Multi-Resident Activity Recognition with Unseen Classes in Smart Homes

Wei Wang
LILY, Interdisciplinary Graduate School
Nanyang Technological University
Singapore
wwang008@e.ntu.edu.sg

Chunyan Miao
School of Computer Science and Engineering
Nanyang Technological University
Singapore
ascymiao@ntu.edu.sg

***Abstract*—Multi-resident activity recognition is important to many applications in smart homes. Existing works in this area are usually based on supervised classification methods, and can only recognize the predefined activity classes that are labeled in the training data. However, in many practical applications, the activities needed to recognize contain not only the predefined activities but also the previously unseen activities. In this paper, we propose a method to solve the problem of multi-resident activity recognition with the previously unseen activity classes. Our method utilizes the techniques of multi-task learning and zero-shot learning. It regards the activity recognition of each resident as a learning task, and learns all tasks jointly. By utilizing zero-shot learning techniques, previously unseen activity classes can be recognized. We conduct extensive experiments on real-world datasets. The experimental results show the effectiveness of our method.**

***Index Terms*—Activity recognition, multiple residents, smart homes, unseen activity classes**

I. Introduction

Smart homes aim to assist and facilitate the living of the residents through providing context-aware services [1]. To provide these services, being aware of the residents' activities is needed, as many services are based on the activities. Thus, activity recognition plays an important role in many applications in smart homes.

To collect information about the activities, various kinds of devices can be used. One kind of the devices is the cameras, which have been widely used in the public environments [2]. However, in the home environment, as the privacy of the residents is important, it is not suitable to use them [2], [3]. Another kind of the devices is the wearable sensors. For these devices, the residents always need to remember to wear them. Thus, they can bring inconvenience to the residents [2], [4]. In existing works on smart home activity recognition, the devices widely used are the ambient sensors deployed at fixed locations in the house. Examples of these sensors are the motion sensors fixed on the ceiling, switch sensors attached on the doors and pressure sensors attached on the seats [5], [6]. As these sensors are deployed at fixed locations in the house, the residents don't need to wear them. In addition, they are considered as less intrusive [7] and more privacy preserving [8]. Because of these advantages, the ambient sensors obtain wide adoption in smart homes.

There have been many works on smart home activity recognition based on the ambient sensors [7], [8]. However, these works mainly focus on the single-resident setting. Under this setting, there is only one resident in the home environment. All sensor readings are solely about this resident. However, in reality, the residents usually do not live alone. In a home environment, there are usually multiple residents. Activity recognition with multiple residents is a more practical setting. Compared with the single-resident setting, activity recognition under the multi-resident setting is more challenging. Under this setting, the sensor readings about different residents are mixed together. In addition, the activities of one resident may be influenced by the activities of other residents. Due to these reasons, the number of research works on multi-resident activity recognition is much smaller. In recent years, it obtains increasing interest in the area of smart home activity recognition and gets some developments [9].

In activity recognition problems, to learn the recognition model, sufficient labeled training instances are needed. To get these instances, adequate sensor readings are required to be collected and labeled. However, during data labeling, only predefined activities that are of interest are labeled, and classification model for these predefined activities is learned with the labeled instances. In many practical applications, when we use the learned model to recognize activities, not only the predefined activities, but also some previously unseen activities are needed to recognize. This phenomenon also occurs in many applications related to multi-resident activity recognition. In these cases, as the process of sensor reading collecting and labeling is costly [10], it is not practical to recollect sensor readings and retrain a new recognition model. Methods that both can recognize instances belonging to the predefined activities and can recognize instances belonging to the previously unseen activities are needed. Traditional classification methods, such as SVM and decision tree, can not achieve this. Because in these methods, for each class to recognize, labeled training instances belonging to this class are needed in the model learning phase. As an emerging learning paradigm, zero-shot learning [11] can be used to address this problem. Zero-shot learning aims to recognize instances belonging to the previously unseen classes.

In this paper, we propose to address the problem of multi-

resident activity recognition with the previously unseen activity classes in the smart home environment. This is a problem of practical value and is also challenging. In this problem, not only activities coming from multiple residents are needed to recognize, but also these activities contain the previously unseen activities. There are existing works on multi-resident activity recognition [9], but they are not for the case of unseen activities. On the other hand, existing works on sensor-based unseen activity recognition generally focus on the setting of single person [10], [12]. In this paper, we propose a method to address the problem of multi-resident activity recognition with unseen activity classes. Based on the problem setting, we refer to our method as "multi-resident with unseen-activity-class recognition method (MRUA)". Our MRUA method is based on multi-task learning and zero-shot learning techniques. By regarding the activity recognition of each resident as a task, all tasks are learned jointly.

The primary contribution we made in this paper is proposing to tackle the problem of multi-resident activity recognition with unseen classes through the multi-task learning and zero-shot learning techniques. In this way, we propose a method MRUA to address this problem. We conduct extensive experiments on real world datasets, and show the effectiveness of our method.

II. Related Work

In this section, we briefly review related works in two research areas: multi-resident activity recognition in smart homes, and zero-shot learning.

A. Multi-Resident Activity Recognition in Smart Homes

Activity recognition in smart homes is an area that has received much research. However, for the sub-area of multi-resident activity recognition, as it is more challenging, there are fewer related works. In recent years, it obtains increasing attention. Benmansour et al. give a survey of existing works in [9]. In this area, some works are based on the wearable sensors. In [13], [14], wearable sensors are worn on the body of the subjects, and activity recognition is based on the sensor readings. In these works, as each sensor is known beforehand which subject it is worn by, the sensor readings about different subjects can be separated naturally.

In most of existing works, because of the advantages introduced in Section I, the ambient sensors are adopted. These ambient sensors are usually binary sensors [5], [15], the readings of which are the sensor events denoting activation information of the sensors. With the readings of the binary ambient sensors, some works perform activity recognition for each sensor event. In these works, when a sensor event is observed, they determine it belongs to which resident and which activity. Some of these works just focus on one aspect of the problem: in some works, they just focus on determining an event belongs to which resident (this is referred to as data association) [16], [17]; in other works, the data association is assumed to be known, and they just focus on activity recognition [3], [4]. In other works, they do not just focus on one aspect and solve the whole problem. In [2], a CRF is used to solve the problem of data association and activity recognition jointly in an iterative manner. In [18], the Cartesian product of activity labels of different residents are used to form the combined labels. In this way, they convert the multi-resident activity recognition problem into a single-label classification problem, and learn an HMM or a CRF to classify the combined labels. Besides the works performing activity recognition for each sensor event, there are works performing activity recognition based on fixed-length time slices. This is in accordance with many works in other sensor-based activity recognition problems [7]. In these works, for a time slice, they determine the activities of each resident within this time slice. In [19], incremental decision trees are used for activity recognition. In [20], an FHMM and a tracking based approach are used.

In area of multi-resident activity recognition, there are also works utilizing active learning techniques [21], and works based on both the wearable and the ambient sensors [22]. Almost all of the existing works are introduced in the situation of two residents, and are not evaluated in situations of more than two residents. However, most of the methods in these works can be extended to situations of more than two residents.

B. Zero-Shot Learning

In our problem, as we need to recognize the previously unseen activities, zero-shot learning methods are needed. Zero-shot learning is a learning paradigm which aims to recognize the previously unseen classes [11]. In zero-shot learning, the classes contained in the labeled training data are referred to as the seen classes; the classes we need to classify in the testing data are referred to as the unseen classes. The seen and the unseen classes are disjoint. To be able to recognize instances belonging to the unseen classes, some semantic information is involved and it forms a semantic space. Each class in the seen and the unseen classes is represented as a vector in the semantic space, and referred to as the class prototype (or prototype for short) of this class. In existing works, from different information sources, different semantic spaces can be involved. Common widely used semantic spaces include the semantic attribute spaces and the word embedding spaces. Zero-shot learning is a sub-area of transfer learning [23], in which the information contained in the seen classes (source classes) is transferred to the task of classifying instances belonging to the unseen classes (target classes).

Although existing works on zero-shot learning are mainly in the area of computer vision [11]. In other areas, there have also been some works. In the area of sensor-based activity recognition, there are some works on different zero-shot recognition tasks. In [10], the semantic attributes are involved for wearable sensor based activity recognition. In [12], the keywords extracted from the text descriptions of the activities form the semantic space and are used for ambient sensor based activity recognition.

The multi-resident activity recognition problem is a multi-label classification problem. In zero-shot learning, there are

some works on multi-label zero-shot learning. In some works [24], each label is treated separately, and the multi-label classification problem is converted into a series of binary classification problems. In other works [25], combinations of the labels in the seen and the unseen classes are regarded as the combined labels. The multi-label classification problem is converted into a single-label classification problem.

III. Proposed Method

In this section, we introduce our proposed MRUA method. Following most of existing works, we assume the ambient sensors are used as data collecting devices. For the convenience of illustration, like in existing works, we introduce our method in the situation of two residents. Our method can be easily extended to situations of more than two residents.

A. Problem Statement

In our problem, the activity classes covered by the training and the testing data are different. From the view of transfer learning [23], the aim of our method is to transfer the information contained in the training data to the task of testing data classification. Following the general way in transfer learning, we refer to the class label space of the training data as the source label space, and the class label space of the testing data as the target label space.

In the training phase, we have the sensor readings got from the ambient sensors deployed in a smart home environment of two residents. We refer to these two residents as Resident1 and Resident2. The activities of these two residents are labeled. These labels belong to the source label space $\mathcal{C}^s = \{c_i^s | i = 1, \cdots, N^s\}$. In the testing phase, we have the unlabeled sensor readings also got from these ambient sensors, and these sensor readings are also about these two residents. These sensor readings belong to the target label space $\mathcal{C}^t = \{c_i^t | i = 1, \cdots, N^t\}$. The source and the target label spaces are different, i.e., $\mathcal{C}^s \neq \mathcal{C}^t$. The target label space contains activity classes not in the source label space. With the labeled sensor readings in the training phase, we aim to learn a classification model that can classify the unlabeled sensor readings in the testing phase.

Our problem setting is a little different from the setting of zero-shot learning. In zero-shot learning, the source and the target label spaces are disjoint: the source label space just contains the seen classes; the target label space just contains the unseen classes. However, in our problem, the source label space contains the seen classes; the target label space contains not only the unseen classes but also the seen classes. This problem setting is more practical in activity recognition applications. Because in practical applications, instances of both the seen and the unseen activity classes can be encountered when the recognition model is used. This setting is referred to as generalized zero-shot learning [11] in recent works on zero-shot learning.

B. Data Preprocessing

Feature Extraction. The raw data we have are the sensor readings. They are time series data. Following the general way of dealing with the sensor readings [7], we adopt the sliding window strategy to segment the sensor readings into time slices with fixed length. The classification is based on these time slices. As introduced in Section II-A, in existing works on multi-resident activity recognition, there are also many works performing the classification based on the sensor events [2], [18]. However, in our method, like in the works [19], [20], we choose to perform classification based on the time slices. This is because, in practical applications, we often need to get aware of the residents' activities not necessarily at the time when there are sensor events occur, but also at the time when there are no sensor events. After data segmentation, for each time slice, one feature space instance is extracted.

Label Embedding. The testing data contain unseen activities. To be able to recognize the different activities in the testing data, following the general way adopted in zero-shot learning, we embed the class labels into the semantic space $\mathcal{T}$. For a class c_i in the source or the target label space, it is embedded into the semantic space, and has a corresponding prototype $\mathbf{t}_i$ in this space, i.e., $c_i \rightarrow \mathbf{t}_i$.

C. Proposed MRUA Method

After the above data preprocessing steps, in the training phase, we have N^{tr} labeled training instances $\{\mathbf{x}_i^{tr}, y_i^{tr(1)}, y_i^{tr(2)}\}_{i=1}^{N^{tr}}$. Each instance $\{\mathbf{x}_i^{tr}, y_i^{tr(1)}, y_i^{tr(2)}\}$ is in the feature space $\mathcal{X}$ (which is a real number space of dimension D), i.e., $\mathbf{x}_i^{tr} \in \mathcal{X}$; $y_i^{tr(1)} \in \mathcal{C}^s$ is the activity label of Resident1; $y_i^{tr(2)} \in \mathcal{C}^s$ is the activity label of Resident2. In the semantic space $\mathcal{T}$ (which is a real number space of dimension M), we have the prototypes $\{\mathbf{t}_i^s\}_{i=1}^{N^s}$ of the classes $\{c_i^s\}_{i=1}^{N^s}$ in the source label space $\mathcal{C}^s$. In the testing phase, we have N^{te} unlabeled testing instances $\{\mathbf{x}_i^{te}\}_{i=1}^{N^{te}}$ in the feature space $\mathcal{X}$. In the semantic space $\mathcal{T}$, we have the prototypes $\{\mathbf{t}_i^t\}_{i=1}^{N^t}$ of the classes $\{c_i^t\}_{i=1}^{N^t}$ in the target label space $\mathcal{C}^t$. For each testing instance $\mathbf{x}_i^{te}$, we need to predict the activity label $y_i^{te(1)}$ of Resident1 and the activity label $y_i^{te(2)}$ of Resident2, both of which belong to the target label space $\mathcal{C}^t$, i.e., $y_i^{te(1)} \in \mathcal{C}^t$ and $y_i^{te(2)} \in \mathcal{C}^t$.

In our problem, for each instance $\mathbf{x}_i$, there are two corresponding activity labels $y_i^{(1)}$ and $y_i^{(2)}$ for the two residents respectively. It is a multi-label classification problem [26]. However, our problem setting is a little different from the general multi-label classification problem settings. In the general multi-label classification problems, for an instance $\mathbf{x}_i$, the number of related labels is uncertain. The related labels can be any label in the label space. In our problem, for each instance $\mathbf{x}_i$, the number of labels is fixed: $y_i^{(1)}$ is the label of Resident1 and is one label from the label space; similarly, $y_i^{(2)}$ is the label of Resident2 and is also one label from the label space.

In view of this, we propose to address this problem from the view of multi-task learning. Multi-task learning [27] is a learning paradigm in which there are multiple learning tasks. By jointly learning these tasks and exploiting the relations among them, multi-task learning aims to improve the learning of all these tasks. Under our problem setting, the activity

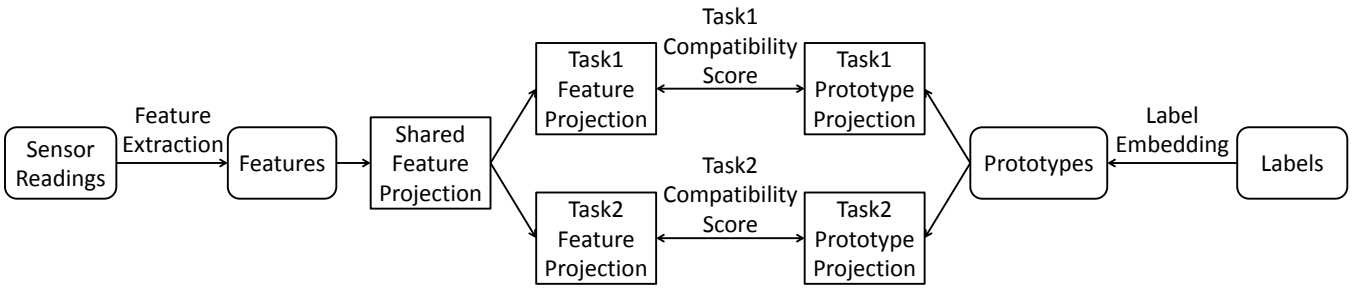

Fig. 1: Overall Process of the Proposed MRUA Method.

recognition of each resident can be regarded as a task. Our goal is the activity recognition for both of the two residents. In the area of sensor-based activity recognition, there have been works solving the multi-label classification problems through multi-task learning. Such as in [28], the problem setting is single-person activity recognition, in which the person can perform more than one activities at the same time. The recognition of each activity is regarded as a task, and all tasks are learned jointly. However, to the best of our knowledge, we have not seen works solving the multi-resident activity recognition problems from the view of multi-task learning, especially under the setting with unseen classes.

To be able to recognize the classes in the target label space, we use the zero-shot learning techniques. In many works on zero-shot learning [11], a compatibility function F between and feature space instances and the semantic space prototypes is learned. This function takes an instance $\mathbf{x}_i$ and a prototype $\mathbf{t}_j$ as input, and outputs a score denoting the degree of compatibility, i.e., $F(\mathbf{x}_i, \mathbf{t}_j) : \mathcal{X} \times \mathcal{T} \rightarrow \mathbb{R}$. For a testing instance $\mathbf{x}_i^{te}$, it is classified to the class in the target label space $\mathcal{C}^t$ with the highest compatibility score.

The overall process of our MRUA method is shown in Fig. 1. In our method, activity recognition of each resident is regarded as a task, and the recognition is achieved through the compatibility function. For task $r(r = 1, 2)$, the compatibility function is in form of

$$F_r(\mathbf{x}_i, \mathbf{t}_j) = \delta_r(\mathbf{x}_i)^T \varphi_r(\mathbf{t}_j). \tag{1}$$

In this function, the instance $\mathbf{x}_i$ and the prototype $\mathbf{t}_j$ are projected into a hidden space[1] $\mathcal{H}$ through the projection functions $\delta_r(\cdot)$ and $\varphi_r(\cdot)$ respectively, and their dot product is used as the compatibility score. The projection function $\delta_r(\cdot)$ for the instances is in form of

$$\delta_r(\mathbf{x}_i) = \xi(W_r^{tf} \phi(W^{sf} \mathbf{x}_i + \mathbf{b}^{sf}) + \mathbf{b}_r^{tf}), \tag{2}$$

which consists of functions $\phi(\cdot)$ and $\xi(\cdot)$ for two projections. The first projection is shared by the two recognition tasks. We refer to this projection as the shared feature projection. $\phi(\cdot)$ is the nonlinear function of this projection. Assume the output of this function is of dimension K. In this function, $W^{sf} \in \mathbb{R}^{K \times D}$ and $\mathbf{b}^{sf} \in \mathbb{R}^K$ are the parameters, and shared by the two tasks. After that is the second projection. For this projection, each task has its own projection function. For task r, we refer to the corresponding projection as task r feature projection. Assume the hidden space $\mathcal{H}$ is of dimension H, in the nonlinear function $\xi(\cdot)$ of this projection, $W_r^{tf} \in \mathbb{R}^{H \times K}$ and $\mathbf{b}_r^{tf} \in \mathbb{R}^H$ are the parameters, and specific for the recognition task r.

The projection of the prototypes is specific for each task. For task r, we refer to this projection as task r prototype projection. The projection function $\varphi_r(\cdot)$ is in form of

$$\varphi_r(\mathbf{t}_j) = \theta(W_r^{tp} \mathbf{t}_j + \mathbf{b}_r^{tp}), \tag{3}$$

in the nonlinear function $\theta(\cdot)$ of this projection, $W_r^{tp} \in \mathbb{R}^{H \times M}$ and $\mathbf{b}_r^{tp} \in \mathbb{R}^H$ are the parameters, and specific for the recognition task r.

In our method, the model parts of task1 and task2 are learned jointly. To learn the whole model of the two residents, with the training instances and the source label space class prototypes, we optimize the objective function in form of

$$\min \frac{1}{R} \sum_{r=1}^{R} \left(\frac{1}{N^{tr}} \frac{1}{N^s} \sum_{i=1}^{N^{tr}} \sum_{j=1}^{N^s} \ell_r(\mathbf{x}_i^{tr}, \mathbf{t}_j^s) \right), \tag{4}$$

where $R = 2$ is the number of tasks, ℓ_r is the loss of classifying instance $\mathbf{x}_i^{tr}$ to class c_j^s (the class of $\mathbf{t}_j^s$) for recognition task r. In our method, we use the hinge loss as the loss function ℓ_r, which is in form of

$$\ell_r(\mathbf{x}_i^{tr}, \mathbf{t}_j^s) = \max(0, 1 - \mathbb{I}_{rij} F_r(\mathbf{x}_i^{tr}, \mathbf{t}_j^s)), \tag{5}$$

where $\mathbb{I}_{rij}$ is an indicator function. If for task r, the class of $\mathbf{x}_i^{tr}$ (i.e., $y_i^{tr(r)}$) and the class of $\mathbf{t}_j^s$ (i.e., c_j^s) are the same, it evaluates to 1; otherwise it evaluates to -1. To learn this model, we use the stochastic gradient descent based method, and perform the optimization process to fixed epochs.

In our method, the shared feature projection for the instances is the shared part of the two tasks. For each task r, the task r feature projection and the task r prototype projection are the parts specific for this task. This model structure is a widely used structure in multi-task learning. In this model,

[1]In our method, to avoid the model to be too complex, we use the same hidden space for the two tasks.

the shared information of the two tasks is captured by the shared part, and the task specific information is captured by the task specific parts. From the above illustration we can see, our method can be easily extended to the situations of more than two residents. By adding more recognition tasks to the model, the model can be extended.

IV. Experiments

A. Datasets

In the experiment, we use the publicly available ARAS datasets[2] [15]. We choose these datasets because they are collected in the real world settings, and they are widely used in existing works [20], [21] on multi-resident activity recognition. ARAS consist of two datasets collected from two houses: HouseA and HouseB. In HouseA, 20 binary ambient sensors are deployed in the house, collecting data of 30 days. Two male residents live in the house. The sensor readings and the activity labels of the two residents are recorded per second. There are 27 different activities labeled. In HouseB, also 20 binary ambient sensors (but not the same sensors as in HouseA) are deployed in the house. The duration of data collection is also 30 days. The residents in this house are a married couple. The sensor readings and the activity labels of the two residents are also recorded per second. The same 27 activities as in the HouseA dataset are labeled.

B. Experimental Setup

1) Preprocessing: We preprocess the raw data in the HouseA and the HouseB datasets following the way adopted in [20]. Adopting the sliding window strategy, we segment the sensor readings into fixed-length time slices. The length of the sliding window is set to be 60 seconds. There are no overlaps between adjacent time slices. For each time slice, an instance is extracted from the sensor readings. In each dataset, as there are 20 sensors, the extracted instance is a vector of dimension 20, with each dimension corresponding to a sensor. If the sensor is activated at least once during this time slice, the corresponding value is set to be 1, otherwise is set to be 0. The label for each resident is set to be the label with the longest duration in this time slice for each of them.

In these two datasets, there are 27 labeled activity classes. Following the approaches adopted in [15], [29], we group similar activities into more general activities, and form 10 activities. In this way to let the activities we dealt with to be more representative. When deciding these 10 activity classes, we refer to the way of grouping activities used in [15] and [29]. In addition, we refer to the taxonomy of activities listed in [6]. The 10 activities are: other, going out, preparing a meal (including preparing breakfast, preparing lunch, and preparing dinner), eating (including having breakfast, having lunch, having dinner, and having snack), doing cleaning (including washing dishes, laundry, and cleaning), sleeping/napping, relaxing (including watching TV, using Internet, reading book, and listening to music), studying, personal hygiene (including having shower, toileting, shaving, brushing teeth, and changing clothes), and social (including talking on the phone, having conversation, and having guest).

TABLE I: Number of Days Each Activity Occurs*

Activity	A1	A2	B1	B2	Total
other	30	24	30	25	109
going out	27	30	25	28	110
preparing a meal	30	14	23	1	68
eating	30	26	25	20	101
doing cleaning	30	21	13	0	64
sleeping/napping	30	27	28	25	110
relaxing	30	27	27	26	110
studying	18	2	19	2	41
personal hygiene	30	27	30	27	114
social	30	24	11	10	75

*In this table and in the following, we use A1 stands for the Resident1 in HouseA, A2 stands for the Resident2 in HouseA, B1 stands for the Resident1 in HouseB, and B2 stands for the Resident2 in HouseB.

For the semantic space, we use the word embedding space [30]. This semantic space is based on the widely adopted word embedding technique in the area of natural language processing. In the word embedding space, each word or phrase has a corresponding vector. The vector contains semantic information. It is an efficient way to get the semantic information about the seen and the unseen classes. In many works on zero-shot learning [11], word embedding space is used as the semantic space. In our experiment, the word embedding space we used is got by running the skip-gram model [30] on the English Wikipedia dump corpus[3] on December 1, 2017. We use gensim [31] to perform this. The word embedding space we got is of dimension 100. For activity labels with more than one words, we use the average of the word embedding vector of each constituent word.

2) Evaluation Metrics: In smart home activity recognition problems, the distribution of classes is usually imbalanced [3], [7], [20]. During activity recognition, we are not only interested in activities with large number of instances, but also activities with small number of instances. In view of this, we adopt the metrics averaged over different classes: $Precision = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i+FP_i}$, $Recall = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i+FN_i}$, and their harmonic mean $F1-score = \frac{2\times Precision\times Recall}{Precision+Recall}$ as the evaluation metrics, where N is the number of classes during training or testing, TP_i, FP_i and FN_i are the number of true positive, false positive and false negative instances for class i respectively. These metrics are not influenced by the distribution of classes.

C. Effectiveness Evaluation

1) Evaluation Design: In each dataset, there are instances from 30 days. We use the instances from Day1 to Day24 as the training instances, and the instances from Day25 to Day30 as the testing instances. Then we decide the seen and the unseen activity classes. We perform this based on the

[2]https://www.cmpe.boun.edu.tr/aras/

[3]https://dumps.wikimedia.org/

occurrence frequencies of the activities. Table I is the statistics of the number of days in which each activity occurs. From this table we can see, the total number of occurrence days of the activities "other", "going out", "eating", "sleeping/napping", "relaxing" and "personal hygiene" is much larger than that of the other four activities (preparing a meal, doing cleaning, studying and social). This is in accordance with our common sense. These six activities are the activities that we almost need to perform every day, and they are more likely to be the activities of interest during training data labeling. We select these six as the seen activities. For training instances belonging to the other four activities, we relabel them as belonging to the "other" activity. The labels of the testing instances retain unchanged.

We do this evaluation design with the aim of to be in accordance with the practical applications. In practical applications, we often collect and label sensor readings of some days, and learn a classification model with them. In the following days, we use this model to classify the observed sensor readings. During data labeling in the training phase, just the predefined activities that are of interest are labeled, and all other readings are labeled as "other". When we use this classification model, some of the activities we need to recognize may not be within the predefined activities and we also need to recognize them.

With this evaluation design, for each dataset, we have 34,560 training instances and 8,640 testing instances. In the training instances, 6 activity classes are labeled; in the testing instances, 10 activity classes are needed to classify.

2) Implementation Details: The nonlinear functions $\phi(\cdot)$, $\xi(\cdot)$ and $\theta(\cdot)$ are implemented by the hyperbolic tangent function. Each prototype in the semantic space is ℓ_2 normalized. Adam [32] is adopted as the optimization method. During model learning, we set the batch size to be 200, and the number of training epochs to be 300. The learning rate η is set to be 0.001. The dimension K of the output of the shared feature projection and the dimension H of the hidden space are determined through five-fold cross validation. The method is implemented using Theano [33]. We learn the classification model for each dataset separately.

3) Comparison Methods: Existing works on multi-resident activity recognition are not under the setting in which there are unseen classes. In addition, existing works on multi-label zero-shot learning are not for sensor-based activity recognition and existing works on sensor-based zero-shot activity recognition are for single-person setting. Thus, we do not have explicit existing methods to compare with. However, there exist some general approaches in multi-label classification [26], and they have been implemented in existing works on multi-resident activity recognition or multi-label zero-shot learning. We implement these approaches under our problem setting, and get the following methods. We compare our method with them. To let the comparison be more persuasive, we use the implementations similar to our method.

Respective model for each resident (RMER). In this method, the recognition models are learned for each resident separately. When learning the recognition model for one resident, information about another resident in the training instances is regarded as noise [20]. This kind of method is widely used in multi-label classification problems [26], and is referred to as the overlaid method in [20]. Under our problem setting, we implement the RMER method using the same implementation of our method, with the difference that when learn the model for each resident, just the related one task is involved.

TABLE II: Comparison of Different Methods (in %)*

Method	House and Resident	Precision	Recall	F1-score
RMER	A1	**39.57**	34.91	37.09
	A2	34.83	**32.47**	33.61
	avg A1 & A2	37.20	33.69	35.35
	B1	**46.29**	**45.64**	**45.96**
	B2	57.58	48.07	52.40
	avg B1 & B2	51.94	46.85	49.18
CPAvg	A1	30.44	22.11	25.61
	A2	27.07	13.16	17.71
	avg A1 & A2	28.76	17.63	21.66
	B1	36.18	33.20	34.62
	B2	54.30	48.92	51.47
	avg B1 & B2	45.24	41.06	43.05
CPCon	A1	32.08	30.49	31.27
	A2	26.10	22.56	24.20
	avg A1 & A2	29.09	26.52	27.73
	B1	44.54	44.41	44.47
	B2	66.90	54.71	60.19
	avg B1 & B2	55.72	49.56	52.33
MRUA	A1	39.36	**36.57**	**37.91**
	A2	**44.55**	31.15	**36.67**
	avg A1 & A2	**41.96**	**33.86**	**37.29**
	B1	45.50	44.62	45.06
	B2	**67.54**	**56.80**	**61.70**
	avg B1 & B2	**56.52**	**50.71**	**53.38**
MRUA with all class labels	A1	52.49	48.20	50.25
	A2	37.91	34.31	36.02
	avg A1 & A2	45.20	41.25	43.14
	B1	61.96	63.88	62.91
	B2	69.40	64.27	66.73
	avg B1 & B2	65.68	64.08	64.82

*In this table, avg A1 & A2 stands for the average results of A1 and A2, avg B1 & B2 stands for the average results of B1 and B2. The best evaluation results among the four methods are shown in bold.

Cartesian product and average prototype method (CPAvg). In this method, the Cartesian product of the labels of the two residents are used as the new combined class labels. It converts the multi-label classification problem into a single-label classification problem. This approach is also widely used in multi-label classification problems. For the prototype of each combined class, we use the average of the prototypes of the two corresponding activity classes. We implement the CPAvg method using the same implementation of our method, with the difference that just one task is involved for the classification of the combined classes.

Cartesian product and concatenation prototype method (CPCon). This method is similar to the CPAvg method. The difference is that, for each combined class, the prototype is the concatenation of the prototypes of the two corresponding activity classes, instead of the average.

4) Results and Analysis: The first four parts of Table II are the results of these methods. From the results we can see,

in general, compared with other three methods, our MRUA method achieves the best results. Even though a few evaluation results of our MRUA method is not the best, they are very close to the best results. Other methods cannot achieve this performance on all of the evaluation results. The results of all these four methods are significantly better than the chance level results of 10%. This illustrates the adopted zero-shot learning technique is effective under this problem setting, and the word embedding space is an effective semantic space.

In HouseA, the recognition results of A1 are better than that of A2 in all of these four methods. We consider this is because, in HouseA, A1's duration of in the house is much longer than that of A2. In the training dataset, for A1, the number of instances belonging to the activity "going out" (not in the house) accounts 12.06% of the whole instances; while for A2, this proportion is 48.47%. Thus, in HouseA, more sensor readings are about A1 and effective for activity recognition of A1. In HouseB, in all of these four methods, the recognition results of B2 are better than that of B1. In the training dataset, although the going out time of B2 (accounts 55.28% of the whole instances) is longer than that of B1 (accounts 37.95%), the number of activities contained in the testing instances of B2 is only seven, with six among which are the seen classes. We consider this is the reason that the results of B2 are better than that of B1.

Among these four methods, the results of the CPAvg method are the lowest. We consider this is because, in this method, when forming the prototypes of the combined classes, the average of the prototypes of the two corresponding activity classes are used. In this way, information contained in these two prototypes has lost to some extent. This results in the poor performance. On the contrast, in the CPCon method, the prototypes of the combined classes are formed by concatenation, and the information loss is less. Thus, the results are much better. The results of MRUA and RMER for A1 and B1 are comparable. However, for A2 and B2, the results of MRUA are generally better than that of RMER. This shows that, by utilizing the multi-task learning strategy, our MRUA method is effective in improving the recognition results of residents with relatively less effective sensor readings.

To further analyze our method, we also compare it with the case in which in the training datasets, all of the ten activity classes are labeled. The results are shown at the bottom of Table II. From the results we can see, with more labeled activity classes, all of the recognition results improve a lot. Except the precision of A2. We consider this decrease is mainly because the duration of in the house for A2 is much shorter than that of A1. Thus, even more label information does not necessarily bring increase of the recognition results. The results show that, compared with the cases in which there are no unseen classes during testing, there exists room for improvement for the cases with the unseen classes.

D. Further Analysis

1) Results on Each Testing Day: The testing results on each testing day are shown in Fig. 2. From the figure we can see,

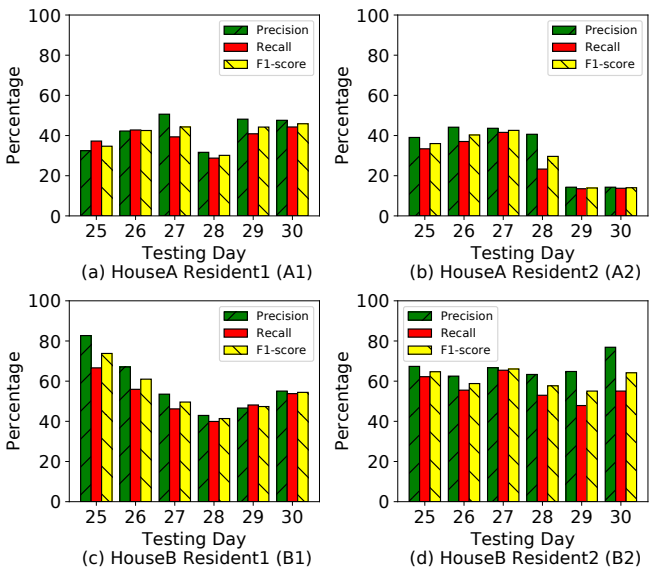

Fig. 2: Results on Each Testing Day.

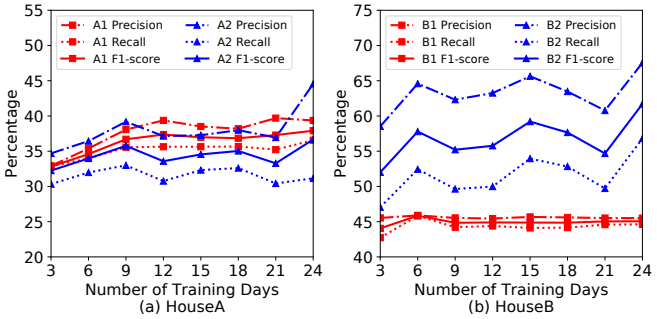

Fig. 3: Effect of the Amount of Training Data.

for each house and each resident, the recognition results on different days vary. These results are in accordance with the results in [20]. It shows that, the activities of the residents are complex. In different days, the number of performed activities is different, and the way of performing activities can also be different. This results in the differences on the recognition results on different days. For A2, the recognition results on Day29 and Day30 are extremely low. This is because, in these two days, there is only one activity "going out" for this resident. Classifying some instances to other activity classes easily causes the decrease of the performance.

2) Effect of the Amount of Training Data: We evaluate the effect of the amount of training data by varying the number of days from which to get the training instances. When we use the training instances of n days, we randomly select n days from Day1 to Day24 to get the training instances. We repeat this ten times and report the average results. The results are shown in Fig. 3. From this figure we can see, the results of A1 and B1 are relatively stable. When the amount of training data increasing to nine days for A1 and six days for B1, the results do not change much with the increase of the training data.

These results are in accordance with the results in other smart home activity recognition problems [7]. However, for A2 and B2, the results are not so stable. We consider this difference is because, as stated in Section IV-C4, in these two datasets, the duration of in the house of A2 and B2 is shorter than that of A1 and B1. Thus, with the increase of training data, the effective amount of training instances for A2 and B2 are not necessarily increased as that of A1 and B1.

V. Conclusion

In this paper, we propose a method to solve the problem of multi-resident activity recognition with unseen activity classes in smart homes. It is a challenging problem and can often be encountered in practical applications. Our method utilizes the multi-task learning and zero-shot learning techniques, and addresses the recognition tasks of different residents jointly. Through extensive experiments, we evaluate the effectiveness of our method. As future work, we plan to utilize more sequential information contained in the data to facilitate the activity recognition. In addition, we plan to explore methods for more complex situations such as situations in which there are concurrent activities for each resident.

Acknowledgment

This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its IDM Futures Funding Initiative; and the Interdisciplinary Graduate School, Nanyang Technological University, Singapore. It is also partially supported by the NTU-PKU Joint Research Institute, a collaboration between Nanyang Technological University and Peking University that is sponsored by a donation from the Ng Teng Fong Charitable Foundation.

References

[1] M. R. Alam, M. B. I. Reaz, and M. A. M. Ali, "A review of smart homes - past, present, and future," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1190–1203, 2012.

[2] K.-C. Hsu, Y.-T. Chiang, G.-Y. Lin, C.-H. Lu, J. Y.-J. Hsu, and L.-C. Fu, "Strategies for inference mechanism of conditional random fields for multiple-resident activity recognition in a smart home," in *IEA/AIE*. Springer, 2010, pp. 417–426.

[3] A. Benmansour, A. Bouchachia, and M. Feham, "Modeling interaction in multi-resident activities," *Neurocomputing*, vol. 230, pp. 133–142, 2017.

[4] Y.-T. Chiang, K.-C. Hsu, C.-H. Lu, L.-C. Fu, and J. Y.-J. Hsu, "Interaction models for multiple-resident activity recognition in a smart home," in *IROS*. IEEE, 2010, pp. 3753–3758.

[5] G. Singla, D. J. Cook, and M. Schmitter-Edgecombe, "Recognizing independent and joint activities among multiple residents in smart environments," *Journal of ambient intelligence and humanized computing*, vol. 1, no. 1, pp. 57–63, 2010.

[6] D. J. Cook and N. C. Krishnan, *Activity learning: discovering, recognizing, and predicting human behavior from sensor data*. John Wiley & Sons, 2015.

[7] T. van Kasteren, A. Noulas, G. Englebienne, and B. Kröse, "Accurate activity recognition in a home setting," in *UbiComp*. ACM, 2008, pp. 1–9.

[8] E. M. Tapia, S. S. Intille, and K. Larson, "Activity recognition in the home using simple and ubiquitous sensors," in *Pervasive*. Springer, 2004, pp. 158–175.

[9] A. Benmansour, A. Bouchachia, and M. Feham, "Multioccupant activity recognition in pervasive smart home environments," *ACM Computing Surveys (CSUR)*, vol. 48, no. 3, p. 34, 2016.

[10] H.-T. Cheng, F.-T. Sun, M. Griss, P. Davis, J. Li, and D. You, "Nuactiv: recognizing unseen new activities using semantic attribute-based learning," in *MobiSys*. ACM, 2013, pp. 361–374.

[11] Y. Xian, B. Schiele, and Z. Akata, "Zero-shot learning - the good, the bad and the ugly," in *CVPR*, 2017, pp. 4582–4591.

[12] V. W. Zheng, D. H. Hu, and Q. Yang, "Cross-domain activity recognition," in *UbiComp*. ACM, 2009, pp. 61–70.

[13] T. Gu, Z. Wu, L. Wang, X. Tao, and J. Lu, "Mining emerging patterns for recognizing activities of multiple users in pervasive computing," in *MobiQuitous*. IEEE, 2009, pp. 1–10.

[14] L. Wang, T. Gu, X. Tao, and J. Lu, "Sensor-based human activity recognition in a multi-user scenario," in *AmI*. Springer, 2009, pp. 78–87.

[15] H. Alemdar, H. Ertan, O. D. Incel, and C. Ersoy, "Aras human activity datasets in multiple homes with multiple residents," in *7th International Conference on Pervasive Computing Technologies for Healthcare and Workshops*, 2013, pp. 232–235.

[16] A. S. Crandall and D. J. Cook, "Coping with multiple residents in a smart environment," *Journal of Ambient Intelligence and Smart Environments*, vol. 1, no. 4, pp. 323–334, 2009.

[17] ——, "Using a hidden markov model for resident identification," in *International Conference on Intelligent Environments*. IEEE, 2010, pp. 74–79.

[18] R. Chen and Y. Tong, "A two-stage method for solving multi-resident activity recognition in smart environments," *Entropy*, vol. 16, no. 4, pp. 2184–2203, 2014.

[19] M. Prossegger and A. Bouchachia, "Multi-resident activity recognition using incremental decision trees," in *ICAIS*. Springer, 2014, pp. 182–191.

[20] H. Alemdar and C. Ersoy, "Multi-resident activity tracking and recognition in smart environments," *Journal of Ambient Intelligence and Humanized Computing*, vol. 8, no. 4, pp. 513–529, 2017.

[21] I. A. Emi and J. A. Stankovic, "Sarrima: smart ADL recognizer and resident identifier in multi-resident accommodations," in *Proceedings of the conference on Wireless Health*. ACM, 2015, pp. 4:1–4:8.

[22] N. Roy, A. Misra, and D. Cook, "Ambient and smartphone sensor assisted ADL recognition in multi-inhabitant smart environments," *Journal of ambient intelligence and humanized computing*, vol. 7, no. 1, pp. 1–19, 2016.

[23] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[24] T. Mensink, E. Gavves, and C. G. Snoek, "Costa: co-occurrence statistics for zero-shot classification," in *CVPR*. IEEE, 2014, pp. 2441–2448.

[25] Y. Fu, Y. Yang, T. Hospedales, T. Xiang, and S. Gong, "Transductive multi-label zero-shot learning," in *BMVC*, 2014.

[26] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE transactions on knowledge and data engineering*, vol. 26, no. 8, pp. 1819–1837, 2014.

[27] R. Caruana, "Multitask learning," *Machine Learning*, vol. 28, no. 1, pp. 41–75, 1997.

[28] Y. Vaizman, N. Weibel, and G. Lanckriet, "Context recognition in-the-wild: unified model for multi-modal sensors and multi-label classification," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, no. 4, pp. 168:1–168:22, 2018.

[29] C. Tunca, H. Alemdar, H. Ertan, O. D. Incel, and C. Ersoy, "Multimodal wireless sensor network-based ambient assisted living in real homes with multiple residents," *Sensors*, vol. 14, no. 6, pp. 9692–9719, 2014.

[30] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *ICLR Workshops*, 2013.

[31] R. Řehůřek and P. Sojka, "Software framework for topic modelling with large corpora," in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, 2010, pp. 45–50.

[32] D. P. Kingma and J. L. Ba, "Adam: a method for stochastic optimization," in *ICLR*, 2015.

[33] Theano Development Team, "Theano: a Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, 2016.