Stratified Transfer Learning for Cross-domain Activity Recognition

Jindong Wang*†, Yiqiang Chen*†✉, Lisha Hu‡, Xiaohui Peng*†, Philip S. Yu§
*Beijing Key Laboratory of Mobile Computing and Pervasive Device
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
†University of Chinese Academy of Sciences, Beijing, China
‡Institute of Information Technology, Hebei University of Economics and Business, Shijiazhuang, China
§Department of Computer Science, University of Illinois at Chicago, IL, USA
Email:{wangjindong,yqchen,pengxiaohui}@ict.ac.cn, hulisha@heuet.edu.cn, psyu@uic.edu

***Abstract*— In activity recognition, it is often expensive and time-consuming to acquire sufficient activity labels. To solve this problem, transfer learning leverages the labeled samples from the source domain to annotate the target domain which has few or none labels. Existing approaches typically consider learning a global domain shift while ignoring the intra-affinity between classes, which will hinder the performance of the algorithms. In this paper, we propose a novel and general cross-domain learning framework that can exploit the intra-affinity of classes to perform intra-class knowledge transfer. The proposed framework, referred to as Stratified Transfer Learning (STL), can dramatically improve the classification accuracy for cross-domain activity recognition. Specifically, STL first obtains pseudo labels for the target domain via majority voting technique. Then, it performs intra-class knowledge transfer iteratively to transform both domains into the same subspaces. Finally, the labels of target domain are obtained via the second annotation. To evaluate the performance of STL, we conduct comprehensive experiments on three large public activity recognition datasets (i.e. OPPORTUNITY, PAMAP2, and UCI DSADS), which demonstrates that STL significantly outperforms other state-of-the-art methods w.r.t. classification accuracy (improvement of 7.68%). Furthermore, we extensively investigate the performance of STL across different degrees of similarities and activity levels between domains. And we also discuss the potential of STL in other pervasive computing applications to provide empirical experience for future research.**

I. Introduction

Human activity recognition (HAR) is a hot research topic in pervasive computing. HAR has been widely applied to many applications such as indoor localization [1], sleep state detection [2], and smart home sensing [3]. The key to successful HAR is to build accurate and robust models using sufficient labeled activity data. Unfortunately, it is often expensive and time-consuming to obtain enough labeled data. In this case, it is necessary to perform *cross-domain activity recognition* (CDAR) by exploiting the labeled activity data from one auxiliary domain to annotate the unlabeled activities in another domain.

An intuitive example of CDAR is given in Fig. 1, which shows two accelerometer readings on the chest (C) and the leg (L) from one person. Such scenarios happen very often: If we only have the activity data and labels on C but labels on L are missing or wrongly labeled, we can certainly use the data from C to recognize the activity on L. Because the signals of C and L can be related according to the body structure. Then the challenge arises: It can be seen that the two sensor readings are under two totally different distributions from Fig. 1. Therefore, it is challenging to design algorithms to tackle CDAR problem.

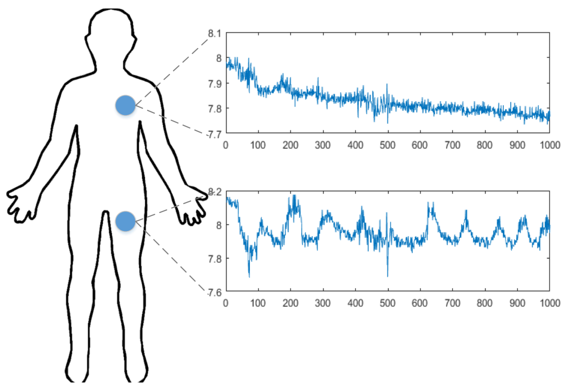


Fig. 1. Two accelerometer readings of the same device when a person is walking. It is obvious that those two readings follow different distributions.

Several dimensionality reduction methods have been proposed to resolve CDAR, such as principal component analysis (PCA), locally linear embedding (LLE), and kernel principal component analysis (KPCA) [4]. Dimensionality reduction does not require domain knowledge, but it ignores the divergence between domains. To this end, transfer learning [5] has made considerable progress for cross-domain tasks by leveraging the labeled samples from other auxiliary domains. The key to transfer learning is to utilize the *similarity* between those different but related domains. Existing transfer learning approaches disentangled the cross-domain learning problems either by exploiting the correlations between features [6], [7], or transforming both the source and target domains into a new shared feature space [8], [9], [10].

Those approaches tend to learn a global domain shift by projecting all samples in both domains into a single subspace. However, they fail to consider the intra-affinity within classes [11]. According to Fig. I, learning the global domain shift (Fig. 2(b)) can only learn a general hyperplane between classes, and this hyperplane is loose; while by exploiting intra-class affinity (Fig. 2(c)), classes can further be clustered more

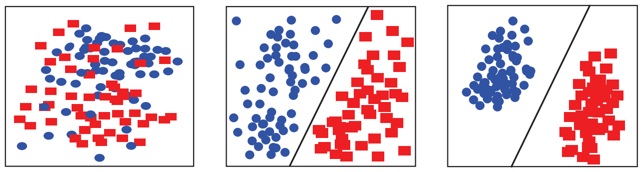

(a) Original target (b) Learn global shift (c) Exploit intra-affinity

Fig. 2. Illustration of learning global domain shift (existing approaches) and exploiting intra-affinity (STL) for the same original target domain.

tightly. Therefore, it is necessary to exploit the intra-affinity of classes to overcome the limitation of global domain shift.

According to [12], the data samples from the same class should lay on the same subspace, even if they belong to different domains. By using this assumption, in this paper, we propose a novel and general Stratified Transfer Learning (STL) framework. STL can simultaneously transform the same classes of the source and target domains into the same subspaces. Compared to existing methods, STL exploits the intra-affinity of classes to perform intra-class knowledge transfer. Thus, the global domain shift can be alleviated. Concretely speaking, STL first obtains pseudo labels for the target domain via a majority voting technique. Then, it performs intra-class knowledge transfer to transform the source and target domain into the same subspaces. Finally, the labels of target domain are obtained through a second annotation step. Comprehensive experiments on three large public activity recognition datasets (i.e. OPPORTUNITY, PAMAP2, and UCI DSADS) demonstrate that STL outperforms other five state-of-the-art methods with a significant improvement of **7.68%** w.r.t. classification accuracy.

Contributions. Our contributions are mainly three-fold:

1) We propose a novel and general cross-domain learning framework STL. Different from existing methods that only obtain a global domain shift, STL is capable of exploiting the intra-affinity of classes to perform intra-class knowledge transfer. STL is a general framework and can be tailored according to specific applications.

2) We conduct comprehensive experiments for cross-domain activity recognition on three public datasets (i.e. OPPORTUNITY, PAMAP2, and UCI DSADS) to evaluate STL. Experiments demonstrate the superiority of STL over other state-of-the-art methods w.r.t. classification accuracy.

3) We extensively investigate the performance of STL on different degrees of task similarity and different levels of activities. And we additionally explore the potential of STL in other pervasive computing applications, providing experience for future research.

The rest of this paper is organized as follows. In Section II, we review the related work. Section III introduces the proposed STL framework. In Section IV we present experimental evaluation and analysis of STL and other comparison methods. In Section V, we discuss the potential of STL in other real applications. Finally, the conclusion and future work are presented in Section VI.

II. Related Work

Our work is mainly related to activity recognition and transfer learning. In this section, we will discuss those two areas and their intersections.

A. Activity Recognition

Human Activity recognition has been a popular research topic in pervasive computing [13] for its competence in learning profound high-level knowledge about human activity from raw sensor inputs. Several survey articles have elaborated the recent advance of activity recognition using conventional machine learning [13], [14] and deep learning [15] approaches.

Conventional machine learning approaches have made tremendous progress on HAR by adopting machine learning algorithms such as similarity-based approach [16], [17], active learning [18], crowdsourcing [19], and other semi-supervised methods [20], [21]. Those methods typically treat HAR as a standard time series classification problem. And they tend to solve it by subsequently performing preprocessing procedures, feature extraction, model building, and activity inference. However, they all assume that the training and test data are with the same distribution. As for CDAR where the training (source) and the test (target) data are from different feature distributions, those conventional methods are prune to underfitting since their generalization ability will be undermined [5].

Deep learning based HAR [15] achieves the state-of-the-art performance than conventional machine learning approaches. The reason is that deep learning is capable of automatically extracting high-level features from the raw sensor readings [22]. Therefore, the features are likely to be more domain-invariant and tend to perform better for cross-domain tasks. A recent work evaluated deep models for cross-domain HAR [23], which provides some experience for future research on this area. There are stll many open problems for deep learning based CDAR. In this paper, we mainly focus on the traditional approaches.

B. Transfer Learning

Transfer learning has been successfully applied in many applications such as Wi-Fi localization [24], natural language processing [6], and visual object recognition [25]. According to the literature survey [5], transfer learning can be categorized into 3 types: instance-based, parameter-based, and feature-based methods.

Instance-based methods perform knowledge transfer mainly through instance re-weighting techniques [26], [27]. Parameter-based methods [28], [29] first train a model using the labeled source domain, then perform clustering on the target domain.

Our framework belongs to the feature based category, which brings the features of source and target domain into the same subspace where the data distributions can be the same. A fruitful line of work has been done in this area [30], [31], [10]. The proposed STL differs from existing feature-based methods in the following aspects:

Exploit the correlations between features. [6] proposed structural correspondence learning (SCL) to generatively learn the relation of features. [7] applied a feature-level transfer model to learn the dependence between domains, then trained a domain-adapted classifier. Instead of modeling the relationship of domain features, STL transforms the domain data into a new subspace, which does not depend on the domain knowledge in modeling features.

Transform domains into new feature space. [24] proposed maximum mean discrepancy embedding (MMDE) to learn latent features in the reproducing kernel Hilbert space (RKHS). MMDE requires solving a semidefinite programming (SDP) problem, which is computationally prohibitive. [8] extended MMDE by Transfer Component Analysis (TCA), which learns a kernel in RKHS. [32] adopted a similar idea. [33] learns target predictive function with a low variance. [34] sampled the domain features by viewing the data in a Grassmann manifold to obtain subspaces. [9] exploited the low-dimensional structure to integrate the domains according to geodesic flow kernel (GFK). Long *et al.* proposed joint distribution adaptation (JDA) based on minimizing joint distribution between domains, while STL focuses on the marginal distribution. [10] proposed transfer kernel learning (TKL), which learned a domain-invariant kernel in RKHS. [35] studied the conditional transfer components between domains. Methods in those literature tend to learn some common representations in the new feature space, then a global domain shift can be achieved. However, the difference between individual classes is ignored.

C. Transfer Learning based Activity Recognition

Some existing work also focused on transfer learning based HAR. A detailed survey is provided in [36]. Among existing work, Zhao *et al.* proposed a transfer learning method called TransEMDT [29] using decision trees, but it ignored the intra-class similarity within classes. [37] proposed the TransAct framework, which is a boosting-based method and ignores the feature transformation procedure. Thus it is not feasible in most activity cases. Feuz *et al.* [38] proposed a heterogeneous transfer learning method for HAR, but it only learns a global domain shift.

To circumvent the global domain shift, [12] clustered the data points in a low-dimensional space using sparse subspace clustering. [11] identified a compact joint subspace for each class, then measured the distance between classes using principal angle. The main difference between STL and [11] is that STL learns pseudo labels using majority voting technique on both domains, instead of performing metric clustering on the target domain alone. Thus, STL can obtain more reliable candidates by taking advantage of both domains.

III. Stratified Transfer Learning

In this section, we introduce the proposed Stratified Transfer Learning (STL) framework. First, we present the problem definition of CDAR and the main idea of STL. Then, each step of STL is presented in detail.

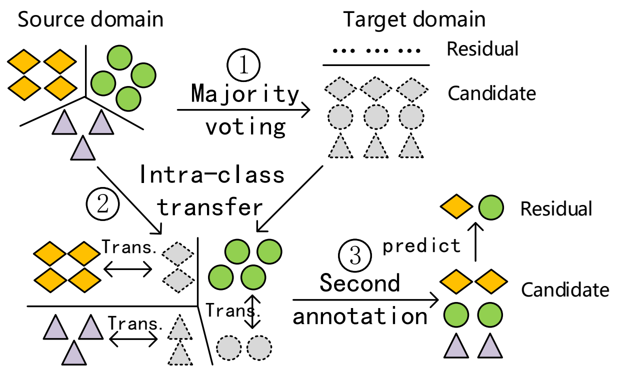


Fig. 3. Main idea of stratified transfer learning (STL) framework. There are mainly three steps: (1) Candidates generating through majority voting technique; (2) Perform intra-class transfer between source domain and candidates; (3) Fully label the target domain via second annotation.

A. Problem Definition

CDAR is a typical cross-domain learning problem, which often consists of a labeled source domain $\mathcal{D}_s = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n_s}$, and an unlabeled target domain $\mathcal{D}_t = \{\mathbf{x}_j\}_{j=1}^{n_t}$. $\mathcal{D}_s$ and $\mathcal{D}_t$ have the same dimensionality and label spaces, i.e. $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$ where d is the dimensionality, and $\mathcal{Y}_s = \mathcal{Y}_t$. In cross-domain learning, they follow different feature distributions. Concretely speaking, we use P and Q to denote the marginal and conditional distributions, respectively. Then, $P(\mathbf{x}_s) \neq P(\mathbf{x}_t)$, and $Q(y_s|\mathbf{x}_s) \neq Q(y_t|\mathbf{x}_t)$. The goal of cross-domain learning is to obtain the labels $\mathbf{y}_t$ for the target domain $\mathcal{D}_t$ using the labeled source domain $\mathcal{D}_s$. Specifically in CDAR, the goal is to use the labeled activity information in one domain to learn the labels for another domain.

STL can simultaneously transform the individual classes of the source and target domains into the same subspaces by exploiting the intra-affinity of classes. Compared to existing approaches which only learn a global domain shift, we adopt the assumption that data samples from the same class should lay on an intrinsic subspace [12]. Fig. 3 illustrates the main idea of STL. There are mainly three steps. Initially, STL generates pseudo labels for the target domain through majority voting technique. Then, it performs intra-class knowledge transfer between domains. Finally, a second annotation step is performed on the newly transformed subspaces.

Now we elaborate each step in detail.

B. Majority Voting

As a preprocessing step of STL, this step generates pseudo labels for the target domain based on several classifiers trained on source domain. Those samples with pseudo labels are called *candidates*. To this end, *majority voting* technique is developed to exploit the knowledge from the crowd [39]. Specifically, STL makes use of some base classifiers learned on $\mathcal{D}_s$ to collaboratively learn the labels for $\mathcal{D}_t$.

Let $A_j (j = 1, 2, \cdots, n_2)$ denote the final result of majority voting on $\mathbf{x}_{t_j}$, and $f_t(j)$ denotes the prediction of the j-th sample by the t-th classifier $f_t(\cdot)$, we have

$$A_j = \begin{cases} \text{majority}(\{f_t(j), t) & \text{if majority holds} \\ -1 & \text{otherwise,} \end{cases} \tag{1}$$

where $t \in \{1, 2, \cdots\}$ denotes the index of classifier.

Majority voting technique generally ensembles all the classifiers learned in the source domain. The condition "majority holds" refers to any potential scheme that helps to generate a better solution such as simple voting and weighted voting. Specifically, A_i could be defined as a) if most classifiers have same results on a sample, we take its label, else we label it '-1'; b) same as a) with voting weights to classifiers; c) the stacking of some base classifiers (similar to the deep forest proposed by Zhou and Feng [40]). In theory, the classifiers can be of any type in our framework.

Using majority voting, STL can generate pseudo labels for the target domain. Label '-1' means that there is no majority consensus for that sample. The samples with this label are called *residuals*, and they will be annotated later. In the sequel, we will use $\mathbf{X}_{can}$ and $\mathbf{X}_{res}$ to denote the candidates and residuals, respectively. And $\widetilde{\mathbf{y}}_{can}$ denotes the pseudo labels for the candidates.

C. Intra-class Transfer

In this step, STL exploits the intra-affinity of classes to further transform each class the candidates and source domain into the same subspace. Initially, $\mathcal{D}_s$ and $\mathbf{X}_{can}$ are divided to C groups according to their (pseudo) labels, where C is the total number of classes. Then, feature transformation is performed within each class of both domains. Finally, the results of distinct subspaces are merged.

The key to successful knowledge transfer is to utilize the *similarity* between the source and the target domain. Hence, how to measure the similarity (or divergence) is critical. We adopt maximum mean discrepancy (MMD) [41] as the measurement. MMD is a nonparametric method to measure the divergence between two distinct distributions and it has been widely applied to many transfer learning methods [8], [10]. The MMD distance between two domains can be formally computed as

$$D(\mathcal{D}_s, \mathcal{D}_t) = \left\| \tfrac{1}{n_s} \sum_{\mathbf{x}_i \in \mathcal{D}_s} \phi(\mathbf{x}_i) - \tfrac{1}{n_t} \sum_{\mathbf{x}_j \in \mathcal{D}_t} \phi(\mathbf{x}_j) \right\|_{\mathcal{H}}^2 \tag{2}$$

where $\mathcal{H}$ denotes reproducing kernel Hilbert space (RKHS). Here $\phi(\cdot)$ denotes some feature map to map the original samples to RKHS. The reason we do not use the original data is that the features are often distorted in the original feature space, and it can be more efficient to perform knowledge transfer in RKHS [8].

The above Eq. 2 derives a global domain shift for the source and target domain just similar to many existing methods [8], [10]. In order to achieve *intra-class transfer*, we need to calculate the MMD distance between *each class*. Since the target domain has no labels, we use the pseudo labels from majority voting. For the candidates and the source domain, we present their *intra-class* MMD distance as

$$\begin{aligned} &D(\mathcal{D}_s, \mathbf{X}_{can}) \\ &= \sum_{c=1}^{C} \left\| \frac{1}{n_s^{(c)}} \sum_{\mathbf{x}_i \in \mathcal{D}_s^{(c)}} \phi(\mathbf{x}_i) - \frac{1}{n_t^{(c)}} \sum_{\mathbf{x}_j \in \mathbf{X}_{can}^{(c)}} \phi(\mathbf{x}_j) \right\|_{\mathcal{H}}^2 \end{aligned} \tag{3}$$

where $c \in \{1, 2, \cdots, C\}$ denote the class label. $\mathcal{D}_s^{(c)}$ and $\mathbf{X}_{can}^{(c)}$ denote the samples belonging to class c in the source and *candidates*, respectively. $n_s^{(c)} = |\mathcal{D}_s^{(c)}|$, and $n_t^{(c)} = |\mathbf{X}_{can}^{(c)}|$.

Unfortunately, it is non-trivial to solve Eq. (3) directly since the mapping function $\phi(\cdot)$ is to be determined. Thus, we turn to some kernel methods. We define a kernel matrix $\mathbf{K} \in \mathbb{R}^{(n_1+n_2)\times(n_1+n_2)}$, which can be constructed by the inner product of the mapping

$$\mathbf{K}_{ij} = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) \tag{4}$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ are samples from either $\mathcal{D}_s$ or $\mathbf{X}_{can}$.

We further introduce a feature transformation matrix $\mathbf{W} \in \mathbb{R}^{(n_1+n_2)\times m}$ to transform the samples of both domains from the original space to the RKHS. Here $m \ll d$ denotes the dimension after feature transformation. Then, by applying kernel tricks, Eq. 3 can be eventually formulated as the following trace optimization problem:

$$\begin{aligned} \min_{\mathbf{W}} \quad & \sum_{c=1}^{C} \text{tr}(\mathbf{W}^\top \mathbf{K} \mathbf{L}_c \mathbf{K} \mathbf{W}) + \lambda \text{tr}(\mathbf{W}^\top \mathbf{W}) \\ \text{s.t.} \quad & \mathbf{W}^\top \mathbf{K} \mathbf{H} \mathbf{K} \mathbf{W} = \mathbf{I} \end{aligned} \tag{5}$$

There are two terms in the objective function of Eq. (5). The first term ($\sum_{c=1}^{C} \text{tr}(\mathbf{W}^\top \mathbf{K} \mathbf{L}_c \mathbf{K} \mathbf{W})$) denotes the MMD distance of each class between source and target domain, and the second one ($\lambda \text{tr}(\mathbf{W}^\top \mathbf{W})$) denotes the regularization term to ensure the problem is well-defined with λ the trade-off parameter. The constraint in Eq. 5 is used to guarantee that the transformed data ($\mathbf{W}^\top \mathbf{K}$) will still preserve some structure property of the original data. $\mathbf{I}_{n_1+n_2}$ is the identical matrix, and $\mathbf{H} = \mathbf{I}_{n_1+n_2} - 1/(n_1+n_2)\mathbf{1}\mathbf{1}^\top$ is the centering matrix. For notational brevity, we will drop the subscript for $\mathbf{I}_{n_1+n_2}$ in the sequel. $\mathbf{L}_c$ is the intra-class MMD matrix, which can be constructed as:

$$(\mathbf{L}_c)_{ij} = \begin{cases} \frac{1}{(n_1^{(c)})^2} & \mathbf{x}_i, \mathbf{x}_j \in \mathcal{D}_s^{(c)} \\ \frac{1}{(n_2^{(c)})^2} & \mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}_{can}^{(c)} \\ -\frac{1}{n_1^{(c)} n_2^{(c)}} & \begin{cases} \mathbf{x}_i \in \mathcal{D}_s^{(c)}, \mathbf{x}_j \in \mathbf{X}_{can}^{(c)} \\ \mathbf{x}_i \in \mathbf{X}_{can}^{(c)}, \mathbf{x}_j \in \mathcal{D}_s^{(c)} \end{cases} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Learning algorithm: Acquiring the solution of Eq. 5 is non-trivial. To this end, we adopt Lagrange method. We denote $\mathbf{\Phi}$ the Lagrange multiplier, then the Lagrange function can be derived as

$$\begin{aligned} L = \; & \text{tr}\left(\mathbf{W}^\top \mathbf{K} \sum_{c=1}^{C} \mathbf{L}_c \mathbf{K}^\top \mathbf{W} \right) + \lambda \text{tr}(\mathbf{W}^\top \mathbf{W}) \\ & + \text{tr}\left((\mathbf{I} - \mathbf{W}^\top \mathbf{K} \mathbf{H} \mathbf{K}^\top \mathbf{W}) \mathbf{\Phi} \right) \end{aligned} \tag{7}$$

Setting the derivative $\partial L/\partial \mathbf{W} = 0$, Eq. 7 can be finally formalized as an generalized eigen-decomposition problem

$$\left(\mathbf{K}\sum_{c=1}^{C}\mathbf{L}_c\mathbf{K}^\top + \lambda\mathbf{I}\right)\mathbf{W} = \mathbf{K}\mathbf{H}\mathbf{K}^\top\mathbf{W}\mathbf{\Phi} \tag{8}$$

Solving Eq. 8 refers to solve this generalized eigen-decomposition problem and take the m smallest eigenvectors to construct $\mathbf{W}$. $\mathbf{W}$ can transform both domains into the same subspace with minimum domain distance while preserving their properties. Since the knowledge transfer pertains to each class, we call this step *intra-class transfer*, and that is where the name *stratified* originates from. After this step, the source and target domains belonging to the same class are simultaneously transformed into the same subspaces.

D. Second annotation

The objective of this step is to annotate the *residual* part using the transformed source domain and candidates. The majority voting provides pseudo labels ($\widetilde{\mathbf{y}}_{can}$) for the *candidates*, which can be transformed into the same subspaces with the target domain after intra-class transfer. Under this circumstance, it is easy to get more reliable predictions ($\hat{\mathbf{y}}_{can}$) of the *candidates*. Specifically, we train a standard classifier using $\{[\mathbf{W}^\top\mathbf{K}]_{1:n_1,:}, \mathbf{y}_s\}$ and apply prediction on $[\mathbf{W}^\top\mathbf{K}]_{n_1+1:n_2,:}$. Finally, the labels of *residuals* can be obtained by training classifier on instances $\{\mathbf{X}_{can}, \hat{\mathbf{y}}_{can}\}$.

Since this step requires annotating the candidates again with more concrete labels, we call it *second annotation*. The labels of candidates can be correspondingly close to the ground truth by annotating twice since the domains are now in the same subspace after intra-class transfer.

E. Iterative refinement

It should be noted that STL could achieve a better prediction if we use the result of *second annotation* as the initial state and run *intra-class transfer* iteratively. This *EM-like* algorithm is empirically effective and will be validated in the following experiments. Additionally, we *only* use majority voting in the first round of the iteration, then the proposed STL could iteratively refine the labels for the target domain by *only* using its previous results.

The overall process of STL is described in Algorithm 1.

Remark: STL is a *general* framework for transfer learning and it can be implemented in different ways according to the specific applications. We only provide one feasible implementation of STL in this section. Meanwhile, each step of STL can be tailored according to certain applications: 1) majority voting could have different classifiers and voting techniques; 2) intra-class transfer could use existing transfer learning approaches such as GFK [9] and TransACT [37]; 3) second annotation could use different classifiers. Therefore, by considering the characteristics of certain applications, STL can be more effective and efficient in solving these problems.

Algorithm 1 STL: Stratified Transfer Learning

```
Input: Source domain $\mathcal{D}_s = \{\mathbf{X}_s, \mathbf{y}_s\}$, target domain $\mathcal{D}_t = \{\mathbf{X}_t\}$, #dimension $m$.
Output: The labels for the target domain: $\{\mathbf{y}_t\}$.
1: Perform majority voting on $\mathcal{D}_t$ using Eq. 1 to get $\{\mathbf{X}_{can}, \widetilde{\mathbf{y}}_{can}\}$ and $\mathbf{X}_{res}$;
2: Construct kernel matrix $\mathbf{K}$ according to Eq. (4) using $\mathbf{X}_{src}$ and $\mathbf{X}_{can}$, and compute the intra-class MMD matrix $\mathbf{L}_c$ using Eq. 6;
3: repeat
4:     Solve the eigen-decomposition problem in Eq. 8 and take the $m$ smallest eigen-vectors to obtain the transformation matrix $\mathbf{W}$;
5:     Transform the same classes of $\mathbf{X}_s$ and $\mathbf{X}_{can}$ into the same subspaces using $\mathbf{W}$, and then merge them;
6:     Perform second annotation to get $\{\hat{\mathbf{y}}_{can}\}$ and $\{\hat{\mathbf{y}}_{res}\}$;
7:     Construct kernel matrix $\mathbf{K}$ and compute the intra-class MMD matrix $\mathbf{L}_c$ using Eq. 6;
8: until Convergence
9: return $\{\mathbf{y}_t\}$.
```

IV. Experimental Evaluation

In this section, we evaluate the performance of STL via extensive experiments on cross-domain activity recognition.

A. Cross-domain activity recognition

Cross-domain activity recognition (CDAR) aims at labeling the activity of one domain using the labeled data from another related domain. There are several types of CDAR: cross-person, cross-device, cross-environment activity recognition, and so on. In our experiments, we choose *cross-position activity recognition*. Specifically, it refers to the situation where the activity labels of some body parts are missing, so it is necessary and feasible to leverage the labeled data from other similar body parts to get the labels of those body parts.

Why cross-position activity recognition: We need to extensively investigate the performance of transfer learning at different degrees of similarities between the source and target domain. In cross-position tasks, there are different degrees of similarities, since body parts are different at certain degrees. For instance, there is more similarity between both arms than between arm and torso. Compared to other types of CDAR such as cross-person or cross-device, cross-position tasks can provide more detailed information about domain similarities, thus more experience about activity transfer can be revealed.

Why knowledge can be transferred between body parts: First of all, different body parts tend to share similar structure and functions. Then, the activity patterns of different body parts are often related. Thus, annotating the activity becomes a cross-domain learning problem because the data distribution is different on body parts.

B. Datasets and Preprocessing

Three large public datasets are adopted in our experiments. TABLE I provides a brief introduction to those three datasets. In the following, we briefly introduce those datasets, and more information can be found in their original papers. OPPORTUNITY dataset (**OPP**) [42] is composed of 4 subjects executing different levels of activities with sensors tied to more than 5

TABLE I

BRIEF INTRODUCTION OF THREE PUBLIC DATASETS FOR ACTIVITY RECOGNITION

Dataset	#Subject	#Activity	#Sample	#Feature	Body parts
OPPORTUNITY	4	4	701,366	459	Back (B), Right Upper Arm (RUA), Right Left Arm (RLA), Left Upper Arm (LLA), Left Lower Arm (LLA)
PAMAP2	9	18	2,844,868	243	Hand (H), Chest(C), Ankle (A)
DSADS	8	19	1,140,000	405	Torso (T), Right Arm (RA), Left Arm (LA), Right Leg (RL), Left Leg (LL)

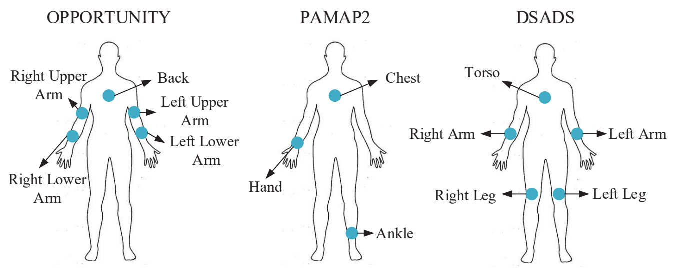


Fig. 4. Different positions on OPPOPTUNITY, PAMAP2 and DSADS.

body parts. PAMAP2 dataset (**PAMAP2**) [43] is collected by 9 subjects performing 18 activities with sensors on 3 body parts. UCI daily and sports dataset (**DSADS**) [44] consists of 19 activities collected from 8 subjects wearing body-worn sensors on 5 body parts. Accelerometer, gyroscope, and magnetometer are all used in three datasets.

Fig. 4 illustrates the positions we investigated in three datasets. In our experiments, we use the data from all three sensors in each body part since most information can be retained in this way. For one sensor, we combine the data from 3 axes together using $a = \sqrt{x^2 + y^2 + z^2}$. Then, we exploit the sliding window technique to extract features (window length is 5s). The feature extraction procedure is mainly executed according to existing work [45]. In total, 27 features from both time and frequency domains are extracted for a single sensor (see TABLE II). Since there are three sensors (i.e. accelerometer, gyroscope, and magnetometer) on one body part, we extracted 81 features from one position.

In order to investigate more detailed information during activity knowledge transfer, we perform CDAR in the following three scenarios according to the different similarities between body parts. Those three scenarios represent three different degrees of similarities between domains. In the sequel, we use the notation $A \rightarrow B$ to denote labeling the activity of domain B using the labeled domain A.

- a) *similar body parts of the same person*, which refers to highly similar body parts such as left and right arms on the same person. Specifically, we extracted the data from OPP and DSADS (PAMAP2 is not included since there are no similar parts), and constructed 8 tasks: RA $\rightarrow$ LA, RL $\rightarrow$ LL, RUA $\rightarrow$ LUA, RLA $\rightarrow$ LLA, and vice versa.
- b) *different body parts of the same person*, which refers to different body parts like torso and arm of the same person. Specifically, we constructed 8 tasks from three datasets: RA $\rightarrow$ T, H $\rightarrow$ C, RLA $\rightarrow$ T, RUA $\rightarrow$ T, and vice versa.
- c) *similar body parts of different person*, which refers to activity recognition on same body parts across different datasets. Specifically, we constructed 6 tasks for T $\rightarrow$ T across three datasets [1].

TABLE II

FEATURES EXTRACTED PER SENSOR ON EACH BODY PART

ID	Feature	Description
1	Mean	Average value of samples in window
2	STD	Standard deviation
3	Minimum	Minimum
4	Maximum	Maximum
5	Mode	The value with the largest frequency
6	Range	Maximum minus minimum
7	Mean crossing rate	Rate of times signal crossing mean value
8	DC	Direct component
9-13	Spectrum peak position	First 5 peaks after FFT
14-18	Frequency	Frequencies corresponding to 5 peaks
19	Energy	Square of norm
20-23	Four shape features	Mean, STD, skewness, kurtosis
24-27	Four amplitude features	Mean, STD, skewness, kurtosis

In total, we constructed 22 tasks. Note that there are different activities in three datasets. For scenario a) and b), we simply use all the classes in each dataset; for scenario c) which is cross-dataset, we extract 4 common classes for each dataset (i.e. *Walking, Sitting, Lying*, and *Standing*). In addition, we did not include the scenario '*different body parts of different person*' since 1) all the methods perform poorly in that scenario, and 2) that scenario does not have reasonable feasibility in real applications.

C. Comparison Methods and Implementation Details

We adopt five state-of-the-art comparison methods:

- PCA: Principal component analysis [4].
- KPCA: Kernel principal component analysis [4].
- TCA: Transfer component analysis [8].
- GFK: Geodesic flow kernel [9].
- TKL: Transfer kernel learning [10].

PCA and KPCA are classic dimensionality reduction methods, while TCA, GFK, and TKL are representative transfer learning approaches. The codes of PCA and KPCA are provided in Matlab. The codes of TCA, GFK, and TKL can be obtained online [2]. The constructed datasets and STL code are released online [3].

We construct the tasks according to each scenario and use the labels for target domain only for testing. Then we perform CDAR using STL and all comparison methods. Other than TKL, all other methods require dimensionality reduction. therefore, they were tested using the same dimension. After that, a classifier with the same parameter is learned using the source domain and then the target domain can be

[1]In this paper, for brevity, we use "T" to denote the Torso / Back / Chest in three datasets, respectively.

[2]https://tinyurl.com/y79j6twy

[3]https://tinyurl.com/y7en6owt

TABLE III

CLASSIFICATION ACCURACY (%) OF STL AND OTHER COMPARISON METHODS ON DIFFERENT CDAR TASKS.

Scenario	Dataset	Task	PCA	KPCA	TCA	GFK	TKL	STL
Similar body parts on same person	DSADS	RA $\rightarrow$ LA	59.91	62.17	66.15	**71.07**	54.10	**71.04**
		RL $\rightarrow$ LL	69.46	70.92	75.06	79.71	61.63	**81.60**
	OPP	RUA $\rightarrow$ LUA	76.12	65.64	76.88	74.62	66.81	**83.96**
		RLA $\rightarrow$ LLA	62.17	66.48	60.60	74.62	66.82	**83.93**
Different body parts on same person	DSADS	RA $\rightarrow$ T	38.89	30.20	39.41	44.19	32.72	**45.61**
	PAMAP2	H $\rightarrow$ C	34.97	24.44	34.86	36.24	35.67	**43.47**
	OPP	RLA $\rightarrow$ T	**59.10**	46.99	55.43	48.89	47.66	56.88
		RUA $\rightarrow$ T	67.95	54.52	67.50	66.14	60.49	**75.15**
Similar body parts on different person	PAMAP2 $\rightarrow$ OPP	T $\rightarrow$ T	32.80	**43.78**	39.02	27.64	35.64	40.10
	DSADS $\rightarrow$ PAMAP	T $\rightarrow$ T	23.19	17.95	23.66	19.39	21.65	**37.83**
	OPP $\rightarrow$ DSADS	T $\rightarrow$ T	44.30	49.35	46.91	48.07	52.79	**55.45**
Average			51.71	48.40	53.23	53.69	48.73	**61.37**

labeled. To be more specific, we use the random forest classifier ($\#Tree = 30$) as the final classifier for all the 6 methods. For majority voting in STL, we simply use SVM ($C = 100$), kNN ($k = 3$), and random forest ($\#Tree = 30$) as the base classifiers. Other parameters are searched to achieve their optimal performance. The #iteration is set to be $T = 10$ for STL. It is noticeable that we randomly shuffle the experimental data for 5 times in order to gain robust results.

Classification $accuracy$ on the target domain is adopted as the evaluation metric, which is widely used in existing transfer learning methods [10], [8]

$$Accuracy = \frac{|\mathbf{x} : \mathbf{x} \in \mathcal{D}_t \wedge \hat{y}(\mathbf{x}) = y(\mathbf{x})|}{\mathbf{x} : \mathbf{x} \in \mathcal{D}_t} \tag{9}$$

where $y(\mathbf{x})$ and $\hat{y}(\mathbf{x})$ are the truth and predicted labels, respectively.

D. Classification Accuracy of STL

We run STL and other methods on all CDAR tasks and report the classification accuracy in TABLE III. For brevity, we only report the results of $A \rightarrow B$ since its result is close to $B \rightarrow A$. It is obvious that STL significantly outperforms other methods in most cases (with a remarkable improvement of $\mathbf{7.68}\%$ over the best baseline GFK). Compared to traditional dimensionality reduction methods (PCA and KPCA), STL improves the accuracy by $10\% \sim 20\%$, which implies that STL is better than typical dimensionality reduction methods. Compared to transfer learning methods (TCA, GFK, and TKL), STL still shows an improvement of $5\% \sim 15\%$. Therefore, STL is more effective than all the comparison methods in most cases.

The performance of TKL is the worst, because of the instability of the transfer kernel. TCA only learns a global domain shift, thus the similarity within classes is not fully exploited. The performance of GFK is second to STL, even if GFK also learns a global domain shift. Because the geodesic distance in high-dimensional space is capable of preserving the intra properties of domains.

The differences between STL and GFK are: 1) STL outperforms GFK in most cases with significant improvement; 2) STL strongly outperforms GFK in *less* similar scenarios (scenarios a) and b)), indicating that STL is more robust in recognizing different levels of activities. For STL, it performs intra-class knowledge transfer after generating pseudo labels for candidates. Thus, better performance can be achieved by exploiting the intra-affinity of classes.

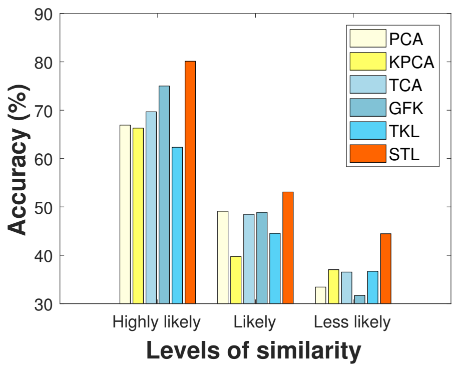
E. Performance on Different Degrees of Similarities

We explore the performance of the transfer learning methods under different degrees of similarities between the source and target domain. The average classification accuracy of each method is shown in Fig. 5(a). For all the methods, the accuracy drops as the domain similarity becomes less. Additionally, the performance of STL is the best in all scenarios.

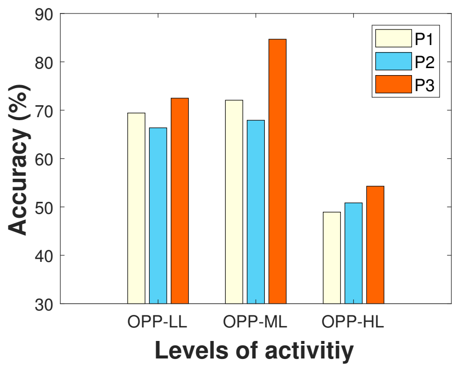
More elaborate observations can be given combining TABLE III and Fig. 5(a). In different scenarios, the change of classification accuracy of each method follows the same tendency. Specifically, the performance of all the methods is the best between similar body parts for the same person (e.g. RA $\rightarrow$ LA). The performance becomes worse for different body parts (e.g. RA $\rightarrow$ T). This is because Right Arm (RA) is more similar to Left Arm (LA) than to Torso (T). For a different person, all methods produce the worst results because different people have exactly different body structure and moving patterns (e.g. T $\rightarrow$ T across datasets).

At the same degree of similarity, the results are also different. For example, the performance of RUA $\rightarrow$ T is better than RLA $\rightarrow$ T in the same dataset. Because there is more similarity between Right Upper Arm (RUA) and Torso (T) than between Right Lower Arm (RLA) and Torso (T). Other positions also share the similar discovery.

All the experimental results indicate that the *similarity* between the source and target domain is important for cross-domain learning. It is critical to find the **right** auxiliary domain to perform successful knowledge transfer. In our CDAR experiments, the body structure and moving patterns contribute to the similarity. In real life, other factors such as age and hobby also help define the similarity of activities. For other cross-domain tasks (image classification etc.), finding the relevant domain is also important. For example, the most similar image set for a dog is probably a cat, since those two kinds of animals have similar body structures, moving patterns, and living environments.



(a) Different degrees of similarities



(b) Different levels of activities of 3 persons

Fig. 5. Classification accuracy of (a) different degrees of similarities and (b) different levels of activities in transfer learning.

F. Performance on Different Levels of Activities

Different levels of activities imply the different depths of activity granularities. In this section, we extensively investigate the performance of our proposed STL on CDAR tasks at different activity levels. By taking advantage of the diverse activity classes in OPP dataset [42], we analyze the transfer learning performance on low-level (**OPP-LL**), middle-level (**OPP-ML**), and high-level (**OPP-HL**) activities. The results are presented in Fig. 5(b). It indicates the best performance is achieved at *middle-level* activities, while it suffers from low-level and even worse at high-level activities.

Low-level activities such as *Walking* and middle-level activities such as *Closing* are mostly contributed by the atomic movements of body parts and they are likely to achieve better transfer results than the high-levels. On the other hand, high-level activities such as *Coffee Time* not only involve basic body movements, but also contain *contextual* information like ambient or objects, which is difficult to capture only by the body parts. Since the bridge of successful cross-position transfer learning is the similarity of body parts, it is not ideal to achieve good transfer performance by only using body parts. The reason why results on OPP-ML are better than OPP-LL is that activities of OPP-ML are more fine-grained than OPP-LL, making it more capable of capturing the similarities between the body parts.

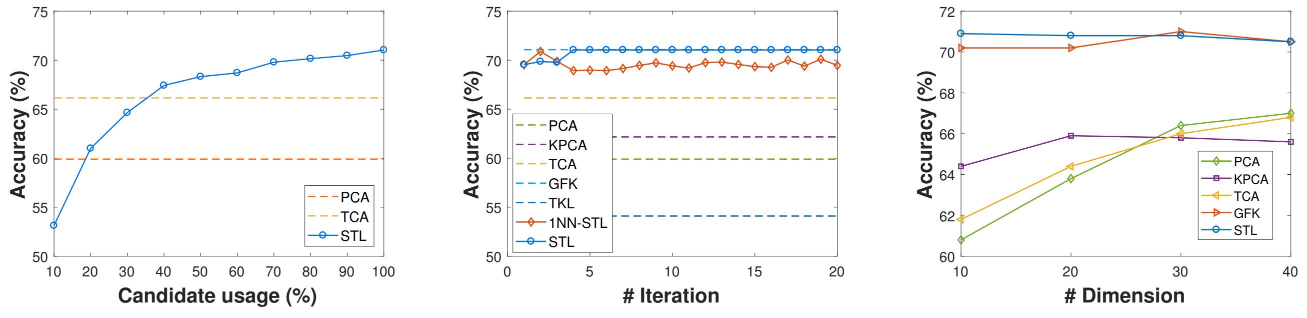
G. Effectiveness verification of STL

In this section, we verify the effectiveness of STL in several aspects. The core idea of STL is *intra-class transfer*, where the *pseudo* labels of the *candidates* are acting as the evidence of transfer learning. It is intuitive to ask the following three questions. Firstly, *Can the confidence of the pseudo labels have an influence on the algorithm?* It seems that STL will not achieve good performance if there is not enough *reliable* candidates available. Secondly, *Can the choice of majority voting classifiers affect the framework?* If we use different classifiers, the performance is likely to vary. Thirdly, *Can intra-class transfer be implemented in parallel?* Since in real applications, there are often several classes, which would require a large amount of training time if not in parallel. In this part, we answer those questions through the following experiments.

1) The confidence of the candidates: We control the percentage of the *candidates* from 10% to 100% in every trial and make the rest belong to the *residual* part. Then we test the performance of STL. We test on the task LA $\rightarrow$ RA on DSADS and compare with other 2 methods (PCA and TCA). The result is shown in Fig. 6(a). For simplicity, we only compare STL with PCA and TCA, since they are both classic dimensionality reduction methods. From the results, we can observe that the performance of STL is increasing along with the increment of candidates percentage. More importantly, STL outperforms the other two methods with **less than** 40% of candidates. It reveals that STL does not largely rely on the confidence of the candidates and can achieve good performance even with fewer candidates.

2) Majority voting classifiers and iteration: To test the effectiveness of majority voting classifiers, we choose 1 nearest neighbor (1NN) as the base classifier of majority voting. Then we run STL iteratively. The results are shown in Fig. 6(b), where '1NN-STL' and 'STL' denotes the result of STL using 1NN and random forest as the base majority voting classifier, respectively. From those results, we can observe: 1) STL can iteratively improve the classification accuracy even with some weak majority voting classifier. 1NN-STL achieves slightly worse than STL, indicating that STL is rather *robust* to the base classifiers. Since more powerful classifiers would lead to better performance, we strongly suggest using more reliable classifiers for majority voting in real problems. 2) STL can reach a quick convergence within *fewer than* 10 iterations. This indicates that STL can be efficiently trained.

3) Potential of parallel deployment: STL has another significant advantage: it can easily be implemented in *parallel*, which is rather important in real applications. Two steps of STL intuitively have that excellence: 1) majority voting involves more than one classifier, where each classifier can be trained and used individually; 2) the intra-class transfer can be performed within each class separately. Those two properties demonstrate that STL could be more powerful and efficient if deployed in parallel in real applications. We will continue to explore that advantage in the future.



(a) Candidate usage (b) Iteration (c) Parameter sensitivity

Fig. 6. Detailed results of a) STL with increasing usage of candidates for RA $\rightarrow$ LA on DSADS dataset; b) 1NN-STL and STL with other methods; c) parameter sensitivity of STL in comparison with other methods

H. Parameter Sensitivity

STL involves two parameters: the dimension m, and trade-off parameter λ. In this experiment, we empirically evaluate the sensitivity of m. The evaluation of λ is omitted due to page limit and our experiments have verified its robustness.

We set $m \in \{10, 20, 30, 40\}$ and test the performance of STL and other dimensionality reduction methods. As shown in Fig. 6(c), STL achieves the best accuracy under different dimensions. Meanwhile, the accuracy of STL almost does not change with the decrement of m. The results reveal that STL is much more effective and robust than other methods under different dimensions. Therefore, STL can be easily applied to many cross-domain tasks which require robust performance w.r.t. different dimensions.

V. Discussion

We studied cross-domain activity recognition via the STL framework and evaluated its performance on cross-position HAR tasks. There are more applications in pervasive computing that STL could be applied to. In this section, we discuss the potential of STL in other applications and give some empirical suggestions.

1) Activity recognition. The results of activity recognition can be different according to different *devices*, *users*, and wearing *positions*. STL makes it possible to perform cross-device/user/position activity recognition with high accuracy. In case cross-domain learning is needed, finding and measuring the similarity between the device/user/position is critical.

2) Localization. In WiFi localization, the WiFi signal changes with the *time*, *sensor*, and *environment*, causing the distributions different. So it is necessary to perform cross-domain localization. When applying STL to this situation, it is also important to capture the similarity of signals according to time/sensor/environment.

3) Gesture recognition. For gesture recognition, due to differences in hand structure and moving patterns, the model cannot generalize well. In this case, STL can also be a good option. Meanwhile, special attention needs to be paid to the divergence between the different characteristic of the subjects.

4) Other context-related applications. Other applications include smart home sensing, intelligent city planning, healthcare, and human-computer interaction. They are also context-related applications. Most of the models built for pervasive computing are only *specific* to certain contexts. Transfer learning makes it possible to transfer the knowledge between related contexts, of which STL can achieve the best performance. But when recognizing high-level contexts such as *Coffee Time*, it is rather important to consider the relationship between different contexts in order to utilize their similarities. The research on this area is still on the go.

5) Suggestions for using STL. Firstly, before using STL and other transfer learning approaches, it is critical to investigate the similarity between the source and target domain in advance. Secondly, it is better to use rather strong classifiers for majority voting since it makes the convergence quick. Thirdly, STL can be more efficient if deployed in parallel. Additionally, each step of STL can be tailored according to specific applications.

VI. Conclusion and Future Work

The label scarcity problem is very common in pervasive computing. Transfer learning addresses this issue by leveraging labeled data from auxiliary domains to annotate the target domain. In this paper, we propose a novel and general Stratified Transfer Learning (STL) framework for cross-domain learning in pervasive computing. Compared to existing approaches which only obtain a global domain shift, STL can exploit the intra-affinity of each class between different domains. We also provide a solution to implement STL. Experiments on three large public datasets demonstrate the significant superiority of STL over existing five state-of-the-art methods. In addition, we extensively analyze the performance of transfer learning under different degrees of similarities and different levels of activities. We also extensively discuss the potential of STL in other pervasive computing applications. In addition, STL is a general framework where each step can be tailored according to specific applications, making STL more effective and efficient in solving other problems.

In the future, we plan to extend STL using deep learning, as well as evaluating STL through more cross-domain learning problems in pervasive computing.

Acknowledgments

J. Wang would like to thank Dr. Shuji Hao from A*star, Singapore, and Dr. Mingsheng Long from THU, China, for their valuable comments. This work is supported in part by National Key R & D Plan of China (No.2016YFB1001200), NSFC (No.61572471), and Science and Technology Planning Project of Guangdong Province (No.2015B010105001).

References

[1] H. Xu, Z. Yang, Z. Zhou, L. Shangguan, K. Yi, and Y. Liu, "Indoor localization via multi-modal sensing on smartphones," in *UbiComp*. ACM, 2016, pp. 208–219.

[2] M. Zhao, S. Yue, D. Katabi, and T. Jaakkola, "Learning sleep stages from radio signals: A deep adversarial architecture," in *ICML*, 2017.

[3] J. Wen, J. Indulska, and M. Zhong, "Adaptive activity learning with dynamically available context," in *2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 2016, pp. 1–11.

[4] I. K. Fodor, "A survey of dimension reduction techniques," *Center for Applied Scientific Computing, Lawrence Livermore National Laboratory*, vol. 9, pp. 1–18, 2002.

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

[6] J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," in *Proceedings of the 2006 conference on empirical methods in natural language processing*. Association for Computational Linguistics, 2006, pp. 120–128.

[7] W. M. Kouw, L. J. van der Maaten, J. H. Krijthe, and M. Loog, "Feature-level domain adaptation," *Journal of Machine Learning Research*, vol. 17, no. 171, pp. 1–32, 2016.

[8] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199–210, 2011.

[9] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *CVPR*, 2012, pp. 2066–2073.

[10] M. Long, J. Wang, J. Sun, and S. Y. Philip, "Domain invariant transfer kernel learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 6, pp. 1519–1532, 2015.

[11] Y. Lin, J. Chen, Y. Cao, Y. Zhou, L. Zhang, Y. Y. Tang, and S. Wang, "Cross-domain recognition by identifying joint subspaces of source domain and target domain," *IEEE transactions on cybernetics*, 2016.

[12] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 11, pp. 2765–2781, 2013.

[13] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Computing Surveys (CSUR)*, vol. 46, no. 3, p. 33, 2014.

[14] O. D. Lara and M. A. Labrador, "A survey on human activity recognition using wearable sensors," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1192–1209, 2013.

[15] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *arXiv preprint arXiv:1707.03502*, 2017.

[16] V. W. Zheng and Q. Yang, "User-dependent aspect model for collaborative activity recognition," in *IJCAI*, vol. 22, no. 3, 2011, pp. 2085–2090.

[17] Y. Chen, Y. Gu, X. Jiang, and J. Wang, "Ocean: A new opportunistic computing model for wearable activity recognition," in *UbiComp Adjunct*. ACM, 2016, pp. 33–36.

[18] H. S. Hossain, N. Roy, and M. A. A. H. Khan, "Active learning enabled activity recognition," in *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2016, pp. 1–9.

[19] W. S. Lasecki, Y. C. Song, H. Kautz, and J. P. Bigham, "Real-time crowd labeling for deployable activity recognition," in *CSCW*. ACM, 2013, pp. 1203–1212.

[20] L. T. Nguyen, M. Zeng, P. Tague, and J. Zhang, "I did not smoke 100 cigarettes today!: avoiding false positives in real-world activity recognition," in *UbiComp*. ACM, 2015, pp. 1053–1063.

[21] L. Hu, Y. Chen, S. Wang, J. Wang, J. Shen, X. Jiang, and Z. Shen, "Less annotation on personalized activity recognition using context data," in *Proceedings of the 2016 International IEEE Conference on Ubiquitous Intelligence Computing (UIC)*, July 2016, pp. 327–332.

[22] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[23] F. J. O. Morales and D. Roggen, "Deep convolutional feature transfer across mobile activity recognition domains, sensor modalities and locations," in *Proceedings of the 2016 ACM International Symposium on Wearable Computers*. ACM, 2016, pp. 92–99.

[24] S. J. Pan, J. T. Kwok, and Q. Yang, "Transfer learning via dimensionality reduction," in *AAAI*, vol. 8, 2008, pp. 677–682.

[25] L. Duan, I. W. Tsang, and D. Xu, "Domain transfer multiple kernel learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 465–479, 2012.

[26] B. Tan, Y. Zhang, S. J. Pan, and Q. Yang, "Distant domain transfer learning," in *AAAI*, 2017.

[27] R. Chattopadhyay, Q. Sun, W. Fan, I. Davidson, S. Panchanathan, and J. Ye, "Multisource domain adaptation and its application to early detection of fatigue," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 4, p. 18, 2012.

[28] Y. Yao and G. Doretto, "Boosting for transfer learning with multiple sources," in *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*. IEEE, 2010, pp. 1855–1862.

[29] Z. Zhao, Y. Chen, J. Liu, Z. Shen, and M. Liu, "Cross-people mobile-phone based activity recognition," in *IJCAI*, vol. 11, 2011, pp. 2545–2550.

[30] J. Wang, Y. Chen, S. Hao, W. Feng, and Z. Shen, "Balanced distribution adaptation for transfer learning," in *The IEEE International conference on data mining (ICDM)*, 2017, pp. 1129–1134.

[31] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2200–2207.

[32] F. Dorri and A. Ghodsi, "Adapting component analysis," in *ICDM*. IEEE, 2012, pp. 846–851.

[33] C.-W. Seah, I. W.-H. Tsang, Y.-S. Ong, and Q. Mao, "Learning target predictive function without target labels," in *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, 2012, pp. 1098–1103.

[34] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *ICML*, 2011, pp. 513–520.

[35] M. Gong, K. Zhang, T. Liu, D. Tao, C. Glymour, and B. Schölkopf, "Domain adaptation with conditional transferable components," in *ICML*, 2016, pp. 2839–2848.

[36] D. Cook, K. D. Feuz, and N. C. Krishnan, "Transfer learning for activity recognition: A survey," *Knowledge and information systems*, vol. 36, no. 3, pp. 537–556, 2013.

[37] M. A. A. H. Khan and N. Roy, "Transact: Transfer learning enabled activity recognition," in *PerCom Workshops*. IEEE, 2017, pp. 545–550.

[38] K. D. Feuz and D. J. Cook, "Collegial activity learning between heterogeneous sensors," *Knowledge and Information Systems*, pp. 1–28, 2017.

[39] D. Prelec, H. S. Seung, and J. McCoy, "A solution to the single-question crowd wisdom problem," *Nature*, vol. 541, no. 7638, pp. 532–535, 2017.

[40] Z.-H. Zhou and J. Feng, "Deep forest: Towards an alternative to deep neural networks," in *IJCAI*, 2017.

[41] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *Journal of Machine Learning Research*, vol. 13, no. Mar, pp. 723–773, 2012.

[42] R. Chavarriaga, H. Sagha, A. Calatroni, S. T. Digumarti, G. Tröster, J. d. R. Millán, and D. Roggen, "The opportunity challenge: A benchmark database for on-body sensor-based activity recognition," *Pattern Recognition Letters*, vol. 34, no. 15, pp. 2033–2042, 2013.

[43] A. Reiss and D. Stricker, "Introducing a new benchmarked dataset for activity monitoring," in *Wearable Computers (ISWC), 2012 16th International Symposium on*. IEEE, 2012, pp. 108–109.

[44] B. Barshan and M. C. Yüksek, "Recognizing daily and sports activities in two open source machine learning environments using body-worn sensor units," *The Computer Journal*, vol. 57, no. 11, pp. 1649–1667, 2014.

[45] L. Hu, Y. Chen, J. Wang *et al.*, "Okrelm: online kernelized and regularized extreme learning machine for wearable-based activity recognition," *Int. J. of Machine Learning and Cybernetics*, pp. 1–14, 2017.