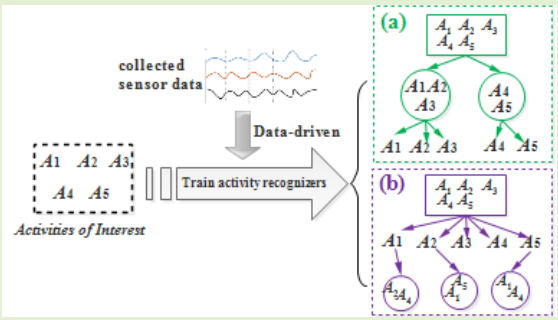
HierHAR: Sensor-Based Data-Driven Hierarchical Human Activity Recognition

Aiguo Wang, Shenghui Zhao, Chundi Zheng, Huihui Chen, Li Liu, and Guilin Chen

***Abstract*—Pervasive computing greatly advances the automatic recognition and understanding of human activities and effectively bridges the gap between the low-level sensor signals and high-level human-centric applications. The inherent complexity of human behavior, however, inevitably poses a huge challenge to the design of a robust activity recognizer, especially in classifying similar activities. In this study, we present a hierarchical framework, named HierHAR, that infers on-going activities in a multi-stage process for better distinguishing similar activities and improving the overall performance. Specifically, we propose a data-driven approach, rather than heavily rely on prior domain knowledge, to automatically determining the relationships among activities. Afterwards, we use the relationships to organize the activities into a tree structure and accordingly construct and train a tree-based activity recognition model. Furthermore, we train a graph-based model that aims to reduce the compounding errors induced by the prediction process of the tree-based model. Finally, extensive comparative experiments are conducted on public datasets and results demonstrate the power of HierHAR in facilitating the automatic organization of activities and the design of hierarchical recognizers without prior knowledge about activities. Besides, the graph-based activity recognizer generally generalizes better across different scenarios and outperforms the tree-based model.**



***Index Terms*— Pervasive computing, activity recognition, data-driven, information fusion.**

I. Introduction

THE rapid development and seamless integration of pervasive computing and sensing technology has greatly facilitated the automatic monitoring and understanding of human behavior and further enabled the provision of meaningful human-centric applications that range from chronic disease management, rehabilitation, wellness evaluation, fitness track, to medication adherence and building energy management in the context of smart homes [1], ambient assisted living systems [2], smart building [3], sports and exercise [4], and among others [5]. One of the key components of these applications is to accurately and automatically infer the on-going human activities, where activity recognition functions as middleware between the low-level sensor readings and various high-level services [6], [7]. However, due to the inherent complexity of human behavior, human activities are characteristically associated with diversity, concurrency, and similarity, which makes it difficult to develop satisfactory activity recognizers [8], [9]. Typically, there exists *inter-subject variation* in the way of doing an activity, where an activity recognizer that works well on one user probably fails to generalize well to others [10]. We also suffer from *intra-subject variation* [7], and one may perform concurrent and interleaving activities [11]. Furthermore, there are activities that can trigger similar sensor readings, even though the activities have different semantics [8]. Consequently, this would confuse an activity recognizer and lead to degraded performance. Therefore, how to automate the recognition of activities remains a challenging yet rewarding topic that has attracted considerable attention from many fields and deserves further investigation [12].

With the aim to achieve higher recognition rates and better adapt to different demand-oriented scenarios, researchers have explored a variety of sensing technologies and a wealth of models [6]. According to the used sensing units, we group the

Manuscript received August 20, 2020; revised September 3, 2020; accepted September 10, 2020. Date of publication September 14, 2020; date of current version January 6, 2021. This work was supported in part by the Natural Science Foundation of China under Grant 61902068 and Grant 61972092, in part by the Major Special Projects of Anhui Province under Grant 201903A06020026, in part by the Key Research and Development Project of Anhui Province under Grant KJ2019ZD44, and in part by the Foshan Self-funded S&T Innovation Plan under Grant 1920001001001. The associate editor coordinating the review of this article and approving it for publication was Prof. Aime (GE) Lay-Ekuakille. *(Corresponding author: Guilin Chen.)*

Aiguo Wang, Chundi Zheng, and Huihui Chen are with the School of Electronic Information Engineering, Foshan University, Foshan 528225, China (e-mail: wangaiguo2546@163.com; cdzheng@fosu.edu.cn; hhchen@fosu.edu.cn).

Shenghui Zhao and Guilin Chen are with the School of Computer and Information Engineering, Chuzhou University, Chuzhou 239000, China (e-mail: zsh@chzu.edu.cn; glchen@chzu.edu.cn).

Li Liu is with the School of Big Data and Software Engineering, Chongqing University, Chongqing 401331, China (e-mail: dcsliuli@cqu.edu.cn).

Digital Object Identifier 10.1109/JSEN.2020.3023860

existing methods into three categories: wearable sensor-based methods [7], vision-based methods [13], and environment sensor-based methods [14]. Compared with the vision-based methods that exploit a camera or video to capture a series of images and the environment sensor-based methods that place sensing units on household objects to capture the interaction between an individual and the surrounding objects, the wearable sensor-based methods recognize human activities by recording the sensor signals when one performs activities and training an activity recognizer. Particularly, the miniature of sensing units and the increased processing power of mobile devices enables us to wear one or more sensing units [15]. Those devices can be worn on different parts of the human body (e.g., the finger, waist, wrist, leg, and arm) and one can simultaneously wear multiple homogeneous or heterogeneous sensors [16]. In addition, such methods are suitable for both the outdoor and indoor scenarios and can support applications closely related to healthcare and body sensor networks [17].

In terms of training an activity recognizer, researchers have conducted considerable work and proposed a large number of models, which range from discriminant models to generative models, to associate raw sensor signals with their corresponding activity labels [18]. Researchers have also used the end-to-end deep learning models (e.g., convolutional neural networks, long short-term memory networks, and stacked autoencoder) to capture the underlying non-linear relationships and to jointly optimize the feature representations and model training [19], [20]. Undoubtedly, they have improved the recognition performance. One common feature of most of the existing activity recognizers is that they adopt a flat structure to recognize all the predefined activities in a single step using a multi-class classification model [8], [21]. For simplicity, we call such a scheme the *flat model*. Noteworthily, there are human activities that can trigger similar sensor readings in a natural setting, which would confuse an activity recognizer and weaken its discriminant ability. One possible solution is that we apply a divide and conquer strategy and design a finer-grained model to find a better decision boundary between similar activities and to gradually infer the activity labels [8], [22]. Accordingly, there are points that need further investigation. First, in terms of simple and well-studied activities, we can use prior domain knowledge to group them. For example, we can divide *sitting* and *standing* into stationary activity and group *walking* and *running* into dynamic activity. However, how to categorize activities in a complex scenario without much reliance on domain knowledge remains unsolved, since experts probably fail to describe the complex activities of interest. Second, how to reduce the confusion among similar activities is an important factor that affects the performance of an activity recognizer. To this end, we present a data-driven hierarchical framework, named HierHAR, that helps distinguish similar activities. Specifically, the predefined activities are first automatically organized into clusters without prior knowledge. We then construct a hierarchical model to predict the specific activity labels in a multi-stage process. Particularly, the main contributions of this study lie in the following aspects. (1) It is not trivial and error-prone to identify similar activities in a relatively complex and new scenario for domain experts. We present a data-driven method to obtain the relationships among activities. This greatly reduces the dependency on prior expert knowledge and contributes to the automatic construction of a hierarchical activity recognition framework. (2) We utilize the relationships to guide the construction of hierarchical activity recognition models and accordingly develop two specific ones: *tree-based model* and *graph-based model*. The former organizes the predefined activities into a tree structure and makes predictions in a top-down fashion along with the tree, while the latter builds a graph with connections between any two activities instead of restricting connections of activities to a hierarchy of disjoint groups. This facilitates the graph-based model to reduce the compounding errors of the tree-based one. Notably, both models have a hierarchical structure and belong to data-driven approaches as they utilize the sensor data to train an activity recognizer. (3) We implement the proposed models and conduct extensive comparative experiments. The results demonstrate the power of HierHAR in quantitatively measuring the relationships among activities and the better generalization of the graph-based model across different scenarios.

The remainder of this paper is organized as follows. In section II, we discuss related work about the exploration of different sensing units and activity recognition models. Section III illustrates how to construct the tree-based and graph-based activity recognition models. Experimental setup is introduced in section IV. Section V presents the experimental results and analyses. Finally, we conclude the paper with a summary and a discussion of future work.

II. Related Work

A. Sensing Units

To adapt to different demand-oriented application scenarios, researchers have explored different sensing technologies and proposed a wealth of models for activity recognition [23], [24]. According to the used sensing units, we broadly categorize them into vision-based, environmental sensor-based, and wearable sensor-based methods [6]. As we discussed in the introduction section, wearable sensor-based methods have the advantage of easy configuration, a low cost, a wide range of applications, robustness to background change, and high degree of portability [8], [25]. Nowadays, commonly used wearable sensors include, but not limited to, accelerometer, gyroscope, electrocardiograph, Radio Frequency Identification (RFID), Global Positioning System (GPS), and light sensors [26]. Particularly, one can wear or carry multiple devices on different parts of the human body [27], [28]. Among these sensors, the accelerometer that can measure acceleration information is among the most widely used ones. For example, Bao and Intille implemented an activity recognizer with five small biaxial accelerometers that were worn on the right hip and four limb positions and they used it to recognize twenty daily activities [29]. They did experiments on the dataset that were collected by 20 volunteers and obtained an accuracy of 84.0% with a decision tree classifier. Besides the accelerometer, there are studies that use other sensing units to recognize activities and locomotion. For example, Kim *et al.*

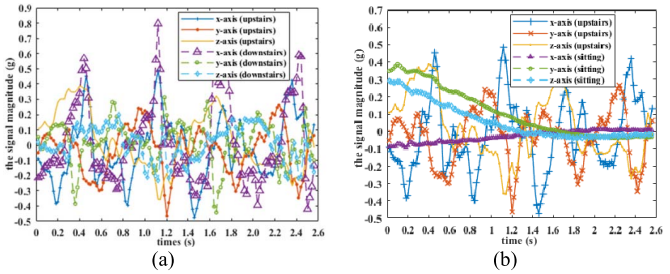
used RFID techniques to build an indoor healthcare monitoring system, where the elderly wore RFID tags and got detected by an RFID reader to infer the on-going activities and location [2]. Experimental results show the superiority over its competitors.

Besides the use of single type of sensing units, researchers explore the combination of different types of sensing units. For example, Peng *et al.* used the acceleration, location data, and vital sign to train an activity recognizer [27]. They conducted experiments on the dataset that was collected by asking volunteers to perform predefined activities. Experimental results demonstrate the superiority of the proposed method over its competitors and show that the inclusion of location data helps alleviate the cold-start problem. Garcia-Ceja *et al.* combined the sound and accelerometer data to recognize home task activities [30]. The results show that the combination of heterogeneous sensors leads to a higher recognition rate. Furthermore, the miniature of sensing units and increasing power of processors makes it possible to integrate multiple sensing units into a device. One typical example is the smartphone that is typically embedded with, among others, an accelerometer, a gyroscope, GPS, and light sensors [31], [32]. Obviously, the use of a smartphone releases users from taking extra devices and remains a priority for activity recognition due to its high adherence. For example, Wang *et al.* investigated the power of a tri-axial accelerometer and a tri-axial gyroscope in a smartphone when they were used simultaneously or separately, which shows that their fusion contributes to better performance [33].

B. Activity Recognition Model

Generally, activity recognition models can be divided into two categories: knowledge-driven model [34] and data-driven model [35]. Specifically, knowledge-driven methods rely on an abstract model of domain knowledge (e.g., logical modelling, evidential theory, and ontology modelling) to define the specification of activity and they are robust to noise and have an advantage of easy interpretation [36]. For example, Chen *et al.* applied the logical knowledge and reasoning to model human activities and the context and used it to infer the occurrence of activity [34]. However, to define the reasonable specification of activity of a new domain is not trivial. In contrast, data-driven methods utilize the collected sensor data to train an activity recognizer and to associate the sensor readings with activity labels [35], [37]. Accordingly, researchers have explored a wealth of models that include discriminant methods (e.g., support vector machine, decision tree, and conditional random field) [38], generative methods (e.g., naïve Bayes and hidden Markov model) [39], semi-supervised methods, and ensemble methods (e.g., boosting, bagging, and stacking) [30], [40]. Furthermore, to jointly optimize the feature representation and classifier training, researchers have explored the use of end-to-end deep learning to activity recognition, such as restricted Boltzmann machine (RBM), deep belief network (DBN), and convolution neural network (CNN) [19], [41], [42]. For example, Ronao and Cho proposed a deep CNN-based model to infer the on-going activities from smartphone data [20]. To capture the temporal dependency, Ordóñez and Roggen presented a convolutional long short-term memory (LSTM) network to infer activities [43]. Experimental results show the superiority over its competitors.

In contrast to the above researches, where most of them adopt a *flat model*, a *hierarchical model* considers the similarity between activities and breaks down a multi-class classification problem into a multi-stage sub-classification problem towards a better decision boundary. For example, Wang *et al.* proposed a model to first recognize human gestures at each sensor node and then infer the specific activity with the readings of other sensors and the output of the first step at a centralized device [44]. From the view of classification, the above studies basically determine the activity label of a test sample in a *down-top* fashion. On the contrary, researchers propose to construct a tree-based activity recognition model that first infer the abstract activity and then the specific activity, which works in a *top-down* scheme. Particularly, such a strategy helps better discriminate similar activities and no assumption about the temporal relations between actions and activities is explicitly made. For example, Khan *et al.* presented a two-stage activity recognizer to first classify static, dynamic or transition states and then recognize the specific activity using the linear discriminant analysis and artificial neural network [24]. Wang *et al.* relied on prior knowledge to group the predefined activities and presented a tree-based activity recognizer to organize the activities into a tree structure according to the activity characteristics [8]. Cho and Yoon presented the two-stage 1D CNN model that first optimized a CNN to classify abstract activities and then trained two CNNs to obtain the specific activity [45]. They conducted experiments to infer six simple activities, where they manually divided them into two groups. Although the existing tree-based models obtain improved performance, however, one major limitation is that most of them rely on prior knowledge to train a tree-based activity recognizer [8], [45]. Unfortunately, it is quite difficult, if not impossible, to determine the hierarchy by using the expert knowledge for new and complex scenarios, where little and even no prior knowledge is available. Even worse, inappropriate grouping of the activities could lead to degraded performance. Hence, this is quite different from our proposed *tree-based model*, where we adopt a data-driven approach to determining the relationships among the predefined activities and automatically organizing them in a tree structure. This contributes to the building of an application-dependent model. However, for a tree-based activity recognizer, if we misclassify a test sample at the first level, we do not have the chance to correct the prediction in the following steps. This requires us to explore a way that has the ability to mitigate the compounding of errors induced in the tree process. Accordingly, we propose a *graph-based model* that considers connections between any two activities instead of restricting connections of activities to a hierarchy of disjoint groups. Similar to the tree-based model, we also use the data-driven approach to obtaining a graph of the activities. Particularly, the proposed tree-based and graph-based models are general data-driven hierarchical frameworks that can take as the building blocks existing classification models to train an activity recognizer.



(a) (b)

Fig. 1. Comparison of the sensor readings of a tri-axial accelerometer (x-axis, y-axis, and z- axis) associated with three different activities. (a) *upstairs* vs. *downstairs*. (b) *upstairs* vs. *sitting*.

III. The Proposed Activity Recognition Framework

The complexity of human behavior poses a great challenge to the design of an activity recognizer for real-world applications, especially in classifying activities that can trigger similar sensor readings. Obviously, this would cause the confusion between activities and further leads to degraded discriminant capabilities. For example, Fig. 1 presents the sensor signals of a three-axis accelerometer that are associated with three different activities (*upstairs*, *downstairs*, and *sitting*). The X-axis denotes different sampling points and the Y-axis represents the magnitude of the sensor signal. We observe that *upstairs* and *downstairs* have similar sensor readings compared to the case of *upstairs* and *sitting*. Consequently, training a flat classifier to classify the three activities probably favors the discrimination of static activity (*sitting*) and dynamic activity (*upstairs* and *downstairs*) and fails to derive a good decision boundary between *upstairs* and *downstairs*. One feasible solution is to recognize activities hierarchically, where we organize the predefined activities into multiple clusters and infer activity labels within each cluster in a coarse-to-fine fashion.

Particularly, as for the hierarchical model, the key is how to determine the hierarchical structure of the activities of interest. For simple and well-studied cases, we rely on prior knowledge to organize these activities into groups and get the relationships among activities. For example, it is reasonable to categorize *sitting* and *lying* as stationary activity and *walking* and *running* as dynamic activity. However, it is not easy to obtain the hierarchical structure in the situations where we need to handle a large number of activities and expert knowledge is not available. This motivates us to automate the organization of activities and construct data-driven hierarchical models. Herein, after giving the approach to measuring the similarity between activities, we present the *tree-based model* and *graph-based model*, where the former has a tree structure and the latter is a graph.

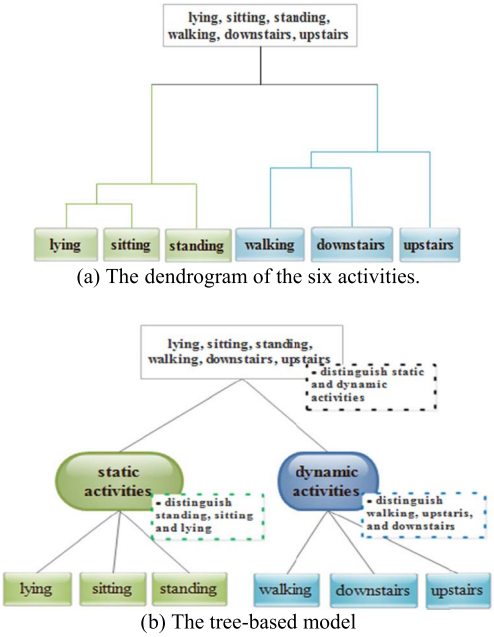
A. Tree-Based Activity Recognition Model

The core idea of the tree-based activity recognition model is to construct a tree of the predefined activities according to the similarity among activities and train classifiers for each non-leaf node. In the prediction phase, it classifies a test sample in a top-down fashion along with the tree structure and returns the predicted label at the leaf-node. As we discussed above, it is not trivial for users to determine the confusion between activities and organize the activities into a tree structure when they handle a new and complex scenario. Hence, it motivates us to explore a data-driven method to automate the process. Since similarity is associated with the recognition errors among different activities, we herein explore the use of a confusion matrix towards improving an activity recognizer in a supervised learning setting. A confusion matrix, with columns representing the instances of the predicted classes and rows denoting the instances of the actual classes (or vice versa), indicates the power of a classifier in making correct predictions and its confusion in distinguishing different classes [46]. The value CM_{ij} of the i-th row and j-th column in a confusion matrix denotes that the number of instances from the i-th activity is misclassified as the j-th activity. A larger CM_{ij} indicates greater similarity between the two activities. Accordingly, we use a confusion matrix that is associated with an activity recognizer to measure how much an activity confuses with other activities. For example, we conduct cross-validation on the UCI-HAR training set using naïve Bayes [40], and obtain the corresponding confusion matrix in Table I, where rows denote the actual activity labels and columns refer to the predicted activity labels. From Table I, we observe that *walking*, *upstairs*, and *downstairs* confuse each other and *sitting*, *standing*, and *lying* are close to each other. We also observe that the classifier seldom classifies an instance from static activity into dynamic activity (or vice versa). For example, the classifier correctly predicts all samples from static activity. Particularly, it tends to make wrong predictions within the same activity group, which indicates the confusion of within-group activities.

TABLE I
Confusion Matrix on UCI-HAR With Naïve Bayes

	Walking	Upstairs	Downstairs	Sitting	Standing	Lying
Walking	0.727	0.165	0.109	0	0	0
Upstairs	0.021	0.901	0.077	0	0	0
Downstairs	0.038	0.173	0.789	0	0	0
Sitting	0	0.012	0	0.750	0.223	0.014
Standing	0.001	0.017	0	0.256	0.722	0.005
Lying	0	0.016	0	0.422	0	0.563

To better understand the relationships of the activities, we apply a clustering algorithm to the confusion matrix and get a dendrogram that determines the clusters of activities. First, each row of the confusion matrix is first normalized to keep the values on the same scale. We then compute the distance between different rows and apply a hierarchical agglomerative clustering algorithm with the Ward's minimum variance criterion to the distances, which returns a dendrogram. Finally, we clip the dendrogram to form a two-level tree structure. For example, Fig. 2(a) presents the tree-based activity relationships associated with Table I. It reflects the similarity between different activities. The smaller the distance between two activities is, the more similar they are. We observe that *lying*, *sitting*, and *standing* form one cluster and *walking*, *downstairs*, and *upstairs* are grouped together. Also, *lying* is more similar to *sitting* than *standing*, and *walking* is more similar to *downstairs* than *upstairs*. In line with the tree, we train classifiers for each of the internal nodes. Specifically,



(a) The dendrogram of the six activities.

(b) The tree-based model

Fig. 2. Tree-based activity recognizer obtained on the UCI-HAR training set.

we train a top-level classifier to classify the top-level clusters of labels and train a second-level classifier within each cluster of activities to predict the specific activity labels. Obviously, the output of the top-level classifier determines the choice of the second-level classifier. Fig. 2(b) gives the tree-based activity recognizer that corresponds to Fig. 2(a), where the leaf-nodes denote specific activities and each internal node represents a classifier. Particularly, we create a meta-class for each internal node. The number of classes of a meta-class equals the number of its children. For example, in Fig. 2(b), we name the second-level internal nodes as "static activity" and "dynamic activity", which are not the predefined activities. The root node is a binary classification problem, and the "static activity" node is a three-classification problem.

As for the prediction, we gradually infer the specific activity label of a test sample along with the tree in a top-down scheme. Specifically, we first use the top-level classifier to predict the label of top-level clusters and then use the output to direct the selection of a second-level classifier. Afterwards, we use the selected second-level classifier to get activity labels. Algorithm 1 presents the pseudo-code of how to train a tree-based activity recognition model and use it to infer the label of a test sample. Moreover, for a dataset D with C labels, we use the notations given in Table II for better illustrations. Besides, we also present the flowchart of Algorithm 1, as shown in Fig. 3. The arrow with solid line represents the training phase and the arrow with dash line denotes the test phase. Particularly, in training a classifier *cls*, the component (a) is associated with the tree-based model.

B. Graph-Based Activity Recognition Model

Although tree-based activity recognition models enable us to automate the organization of activities, however, one drawback is that the misclassification of the top-level classifier jeopardizes the performance of the second-level classifiers. That is,

Algorithm 1 Tree-Based Activity Recognition Model

Input: a labeled train set D, activity labels L, a test sample x
Output: the activity label of x

// the construction of a tree-based activity recognition model
1. obtain the confusion matrix CM on D;
2. construct a two-level tree T of L on CM;
3. $cls\_fs = \{\}$; // a set used for storing classifiers
4. **for** each non-leaf node nd of T **do**
   4.1) search the child nodes $\downarrow(nd)$ of nd within T;
   4.2) obtain the training set $D(nd)$ using $\downarrow(nd)$;
   4.3) train a classifier cls_{nd} using $D(nd)$;
   4.4) cls_fs.add(cls_{nd}); // add it to cls_fs

// activity recognition using the tree-based model
5. set the root node of T as current node nd; //initialization
6. obtain the number of children of nd, and note it as $|\downarrow(nd)|$;
7. use cls_{nd} on x to get the next-level label A and corresponding next-level node cnd; ($1 \leq cnd \leq |\downarrow(nd)|$ and A corresponds to the maximal probability output of cls_{nd})
8. **if** $is\_leaf\_node(cnd)$ **do** // cnd is a leaf node?
   return PL as the predicted activity label; // the activity label of interest
9. **else**
   set node pnd as current node nd, and go to step 6;

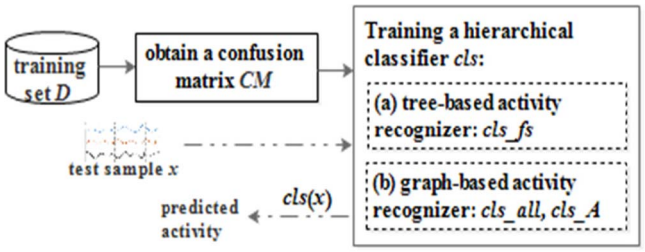


Fig. 3. Flowchart of the proposed activity recognition framework.

TABLE II
NOTATION AND CORRESPONDING MEANING

Symbol	Meaning	Symbol	Meaning
D	a dataset set	$\Downarrow(nd)$	the set of descendants of nd
$\uparrow(nd)$	the parent of nd	$\vdash(nd)$	the set of siblings of nd
$\downarrow(nd)$	the set of children of nd	$D(nd)$	the training set of nd
$\Uparrow(nd)$	the set of ancestors of nd	$\|\downarrow(nd)\|$	the number of children of nd

if a test sample is classified into the wrong clusters by the top-level classifier, the second-level classifier makes wrong decisions as well. For example, as shown in Table I, 1.7% *standing* instances are classified as *upstairs*. If an instance of *standing* is classified as dynamic activity by the top-level classifier, the second-level classifier can only classify it as *walking*, *upstairs*, or *downstairs*. To mitigate it, we present a graph-based model that can connect an activity with any other activities rather than the cluster of a subset of activities in the same tree branch. Specifically, according to the given confusion matrix, for each activity A, we find the set of activities $S(A)$ that are more easily misclassified as activity A. We herein define a confusion threshold to obtain the confusing activities of A. For example, according to Table I, if we use a confusion

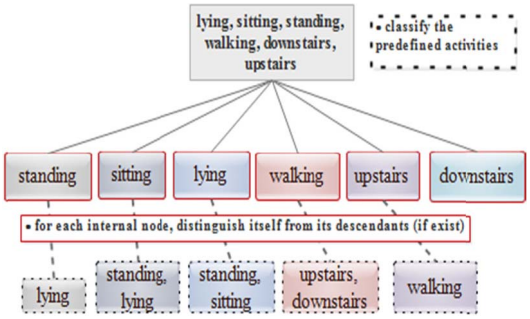


Fig. 4. Graph-based activity recognizer on UCI-HAR obtained by naïve Bayes.

threshold of 3%, the confusing activities of *downstairs* include *walking* and *upstairs*. If we use 1%, the confusing activities of *go upstairs* include *walking*, *standing*, *go downstairs*, *sitting*, and *lying*. This enables us to make correct predictions despite of the misclassification of the top-level classifier. The above example indicates that if the inferred activity label is A, its true label possibly comes from $S(A)$, and we need an activity recognizer to distinguish between $S(A)$ and A. Fig. 4 presents the corresponding graph-based model of Table I that uses a 3% confusion threshold, where we determine and give the confusing classes of the second-level nodes. That is, a second-level node has confusing activities if it has its child node. For example, the confusing activities of *standing* include *lying*, the confusing activities of *lying* include *standing* and *sitting*, and *downstairs* does not have confusing activities.

Different from the tree-based model, the graph-based model first trains a top-level classifier to distinguish all the predefined activities. Afterwards, for each activity A that has non-empty $S(A)$, we train a second-level classifier to distinguish between A and $S(A)$. In classifying an unseen sample, we first classify it using the top-level classifier. If the set $S(A)$ of the top-level prediction A is not empty, we use the second-level classifier associated with A and $S(A)$ to get the final prediction; otherwise, we report the top-level result. For Fig. 4, we use the top-level classifier to classify the six activities and use the corresponding second-level classifier to get specific activity labels. Compared with the tree-based model, the graph-based model does not involve meta-classes. Algorithm 2 presents the pseudo-code of how to train a graph-based activity recognition model and use it to make predictions, where lines 1-4 gives the training process and lines 5-7 shows how to infer the activity label of a test sample. Besides, we present the flowchart of Algorithm 2, as shown in Fig. 3, where the component (b) is associated with the graph-based model in training a classifier *cls*. The main difference between Algorithms 1 and 2 is the way of getting *cls*.

Algorithm 2 Graph-Based Activity Recognition Model

Input: a labeled train set D, activity labels L, a confusion threshold θ, a test sample x
Output: the activity label A of x

// the training of graph-based activity recognition model
1. calculate the confusion matrix CM on D; // return confusion matrix
2. f**or** each activity A of L **do**
   2.1) $S(A) = \{\}$; // initialize the set of confusing activities of A
3. **for** each activity A of L **do**
   3.1) **for** each activity B of L **do**
      if $A! = B$ and $CM(A, B) >= \theta$ **do**
         $S(A)$.add(B); // B is the confusing activity of A and add it to $S(A)$
   3.2) **if** not_empty($S(A)$) **do**
         train a classifier *cls_A* to distinguish between A and $S(A)$;
4. train a classifier *cls_all* on D to distinguish all activities;

// activity recognition using the graph-based model
5. $A = cls\_all(x)$; // return the activity label of x using the first-level classifier
6. **if** not_empty($S(A)$) **do**
   $A = cls\_A(x)$; // return the label of x using the second-level classifier
7. **return** A

IV. Experimental Setting and Results

A. Experimental Datasets

To evaluate HierHAR, we conduct comparative experiments on two public activity recognition datasets. The first dataset UCI-HAR consists of six human activities (*walking*, *standing going downstairs*, *going upstairs*, *sitting*, and *lying*) performed by thirty volunteers with a smartphone attached to their waist [31]. The used smartphone was embedded with a 3-axis accelerometer and a 3-axis gyroscope and worked at a 50 Hz sample rate. The streaming sensor readings were divided into segments with a 2.56s half-overlap sliding window. That is, each segment has 128 sensor readings. For each segment, 561 features (272 time-domain features and 289 frequency-domain features) were extracted. The second dataset Skoda Mini Checkpoint (SkodaMiCP) contains the data of ten manipulative gestures performed by the assembly-line worker in a car maintenance environment [28]. The ten gestures of interest include *write on notepad* (*WN*), *open hood* (*OH*), *close hood* (*CH*), *check gaps on the front* (*CG*), *open left front door* (*OL*), *close left front door* (*CL*), *close both left door* (*CB*), *check trunk gaps* (*CT*), *open and close trunk* (*OCT*), and *checking steering wheel* (*CSW*). The dataset was recorded for about three hours with USB sensors placed on the right and left lower and upper arm. Each USB sensor is a 3-axis accelerometer working at a 64 Hz sample rate. The sensor data were divided into 1s segments with 50% overlap between two adjacent windows. Various time-domain and frequency-domain features are extracted from the segments, such as mean, variance, median, maximum, minimum, mean cross rate, skewness, mean absolute deviation, 25^{th} percentile, 75^{th} percentile, and Hjorth parameters. The aim is to automate the recognition of activities from the wearable sensor.

For the two datasets, we can group the activities of UCI-HAR into stationary activity and dynamic activity based on the prior knowledge of human movement states. In contrast, SkodaMiCP involves a larger number of activities,

which poses a challenge to the identification of similar activities for knowledge-driven methods and also makes the recognition of activities much more challenging.

B. Constructing an Activity Recognizer

As we discussed above, the tree-based model and graph-based model are general frameworks that can take as the building blocks various classification models. Herein, we explore two different ways to design the hierarchical model and to explore the combination of different classification models. First, we use the same classification model at the top level and the second level. For illustration purpose, we call such a scheme the homogeneous mode, including the homogeneous tree-based model (HoT) and homogeneous graph-based model (HoG). Moreover, we can use different classification models in the two levels and we name it the heterogeneous mode, including the heterogeneous tree-based model (HeT) and heterogeneous graph-based model (HeG). Besides, we name the non-hierarchical model a flat model. For a flat model, it only uses one single type of classification model. As for the choice of a classifier, we explore four classification models that have different metrics, i.e., naïve Bayes (NB), k nearest neighbor with $k = 1$ (KNN), decision tree (DT), and support vector machine (SVM). These classifiers are commonly used in previous studies for activity recognition [31], [33], [38], [40]. One-versus-one strategy is used in multi-class SVM to infer the activity label of a test sample. For the performance evaluation metrics, we use *accuracy* (*Acc*), *precision* (*Prec*), *recall* (*Rec*), $F1$, and *G-mean* (*Gm*) to evaluate the activity recognizers [43]. Since $F1$ is a combination of *precision* and *recall*, we here present *precision* and $F1$ to show the recognition performance.

Precision denotes the weighted average of the correctly predicted instances for each activity label.

$$Precision = \frac{1}{C}\sum_{i=1}^{C}\frac{T_{ii}}{NP_i}, \tag{1}$$

where C is the number of classes, T_{ii} indicates the number of samples from label i that are correctly classified, and NP_i means the number of samples that are predicted with label i.

Recall indicates the fraction of correctly retrieved samples for each activity class.

$$Recall = \frac{1}{C}\sum_{i=1}^{C}\frac{T_{ii}}{NT_i}, \tag{2}$$

where NT_i means the number of instances with true label i.

To account for the class imbalance problem, $F1$ is the harmonic mean of precision and recall.

$$F1 = \frac{2^{*}precision^{*}recall}{precision + recall} \tag{3}$$

Similarity, G-mean also considers the case of class imbalance and equals the root of the product of recall $recall_i$ for each class i.

$$G - mean = \sqrt{\prod_{i=1}^{C} recall_i} \tag{4}$$

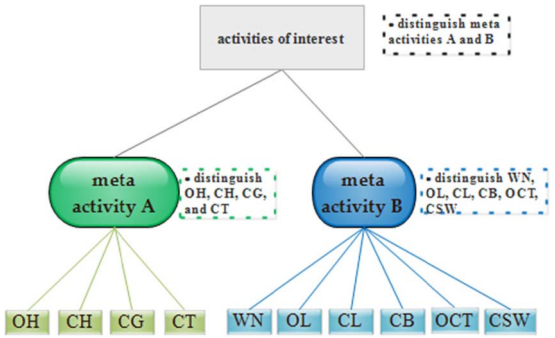


Fig. 5. Tree-based activity recognizer on SkodaMiCP obtained with NB.

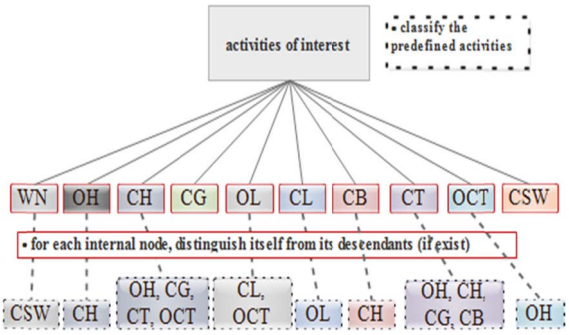


Fig. 6. Graph-based activity recognizer on SkodaMiCP obtained with NB.

V. Experimental Results and Analysis

A. Tree-Based and Graph-Based Activity Recognizers

According to the steps of Algorithms 1 and 2, we construct the tree-based as well as graph-based activity recognizers for each of the experimental datasets. For UCI-HAR, Figs. 2 and 4 present the activity recognizers for UCI-HAR using NB, where the confusion threshold is 0.03 for the graph-bases model. Similarly, we can obtain the tree-based and graph-based models that are associated with KNN, DT, and SVM, respectively. Due to space limitations, we omit the models here. For SkodaMiCP, we give the tree-based and graph-based activity recognizers when NB is used. Fig. 5 presents the tree-based activity recognizer and Fig. 6 gives the graph-based one with a confusion threshold of 3%. Similar to Fig. 2, the task of the root node in Fig. 5 is to classify meta activity A and meta activity B, and the task of meta activity A (meta activity B) is to distinguish *OH*, *CH*, *CG*, and *CT* (*WN*, *OL*, *CL*, *CB*, *OCT*, and *CSW*). For Fig. 6, the child node of a second-level node, if exists, denotes the confusing activities of the second-level node. For example, the confusing activities of the second-level node *WN* include *CSW*, and the confusing activities of *OL* include *CL* and *OCT*. From Fig. 6, we see that there are connections between activities (e.g., the connection between *CH* and *OCT*) that are in disjoint groups in the tree-based model. This indicates the power of graph-based model in capturing the confusion from different groups. Similarly, we can optimize activity recognizers for KNN, DT, as well as SVM.

B. Recognition Performance

We herein evaluate both homogeneous and heterogeneous models on test sets with the tree-based and graph-based

TABLE III
RECOGNITION PERFORMANCE ON UCI-HAR OF FLAT, TREE-, AND GRAPH-BASED MODELS

Classifier		NB				KNN				DT				SVM			
Metrics (%)		*Acc*	*Prec*	*F1*	*Gm*	*Acc*	*Prec*	*F1*	*Gm*	*Acc*	*Prec*	*F1*	*Gm*	*Acc*	*Prec*	*F1*	*Gm*
AR	Flat	76.99	41.03	55.10	79.63	87.85	58.54	72.55	90.74	86.36	55.82	69.17	88.14	96.40	82.83	82.83	82.83
	HoT	76.86	40.86	54.95	79.54	87.85	58.54	72.55	90.74	86.29	55.71	68.97	87.94	96.44	82.97	90.36	97.52
	HeT	**95.69**	**80.00**	**88.57**	**97.07**	**96.40**	**82.83**	**90.28**	**97.50**	**96.44**	**82.97**	**90.36**	**97.52**	**96.44**	**82.97**	**90.36**	**97.52**
	HoG	76.99	41.03	55.10	79.63	85.34	53.62	68.70	89.21	83.88	51.55	59.57	78.16	96.40	82.83	90.28	97.50
	HeG	95.11	77.76	87.26	96.79	89.72	62.78	75.78	91.98	95.42	80.13	87.67	95.96	96.40	82.83	90.27	97.50

TABLE IV
RECOGNITION PERFORMANCE ON SKODAMICP OF FLAT, TREE-, AND GRAPH-BASED MODELS

Classifier		NB				KNN				DT				SVM			
Metrics (%)		*Acc*	*Prec*	*F1*	*Gm*	*Acc*	*Prec*	*F1*	*Gm*	*Acc*	*Prec*	*F1*	*Gm*	*Acc*	*Prec*	*F1*	*Gm*
AR	Flat	73.68	30.66	46.90	83.67	78.83	34.05	48.90	82.16	92.91	62.37	76.54	95.52	25.52	0	-	0
	HoT	62.94	23.91	38.57	76.13	72.49	28.23	42.77	78.79	92.47	60.95	75.38	95.14	12.79	0	-	0
	HeT	42.72	0	-	0	12.86	0	-	0	35.76	0	-	0	12.79	0	-	0
	HoG	73.68	30.66	46.90	83.67	79.25	34.53	49.40	82.41	93.08	62.93	76.96	95.62	25.52	0	-	0
	HeG	**82.37**	**39.78**	**56.84**	**89.28**	**82.24**	**38.39**	**53.10**	**83.89**	**93.23**	**63.46**	**77.35**	**95.71**	**25.52**	0	-	0

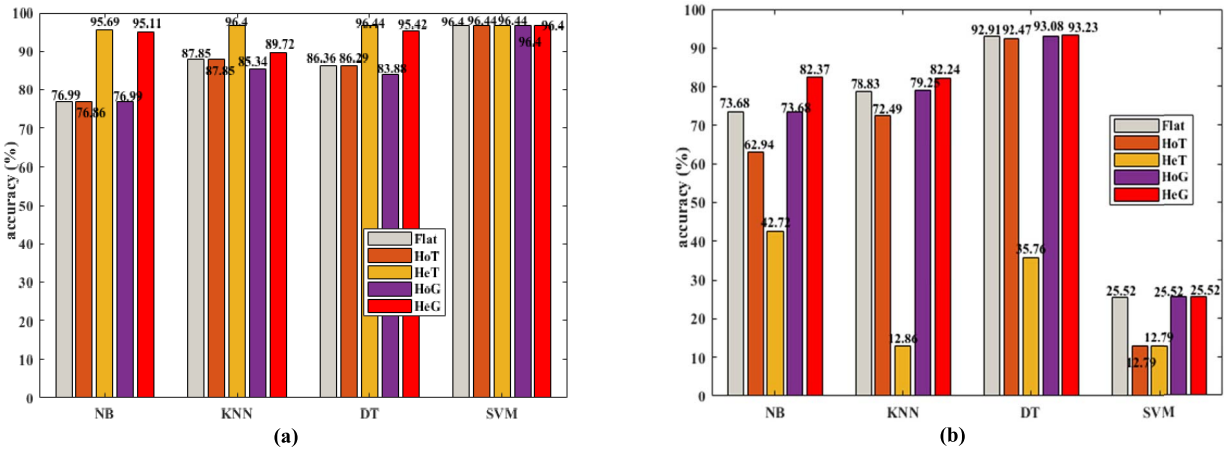


Fig. 7. Accuracy of different activity recognition models on the two datasets. (a) UCI-HAR; (b) SkodaMiCP.

models. Specifically, for the homogeneous models, we use NB, KNN, DT, and SVM. For the heterogeneous models, we choose SVM at the second level and other classifier (i.e., NB, KNN, DT, or SVM) at the top level. Notably, we evaluate the combination of different classification models and present the results in the following section. Tables III-IV show the results on UCI-HAR and SkodaMiCP, where AR denotes an activity recognizer. To facilitate the comparison, Fig. 7 shows the accuracy of different methods on UCI-HAR and SkodaMiCP. First, according to the results, we observe that the tree-based model has mixed results. Specifically, HeT outperforms HoT on UCI-HAR, while HoT performs better than HeT on SkodaMiCP. For the graph-based model, HeG consistently performs better than HoG. Second, in terms of the tree-based model and flat model, the flat model achieves a higher recognition rate in some cases. Particularly, the flat model outperforms both HoT and HeT on SkodaMiCP. The main reason is that the tree-based model probably induces compounding errors in its two-stage prediction process. Third, when we compare the flat model and graph-based model, we see that the graph-based model performs better in the majority of cases. This is mainly because the graph-based model can benefit from the finer-step to distinguish confusing activities. Fourth, compared with the tree-based model, the graph-based model obtains consistently better generalization ability. Specifically, both achieve comparable performance on UCI-HAR, while the graph-based model outperforms the tree-based model on SkodaMiCP. The main reason is that, in the case of UCI-HAR, the graph-based model only has connections between activities of the same clusters as the tree-based model and fails to capture the confusion of different types of activities. For SkodaMiCP, however, it has a larger number of activities and the graph-based model better captures the relations of different activities that are in disjoint groups in the tree-based model.

In addition, to investigate the performance improvement, we investigate the confusion matrix. Figs. 8 and 9 present the confusion matrices of the two datasets with NB used in the top-level. The rows refer to the actual labels and the columns denote the predicted labels. From Fig. 8, we can

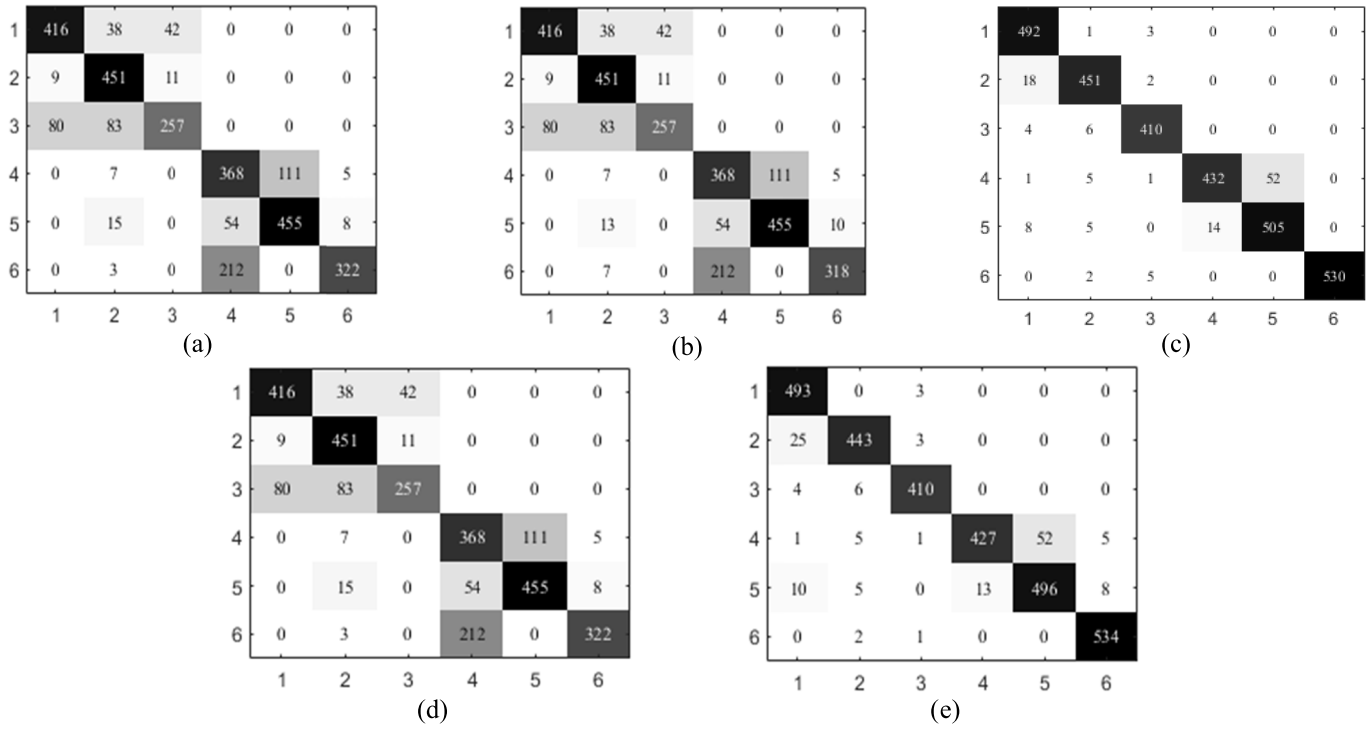


Fig. 8. Confusion matrix on UCI-HAR with NB used at the top-level. (a) Flat; (b) HoT; (c) HeT; (d) HoG; (e) HeG.

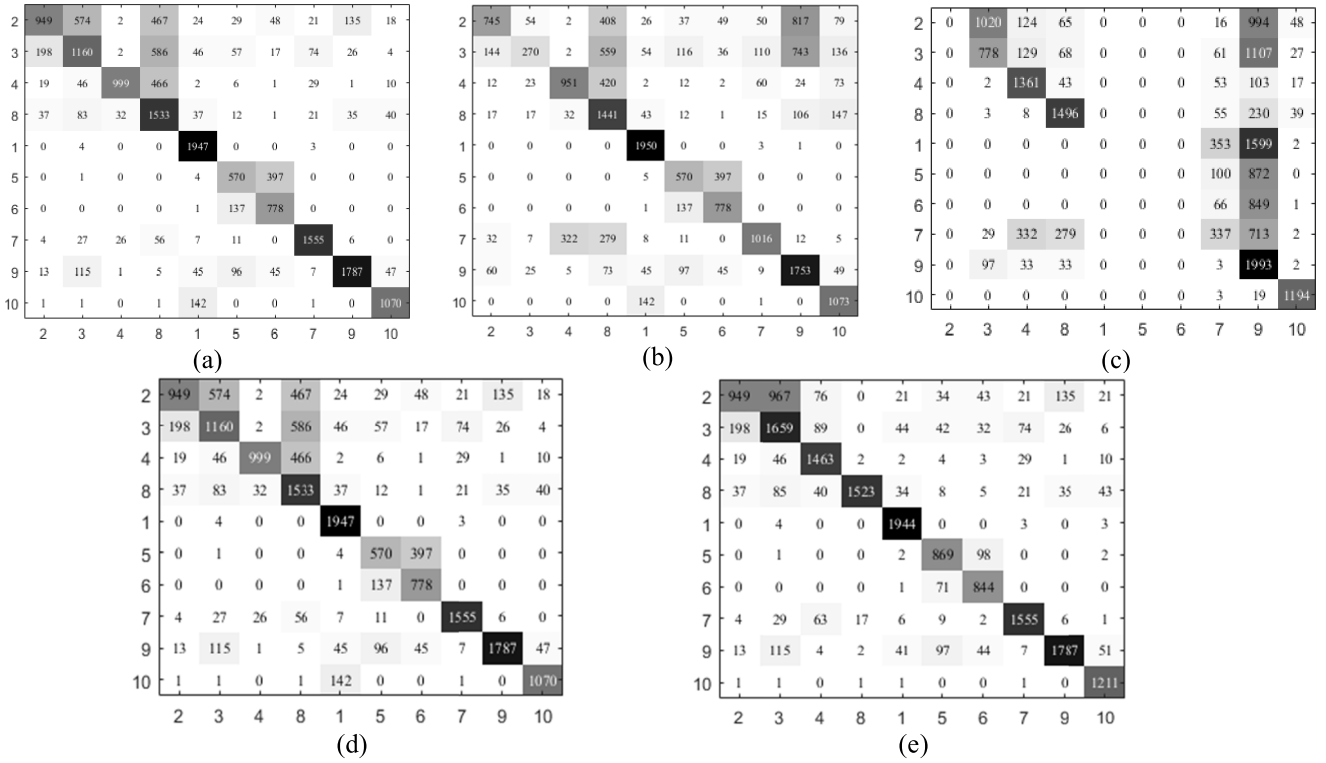


Fig. 9. Confusion matrix on SkodaMiCP with NB used at the top-level. (a) Flat; (b) HoT; (c) HeT; (d) HoG; (e) HeG.

observe that the heterogeneous model performs better than the homogeneous model. In terms of the flat model, HeT and HeG, we observe HeT and HeG better classify the similar activities. For example, the flat model misclassifies 212 *lying* samples as *standing*, while HeT and HeG correctly classify 530 and 534 *lying* test samples. From Fig. 9, we observe that the graph-based model has an advantage of correcting the confusion errors that happen in the tree-based model, which demonstrates the superiority of the graph-based model in complex scenarios.

C. Evaluation of Hyperparameter

The confusion threshold θ is an important parameter of the graph-based model, which determines the construction

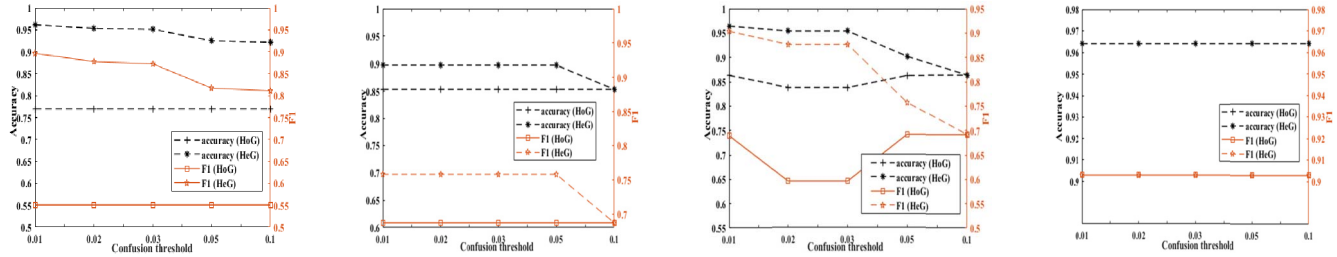


Fig. 10. Recognition accuracy and F1 vs. different confusion thresholds on UCI-HAR using different top-level classifiers. (a) NB; (b) KNN; (c) DT; (d) SVM.

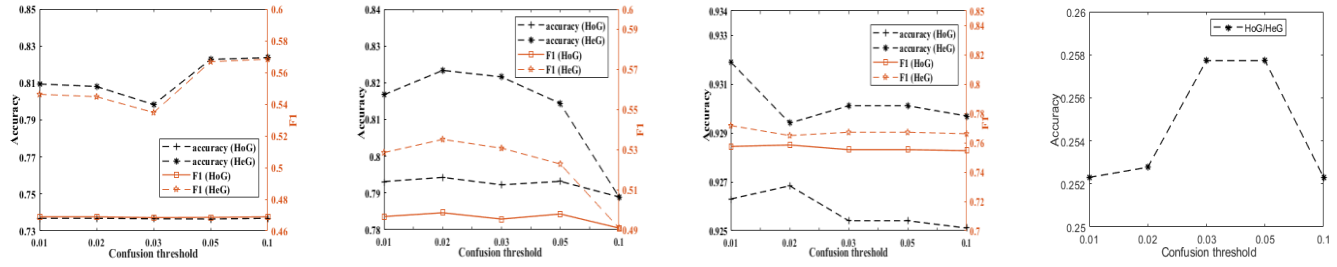


Fig. 11. Recognition accuracy and F1 vs. different confusion thresholds on SkodaMiCP using different top-level classifiers. (a) NB; (b) KNN; (c) DT; (d) SVM.

TABLE V
PERFORMANCE ON UCI-HAR WITH THE COMBINATION OF DIFFERENT CLASSIFIERS

Classifier	NB-NB		NB-KNN		NB-DT		NB-SVM		KNN-NB		KNN-KNN		KNN-DT		KNN-SVM	
Metrics (%)	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
HoT/HeT	76.86	54.95	87.17	71.45	85.48	67.71	**95.69**	88.57	77.30	55.43	87.85	72.55	86.22	68.87	**96.40**	90.28
HoG/HeG	76.99	55.10	86.87	71.10	85.34	67.81	95.11	87.26	86.39	70.27	85.34	68.70	85.95	69.60	89.72	75.78
Classifier	DT-NB		DT-KNN		DT-DT		DT-SVM		SVM-NB		SVM-KNN		SVM-DT		SVM-SVM	
Metrics (%)	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
HoT/HeT	77.30	55.43	87.85	72.55	86.29	68.97	**96.44**	90.36	77.30	55.43	87.85	72.55	86.29	68.97	**96.44**	90.36
HoG/HeG	86.70	68.49	88.60	73.63	83.88	59.57	95.42	87.67	92.78	82.62	92.64	81.93	92.50	81.66	96.40	90.28

of the second-level classifiers. Herein, we evaluate its association with the recognition performance. According to our preliminary work, the candidate values of θ used in this study include 0.01, 0.02, 0.03, 0.05, and 0.1. Accordingly, Figs. 10-11 present the accuracy and $F1$ on UCI-HAR and SkodaMiCP, respectively. The X-axis denotes the optional values of θ and the Y-axis gives the corresponding accuracy. For each dataset, we use SVM at the second level, and four different classification models (i.e., NB, KNN, SVM, and DT) are used to train the top-level classifier. Therefore, we plot four figures for each dataset and compare the results of HoG and HeG. From Figs. 10 and 11, we observe that the value of θ indeed influences the performance of the homogeneous model and heterogeneous model. There is a general trend that the accuracy first increases and then decreases with the increase of the value of θ. Also, 3% is a reasonable choice and the graph-based model works well in the majority of cases. The possible reason is that the value of θ is associated with the graph structure. If θ is set a large value, there will be sparse connections (even no connection) between the activities at the second level. This reduces the graph-based model to the flat model. In contrast, a small value of θ will return a densely connected graph. We also observe that the heterogeneous model performs better than the homogeneous model, which is consistent with previous results.

D. Evaluation of the Combination of Classifiers

We explore the combinations of different classifiers in the tree-based and graph-based models. Specifically, we can use NB, KNN, DT, or SVM at the top level and use NB, KNN, DT or SVM at the second level. Tables V-VI present the results on UCI-HAR and SkodaMiCP, respectively. Experimental results are grouped by the top-level classifier and the best results for each group are shown in bold. For UCI-HAR, we observe that the use of SVM at the second level generally outperforms its competitors. For SkodaMiCP, the homogeneous model is inferior to that of the heterogeneous model that uses SVM at the second level. This is consistent with the experimental results in the previous subsection. Also, we observe the use of decision tree at the second level tends to obtain better performance.

TABLE VI
PERFORMANCE ON SKODAMICP WITH THE COMBINATION OF DIFFERENT CLASSIFIERS

Classifier	NB-NB		NB-KNN		NB-DT		NB-SVM		KNN-NB		KNN-KNN		KNN-DT		KNN-SVM	
Metrics (%)	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
HoT/HeT	62.94	38.57	66.80	38.84	74.82	47.72	42.72	-	67.28	41.09	72.60	42.74	85.42	60.80	12.86	-
HoG/HeG	73.68	46.90	85.58	61.08	**89.93**	69.75	79.97	53.69	74.93	45.00	79.30	49.82	**87.01**	60.98	82.24	53.10
Classifier	DT-NB		DT-KNN		DT-DT		DT-SVM		SVM-NB		SVM-KNN		SVM-DT		SVM-SVM	
Metrics (%)	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
HoT/HeT	73.92	47.05	82.08	53.48	92.53	75.49	35.76	-	66.73	41.10	72.62	42.92	84.61	59.89	12.79	-
HoG/HeG	84.72	60.05	91.55	73.11	**92.73**	75.97	93.23	77.35	30.40	-	25.23	-	**92.43**	75.31	43.32	-

VI. CONCLUSION

From the perspective of data analysis, activity recognition chain (ARC) mainly consists of sensor data acquisition, data preprocessing, feature extraction, and model training and classification, where a classification model is generally first trained on the collected sensor data and then used for making predictions. Accordingly, the training of a discriminant activity recognition model functions as middleware in bridging the gap between the raw sensor data and high-level applications for activity recognition supported Internet of Things (IoT) services. Particularly, the recognition accuracy of an activity recognizer would largely determine the acceptability of an IoT application. Therefore, studies in the model construction guide users in choosing activity recognition models and the improved activity recognizer contributes to the development of an IoT application. Although the development of wearable sensors greatly facilitates the automatic recognition of activities, the complexity of human behavior that is characteristically associated with uncertainty and concurrency poses a serious challenge to the design of an activity recognizer. Particularly, even different activities can trigger similar sensor data, which inevitably degrades the discriminant ability of a classifier. To this end, we present a data-driven hierarchical framework HierHAR for sensor-based activity recognition. Specifically, we propose to automate the organization of similar activities and adopt a hierarchical structure to gradually infer the specific activity label towards a better decision boundary of activities. We then present two hierarchical activity recognition models: tree-based model and graph-based model, where the former organizes the activities into a tree structure and the latter essentially builds a graph to connect activities. Particularly, both are data-driven approaches that use sensor data to train an activity recognizer and the graph-based model helps reduce the compounding error. Finally, we conduct extensive experiments to evaluate HierHAR. The results demonstrate its effectiveness in automating the organization of activities and in training two activity recognition models and also indicate the superiority of graph-based activity recognizer.

One limitation of the study is the offline data analysis, which may be different from the real-time situations under time and resource constraints. On the one hand, since the procedure of both online and offline schemes involves the construction and optimization of an activity recognizer, our study provides an objective metric for the choice of an activity recognizer. On the other hand, this motivates us to conduct further researches on evaluating the performance of HierHAR and co-optimizing the hardware, software, and algorithms in real-time cases. Besides, we plan to work along with the following directions in the future. First, in addition to the applications such as behavior analysis, wellness evaluation and chronic disease management, there are critical tasks, especially in IoT services, that have little tolerance to time-delay in data collection and analysis, hence, a real-time activity recognizer is expected. Considering that the rapid development of edge computing greatly facilitates the implementation of the proposed method into the intelligent edge to better support (near) real-time IoT applications, we plan to explore and evaluate its combination with the edge computing. Particularly, this also involves further studies in the choice of sensing units and edge devices, the selection of informative features, and the optimization of a classification model. Second, we adopt an empirical value of the confusion threshold in the graph-based model, which requires users to set an appropriate value. Hence, how to adaptively estimate its optimal value by utilizing information latent in the dataset requires further study. Third, recent years have witnessed the great success of deep learning in many fields such as computer vision and activity recognition. Considering that the proposed models are general frameworks, we can incorporate deep learning into HierHAR. This remains another topic on conducting a systematic study.

REFERENCES

[1] P. N. Dawadi, D. J. Cook, and M. Schmitter-Edgecombe, "Automated cognitive health assessment from smart home-based behavior data," *IEEE J. Biomed. Health Informat.*, vol. 20, no. 4, pp. 1188–1194, Jul. 2016.

[2] S.-C. Kim, Y.-S. Jeong, and S.-O. Park, "RFID-based indoor location tracking to ensure the safety of the elderly in smart home environments," *Pers. Ubiquitous Comput.*, vol. 17, no. 8, pp. 1699–1707, Dec. 2013.

[3] P. Cottone, S. Gaglio, G. Lo Re, and M. Ortolani, "User activity recognition for energy saving in smart homes," *Pervasive Mobile Comput.*, vol. 16, pp. 156–170, Jan. 2015.

[4] G. Acampora, D. J. Cook, P. Rashidi, and A. V. Vasilakos, "A survey on ambient intelligence in healthcare," *Proc. IEEE*, vol. 101, no. 12, pp. 2470–2494, Dec. 2013.

[5] N. K. Suryadevara and S. C. Mukhopadhyay, "Determining wellness through an ambient assisted living environment," *IEEE Intell. Syst.*, vol. 29, no. 3, pp. 30–37, May 2014.

[6] O. D. Lara and M. A. Labrador, "A survey on human activity recognition using wearable sensors," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1192–1209, 3rd Quart., 2013.

[7] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu, "Sensor-based activity recognition," *IEEE Trans. Syst. Man Cybern. C, Appl. Rev.*, vol. 42, no. 6, pp. 790–808, Nov. 2012.

[8] A. Wang, G. Chen, X. Wu, L. Liu, N. An, and C.-Y. Chang, "Towards human activity recognition: A hierarchical feature selection framework," *Sensors*, vol. 18, no. 11, p. 3629, Oct. 2018.

[9] E. Kim, S. Helal, C. Nugent, and M. Beattie, "Analyzing activity recognition uncertainties in smart home environments," *ACM Trans. Intell. Syst. Technol.*, vol. 6, no. 4, p. 52, 2015.
[10] J.-H. Hong, J. Ramos, and A. K. Dey, "Toward personalized activity recognition systems with a semipopulation approach," *IEEE Trans. Human-Mach. Syst.*, vol. 46, no. 1, pp. 101–112, Feb. 2016.
[11] P. Rashidi, D. J. Cook, L. B. Holder, and M. Schmitter-Edgecombe, "Discovering activities to recognize and track in a smart environment," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 4, pp. 527–539, Apr. 2011.
[12] A. Stisen *et al.*, "Smart devices are different: Assessing and mitigating mobile sensing heterogeneities for activity recognition," in *Proc. 13th ACM Conf. Embedded Netw. Sensor Syst.*, Seoul, South Korea, 2015, pp. 127–140.
[13] R. Poppe, "A survey on vision-based human action recognition," *Image Vis. Comput.*, vol. 28, no. 6, pp. 976–990, Jun. 2010.
[14] G. Chen, A. Wang, S. Zhao, L. Liu, and C.-Y. Chang, "Latent feature learning for activity recognition using simple sensors in smart homes," *Multimedia Tools Appl.*, vol. 77, no. 12, pp. 15201–15219, Jun. 2018.
[15] S. Dernbach, B. Das, N. C. Krishnan, B. L. Thomas, and D. J. Cook, "Simple and complex activity recognition through smart phones," in *Proc. 8th Int. Conf. Intell. Environ.*, Guanajuato, Mexico, Jun. 2012, pp. 214–221.
[16] R. Gravina, P. Alinia, H. Ghasemzadeh, and G. Fortino, "Multi-sensor fusion in body sensor networks: State-of-the-art and research challenges," *Inf. Fusion*, vol. 35, pp. 68–80, May 2017.
[17] E. M. Tapia *et al.*, "Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart monitor," in *Proc. 11th IEEE Int. Symp. Wearable Comput.*, Washington, DC, USA, 2007, pp. 1–4.
[18] F. Ordóñez, P. de Toledo, and A. Sanchis, "Activity recognition using hybrid generative/discriminative models on home environments using binary sensors," *Sensors*, vol. 13, no. 5, pp. 5460–5477, 2013.
[19] T. Plötz, N. Y. Hammerla, and P. L. Olivier, "Feature learning for activity recognition in ubiquitous computing," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, Barcelona, Spain, 2011, pp. 1729–1734.
[20] C. A. Ronao and S.-B. Cho, "Human activity recognition with smartphone sensors using deep learning neural networks," *Expert Syst. Appl.*, vol. 59, pp. 235–244, Oct. 2016.
[21] L. Peng, L. Chen, X. Wu, H. Guo, and G. Chen, "Hierarchical complex activity representation and recognition using topic model and classifier level fusion," *IEEE Trans. Biomed. Eng.*, vol. 64, no. 6, pp. 1369–1379, Jun. 2017.
[22] Y. Liu, L. Nie, L. Liu, and D. S. Rosenblum, "From action to activity: Sensor-based activity recognition," *Neurocomputing*, vol. 181, pp. 108–115, Mar. 2016.
[23] J. Rafferty, C. D. Nugent, J. Liu, and L. Chen, "From activity recognition to intention recognition for assisted living within smart homes," *IEEE Trans. Human-Mach. Syst.*, vol. 47, no. 3, pp. 368–379, Jun. 2017.
[24] A. M. Khan, Y.-K. Lee, S. Y. Lee, and T.-S. Kim, "A triaxial accelerometer-based physical-activity recognition via augmented-signal features and a hierarchical recognizer," *IEEE Trans. Inf. Technol. Biomed.*, vol. 14, no. 5, pp. 1166–1172, Sep. 2010.
[25] A. Mannini and S. S. Intille, "Classifier personalization for activity recognition using wrist accelerometers," *IEEE J. Biomed. Health Informat.*, vol. 23, no. 4, pp. 1585–1594, Jul. 2019.
[26] D. Patterson, D. Fox, H. Kautz, and M. Philipose, "Fine-grained activity recognition by aggregating abstract object usage," in *Proc. IEEE Int. Symp. Wearable Comput.*, Osaka, Japan, Oct. 2005, pp. 44–51.
[27] L. Peng, L. Chen, M. Wu, and G. Chen, "Complex activity recognition using acceleration, vital sign, and location data," *IEEE Trans. Mobile Comput.*, vol. 18, no. 7, pp. 1488–1498, Jul. 2019.
[28] P. Zappi *et al.*, "Activity recognition from on-body sensors: Accuracy-power trade-off by dynamic sensor selection," in *Proc. Eur. Conf. Wireless Sensor Netw.*, Bologna, Italy, 2008, pp. 17–33.
[29] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in *Proc. Int. Conf. Pervasive Comput.*, Vienna, Austria, 2004, pp. 1–17.
[30] E. Garcia-Ceja, C. E. Galván-Tejada, and R. Brena, "Multi-view stacking for activity recognition with sound and accelerometer data," *Inf. Fusion*, vol. 40, pp. 45–56, Mar. 2018.
[31] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," in *Proc. 21th Eur. Symp. Artif. Neural Netw., Comput. Intell. Mach. Learn.*, Bruges, Belgium, 2013, pp. 437–442.
[32] J.-L. Reyes-Ortiz, L. Oneto, A. Sam, X. Parra, and D. Anguita, "Transition-aware human activity recognition using smartphones," *Neurocomputing*, vol. 171, pp. 754–767, Jan. 2016.
[33] A. Wang, G. Chen, J. Yang, S. Zhao, and C.-Y. Chang, "A comparative study on human activity recognition using inertial sensors in a smartphone," *IEEE Sensors J.*, vol. 16, no. 11, pp. 4566–4578, Jun. 2016.
[34] L. Chen, C. D. Nugent, and H. Wang, "A knowledge-driven approach to activity recognition in smart homes," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 6, pp. 961–974, Jun. 2012.
[35] N. C. Krishnan and D. J. Cook, "Activity recognition on streaming sensor data," *Pervasive Mobile Comput.*, vol. 10, pp. 138–154, Feb. 2014.
[36] N. Rodríguez, M. Cuéllar, J. Lilius, and M. Calvo-Flores, "A survey on ontologies for human behavior recognition," *ACM Comput. Surv.*, vol. 46, no. 4, p. 43, 2014.
[37] J. Ye, G. Stevenson, and S. Dobson, "KCAR: A knowledge-driven approach for concurrent activity recognition," *Pervasive Mobile Comput.*, vol. 19, pp. 47–70, May 2015.
[38] V. Vapnik and O. Chapelle, "Bounds on error expectation for support vector machines," *Neural Comput.*, vol. 12, no. 9, pp. 2013–2036, Sep. 2000.
[39] S. Fine, Y. Singer, and N. Tishby, "The hierarchical hidden Markov model: Analysis and applications," *Mach. Learn.*, vol. 32, no. 1, pp. 41–62, 1998.
[40] X. Wu *et al.*, "Top 10 algorithms in data mining," *Knowl. Inf. Syst.*, vol. 14, no. 1, pp. 1–37, 2008.
[41] J. Yang, M. Nguyen, P. San, X. L. Li, and S. Krishnaswamy, "Deep convolutional neural networks on multichannel time series for human activity recognition," in *Proc. 24th Int. Joint Conf. Artif. Intell.*, Buenos Aires, Argentina, 2015, pp. 3995–4001.
[42] Y. Guan and T. Plötz, "Ensembles of deep LSTM learners for activity recognition using wearables," *ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 1, no. 2, 2017, Art. no. 11.
[43] F. Ordóñez and D. Roggen, "Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, Jan. 2016.
[44] L. Wang, T. Gu, X. Tao, and J. Lu, "A hierarchical approach to real-time activity recognition in body sensor networks," *Pervasive Mobile Comput.*, vol. 8, no. 1, pp. 115–130, Feb. 2012.
[45] H. Cho and S. Yoon, "Divide and conquer-based 1D CNN human activity recognition using test data sharpening," *Sensors*, vol. 18, no. 4, p. 1055, Apr. 2018.
[46] S. V. Stehman, "Selecting and interpreting measures of thematic classification accuracy," *Remote Sens. Environ.*, vol. 62, no. 1, pp. 77–89, Oct. 1997.

Aiguo Wang received the B.S. and Ph.D. degrees from the Hefei University of Technology, China, in 2010 and 2015, respectively.

He is currently a Distinguished Researcher with the School of Electronic Information Engineering, Foshan University, Guangdong, China. His current research interests include machine learning, data mining, and pervasive computing.

Shenghui Zhao received the M.S. degree from the Hefei University of Technology, China, in 2003, and the Ph.D. degree from Southeast University, China, in 2013.

She is currently a Professor with the School of Computer and Information Engineering, Chuzhou University, Anhui, China. Her current research interests include trusted computing, healthcare, and the Internet of Things.

Chundi Zheng received the M.S. degree from Xidian University, Xi'an, China, in 2006, and the Ph.D. degree from Tsinghua University, Beijing, China, in 2013.

He is currently an Associate Professor with the School of Electronic Information Engineering, Foshan University, Guangdong, China. His research interests include signal processing and sparse recovery.

Huihui Chen received the Ph.D. degree in computer science from Northwestern Polytechnical University, China, in 2017.

She is currently a Lecturer with the School of Electronic Information Engineering, Foshan University, Guangdong, China. Her research interests include ubiquitous computing and mobile crowdsensing.

Li Liu received the M.S. degree from Lanzhou University, Lanzhou, China, and the Ph.D. degree in computer science from the University of Paris-Sud 11.

He is currently a Professor with the School of Big Data and Software Engineering, Chongqing University, Chongqing, China. His research interests include mobile and ubiquitous computing, data analysis, and their applications on health and behavior.

Guilin Chen received the B.S. degree from Anhui Normal University, China, in 1985, and the M.S. degree from the Hefei University of Technology, in 2007.

He is currently a Professor with the School of Computer and Information Engineering, Chuzhou University, Anhui, China. His current research interests include cloud computing, wireless networks, healthcare, and the Internet of Things.