled sensor data.

2. Related Work

The goal of activity recognition is to identify activities as they occur based on data collected by sensors. There exist a number of approaches to activity recognition [22] that vary depending on the underlying sensor technologies that are used to monitor activities, the machine learning algorithms that are used to model the activities and the realism of the testing environment.

Advances in pervasive computing and sensor networks have resulted in the development of a wide variety of sensor modalities that are useful for gathering information about human activities. Wearable sensors such as accelerometers are commonly used for recognizing ambulatory movements (e.g., walking, running, sitting, climbing, and falling) [24],[32]. More recently, researchers are exploring smart phones equipped with accelerometers and gyroscopes to recognize such movement and gesture patterns [26].

Environment sensors such as infrared motion detectors or magnetic door sensors have been used to gather information about more complex activities such as cooking, sleeping, and eating. These sensors are adept in performing location-based activity recognition in indoor environments [2],[31],[46] just as GPS is used for outdoor environments [29]. Some activities such as washing dishes, taking medicine, and using the phone are characterized by interacting with unique objects. In response, researchers have explored the usage of RFID tags and shimmer sensors for tagging these objects and using the data for activity recognition [33],[36]. Researchers have also used data from video cameras and microphones as well [2].

There have been many varied machine learning models that have been used for activity recognition. These can be broadly categorized into template matching / transductive techniques, generative, and discriminative approaches. Template matching techniques employ a kNN classifier based on Euclidean distance or dynamic time warping. Generative approaches such as naïve Bayes classifiers where activity samples are modeled using Gaussian mixtures have yielded promising results for batch learning. Generative probabilistic graphical models such as hidden Markov models and dynamic Bayesian networks have been used to model activity sequences and to smooth recognition results of an ensemble classifier [27]. Decision trees as well as bagging and boosting methods have been tested [32]. Discriminative approaches, including support vector machines and conditional random fields, have also been effective [23],[46] and unsupervised discovery and recognition methods have also been introduced [15],[38].

Many of these methods analyze pre-segmented activity sequences that have been collected in controlled settings. The work summarized here reports classification accuracies ranging from 88% to 93% for 6-13 classes of pre-segmented activities. A few projects evaluate activity recognition on continuous data streams in real-world settings. These experiments have been evaluated for 7-17 activity classes and report accuracies that are on average 20% lower than the pre-segmented experiments. In this paper we focus on extending this prior work on real-time activity recognition for large numbers of activities gathered from multiple datasets. In addition, there has been work that employs hierarchical models to represent activities. Wang et al. [48] have proposed a framework that maps low-level patterns to high-level activities using a hierarchical framework. Van Kasteren et al. [45] employ a hierarchical hidden Markov model to model motion sensor data.

While some researchers have offered approaches to transfer data between sensor networks [47], environmental settings [8] or activity labels [20],[45] to perform activity recognition across generalized settings, none to date have investigated how data from multiple sources can be leveraged to scale activity recognition methodologies.

While recognizing predefined activities often relies on supervised learning techniques, unsupervised learning is valuable for its ability to discover recurring sequences of unlabeled sensor activities that may comprise activities of interest. Our approach to activity discovery builds on a rich history of discovery research, including methods for mining frequent sequences [1],[15], mining frequent patterns using regular expressions [4], constraint-based mining [35], mining frequent temporal relationships [3], and frequent-periodic pattern mining [18].

More recent work extends these early approaches to look for more complex patterns. Ruotsalainen et al. [41] design the Gais genetic algorithm to detect interleaved patterns in an unsupervised learning fashion.

Other approaches have been proposed to mine discontinuous patterns [7],[34],[50], in different types of sequence datasets and to allow variations in occurrences of the patterns [38]. Aspects of these earlier techniques are useful in analyzing sensor sequence data. We build from these earlier approaches and enhance them in order to perform activity discovery as well as real-time activity recognition and tracking from streaming sensor data.

3. Introduction

We test the ideas described in this paper using 34 sensor event datasets collected from 31 different smart home testbeds operated by 5 research groups. In all of the testbeds, sensor readings were captured while residents lived in the home and performed their normal daily routines. Sensors that are used for these data collections include infrared motion sensors and sensors that monitor door open/shut state. In some cases sensors are to monitor light usage, light levels, temperature, power usage, water usage, burner usage, and object usage.

The physical settings vary between testing sites – some are apartments while others are one story or two story homes. The residential situations vary as well. Some of the sites have one resident, some have two residents, and some include pets. The datasets themselves were collected by different research groups including the MIT Placelab group [38], the University of Amsterdam [46], the Alpen-Adria University, Bosch, and the WSU CASAS group [8]. The timespan of the data varies from one to eight months. Human annotators at the respective labs provided the activity labels for each dataset. Some of the annotators viewed video footage of the house to label activities, while others scanned raw sensor data. Although many of the syntactic labels for the activities are similar, we postulate that there will exist differences in the interpretation of the activity and thus in the sensor data sequences that correspond to even similar activity labels collected at different sites and annotated by different experimenters. We use labels provided with each data set as ground truth to train and test our activity recognition algorithms.

Table 1 lists the activities for which annotations were available. It can be observed that there are many activities that have the same connotation, but slightly different labels. Dataset features are summarized in Table 2. We initially filter out data that has no corresponding activity label, and have a remaining total of 136,529,421 data points to analyze. In the second part of this paper we will describe an approach to discovering activity patterns in the un-annotated data and integrating the corresponding activity models into our hierarchy.

Table 1. Activity labels collected from all datasets.

Bathe	Eat_ Dinner	Phone	Wakeup
Bathing	Eat_ Lunch	Read	Wash_Break-Fast_Dishes
Bed_Toilet_ Transition	Eating	Relax	Wash_Dinner _Dishes
Bed_to_ Toilet	Enter_ Home	Resperate	Wash_Dishes
Brush_ Teeth	Entertain _Guests	Shower	Wash_Lunch _Dishes
Cook	Evening _Meds	Sleep	Watch_TV
Cook_ Breakfast	Exercise	Sleep_Out _Of_Bed	Work
Cook_ Brunch	Groom	Sleeping	Work_At _Computer
Cook_ Dinner	House Keeping	Sleeping _in_Bed	Work_At _Desk
Cook_ Lunch	Leave_ Home	Snack	Work_At _Table
Dress	Meal_ Preparation	Take_ Medicine	Work_On _Computer
Drink	Morning _Meds	Toilet	work
Eat	Night_ Wandering	Toilet_ Downstairs	
Eat_ Breakfast	Personal_ Hygiene	Wake	

While the activity taxonomy can be learned using natural language processing techniques, this mapping may not be necessarily the most accurate across multiple settings because label definitions vary between annotators. To overcome this problem, we propose a data-driven approach to identify the similarity between different labels. The sensor events for an activity are transformed into feature descriptors. Because the datasets use sensors with different identifiers, we map the sensor identifiers to a common vocabulary. Sensors are identified using the labels summarized in Table 3.

Table 2. Description of the 34 smart home datasets used in this paper.

Dataset	#Residents	#Activities	#Sensors	#Months	#Events
Aruba[a]	1	11	39	7	1719553
Cairo[a]	2 (+cat)	8	32	2	725163
HH102[a]	1	35	41	1	133638
HH103[a]	1	35	21	1	79470
HH104[a]	1	35	45	1	181664
HH105[a]	1	35	36	1	58341
HH106[a]	1	35	46	1	171990
HH108[a]	1	35	27	1	149406
HH109[a]	1	35	31	1	184672
HH110[a]	1	35	31	1	99897
HH111[a]	1	35	40	1	168521
HH112[a]	1	35	29	1	205497
HH113[a]	1	35	44	1	226794
HH114[a]	1	35	28	1	192685
HH115[a]	1	35	39	1	118715
HH116[a]	1	35	33	1	255975
HH117[a]	1	35	27	1	142950
HH118[a]	1	35	40	1	525039
Kyoto 2[a]	2	5	72	2	135616
Kyoto 3[a]	2	7	72	2	7770712
Kyoto 4[a]	2	10	72	8	2798552
Milan[a]	1 (+dog)	9	33	2.5	426449
Tulum 1[a]	2	6	20	4	433820
Tulum 2[a]	3	10	36	6	1085900
Klagenfurt 11[b]	1 (+dog)	6	11	1	9601
Klagenfurt 14[b]	1 (+dog)	3	6	5	3818
Klagenfurt 32[b]	1	2	6	5	3137
Bosch 1[c]	1	10	32	6.5	658792
Bosch 2[c]	1	10	32	6.5	572255
Bosch 3[c]	1	11	32	5.5	518759
Placelab[d]	2	10	180	1	3494137
UA1[e]	1	8	14	1	53611
UA2[e]	1	8	23	0.5	118842
UA3[e]	1	8	21	1	286071

[a]Washington State University, [b]Alpen-Adria University, [c]Bosch, [d]Massachusetts Institute of Technology, [e]University of Amsterdam.

While the activity taxonomy can be learned using natural language processing techniques, this mapping may not be necessarily the most accurate across multiple settings because label definitions vary between annotators. To overcome this problem, we propose a data-driven approach to identify the similarity between different labels. The sensor events for an activity are transformed into feature descriptors. Because the datasets use sensors with different identifiers, we map the sensor identifiers to a common vocabulary. Sensors are identified using the labels summarized in Table 3.

Table 3. List of sensor names used in models.

Bath	KitchenDoor
Bathroom	LivingRoom
BathroomDoor	LivingRoomDoor
Bedroom	LoungeChair
BedroomDoor	OutsideDoor
DiningRoom	WorkArea
DiningRoomDoor	OtherRoom
Kitchen	OtherDoor

4. Learning an Activity Taxonomy

4.1. Activity recognition on streaming sensor data

Data collected in a smart home is stored as a tuple *<Date, Time, SensorId, Message>* and are referred to as *events*. Our activity recognition goal is to provide real-time activity labeling as sensor events arrive in a stream. To meet this goal, we formulate the learning problem as that of mapping the sequence of *k* most recent sensor events to a label that indicates the activity that corresponds to the last (most recent) event in the sequence. The sensor events preceding the last event define the context for this last event. For example, the sequence of sensor events:

2011-06-15	*03:38:23.27*	*BedMotionSensor*	*ON*
2011-06-15	*03:38:28.21*	*BedMotionSensor*	*ON*
2011-06-15	*03:38:29.21*	*BedMotionSensor*	*ON*

could be mapped to a *Sleep* activity label. Empirical evidence suggests that a sequence of *k=20* yields consistently good accuracy [23] and this is the size we use for our experiments. We are currently improving this approach to allow *k* to be dynamic based on the most likely activities that are being observed.

To provide input to the classifiers, we define features describing data point *i* that corresponds to a sensor event sequence of length *k*. Vector x_i includes values for the 25 features summarized in Table 4. Each vector x_i is tagged with the label y_i of the last sensor event in the window. The label y_i corresponds to the activity label associated with the last sensor event in the window.

Table 4. The feature vector describing a data point.

Feature #	Value
1..16	#Times each sensor generated an event in the sequence (16 unique sensors)
17..20	Time of day at beginning of sequence (morning, afternoon, evening, night)
21..24	Time of day at end of sequence
25	Time duration of entire sequence

In this paper we use support vector machines (SVMs) and naïve Bayes (NBC) classifiers to learn the activity models. We have tested naïve Bayes, hidden Markov models, and conditional random field models as well on similar data. We found that SVMs achieved consistently better performance than the NBC and HMM models and performed as well as the CRF on average with a reduced training cost. In addition, SVMs offer additional advantages in terms of determining the degree of fit between a data point and a class, which has been used to identify anomalous points [42]. However, SVMs are costly in terms of training time and thus methods are needed to reduce these costs. We use the LibSVM implementation [6] with the one-vs-one paradigm and a radial basis function kernel with default parameter settings. A number of approaches have been explored for adapting SVMs to multi-class learning, including the one-vs-one approach, a one-vs-all formulation, a formulation of a multi-class SVM objective function, and utilization of a DAGSVM. Experiments conducted by Hsu and Lin [19] reveal that the one-vs-one paradigm is one of the most practical multi-class SVM approaches. The other successful approach utilizes a DAG, which we also test in this paper by approximating a DAG with the activity taxonomy. Though we test the SVM and NBC classifiers, our methodology can use any classifier.

4.2. Creating an activity cluster hierarchy

Learning an activity model is time consuming, even more so when the number of activity classes is large and when costly models are used. We hypothesize that insights can be gained about activities and models can be learned effectively with less training time by organizing activities in a cluster hierarchy. In the hierarchy, each leaf node represents a single activity, and internal nodes represent unions of the activities that reside in the subtree rooted at the node.

Multiple approaches have been proposed for creating hierarchical clusters. We propose to use a bottom-up or agglomerative method [13]. At each step of the process the two most similar nodes are merged. When nodes are merged a parent node is created in the hierarchy, which represents a union of the two activities. The original (merged) activities become the children of the new node. Because merging occurs between two nodes at a time the resulting hierarchy is represented as a binary tree.

Central to the problem of clustering is the notion of similarity between two nodes. One of the most popular methods, which we adopt, is to estimate similarity as the inverse mean distance between elements of each cluster. This method, also referred to as average linkage clustering, merges clusters X_1 and X_2 with the smallest average distance between all pairs of their elements, as shown in Equation 1.

$$d_{avg} = \frac{1}{|X_1||X_2|}\sum_{x_1 \in X_1}\sum_{x_2 \in X_2} d(x_1, x_2) \qquad (1)$$

However, we still need to define a distance measure between the data points. We experiment with three alternative distance functions for this study. The first method, ***fmean***, calculates $d(x_1,x_2)$ as the Euclidean distance between the feature vectors for data points x_1 and x_2. The second method, ***smean***, performs spectral decomposition on the feature distances. Spectral clustering makes use of the spectrum of the feature distance matrix to reduce the dimensionality of the space and thus perform cluster merging in fewer dimensions. Let $c_1,..,c_L$ represent the set of nodes (activities). We first compute the pairwise distance matrix $D_{avg} = davg_{mn}$, where $davg_{mn}$ is the distance between activities c_m and c_n as computed by Equation 1. This distance matrix is then transformed into an adjacency matrix, W, by applying the Gaussian / heat kernel, where $w_{mn} = \exp(-d_{mn}^2 / 2\sigma^2)$ and σ is the free parameter representing the kernel width. The normalized Laplacian, L, for this adjacency matrix is computed as defined in Equation 2, where I is the identity matrix and D is the degree matrix.

$$L = I - D^{-1/2} W D^{-1/2} \quad (2)$$

In Equation 2, D is a diagonal matrix where the diagonal elements contain the sum of all of the elements in the corresponding row of W. The eigenvectors of L up to K dimensions (where K corresponds to the index with the maximum eigengap) represent the activity data points in the transformed space. Finally, cluster distances are computed based on the Euclidean distance between the data points in the transformed space. For high-dimensionality data, spectral clustering should generate clusters that reflect the distribution of the data without being sensitive to redundancies in the feature vector description.

The third choice of distance function, ***mm***, again uses spectral clustering to generate the final cluster differences. However, where the *smean* calculation started with Euclidean differences between feature vectors representing the different activities, the *mm* approach computes the differences in the first-order Markov models of the activity sequences. Let (I_m, T_m) and (I_n, T_n) represent the initial probability vector and transitional probability matrices of the Markov models representing the two activities m and n, respectively. We measure the difference between the two Markov models by computing the symmetric Kullback-Leibler divergence measure [25] between the initial (I_m, T_m) and (I_n, T_n) probability distributions. Let D_{KL} represent the distance matrix between the activities, measured in terms of the symmetric KL divergence measure. Each element of the matrix is thus defined as shown in Equation 3.

$$D_{KL_{mn}} = [I_m \log(\frac{I_m}{I_n}) + \sum_{p=1}^{Cols(T_m)} T_m(p) \log(\frac{T_n(p)}{T_m(p)})] * [I_n \log(\frac{I_n}{I_m}) + \sum_{p=1}^{Cols(T_n)} T_n(p) \log(\frac{T_m(p)}{T_n(p)})] \quad (3)$$

Using the *mm* approach, distances between data points are obtained based on the probabilities of the activity sequences. In contrast to the *fmean* and *smean* approaches, *mm* is highly sensitive to the ordering of sensor events that occur within the instances of each activity class. The cluster hierarchy that was generated by the *fmean* approach is shown in Figure 1. This hierarchy represents a fairly balanced tree of height 13. The cluster hierarchy that was generated by the *smean* approach is identical to the *fmean* hierarchy for these datasets, so we do not include discussion of this hierarchy in the remainder of the paper. The *mm* approach generates the maximally unbalanced hierarchy shown in Figure 2.

4.3. Comparing hierarchies

By varying the choice of distance measure, merging criteria, and algorithm direction (top down or bottom up), alternative cluster hierarchies can be generated. Here we compare the hierarchies that are generated using these various methods. A number of cluster quality measures have been introduced in the literature. Internal measures evaluate cluster quality based only on the data that was clustered. Example internal evaluation including measuring compactness within a cluster vs. the separation between clusters [12],[16] and measuring pairwise similarity within a cluster weighted by the size of the cluster [51]. Other measures such as centrality and weakest link are useful when clustering graphs [21], whereas the Davies-Bouldin index [10] and cophenetic distance measure [44], used here, can be applied specifically to evaluate hierarchical clusters. On the other hand, external measures related the quality of the clusters to an external factor such as classification accuracy [14],[37]. Here we compare alternative hierarchies using internal evaluation methods. Later we will perform external evaluation by determining the effectiveness of the hierarchy to perform activity classification.

Examining the hierarchies, we see that both data-driven methods tend to group activities with similar labels together. In the *fmean* cluster, one subtree contains the activities Sleep, Sleeping, Sleeping_in_Bed, Bed_to_Toilet, Bed_Toilet_Transition, and Night_Wandering, and another subtree contains

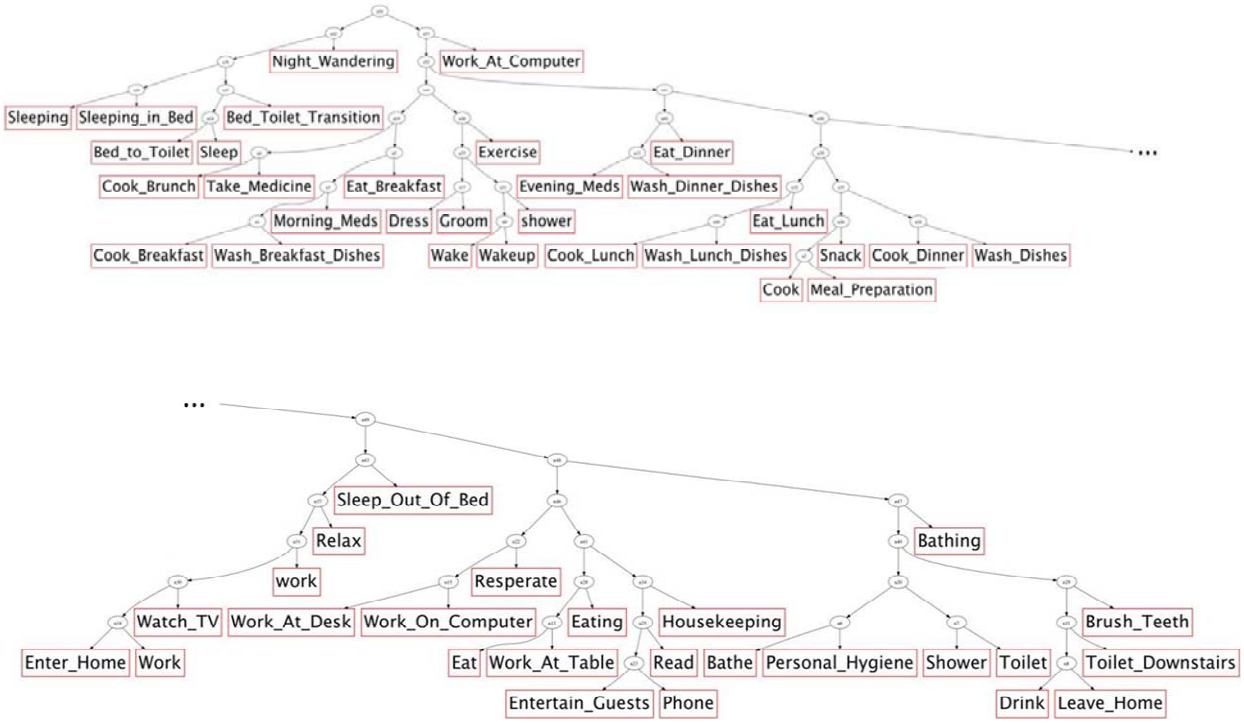

Figure 1. Cluster hierarchy built from 55 activities using the *fmean* method.

the activities Bathe, Personal_Hygiene, Shower, and Toilet. The *mm* hierarchy also groups similar semantic labels together, although the resulting hierarchy is much more imbalanced. These groupings result from the data features, despite the fact the data and the labels originate from different sites and annotators. This provides evidence that 1) activities are performed in similar ways even in different settings with different residents, 2) annotators at different labs take similar approaches to identifying activities, and 3) activities with similar textual labels exhibit similar sensor event sequences. We note, however, that not all similarly-labeled activities appear as siblings in the hierarchy. For example, classes "Shower" and "shower" have virtually identical labels yet have a path length of 14 between their nodes in the hierarchy. Later we will examine differences between the corresponding data patterns.

To compare cluster hierarchy quality using internal evaluation, we compute the Davies-Bouldin index and the cophenetic correlation coefficient for each hierarchy. The Davies-Bouldin index [10] is the ratio of the sum of within-cluster distance to between-cluster separation. Smaller ratios are better, because they indicate that the clusters are compact and far apart. The index is calculated using Equation 4 where n is the number of clusters, S_n is the average distance of cluster points to the cluster centroid, and $S(C_i,C_j)$ is the distance between cluster centroids.

$$DB = \frac{1}{n}\sum_{i=1}^{n}\max_{i\neq j}\left\{\frac{S_n(C_i)+S_n(C_j)}{S(C_i,C_j)}\right\} \qquad (4)$$

The cophenetic correlation coefficient [44] measures how well the cluster hierarchy maintains pairwise distances between the original data points (between individual activities). Specifically, the coefficient c measures how closely the original distances between data points (activities) i and j correlate with their distance in the hierarchy $t(i,j)$, or the height of the node at which points i and j are merged. Calculation of c is shown in Equation 5. The closer the value of c is to 1, the more accurately the cluster hierarchy reflects the similarity of the actual data points (activities).

$$c = \frac{\sum_{i<j}(d_{avg}(i,j) - \bar{d}_{avg})(t(i,j) - \bar{t})}{\sqrt{\sum_{i<j}[(d_{avg}(i,j) - \bar{d}_{avg})^2][(\sum_{i<j} t(i,j) - \bar{t})^2]}} \quad (5)$$

A summary of the cluster internal evaluation scores is provided in Table 5. We note that while both hierarchies provide a better organization than the original (non-clustered) set of activities, the hierarchy that is generated using feature vector distances (*fmean*) yields both a lower Davies-Bouldin index and a higher cophenetic correlation coefficient than the hierarchy generated using Markov model distances (*mm*). Furthermore, the cophenetic correlation coefficient for the *fmean* hierarchy is very close to 1, which indicates that the generated hierarchy provides a taxonomy that accurately reflects the underlying data.

Table 5. Internal cluster hierarchy results.

	Hierarchy type		
Measure	**Flat**	**Fmean**	**Mm**
DB	50.07	**4.03**	36.26
CCC	N/A	**0.93**	0.80

4.4. Hierarchy activity model creation

Our objective is to train a binary classifier at every internal node of the cluster tree to differentiate between the node's children, thus creating a hierarchical activity model. The data samples for a parent node are obtained by combining the instances of its children. Using all of the child data points can lead to a difficulty for activity modeling, however, because the class distribution can become very imbalanced. Consider the root's left child in Figure 1. If all data points are propagated from child to parent nodes, then the model for this node has to discriminate a class (the left subtree) which represents a union of five different activities and 20,914,278 data points with another class (the right subtree) which represents a single activity, Night_Wandering, which contains 123,627 data points. This results in a classic class imbalance situation, which is a challenge for machine learning algorithms [17]. One way to face this challenge is to strategically undersample the majority class (the class with the majority of data points) [30].

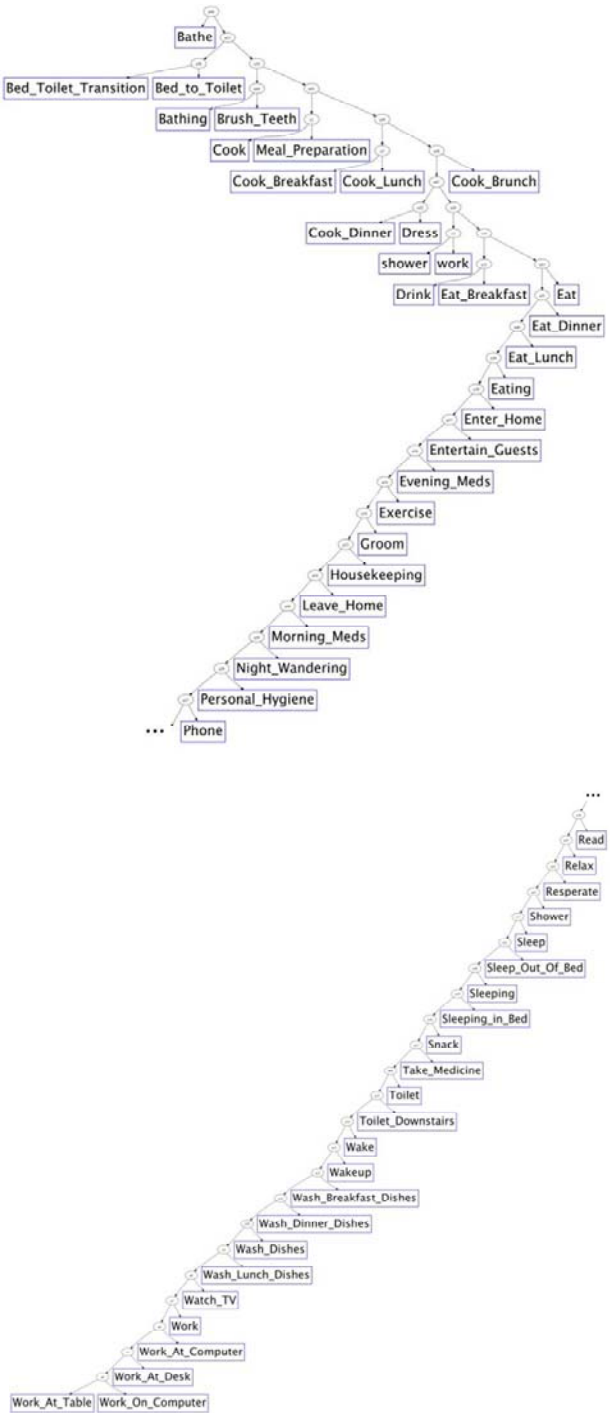

Figure 2. Cluster hierarchy built using *mm*.

We employ an undersampling method in our hierarchical classifiers. Instead of combining all instances from child nodes into the parent node, we selectively sample instances from the children. Consider an internal node N with two children, N_1 and N_2, with n_1 and n_2 numbers of data points, respectively. Our hierarchical classifier selects $max(n_1, n_2)$ data points to assign to parent node N. We select half of these data points to be the ones furthest from the decision boundary separating the instances of N_1 and N_2, and the remaining half are those which are closest to this decision boundary. Samples that are farthest from the decision boundary represent the unique characteristics of the individual children. In contrast, points that are closest to the decision boundary represent the characteristics of the combined dataset. Selecting the data points for the parent in this manner helps the algorithm to partially preserve the data distribution of its children. This is also a good method to even the distribution of activity data that normally occurs.

5. Learning an Activity Taxonomy

We originally postulated that activity modeling would require less training time when activities are organized in a cluster hierarchy rather than as a flat set. We further argue that by using sampling methods activity recognition performance would not be sacrificed. In this section we describe some experiments that are designed to validate these hypotheses. We evaluate the performance of the flat model (no clusters, the model discriminates between all activity classes at once) and the hierarchical model (based on the *fmean* hierarchy). We report the results in terms of model training time and recognition accuracy for the SVM and NBC classifiers. The flat models for these classifiers are denoted as FSVM and FNBC, and the hierarchical models are denoted as HSVM and HNBC.

In addition, we introduce a new measure, a Hierarchical Classification Index (***HC-Index***), and evaluate activity recognition for activity cluster hierarchies using this index. We introduce this alternative performance measure based on the observation that not all recognition errors are alike. Looking at the activities in Figure 1 one can note that if a *Sleeping_in_Bed* activity is mislabeled as *Sleeping* (the sibling node in the hierarchy), this error is not as great as mislabeling a *Sleeping_in_Bed* activity as *Wash_Dishes* (a node far removed in the hierarchy). For some applications a precise activity label is required. On the other hand, in some cases an activity label which is close to the actual activity will suffice.

Let y_i be the ground truth label for a test sample x_i and let l_i be the label predicted by the hierarchical model. HC-index(l_i, y_i) determines the hierarchical classification accuracy of the predicted label l_i by computing the ratio of the depth of the lowest node in the tree that is an ancestor of both l_i and y_i to the maximum depth of l_i and y_i.

To compute this index, the activity recognition algorithm traverses a path from the root node of the hierarchy to a leaf. At each node, the corresponding classifier processes the data to determine which edge to pursue. If an activity is labeled correctly, then the path to the ground truth label (y_i) aligns with the hierarchical model-generated path from the root node to the leaf node (l_i) resulting in an HC-Index(l_i, y_i) of 0. If an activity is not labeled correctly, then at some level in the hierarchy, the model outputs an edge node that differs from the correct path.

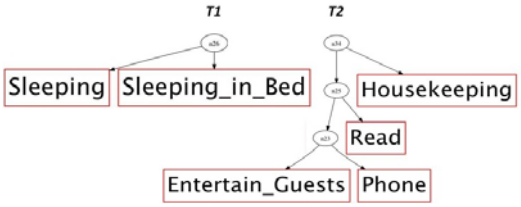

Figure 3. Two sample subtrees. T1 has L=2 leaves and T2 has L=4 leaves.

Activity modeling consists creating an activity hierarchy and then training separate models for each node in the hierarchy in a bottom-up fashion, using undersampling methods to provide data to the higher-level nodes. In contrast, labeling new data involves nagivating the tree in a top-down manner. Consider an example of how an activity is categorized using a portion of the *fmean* cluster hierarchy (see Figure 1). In this example, a sensor sequence is input that corresponds to the activity Meal_Preparation. A dramatic mistake would be to mislabel the sensor sequence as Night_Wandering. In this case, the classifier decides the data point belongs to the left child of the root. The only point of agreement between the classifier-generated label and the true label is at the root node, so the HC-Index for the data point is 0/9 = 0.00. On the other hand, a less obvious mistake would be to classify the point as Cook_Dinner, which is intuitively a subset of Meal_Preparation. In this case, classifiers at four levels correctly label the data point before the path between the true label and the classifier-generated label diverge. Activity recognition perfor-

mance compared with HC-Index for this data point is 6/9 = 0.67. The HC-Index values are averaged over all test data points to determine the HC-Index of the hierarchical model for a particular dataset. We compare HC-Index to classification accuracy for our cluster hierarchies.

In our first experiment we compare a flat SVM (FSVM) and a hierarchical SVM (HSVM) on our targeted dataset. In order to examine how performance is affected by the number of activities, we compare these two approaches for all of the subtrees of the hierarchy shown in Figure 1. We plot training time as a function of the number of leaves (activities) that are found in the corresponding subtree. For example, the subtrees in Figure 3 have sizes *L=2* and *L=3*, respectively. The training times for all subtrees in the *fmean* cluster hierarchy are shown in Figure 4 and are plotted as a function of the number of leaves in the subtree. Each of the times is compared with the corresponding training time if a flat SVM is used.

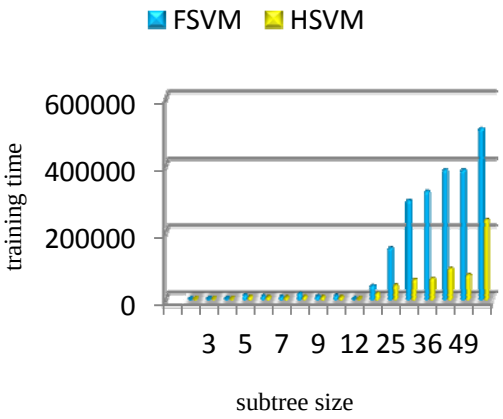

Figure 4. Classifier training time in seconds (y axis) as a function of subtree size (x axis) measured as the number of leaf nodes (activity classes) in the subtree.

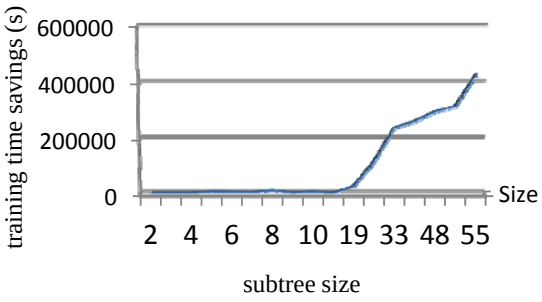

Figure 5. Training time savings in seconds for HSVM as a function of subtree size.

A few observations can be made about these results. First, as is expected, the training time increases superlinearly with the number of activities that are being considered. Second, HSVM requires significantly less training time than FSVM. Not surprisingly, the savings in training time increases superlinearly with the size of the subtree, as is shown in Figure 5. The training time averaged over all subtrees is 10,718.71 seconds for HSVM and 41,629.25 with a statistically significant ($p < 0.01$) improvement using the hierarchical model. The hierarchy generation time is less than 10 seconds for each approach so we do not include this in the summary.

Improvement for the naïve Bayes classifier is not as dramatic, because both the flat and hiearchical classifiers are fast. A flat NBC completes training for this dataset within 1 second on average, while the hierarchical version completes training in 0.76 seconds. However, as we will see next, the naïve Bayes classifiers do not generate as predictive of a model for this type of data.

Table 6. Predictive accuracy for flat and hierarchical models.

Classifier			
FNBC	**HNBC**	**FSVM**	**HSVM**
.409	.763	.777	.690

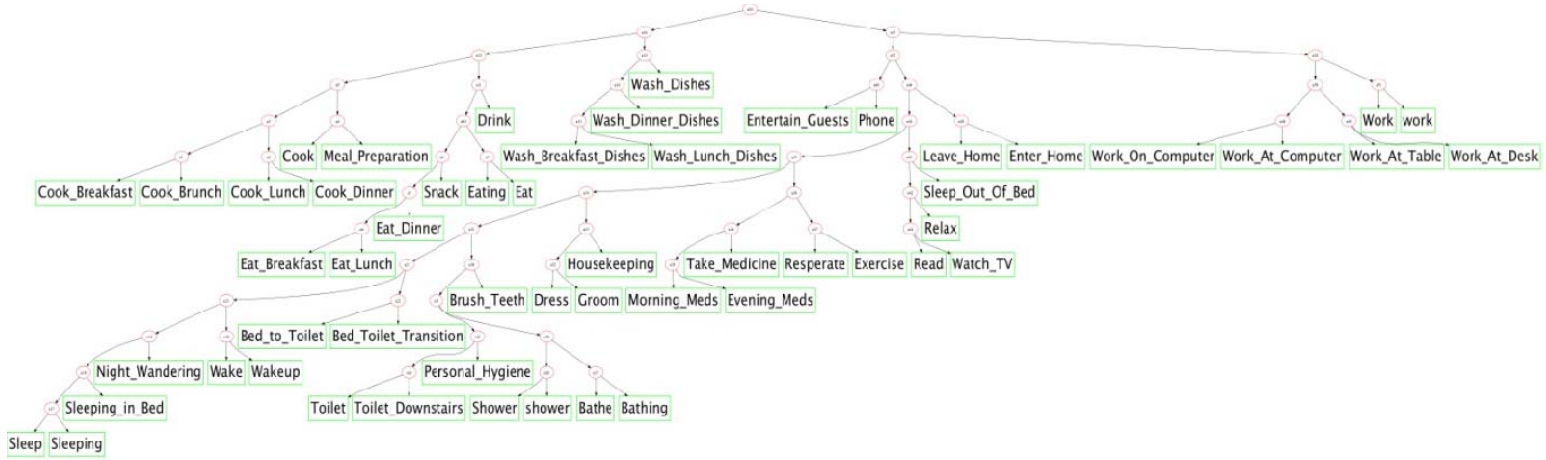

Figure 7. Human-generated activity hierarchy.

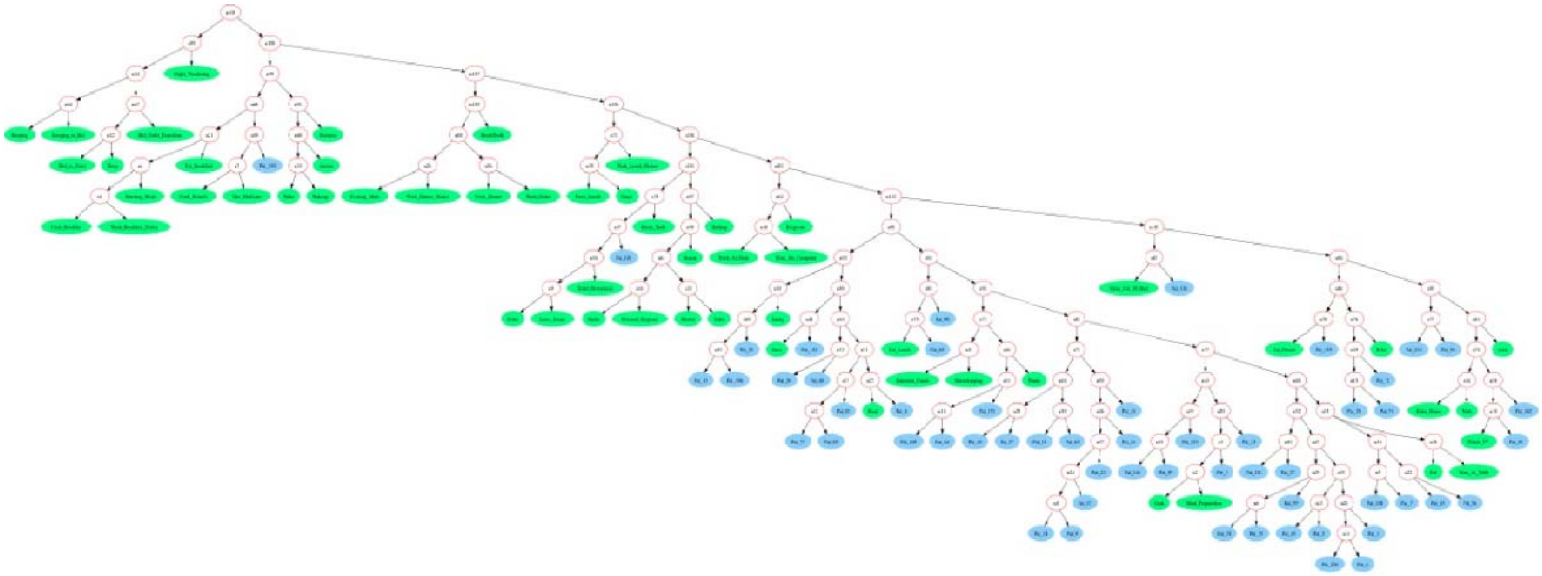

Figure 8. Birds-eye view of a cluster hierarchy combining predefined activities and discovered activities. Green nodes are predefined activities and blue nodes are discovered patterns.

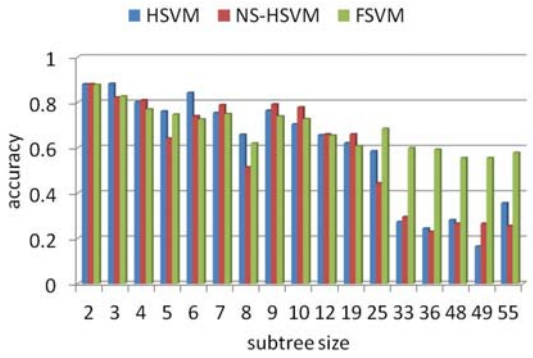

Figure 6. Classification accuracy for HSVM, HSVM-NS, and FSVM as a function of subtree size.

Next, we note that activity recognition performance is not adversely affected using the hierarchy with sampling. Table 6 summarizes the predictive accuracy results obtained for FNBC, HNBC, FSVM, and HSVM using (2/3, 1/3) testing and averaged over all subtrees in the hierarchy. While the hierarchical models have an edge over the flat models in both cases, the difference is not as great for the SVM classifier, although both SVM classifiers outperform the naïve Bayes classifiers. Looking at individual subtree sizes (plotted in Figure 6), we note that the HSVM consistently outperforms the flat model for subtrees of size less than 25. This improvement in the performance is due to several factors, including the smaller number of classes to model for a subtree and the undersampling that evens the class distribution.

Ironically, this undersampling sometimes also causes the HSVM to underperform the FSVM for subtrees of size greater than 25. As we approach these large subtrees the methodology tends to aggressively undersample parent nodes that otherwise would have a large number of data points from very diverse classes. An example of this situation is the subtree rooted at n48, which contains 19 leaves. If all of the data points were considered the model would learn from 1,163,559 data points. However, when sampling is used the SVM at this node learns a model from only 297,396 data points (approximately 1/4 of the total data points). We note that in the case of class imbalance the highest-accuracy approach is generally to label all points as the majority class. However, this lowers the true positive rate for the minority class, which is problematic if detecting the minority class is important.

We test the result of sampling by comparing with performance of FSVM with HSVM and HSVM with no sampling (NS-HSVM). As Figure 6 shows, the lack of sampling does not result in a consistent improvement in performance. In fact, the predictive accuracy averaged over all subtrees in the hierarchy degrades from .790 for HSVM to .742 for NS-HSVM. Additional performance measures would need to be used to capture the affect of the HSVM and FSVM for cases with dramatically imbalanced classes. We will investigate these issues in the future, along with exploring alternate sampling methodologies for selecting data points for the parent node.

Next, we take a look at the relationship between HC-Index and accuracy. Accuracy averaged over all subtree sizes is 0.798 based on HC-Index, as opposed to 0.790 based on accuracy. This difference indicates that many of the errors are closer in the hierarchy that the accuracy measure would indicate. Knowing the similarity of the generated label with the true label provides an application with option to still use the output of the classifier. These near misses may still represent valuable information for many intelligent system applications.

A natural question that arises is how our automatically-generated activity hierarchy compares with other possible hierarchies for the purpose of activity recognition. There is no single hierarchy that is obviously intuitive or can provide a ground truth hierarchy for comparison. Annotators for the various datasets do not maintain definitions for their interpretations of activities that were labeled, and there are obvious differences in interpretations of the activities between datasets. These variances are consistent with the study by Hu et al. [20] in which humans showed tremendous differences in their determination of the similarities of activities. Some participants ranked activities similar based on function while others used spatial relationships, temporal relationships, or other criteria for determining activity similarity.

As a result, we compare the result of our activity taxonomy with two other hierarchies. One is the hierarchy shown in Figure 7 which is built based on human intuition of activity similarity based on the intended function of the activities. In the other hierarchy we utilize the structure of the *fmean* hierarchy but we randomly order the activity classes within the hierarchy. The resulting activity recognition accuracy for a random sample of the entire datasets using the human-generated hierarchy is 35.3%, while the accuracy for the randomly-labeled hierarchy is 29.6% and

the accuracy for our generated hierarchy is 35.6%. These results indicate that well-formed hierarchies do impact the performance of activity recognition algorithms and that the formation can be based on human interpretation of activities or on the data itself.

6. Merging Predefined Activities with Discovered Activities

The generally accepted approach to activity recognition is to design or use machine learning techniques to map a sequence of sensor events to a corresponding activity label. Recognizing activities in real time from streaming data introduces new challenges for this problem because data must be processed that does not belong to any of the targeted activity classes. Such "out of vocabulary" detection is difficult, particularly when the out of vocabulary data represents a majority of the data that is observed. For the datasets that we are considering, more than half of the sensor events are unlabeled and do not correspond to any of the predefined activity classes.

One way to handle unlabeled data is to design an unsupervised learning algorithm to discover activities from the sensor data. We have shown in earlier work [9] that segmenting unlabeled data into smaller classes improves activity recognition performance because the "Other" class is not dominant in terms of size. Another important reason to discover activity patterns from unlabeled data is to characterize and analyze as much behavioral data as possible, not just predefined activity classes. The unlabeled data represents an important part of everyday life that needs to be examined and modeled in order to get a complete view of everyday life.

Our AD algorithm [9] searches the space of candidate patterns which consist of sensor event sequences appear in the raw data. Each potential pattern is evaluated based on its ability to compress the data. Once the best pattern is identified, AD compresses the data using the pattern and repeats the discovery process to find additional patterns in the data.

While our initial experiment only clustered predefined activities into a hiearchy, the same approach can be taken to create a hierarchy that combines predefined and discovered patterns. All of the clustering approaches are data-driven, so the algorithm only needs sensor sequence instances of the predefined patterns and of the discovered patterns in order to create the hiearchy.

One difficulty that we encounter is limiting the number of discovered patterns. Because AD reports as many patterns as possible (until no more compression is achieved), the number of patterns that result from processing 34 datasets is very large. AD discovered an average of 200 patterns in each dataset. If the hiearchy included all of the patterns the hierarchy would be enormous and the discovered patterns would dominate the hierarchy. In addition, some of the discovered patterns might be common to one setting but may not generalize over a population.

We want to target patterns that are frequent over many smart home datasets. We therefore evaluated each pattern in terms of its ability to compress *all* of the unlabeled data in all of the datasets. To do this, a one-class SVM model was trained for each pattern and the number of instances of the pattern for all unlabeled data over all datasets was calculated. Compression of the combined data was then estimated as the number of unlabeled sensor events divided by the number of sensor events that were not instances of the pattern in question. The entire set of patterns was sorted based on this value. For our experiment, we selected the top $n=55$ patterns for clustering, equal to the number of predefined activities that are also processed.

Figure 9. Visualization of AD-discovered patterns: P1 (top left), P2 (top right), P3 (bottom left).

Figure 8 shows a high-level view of the hierarchy that is generated by clustering predefined activities and discovered activities together. Figure 9 provides a visualization of the three top AD-discovered activities. The first one in the upper left contains a sequence consisting of motion in the bedroom followed by the living room (sometimes interspersed with kitchen events) and back to the bedroom, around

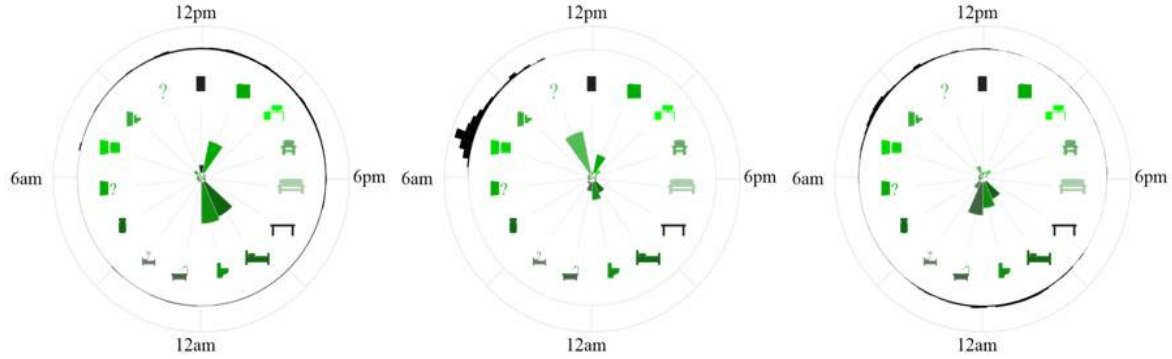

Figure 10. Radial bar charts for Cook (left), Meal_Preparation (middle), and Pat_3 (right).

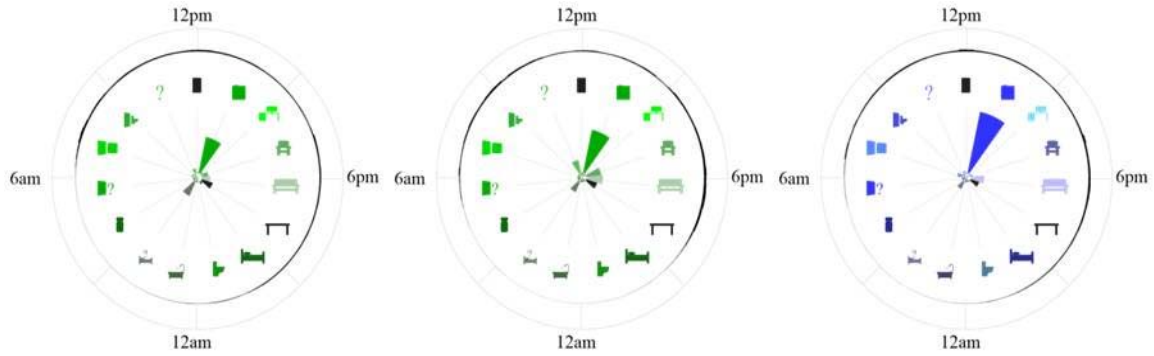

Figure 11. Radial bar charts for Shower (left), shower (middle), and Bathing (right).

10:20 in the evening. Looking at the larger context, we see that many of these events occur prior to sleeping and may represent getting ready for bed. The second pattern in the upper right consists of a front door closing followed by a series of kitchen events and then a living room event, usually in the late morning or mid afternoon. This could represent a number of different activities that occur after returning home, such as putting away groceries or getting a drink. The third pattern in the lower left consists of a sequence of events alternating between the bedroom, a work area, and the living room, shortly after waking up in the morning. This pattern might represent an individual gathering or setting up items in order to get ready for their daily routine. Many of the patterns represent transitions between activities or bursts of activity that are too short to easily label. Others represent activities that are recognizable but do not appear on the list of predefined activities, such as spending extended time in a secondary bedroom that is used for guests or crafts.

Another visualization that allows us to better observe the similarity between nodes in the cluster hierarchy is a radial bar chart. With this chart, bars on the inside of the circle point to locations in the home (the 12:00 position is front door, followed in a clockwise direction by kitchen, work area, lounge chair, living room, dining room, bedroom, bathroom, shower, other room, medicine cabinet, other door,

kitchen door, bathroom door, and other). The icon ordering was generated using an MIC-based distance graph [39] between areas of the home with a least-cost traversal of the graph. Bars on the outside of the circle indicate the times during the day when the activity is performed. Figure 10 shows the radial bar charts for activities Cook, Meal_Preparation, and Pat_3. We observe that all occur predominately in the kitchen and are distributed in time 7 am to 8pm (with peaks around 8am and noon).

We also want to better understand why some activities have similar semantic labels but are positioned far apart in the hierarchy. As mentioned earlier, activities "Shower" and "shower" are not close together while the names are virtually identical. As is shown in Figure 11, the radial bar charts for Shower, shower, and Bathing in fact reveal fairly different patterns. The chart on the left shows a pattern that occurs in the bathroom anytime between 7 am and 7 pm. In contrast, the chart in the middle shows a pattern that trips non-motion sensors (perhaps humidity or temperature sensors) and occurs between 7 am and 8 am. The chart on the right reflects a pattern that occurs in the bathtub between 7 am and 8 am and late in the evening, around 11pm. These charts highlight the fact that there can exist significant differences in the interpretation of an activity name as well as differences in how the activity may be performed at different locations. In order to create activity models that generalize to larger populations, these differences will need to be recognized and standards for activities will need to be defined. While the hierarchical SVM model when tested for all 110 activities yields an accuracy that is 25 times better than random guess, the resulting accuracy of 0.228 highlights the fact that continued research is needed to make these recognition techniques even more robust.

7. Conclusions

In this paper we propose and evaluate methods to analyze the similarity of predefined and discovered activities and to scale activity recognition by creating a hierarchical cluster of activity labels provided by various datasets. Instead of using a single flat classifier to distinguish between a large numbers of activities, we design a hierarchy of classifiers, each of which distinguishes between child nodes at a particular location in the hierarchy. We hypothesize that building the activity hierarchy in this manner will result in a reduction in the training time without losing recognition performance over the flat classifier model. We validate our method for 55 different activities based on data collected from 34 smart home datasets. The activity taxonomy that was generated using our proposed data-driven method did often group activities with similar labels together. On the other hand, some differences between activities with similar names were discovered using this analysis. Our experiments also demonstrate clearly the advantage of employing hierarchical activity organization for modeling activities, resulting significant savings in training time reducing the class imbalance inherent in the datasets.

In addition, an individual's daily routine includes many activities outside the lists of activities that were annotated in these datasets. We introduced a method by which these activities can be extracted using activity discovery algorithms based on frequent pattern mining. In this paper we demonstrated how discovered activities and predefined activities can be merged together into a comprehensive model of smart home behavioral data. Future work may focus on gathering physical interpretations of some of the discovered activities by studying their relationships with other known activities in the taxonomy.

As part of our future work, we plan to explore different pruning strategies, to further improve the current hierarchical taxonomy. In the current model, we do not associate any physical connotation to the intermediate nodes of the hierarchy. It would be an interesting exercise to devise methodologies for associating an "activity label" to the intermediate nodes and thereby study the relationships between the child nodes and its parent.

The activity taxonomy generated by our approach can also be used in the context of transfer learning across generalized settings. For example, consider the challenge of learning an activity model for a new activity in the new setting. We can determine the position of the new activity label in the hierarchy using few labeled samples of the activity, which then can be combined with labeled samples of similar activities as determined by the taxonomy for learning the new activity model.

In addition to exploring the above described directions, we also plan to investigate other undersampling approaches for selecting data samples for the parent node from its children. We will also consider better feature descriptors based on mutual information and time weighting for encoding the information present in the sensor events describing an activity and for handling the interwoven and parallel activities. Finally, we will explore the use of the hier-

archical activity taxonomy with other activity recognition algorithms.

8. Acknowledgements

The authors thank Bosch and Alpen-Adria University for making their datasets available for this project. This work was supported by Grant Number 0852172 from the National Science Foundation and Grant Number R01EB009675 from the National Institutes of Health.

References

[1] R. Agrawal and R. Srikant, Mining sequential patterns. In *Proceedings of the International Conference on Data Engineering*, pages 3-14, 1995.

[2] J. Alon, V. Athitsos, Q. Yuan, and S. Sclaroff, A unified framework for gesture recognition and spatiotemporal gesture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31:1685-1699, 2008.

[3] A. Aztiria, J. Augusto, and D.J. Cook, Discovering frequent user-environment interactions in intelligent environments, *Personal and Ubiquitous Computing*, 16(1):91-103, 2012.

[4] T. Barger, D. Brown, and M. Alwan, Health-status monitoring through analysis of behavioral patterns *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 35(1):22-27, 2005.

[5] O. Brdiczka, J.L. Crowley, and P. Reignier, Learning situation models in a smart home. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 39(1):56-63, 2009.

[6] C.-C. Chang and C.-J. Lin, LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27, 2011.

[7] Y.-I. Chen, S.-S. Chen, and P.-Y. Hsu, Mining hybrid sequential patterns and sequential rules. *Information Systems*, 27(5):345-362, 2002.

[8] D.J. Cook, Learning setting-generalized activity models for smart spaces. *IEEE Intelligent Systems*, 27(1):32-38, 2012.

[9] D.J. Cook, N. Krishnan, and P. Rashidi, Activity discovery and activity recognition: A new partnership. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, to appear.

[10] D.L. Davies and D.W. Bouldin, A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2):224-227, 1979.

[11] S. Dernbach, B. Das, N. Krishnan, B.L. Thomas, and D.J. Cook, Activity recognition on smart phones. In *Proceedings of the IEEE International Conference on Intelligent Environments*, 2012.

[12] D. Dunn, Well separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4:95-104, 1974.

[13] D.H. Fisher, Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2(2):139-172, 1987.

[14] E.B. Fowlkes and C.L. Mallows, A method for comparison of two hierarchical clusterings. *Journal of the American Statistical Association*, 78:553-569, 1983.

[15] T. Gu, S. Chen, X. Tao, and J. Lu, An unsupervised approach to activity recognition and segmentation based on object-use fingerprinters. *Data and Knowledge Engineering*, 69(6):533-544, 2010.

[16] M. Halkidi and M. Vazirgiannis, Clustering validity assessment: Finding the optimal partitioning of a data set. In *Proceedings of the IEEE International Conference on Data Mining*, pages 187-194, 2001.

[17] H. He and E.A. Garcia, Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21:1263-1284, 2009.

[18] E.O. Heierman and D.J. Cook, Improving home automation by discovering regularly occurring device usage patterns. In *Proceedings of the International Conference on Data Mining*, pages 537-540, 2003.

[19] C.-W. Hsu and C.-J. Lin, A comparison of methods for multiclass support vector machines. IEEE Transactions on Neural Networks, 13(2):415-425, 2002.

[20] D.H. Hu, V.W. Zheng, and Q. Yang, Cross-domain activity recognition via transfer learning. *Pervasive and Mobile Computing*, 7(3):344-358, 2011.

[21] W. Hwang, T. Kim, M. Ramanathan, and A. Zhang, Bridging centrality: Graph mining from element level to group level. In *Proceedings of the SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 336-344, 2008.

[22] E. Kim, A. Helal, and D.J. Cook, Human activity recognition and pattern discovery. *IEEE Pervasive Computing*, 9:48-53, 2010.

[23] N. Krishnan and D.J. Cook, Activity recognition on streaming sensor data. *Pervasive and Mobile Computing*, to appear.

[24] N. Krishnan and S. Panchanathan, Analysis of low resolution accelerometer data for human activity recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 3337-3340, 2008.

[25] S. Kullback and R.A. Leibler, On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79-86, 1951.

[26] J.R. Kwapisz, G.M. Weiss, and S.A. Moore, Activity recognition using cell phone accelerometers. In *Proceedings of the International Workshop on Knowledge Discovery from Sensor Data*, pages 10-18, 2010.

[27] J. Lester, T. Choudhury, N. Kern, G. Borriello, and B. Hannaford, A hybrid discriminative / generative approach for modeling human activities. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 766-772, 2005.

[28] V.I. Levenshtein, Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707-710, 1966.

[29] L. Liao, D. Fox, and H. Kautz, Location-based activity recognition using relational Markov networks. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 773-778, 2005.

[30] X.Y. Liu, J. Wu, and Z.-H. Zhou, Exploratory under-sampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 39(2):535-550, 2009.

[31] B. Logan, J. Healey, M. Philipose, E.M. Tapia, and S. Intille, A long-term evaluation of sensing modalities for activity recognition. In *Proceedings of the International Conference on Ubiquitous Computing*, pages 483-500, 2007.

[32] U. Maurer, A. Smailagic, D. Siewiorek, and M. Deisher, Activity recognition and monitoring using multiple sensors on different body positions. In *Proceedings of the International workshop on Wearable and Implanatable Body Sensor Networks*, pages 113-116, 2006.

[33] P. Palmes, H.K. Pung, T. Gu, W. Xue, and S. Chen, Object relevance weight pattern mining for activity recognition and segmentation. *Pervasive and Mobile Computing*, 6:43-57, 2010.

[34] J. Pei, J. Han, M.B. Asl, H. Pinto, Q. Chen, U. Dayal, and M.C. Hsu, Prefixspan: Mining sequential patterns efficiently by prefix projected pattern growth. In *Proceedings of the International Conference on Data Engineering*, pages 215-226, 2001.

[35] J. Pei, J. Han, and W. Wang, Constraint-based sequential pattern mining: The pattern-growth methods. *Journal of Intelligent Information Systems*, 28(2):133-160, 2007.

[36] M. Philipose, K.P. Fishkin, M. Perkowitz, D.J. Patterson, D. Fox, H. Kautz, and D. Hahnel, Inferring activities from interactions with objects. *IEEE Pervasive Computing*, 3:50-57, 2004.

[37] W.M. Rand, Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846-850, 1971.

[38] P. Rashidi, D.J. Cook, L. Holder, and M. Schmitter-Edgecombe, Discovering activities to recognize and track in a smart environment. *IEEE Transactions on Knowledge and Data Engineering*, 23(4):527-539, 2011.

[39] D. Reshef, Y. Reshef, H. Finucane, S. Grossman, G. McVean, P. Turnbaugh, E. Lander, M. Mitzenmacher, and P. Sabetic, Detecting novel associations in large datasets. *Science*, 334:6062, 2011.

[40] J. Rissanen, Stochastic Complexity in Statistical Inquiry. World Scientific Publishing Company, 1989.

[41] A. Ruotsalainen and T. Ala-Kleemola, Gais: A method for detecting discontinuous sequential patterns from imperfect data. In *Proceedings of the International Conference on Data Mining*, pages 530-534, 2007.

[42] B. Scholkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson, Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443-1472, 2001.

[43] G. Singla, D.J. Cook, and M. Schmitter-Edgecombe, Recognizing independent and joint activities among multiple residents in smart environments. *Ambient Intelligence and Humanized Computing Journal*, 1:57-63, 2010.

[44] R.R. Sokal and F.J. Rohif. The comparison of dendrograms by objective methods. *Taxon*, 11:33-40, 1962.

[45] T.L.M. van Kasteren, G. Englebienne, and B.J.A. Kröse, Hierarchical activity recognition using automatically clustered actions. In *Proceedings of the International Conference on Ambient Intelligence*, pages 82-91, 2011.

[46] T.L.M. van Kasteren, G. Englebienne, and B.J.A. Kröse, An activity monitoring system for elderly care using generative and discriminative models. *Personal and Ubiquitous Computing*, 14(6):489-498, 2010.

[47] T.L.M. van Kasteren, G. Englebienne, and B.J.A. Kröse. Transferring knowledge of activity

recognition across sensor networks. In *Proceedings of the International Conference on Pervasive Computing*, pages 283-300, 2010.

[48] L. Wang, T. Gu, X. Tao, and J. Lu, A hierarchical approach to real-time activity recognition in body sensor networks. *Journal of Pervasive and Mobile Computing*, 8(1):115-130, 2012.

[49] C.R. Wren and E. Munguia-Tapia, Toward scalable activity recognition for sensor networks. In *Proceedings of the International Workshop on Location and Context Awareness*, pages 168-185, 2006.

[50] M.J. Zaki, N. Lesh, and M. Ogihara, Planmine: Sequence mining for plan failures. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 369-373, 1998.

[51] Y. Zhao and G. Karypis, Evaluation of hierarchical clustering algorithms for document datasets. In *Proceedings of the Conference on Information and Knowledge Management*, pages 515-524, 2002.