Pervasive and Mobile Computing 57 (2019) 1–13

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc

Cross-position activity recognition with stratified transfer learning

Yiqiang Chen [a,b,*], Jindong Wang [a,b], Meiyu Huang [c], Han Yu [d]

[a] Beijing Key Lab of Mobile Computing and Pervasive Devices, Institute of Computing Technology, Chinese Academy of Sciences, China
[b] University of Chinese Academy of Sciences, China
[c] Qian Xuesen Lab of Space Technology, China Academy of Space Technology, China
[d] School of Computer Science and Engineering, Nanyang Technological University, Singapore

ARTICLE INFO

Article history:
Available online 17 April 2019

Keywords:
Activity recognition
Transfer learning
Domain adaptation
Pervasive computing

ABSTRACT

Human activity recognition (HAR) aims to recognize the activities of daily living by utilizing the sensors attached to different body parts. HAR relies on the machine learning models trained using sufficient activity data. However, when the labels from a certain body position (i.e. *target* domain) are missing, how to leverage the data from other positions (i.e. *source* domain) to help recognize the activities of this position? This problem can be divided into two steps. Firstly, when there are several source domains available, it is often difficult to select the most *similar* source domain to the target domain. Secondly, with the selected source domain, we need to perform accurate knowledge transfer between domains in order to recognize the activities on the target domain. Existing methods only learn the global distance between domains while ignoring the local property. In this paper, we propose a *Stratified Transfer Learning* (STL) framework to perform both source domain selection and activity transfer. STL is based on our proposed *Stratified* distance to capture the local property of domains. STL consists of two components: 1) Stratified Domain Selection (STL-SDS), which can select the most similar source domain to the target domain; and 2) Stratified Activity Transfer (STL-SAT), which is able to perform accurate knowledge transfer. Extensive experiments on three public activity recognition datasets demonstrate the superiority of STL.

1. Introduction

Human activity recognition (HAR) aims to seek high-level knowledge from the low-level sensor inputs. For example, we can detect if a person is walking or running using the on-body sensors such as the smartphone or the wristband. Activities are of great importance to a person's health status. When a person is performing some activities, each of his body parts has certain activity patterns. Thus, sensors can be attached on some body positions to collect activity data which can be used to build machine learning models in order to recognize their activities. The combination of sensor signals from different body positions can be used to reflect meaningful knowledge such as a person's detailed health conditions [1] and working states [2]. However, it is nontrivial to design wearing styles for a wearable device. On the one hand, it is not bearable to equip all the body positions with sensors which makes the activities not natural. Therefore,

* Corresponding author at: Beijing Key Lab of Mobile Computing and Pervasive Devices, Institute of Computing Technology, Chinese Academy of Sciences, China.
E-mail address: yqchen@ict.ac.cn (Y. Chen).

https://doi.org/10.1016/j.pmcj.2019.04.004
1574-1192/

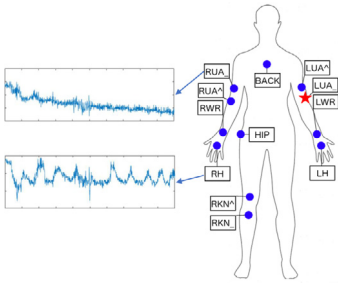


Fig. 1. An example of cross-position activity recognition. How to leverage the well-labeled activity data on other body parts (the blue dots) to acquire the labels of the pentacle part given that their signal distributions are different?

we can only attach sensors on limited body positions. On the other hand, it is impossible to perform HAR if the labels on some body parts are missing, since the activity patterns on specific body positions are significant to capture certain information.

Fig. 1 illustrates this situation. Assume this person is suffering from Small Vessel Disease (SVD) [3], which is a severe brain disease heavily related to activities. However, we cannot equip his all body with sensors to acquire the labels since this will make his activities unnatural. We can only label the activities on certain body parts in reality. If the doctor wants to see his activity information on the arm (the red pentacle, we call it the *target domain*), which only contains sensor readings instead of labels, how to utilize the information on other parts (such as torso or leg, we call them the *source domains*) to help obtain the labels on the target domain? This is referred to as the *cross-position activity recognition (CPAR).*

The problem of CPAR is extremely challenging. Firstly, we do not know which body part is the most similar to the target position since the sensor signals are not independent, but highly correlated because of the shared body structures and functions. If we use all the body parts as the source domain, there is likely to be *negative* transfer [4] because some body parts may be dissimilar. Secondly, we only have the raw activity data on the target domain without the actual activity labels, making it infeasible to measure the similarities between different body positions. Thirdly, even when we know the similar body parts to the target domain, it is still difficult to build a good machine learning model using both the source and the target domains. The reason is that signals from different domains are following different distributions, which means there are distribution discrepancies between them. However, traditional machine learning models are built by assuming that all signals follow the same distribution. Fourthly, when it comes to *multiple* persons, the sensor readings are more different compared to different body parts on one person. This makes the problem more challenging.

To tackle the above challenges, several transfer learning methods have been proposed [5]. The key is to learn and reduce the distribution divergence (distance) between two domains. With the distance, we can perform source domain selection as well as knowledge transfer. Based on this principle, existing methods can be summarized into two categories: exploiting the correlations between features [6,7], or transforming both the source and the target domains into a new shared feature space [8–10].

Existing approaches tend to reduce the *global* distance by projecting all samples in both domains into a single subspace. However, they fail to consider the *local* property within classes [11]. The *global* distance may result in loss of domain local property such as the source label information and the similarities within the same class. Therefore, it will generate a negative impact on the source selection as well as the transfer learning process. It is necessary to exploit the local property of classes to overcome the limitation of global distance learning.

In this paper, we propose a **Stratified Transfer Learning** (STL) framework to tackle the challenges of both source domain selection and knowledge transfer in CPAR. The term *stratified* comes from the notion of *splitting at different levels and then combining*. We adopt the well-established assumption that data samples within the same class should lay on the same subspace, even if they come from different domains [12]. Thus, *stratified* refers to the procedure of transforming features into distinct subspaces. This has motivated us to propose the concept of **Stratified** distance (SD) in comparison to traditional *Global* distance (GD). STL has two components regarding the challenges in CPAR: a **Stratified Domain Selection** (STL-SDS) algorithm to select the most similar source domain to the target domain, and a **Stratified Activity Transfer** (STL-SAT) method to perform activity knowledge transfer between different body parts. Both STL-SDS and STL-SAT are able to exploit the local property of domains, thus they can achieve promising results in CPAR. Comprehensive experiments on 4 large public activity recognition datasets (i.e. OPPORTUNITY, PAMAP2, UCI DSADS, and MHEALTH) demonstrate that STL-SDS is better than the existing global distance used in selecting source domains. STL-SAT outperforms five state-of-the-art methods with a significant improvement of **6.3**% in classification accuracy with improved $F1$ score.

This paper is a substantial extended version of our PerCom paper [13]. To summarize, the contributions are as follows:

(1) We propose the *Stratified Transfer Learning* (STL) framework for source domain selection and knowledge transfer in CPAR. STL is the *first* attempt to exploit the *Stratified* distance (SD) in order to capture the *local* property between domains. SD is a general distance that can be applied to different transfer learning applications.

(2) We propose two novel algorithms: (1) *Stratified Domain Selection* (STL-SDS) algorithm to accurately select the most similar source domain to the target domain, and (2) *Stratified Activity Transfer* (STL-SAT) method to perform knowledge transfer for cross-position activity recognition.

(3) We extensively investigate the performance of STL on four public HAR datasets, indicating its effectiveness in cross-domain HAR applications.

2. Related work

2.1. Activity recognition

Human Activity recognition has been a popular research topic in pervasive computing [14] for its competence in learning profound high-level knowledge about human activity from raw sensor inputs.

Conventional machine learning approaches have made tremendous progress on HAR by adopting machine learning algorithms such as similarity-based approach [15,16], active learning [17], crowdsourcing [18], and other semi-supervised methods [19,20]. 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 and the test data are from different feature distributions, those conventional methods are prune to under-fitting since their generalization ability will be undermined [4].

2.2. Transfer learning

According to the literature survey [4], transfer learning can be categorized into 3 types: instance-based, parameter-based, and feature-based methods. 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. 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, SAT transforms the domain data into a new subspace, which does not depend on the domain knowledge.

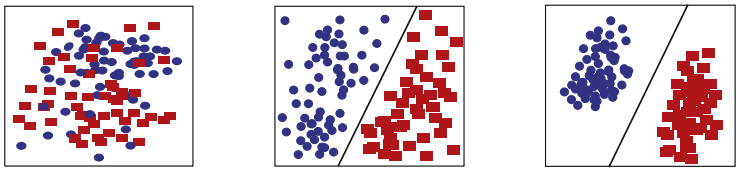
Transform domains into new feature space. [21] proposed maximum mean discrepancy embedding (MMDE) to learn latent features in the reproducing kernel Hilbert space (RKHS). MMDE requires solving a semidefinite programming (SDP) problem, which is computationally prohibitive. [8] extended MMDE by Transfer Component Analysis (TCA), which learns a kernel in RKHS. [22] adopted a similar idea. [23] learns target predictive function with a low variance. [24] 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 SAT focuses on the marginal distribution. [10] proposed transfer kernel learning (TKL), which learned a domain-invariant kernel in RKHS. [25] studied the conditional transfer components between domains. Methods in this 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.

2.3. Transfer learning based activity recognition

Among existing work on transfer learning based HAR [5], Zhao et al. proposed a transfer learning method TransEMDT [26] using decision trees, but it ignored the intra-class similarity within classes. [27] 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. [28] proposed a heterogeneous transfer learning method for HAR, but it only learns a global domain shift. A more recent work of Wang et al. [29] performs activity recognition using deep transfer learning, which also focused on the global distance between domains. To the best of our knowledge, STL is the first attempt towards learning a stratified distance to capture the local property of domains.

3. Stratified transfer learning

In this section, we introduce the Stratified Transfer Learning (STL) framework for cross-position activity recognition.



(a) Original target (b) Global dist. (c) Stratified dist.

Fig. 2. Traditional global distance (GD) and proposed stratified distance (SD).

3.1. Problem definition

The goal of cross-position activity recognition (CPAR) is to predict the activities of one body part (i.e. the *target* domain) using existing labeled data from another body part (i.e. the *source* domain). Formally speaking, we can use $\mathcal{D}_s = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{n_s}$ to denote the labeled source domain, and $\mathcal{D}_t = \{(\mathbf{x}_j^t, y_j^t)\}_{j=1}^{n_t}$ denotes the target domain. Note that y_j^t is missing, i.e. it is the goal of CPAR. In CPAR, we assume the sensor modalities and the activity categories are equal in all body parts, thus the feature space $\mathcal{X}^s = \mathcal{X}^t$ and label space $\mathcal{Y}_s = \mathcal{Y}_t$. Here, d denotes the feature dimension. We use C to denote the total categories of both domains. The different data distributions on different body parts means that the marginal and conditional distributions between domains are different. Therefore, $P(\mathbf{x}_s) \neq P(\mathbf{x}_t)$, and $Q(y_s|\mathbf{x}_s) \neq Q(y_t|\mathbf{x}_t)$.

3.2. Motivation

The key to successful transfer is to utilize the *similarity* between the source and the target domain [4]. It is not difficult to directly measure the similarity between domains using existing distance functions. For instance, we can adopt the Euclidean distance or the Kullback–Leibler divergence [30] to calculate the probability distance between domains. Unfortunately, these computing approaches are only calculating the *global* distance between domains since they are applied to the whole domain. The *global* distance may result in loss of domain local property such as the source label information and the similarities within the same class. Therefore, it will generate a negative impact on the source selection as well as activity transfer.

In order to capture the local property of domains, we adopt the assumption [12]: the data samples from the same class should lay on the *same* subspace, even if they belong to different domains. By following this assumption, we propose the *stratified* distance to capture the local property of domains. The *stratified* distance refers to the class-wise distance between two different domains. For example, if there are 5 classes in two domains: $\mathcal{D}_s = (\mathcal{D}_s^{(1)}, \ldots, \mathcal{D}_s^{(5)})$ and $\mathcal{D}_t = (\mathcal{D}_t^{(1)}, \ldots, \mathcal{D}_t^{(5)})$, then the stratified distance (SD) should be calculated as:

$$SD = \frac{1}{5}\sum_{i=1}^{5} Dist(\mathcal{D}_s^{(i)}, \mathcal{D}_t^{(i)}), \tag{1}$$

where $Dist(\cdot)$ denotes some distance function such as the Euclidean distance. As a comparison, the global distance (GD) can be represented as:

$$GD = Dist(\mathcal{D}_s, \mathcal{D}_t). \tag{2}$$

Fig. 2 briefly illustrates the results of *global* and our *stratified* distance using two classes of the same target domain [13]. It indicates that *SD* could not only help to learn good classification function, but helps to obtain tighter within-class distance.

3.3. The STL framework

In this paper, we propose **Stratified Transfer Learning** (STL) framework to address both of these two challenges. Firstly, STL uses a popular *majority voting* technique to acquire the labels for the target domain. Secondly, STL exploits an effective distance measure called Maximum Mean Discrepancy (MMD) to calculate the SD distance and perform *Intra-class transfer*. Finally, STL could perform source domain selection and activity transfer based on the calculated distance. Fig. 3 illustrates the main idea of the STL framework.

3.3.1. Majority voting

We can easily obtain the *pseudo* labels for the target domain based on a weak classier such as 1-nearest-neighbor (1NN) trained on the source domain by following existing methods [31]. However, since there is a large distribution discrepancy between two domains, a simple 1NN classifier may not work well. Therefore, we propose to use the *majority voting* technique to exploit the knowledge from the crowd [32]. The idea is that one certain classifier may be less reliable,

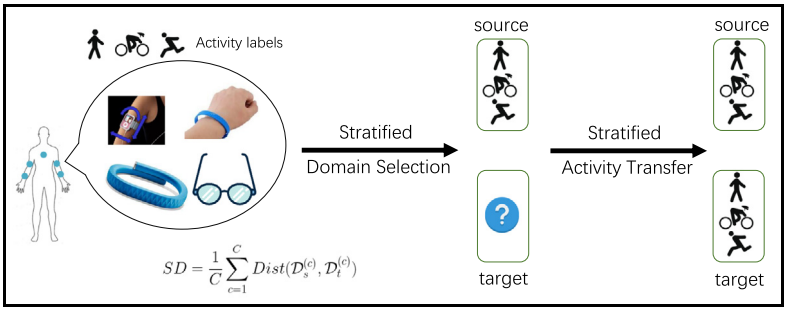


Fig. 3. Main idea of the Stratified Transfer Learning (STL) framework. There are two steps: (1) Stratified Domain Selection (STL-SDS), which can select the most similar source domains to the target domains. (2) Stratified Activity Transfer (STL-SAT), which can perform activity recognition on the target domain based on transfer learning.

so we assemble several different classifiers to obtain more reliable pseudo labels. To this end, STL makes use of some base classifiers learned on $\mathcal{D}_s$ to collaboratively learn the labels for $\mathcal{D}_t$.

Let $A_j(j = 1, 2, \ldots, n_2)$ denotes the final result of majority voting on $\mathbf{x}_{t_j}$, and $f_t(j)$ denotes the prediction of the jth sample by the tth classifier $f_t(\cdot)$, then:

$$A_j = \begin{cases} \text{majority}(f_t(j), t) & \text{if majority holds} \\ -1 & \text{otherwise} \end{cases} \tag{3}$$

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

The 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_j could be defined as (a) if most classifiers have the 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. In theory, the classifiers can be of any type.

Formally, we call the samples with pseudo labels in the target domain *Candidates*, which is denoted as $\mathbf{X}_{can}$. For those samples with label '-1', we call them *Residuals* and denote as $\mathbf{X}_{res}$. Using majority voting, STL can generate reliable *pseudo* labels for the target domain.

3.3.2. Intra-class transfer

In this step, STL exploits the local property of domains to further transform each class of the source and target domains into the same subspace. Since the properties within each class are more similar, the *Intra-class transfer* technique will guarantee that the transformed domains have the minimal distance. In this step, we only consider the *candidates* from the target domain for their reliable pseudo labels.

Initially, $\mathcal{D}_s$ and $\mathbf{X}_{can}$ are divided into 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.

In order to achieve intra-class transfer, we need to calculate the 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 calculate their *intra-class* distance using the *stratified* distance:

$$SD = \frac{1}{C}\sum_{c=1}^{C} Dist(\mathcal{D}_s^{(c)}, \mathbf{X}_{can}^{(c)}), \tag{4}$$

where $\mathcal{D}_s^{(c)}$ and $\mathbf{X}_{can}^{(c)}$ denote the samples from class c in the source and candidates, respectively.

This distance can easily be calculated using existing metrics such as Euclidean distance and Kullback–Leibler (KL) divergence [30]. However, the Euclidean distance and KL divergence are too general for activity recognition problem, which ignores the label information of the source domain. Moreover, the KL divergence needs to first estimate the probability of both domains, which is trivial and not suitable since the target domain has no labels. Therefore, we need to calculate the similarity between domains while considering the label information on the source domain.

In order to calculate the function $Dist(\cdot)$ in Eq. (1), we adopt Maximum Mean Discrepancy (MMD) [33] 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:

$$D(\mathcal{D}_s, \mathbf{X}_{can}) = \sum_{c=1}^{C} \left\| \frac{1}{n_s^{(c)}} \sum_{\mathbf{x}_i^s \in \mathcal{D}_s^{(c)}} \phi(\mathbf{x}_i^s) - \frac{1}{n_t^{(c)}} \sum_{\mathbf{x}_j^t \in \mathbf{X}_{can}^{(c)}} \phi(\mathbf{x}_j^t) \right\|_{\mathcal{H}}^2, \tag{5}$$

where $\mathcal{H}$ denotes reproducing kernel Hilbert space (RKHS). $n_s^{(c)} = |\mathcal{D}_s^{(c)}|$, $n_t^{(c)} = |\mathbf{X}_{can}^{(c)}|$. 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].

Therefore, given a predefined mapping function $\phi(\cdot)$, we can compute the intra-class distance between two domains.

Remark. The majority voting and intra-class transfer are common steps in STL. In the next sections, we will elaborate on how to use the STL framework to perform source domain selection and activity transfer.

3.4. Stratified domain selection

In this paper, based on the STL framework, we propose the *Stratified Domain Selection (STL-SDS)* algorithm to select the most similar domains to the target. Based on the proposed *Stratified* distance, STL-SDS well exploits the local property of different domains as well as the supervised information on the source domain.

We adopt a *greedy* technique in STL-SDS. We know that the most similar body part to the target is the one with the most similar structure and body functions. Therefore, we use the distance to reflect their similarity. We calculate the stratified distance according to Eq. (5) between each source domain and the target domain and select the one with the minimal distance.

Unfortunately, it is non-trivial to solve Eq. (5) directly since the mapping function $\phi(\cdot)$ is to be determined. Simply use a certain function will ruin the local property of domains. 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:

$$\mathbf{K}_{ij} = \langle \phi(\mathbf{x}_i^s), \phi(\mathbf{x}_j^t) \rangle = \phi(\mathbf{x}_i^s)^\top \phi(\mathbf{x}_j^t), \tag{6}$$

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

Therefore, we can easily calculate this equation without losing the local property by giving $\phi(\cdot)$ a certain implementation. In our work, we use the Radical Basis Function (RBF), which can be defined as:

$$K(\mathbf{x}_i^s, \mathbf{x}_j^t) = \exp\left(-\frac{\|\mathbf{x}_i^s - \mathbf{x}_j^t\|_2^2}{2\sigma^2} \right), \tag{7}$$

where σ is the kernel bandwidth. The procedure of STL-SDS is in Algorithm 1.

Algorithm 1 STL-SDS: Stratified Domain Selection

Input: A list of source domains $\mathcal{D}_{s_1}, \cdots, \mathcal{D}_{s_K}$, target domain $\mathcal{D}_t$.
Output: The source domain that has the minimal distance to $\mathcal{D}_t$.

```
Initialize a source domain set S = {};
Perform majority voting on D_t to acquire X_can;
Obtain the pseudo labels y_t for D_t;
for i = 1 to K do
    Compute the distance SD_i between D_{s_i} and X_can using Eq. (5);
    Add SD_i to S;
end for
return Index j of source domain that SD_j = min{S}.
```

3.5. Stratified activity transfer

After source domain selection, we can obtain the most similar body part to the target domain. The next step is to design an accurate transfer learning algorithm to perform activity transfer. In this paper, we propose a Stratified Activity Transfer (STL-SAT) method for activity recognition. STL-SAT is also based on our *stratified* distance, and it can simultaneously transform the individual classes of the source and target domains into the same subspaces by exploiting the local property of domains. After feature learning, STL can learn the labels for the candidates. Finally, STL-SAT will perform a second annotation to obtain the labels for the residuals. The process of STL-SAT can be seen in Fig. 4.

The feature transformation is also based on the intra-class transfer technique in Eq. (5). However, different from STL-SDS where we use a certain mapping function such as RBF for feature mapping, we *learn* this mapping $\phi(\cdot)$ in this step.

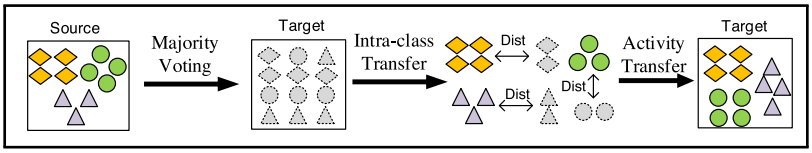


Fig. 4. Main idea of Stratified Activity Transfer (STL-SAT). There are three steps: (1) Candidates generating to generate pseudo labels for the target domain; (2) Perform intra-class transfer between source domain and candidates; (3) Perform activity transfer using the transferred data.

The reason is that STL-SDS only needs one particular feature map to avoid the feature distortion, while STL-SAT will find the *optimal* feature map with the minimal domain distance.

The learning of this feature map also starts with Eq. (5). For efficient learning, we 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. Thus, learning $\phi(\cdot)$ is equal to learning the matrix $\mathbf{W}$. Then, via kernel tricks, Eq. (5) can be formulated as a trace optimization problem:

$$\begin{aligned} \min_{\mathbf{W}} \quad & \sum_{c=1}^{C} \mathrm{tr}(\mathbf{W}^\top \mathbf{K}\mathbf{L}_c\mathbf{K}\mathbf{W}) + \lambda \mathrm{tr}(\mathbf{W}^\top \mathbf{W}) \\ \text{s.t.} \quad & \mathbf{W}^\top \mathbf{K}\mathbf{H}\mathbf{K}\mathbf{W} = \mathbf{I} \end{aligned} \tag{8}$$

There are two terms in the objective function of Eq. (8). The first term ($\sum_{c=1}^{C} \mathrm{tr}(\mathbf{W}^\top \mathbf{K}\mathbf{L}_c\mathbf{K}\mathbf{W})$) denotes the MMD distance of each class between source and target domain, and the second one ($\lambda \mathrm{tr}(\mathbf{W}^\top \mathbf{W})$) denotes the regularization term to ensure the problem is well-defined with λ the trade-off parameter. The constraint in Eq. (8) is used to guarantee that the transformed data ($\mathbf{W}^\top \mathbf{K}$) will still preserve some structure property of the original data. $\mathbf{I}_{n_s+n_t}$ is the identical matrix, and $\mathbf{H} = \mathbf{I}_{n_s+n_t} - 1/(n_s+n_t)\mathbf{1}\mathbf{1}^\top$ is the centering matrix. For notational brevity, we will drop the subscript for $\mathbf{I}_{n_s+n_t}$ in the sequel. $\mathbf{L}_c$ is the intra-class MMD matrix:

$$(\mathbf{L}_c)_{ij} = \begin{cases} \frac{1}{(n_s^{(c)})^2} & \mathbf{x}_i, \mathbf{x}_j \in \mathcal{D}_s^{(c)} \\ \frac{1}{(n_t^{(c)})^2} & \mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}_{can}^{(c)} \\ -\frac{1}{n_s^{(c)} n_t^{(c)}} & \begin{cases} \mathbf{x}_i \in \mathcal{D}_s^{(c)}, \mathbf{x}_j \in \mathbf{X}_{can}^{(c)} \\ \mathbf{x}_i \in \mathbf{X}_{can}^{(c)}, \mathbf{x}_j \in \mathcal{D}_s^{(c)} \end{cases} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

Learning algorithm: Acquiring the solution of Eq. (8) is non-trivial. To this end, we adopt Lagrange method as most of the existing work did [8,10,13,34]. We denote Φ the Lagrange multiplier, then the Lagrange function can be derived as

$$\begin{aligned} L = & \ \mathrm{tr}\left(\mathbf{W}^\top \mathbf{K} \sum_{c=1}^{C} \mathbf{L}_c \mathbf{K}^\top \mathbf{W}\right) + \lambda \mathrm{tr}(\mathbf{W}^\top \mathbf{W}) \\ & + \mathrm{tr}\left((\mathbf{I} - \mathbf{W}^\top \mathbf{K}\mathbf{H}\mathbf{K}^\top \mathbf{W})\Phi\right) \end{aligned} \tag{10}$$

Setting the derivative $\partial L/\partial \mathbf{W} = 0$, Eq. (10) can be finally formalized as an generalized eigen-decomposition problem

$$\left(\mathbf{K} \sum_{c=1}^{C} \mathbf{L}_c \mathbf{K}^\top + \lambda \mathbf{I}\right) \mathbf{W} = \mathbf{K}\mathbf{H}\mathbf{K}^\top \mathbf{W}\Phi \tag{11}$$

Eq. (11) can be easily solved as a 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 transformed into the same subspaces.

After solving the above optimization problem, 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 $\{\mathbf{X}_{can}, \hat{\mathbf{y}}_{can}\}$. 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. The procedure of STL-SAT is in Algorithm 2.

Algorithm 2 STL-SAT: Stratified Activity Transfer

Input: Source domain $\mathcal{D}_s$, target domain $\mathcal{D}_t$, dimension m.
Output: The labels for the target domain: $\{\mathbf{y}_t\}$.
Majority voting on $\mathcal{D}_t$ using Eq. (3) to get $\{\mathbf{X}_{can}, \widetilde{\mathbf{y}}_{can}\}$ and $\mathbf{X}_{res}$;
Construct kernel matrix $\mathbf{K}$ according to Eq. (6) using $\mathbf{X}_{src}$ and $\mathbf{X}_{can}$, and compute the intra-class MMD matrix $\mathbf{L}_c$ using Eq. (9);
repeat
  Solve the eigen-decomposition problem in Eq. (11) and take the m smallest eigen-vectors to obtain the transformation matrix $\mathbf{W}$;
  Transform the same classes of $\mathbf{X}_s$ and $\mathbf{X}_{can}$ into the same subspaces using $\mathbf{W}$, and then merge them;
  Perform second annotation to get $\{\hat{\mathbf{y}}_{can}\}$ and $\{\hat{\mathbf{y}}_{res}\}$;
  Construct $\mathbf{K}$ and compute intra-class MMD matrix $\mathbf{L}_c$ using Eq. (9);
until Convergence
return $\{\mathbf{y}_t\}$.

Table 1
Statistical information of three public datasets for activity recognition.

Dataset	Person	Activity	Sample	Position
OPP	4	4	701K	Back (B), Right Upper Arm (RUA), Right Left Arm (RLA), Left Upper Arm (LUA), Left Lower Arm (LLA)
PAMAP	9	18	284M	Hand (H), Chest(C), Ankle (A)
DSADS	8	19	114M	Torso (T), Right Arm (RA), Left Arm (LA), Right Leg (RL), Left Leg (LL)
MHEALTH	10	12	66M	Chest (C), Right Wrist (RW) and Left Ankle (LAn)

4. Experiments

In this section, we evaluate the performances of STL framework (STL-SDS and STL-SAT) via extensive experiments on public HAR datasets.

4.1. Datasets and preprocessing

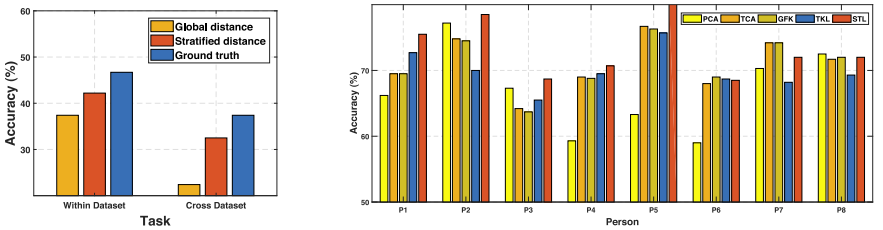
Four large public datasets are adopted in experiments. Table 1 provides a brief introduction to them. In the following, we briefly introduce those datasets, and more information can be found in their original papers. OPPORTUNITY dataset (**OPP**) [35] is composed of 4 subjects executing different levels of activities with sensors tied to more than 5 body parts. PAMAP2 dataset (**PAMAP**) [36] is collected by 9 subjects performing 18 activities with sensors on 3 body parts. UCI daily and sports dataset (**DSADS**) [37] consists of 19 activities collected from 8 subjects wearing body-worn sensors on 5 body parts. **MHEALTH** [38] is composed of 10 subjects performing 12 kinds of activities. Accelerometer, gyroscope, and magnetometer are all used in these 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 [39]. In total, 27 features from both time and frequency domains are extracted for a single sensor. 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 better exploit these three datasets, we perform two aspects of cross-position activity recognition: *Within Dataset* and *Cross dataset*. *Within Dataset* refers to perform CPAR inside a particular dataset. For instance, we can learn the labels for the Right Arm (RA) of the DSADS dataset by using labeled data from the other four body parts. In contrast, *Cross dataset* uses the labeled body parts from another different dataset as the source domains. In our experiments, we use DSADS and OPP datasets for *Within dataset* since they contain more body parts than PAMAP. We use all classes for each dataset in this aspect. For *Cross Dataset*, we use the body parts from DSADS and OPP as the target domain, and use body parts from PAMAP as the source domain since there are more samples in PAMAP than other two datasets (Table 1). Note that there are different activities in three datasets. Thus we extract 4 common classes: *Walking, Sitting, Lying*, and *Standing*.

4.2. Evaluation of stratified domain selection

We perform source domain selection using the proposed STL-SDS (i.e. the stratified distance (SD)) algorithm on both the *Within Dataset* and *Cross Dataset* aspects. The comparison method is the *global* distance (GD) using the MMD. It is worth noting that there is no effective evaluation metric for this domain selection problem. We can *never* quantitatively know the actual distance between two activity domains. Thus, we evaluate the similarity based on *classification accuracy*



(a) Performance of STL-SDS (b) Case study of STL-SAT

Fig. 5. (a) The comparison between global and stratified distance for domain selection. (b) Classification accuracy of different methods of each person on RA → LA task.

Table 2
Comparison of Global (G) and Stratified (S) distance on source domain selection. The symbol *item* denotes that the item is *not* consistent with the ground truth.

Target dataset		DSADS					OPP				
Target position		T	RA	LA	RL	LL	B	RUA	RLA	LUA	LLA
Cross dataset	**Truth**	RA	LA	RA	LL	RL	LUA	RLA	RUA	B	LUA
	GD	*LA*	*T*	*T*	LL	RL	LUA	*LLA*	RUA	*LLA*	*RUA*
	SD	*LA*	LA	*T*	LL	RL	LUA	RLA	RUA	*LLA*	*RUA*
Cross dataset	**Truth**	Chest	Ankle	Chest	Hand	Hand	Chest	Chest	Chest	Chest	Chest
	GD	Chest	*Chest*	Chest	*Chest*	*Chest*	Chest	*Ankle*	*Ankle*	*Ankle*	*Ankle*
	SD	Chest	Ankle	Chest	Hand	*Chest*	Chest	Chest	*Ankle*	Chest	Chest

Table 3
Accuracy (%) of Global (GD) and Stratified (SD) distance on domain selection.

Target dataset		DSADS					OPP					Average
Target position		T	RA	LA	RL	LL	B	RUA	RLA	LUA	LLA	–
Target as source		87.6	83.4	84.8	87.9	88.4	37	37.3	36.7	34.8	38.3	61.6
Within dataset	**Truth**	35.0	69.5	66.9	69.9	77.3	30.9	33.2	32.1	24.6	27.8	**46.7**
	GD	31.5	34.1	36.0	69.9	77.3	30.9	20.3	32.1	22.7	19.6	37.4
	SD	31.5	69.5	36.0	69.9	77.3	30.9	33.2	32.1	22.7	19.6	**42.2**
Cross dataset	**Truth**	23.8	25.0	24.8	25.0	25.0	50.1	50.0	49.9	50.0	50.0	**37.4**
	GD	23.8	0.6	24.8	2.7	2.0	50.1	24.5	24.5	47.3	23.6	22.4
	SD	23.8	25.0	24.8	25.0	2.0	50.1	50.0	24.5	50.0	50.0	**32.5**

using the same classifier following existing work [29]. For instance, for source domains *A* and *B* and target domain *C*, we respectively train a linear SVM classifier using *A* or *B* and apply prediction on *C*. The accuracy acts as the ground *truth* for domain similarity: higher accuracy means shorter distance. Then, the source domain with the highest accuracy is the *right* source for the target domain. Therefore, we can obtain the 'ground truth' for the source domain selection.

Fig. 5(a) shows the average source domain selection accuracy of the global and stratified distance. It is obvious that our proposed SD distance achieves better performance than the traditional global distance. Note that in both *Within Dataset* and *Cross Dataset* scenarios, SD distance outperforms GD. Furthermore, for the even challenging *Cross Dataset* distance where the similarity between the source and the target domains are little, SD could significantly outperform the traditional GD distance. This indicates the effectiveness of our proposed STL-SDS algorithm.

Now we dig deeper into the results. Table 2 shows the selected source domains of GD and SD, and their accuracies are in Table 3. We also report the accuracy when target itself is as the source domain for comparison (which is the ideal state that can never be satisfied). From these tables, we observe: (1) The proposed Stratified distance can select better source domain than the traditional global distance with close performance with the ground truth. (2) The situation when the target domain as the source domain can achieve the best performance. However, it is only the ideal state since there are always the label scarcity problems. It indicates that CPAR is a challenging task since both of the *Within Dataset* and *Cross Dataset* aspects only achieve worse performance, which ensures the necessity of transfer learning algorithm. (2) Generally speaking, the most similar body parts to a side Arm or Leg is its other side. This observation is much easier to understand as common sense. (3) Torso (or Back/Chest) is the most similar body part to other body parts such as Arms and Legs. This is probably because the Torso is physically connected to Arms and Legs, leading to similar moving patterns. Therefore,

Table 4
Classification accuracy (%) and F1 score of STL-SAT and comparison methods.

Dataset		Task	PCA	KPCA	TCA	GFK	TKL	STL-SAT
Within dataset	DSADS	RA → LA	59.9 (0.72)	62.2 (0.72)	66.2 (0.76)	71.0 (0.73)	54.1 (0.56)	**71.1 (0.81)**
		RL → LL	69.5 (0.72)	70.9 (0.73)	75.1 (0.77)	79.7 (0.73)	61.6 (0.67)	**81.6 (0.69)**
		RA → T	38.9 (0.62)	30.2 (0.62)	39.4 (0.56)	44.2 (0.62)	32.7 (0.58)	**45.6 (0.77)**
	OPP	RUA → LUA	76.1 (0.72)	65.6 (0.73)	76.9 (0.79)	74.6 (0.73)	66.8 (0.68)	**84.0 (0.80)**
		RLA → LLA	62.2 (0.60)	66.5 (0.61)	60.6 (0.68)	74.6 (0.62)	66.8 (0.55)	**83.9 (0.70)**
		RLA → T	**59.1** (0.47)	47.0 (0.47)	55.4 (0.56)	48.9 (0.48)	47.7 (0.49)	56.9 **(0.59)**
		RUA → T	68.0 (0.64)	54.5 (0.67)	67.5 (0.70)	66.1 (0.66)	60.5 (0.58)	**75.2 (0.75)**
	PAMAP	H → C	35.0 (0.24)	24.4 (0.25)	34.9 (0.27)	36.2 (0.25)	35.7 (0.26)	**43.5 (0.33)**
	MHEALTH	C → RW	40.5 (0.41)	41.2 (0.43)	43.2 (0.46)	45.6 (0.52)	44.2 (0.47)	**48.7 (0.54)**
Cross dataset	PAMAP → OPP	C → B	32.8 (0.32)	43.8 (0.42)	39.0 (0.37)	27.6 (0.27)	35.6 (0.27)	**40.1 (0.44)**
	DSADS → P	T → C	23.2 (0.22)	18.0 (0.20)	23.7 (0.25)	19.4 (0.24)	21.7 (0.23)	**37.8 (0.39)**
	OPP → D	B → T	44.3 (0.46)	49.4 (0.50)	46.9 (0.48)	48.1 (0.50)	52.8 (0.54)	**55.5 (0.59)**
	MHEALTH → PAMAP	C → C	45.2 (0.46)	46.1 (0.48)	44.9 (0.49)	46.7 (0.52)	46.2 (0.49)	**49.3 (0.55)**
	MHEALTH → OPP	B → C	39.8 (0.39)	42.1 (0.41)	43.2 (0.42)	44.5 (0.46)	44.4 (0.42)	**46.8 (0.49)**
	MHEALTH → DSADS	C → T	42.3 (0.44)	42.8 (0.43)	44.5 (0.45)	44.9 (0.48)	43.1 (0.46)	**46.2 (0.51)**
Average			49.1	47.0	50.7	51.5	47.6	**57.8**

most of the target domains can leverage the labeled information from Torso to build models. (4) It is important to notice that although SD achieves good results, its performance is not 100% right. This indicates that it is extremely difficult to perform source selection. We expect to increase the performance of SD in future research.

4.3. Evaluation of stratified activity transfer

We evaluate the performance of STL-SAT in both *Within Dataset* and *Cross Dataset* aspects. The state-of-the-art comparison methods include: (1) PCA: Principal component analysis [40], (2) KPCA: Kernel principal component analysis [40], (3) TCA: Transfer component analysis [8], (4) GFK: Geodesic flow kernel [9], and (5) TKL: Transfer kernel learning [10]. PCA and KPCA are classic dimensionality reduction methods, while TCA, GFK, and TKL are representative transfer learning approaches.

We construct several CPAR tasks according to each scenario and use the labels for the target domain only for testing by following the common setting in existing work [5,26]. 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 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-SAT, 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 number is set to be $T = 10$ for SAT. Other parameters of STL-SAT are set $\lambda = 1.0$, $d = 30$, $kernel = linear$. It is noticeable that we randomly shuffle the experimental data 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 [8,10]

$$Accuracy = \frac{|\mathbf{x} : \mathbf{x} \in \mathcal{D}_t \wedge \hat{y}(\mathbf{x}) = y(\mathbf{x})|}{|\mathbf{x} : \mathbf{x} \in \mathcal{D}_t|} \tag{12}$$

where $y(\mathbf{x})$ and $\hat{y}(\mathbf{x})$ are the truth and predicted labels, respectively. Additionally, we also use the F1 score as another measurement of the results. F1 score can be calculated as

$$F1 = \frac{2PR}{P + R} \tag{13}$$

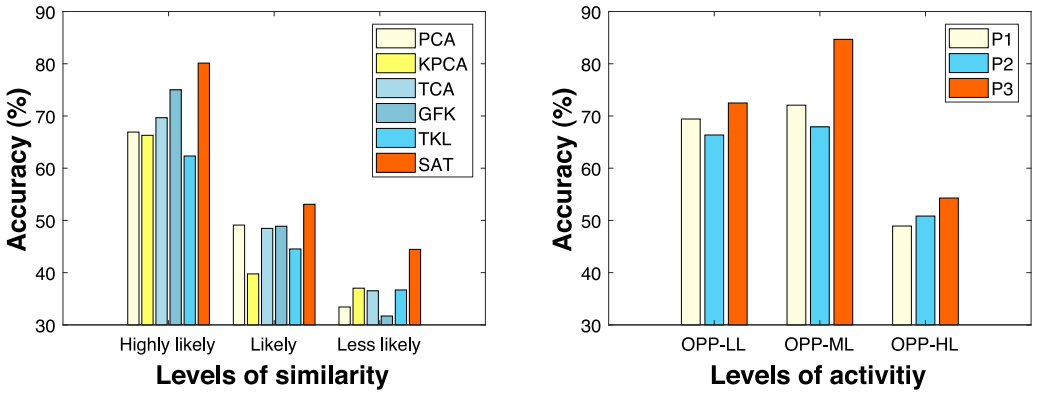
where P, R are the precision and recall, respectively.

We run STL-SAT and other methods on all tasks and report the classification accuracy and F1 score in Table 4. For brevity, we only report the results of $A \rightarrow B$ since its result is close to $B \rightarrow A$. It is obvious that SAT significantly outperforms other methods in most cases (with a remarkable improvement of **6.3**% over the best baseline GFK). Compared to traditional dimensionality reduction methods (PCA and KPCA), STL-SAT improves the accuracy by 10%∼20%, which implies that SAT is better than typical dimensionality reduction methods. Compared to transfer learning methods (TCA, GFK, and TKL), STL-SAT still shows an improvement of 5%∼15%. Therefore, STL-SAT 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 SAT, 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-SAT and GFK are: (1) STL-SAT outperforms GFK in most cases with

Table 5
Confusion matrix on RUA → LUA of OPP dataset.

PCA	Acc = 72.5				KPCA	Acc = 74.5				TCA	Acc = 76.2			
	Walking	Sitting	Lying	Standing		Walking	Sitting	Lying	Standing		Walking	Sitting	Lying	Standing
Walking	**1426**	144	445	18	Walking	**1456**	134	435	8	Walking	**1423**	155	445	10
Sitting	57	**492**	6	0	Sitting	57	**492**	6	0	Sitting	152	**403**	0	0
Lying	278	10	**1013**	34	Lying	278	10	**1013**	34	Lying	369	4	**936**	26
Standing	38	1	54	**121**	Standing	38	1	54	**121**	Standing	65	0	30	**119**
GFK	Acc = 74.3				TKL	Acc = 67.8				STL	Acc = 78.6			
	Walking	Sitting	Lying	Standing		Walking	Sitting	Lying	Standing		Walking	Sitting	Lying	Standing
Walking	**1449**	141	435	8	Walking	**1423**	157	443	10	Walking	**1658**	67	303	5
Sitting	57	**492**	6	0	Sitting	114	**430**	11	0	Sitting	53	**496**	6	0
Lying	272	12	**1015**	36	Lying	397	26	**830**	82	Lying	357	6	**912**	60
Standing	37	1	57	**119**	Standing	50	1	40	**123**	Standing	33	0	58	**123**



(a) Similarity (b) Activity level

Fig. 6. Classification accuracy of (a) different degrees of similarities and (b) different levels of activities in transfer learning.

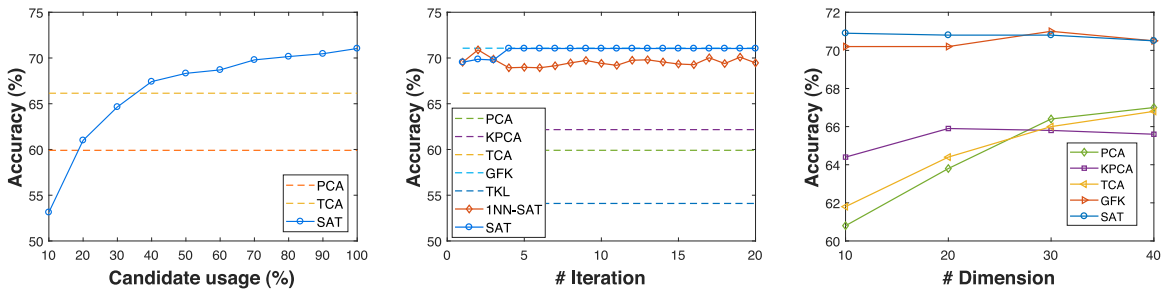
significant improvement; (2) STL-SAT strongly outperforms GFK in *Within Dataset* category, indicating that SAT is more robust in recognizing different levels of activities. For SAT, it performs intra-class knowledge transfer after generating pseudo labels for candidates. Thus, better performance can be achieved by exploiting the local property of classes.

We further did a case study by showing the confusion matrices of all methods for task RUA → LUA on OPP dataset in Table 5 and the performance of each person in Fig. 5(b). The confusion matrix helps to analyze which class is easier to classify. From the results, we can observe that in CPAR, all the classes are easy to be wrongly classified since there is distribution divergence between domains. Of all the classes, *Walking* and *Standing* are two classes that are more easily to be misclassified. In this situation, our proposed STL-SAT could significantly outperform other comparison methods. In other situations, STL-SAT can still obtain comparable performance. Overall, STL-SAT achieves the best classification accuracy. On other tasks, the results are almost the same. It indicates the superiority of STL-SAT. We also noticed that on some persons (P6, P7, and P8), the performance of STL-SAT is worse. This may be because the sensor data of these three persons are a little mixed during the activity transition. We will explore more fine-grained classes for future research.

4.4. Performance on similarity and activity level

We explore the performance on different similarity and activity levels in Fig. 6. For similarity level, we use *Highly Likely* to denote the similar body parts such as RA → LA, *Likely* to denote the dissimilar body parts such as RA → T, and *Less Likely* to denote the *Within Dataset* for notational consistency. For activity level, we analyze low-level (*OPP-LL*), middle-level (*OPP-ML*), and high-level (*OPP-HL*) activities.

Firstly, the accuracies of all methods drop as the domain similarity becomes less and activity level becomes high. Additionally, the performance of STL-SAT is the best in all scenarios. For a different person, all methods produce the worst results because different people have exactly different body structure and moving patterns (e.g. T → T across datasets). At the same degree of similarity, the results are also different. For example, the performance of RUA → T is better than RLA → 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 a similar discovery.



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

Fig. 7. (a) Candidates usage of STL-SAT for RA → LA on DSADS dataset; (b) Convergence of 1NN-SAT and STL-SAT; (c) Parameter sensitivity.

Secondly, 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. 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.

4.5. Effectiveness analysis

In this section, we verify the effectiveness of STL-SAT in several aspects.

(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-SAT on the task LA → RA on DSADS and compare with other 2 methods (PCA and TCA). The result is shown in Fig. 7(a). From the results, we can observe that the performance of STL-SAT is increasing along with the increment of candidates percentage. More importantly, STL-SAT outperforms the other two methods with **less than** 40% of candidates. It reveals that SAT 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: We choose 1 nearest neighbor (1NN) instead of three classifiers as the base classifier. The results are in Fig. 7(b), where '1NN-SAT' and 'SAT' denotes the result of STL-SAT using 1NN and random forest as the base majority voting classifier, respectively. From those results, we can observe: (1) STL-SAT can iteratively improve the classification accuracy even with some weak majority voting classifier. 1NN-SAT achieves slightly worse than STL-SAT, indicating that STL-SAT is rather *robust* to the base classifiers. Since more powerful classifiers would lead to better performance, we strongly suggest using more reliable classifiers. (2) STL-SAT can reach a quick convergence within *fewer than* 10 iterations. This indicates that STL-SAT can be efficiently trained.

(3) Parameter sensitivity. STL-SAT involves two parameters: the dimension m, and trade-off parameter λ. In this experiment, we evaluate the sensitivity of m. We set $m \in \{10, 20, 30, 40\}$ and test the performance of SAT and other dimensionality reduction methods. As shown in Fig. 7(c), STL-SAT achieves the best accuracy under different dimensions. Meanwhile, the accuracy of STL-SAT almost does not change with the decrement of m. The results reveal that STL-SAT is much more effective and robust than other methods under different dimensions.

5. Conclusions and future work

In this paper, we propose a novel Stratified Transfer Learning (STL) framework for cross-domain HAR. STL can address both the source domain selection (by STL-SDS) and knowledge transfer (by STL-SAT) problems in CPAR by exploiting the local property of each class between different domains. Experiments on three large public datasets demonstrate the significant superiority of STL over five state-of-the-art methods. We extensively analyze the performance of transfer learning under different degrees of similarities and different levels of activities.

In the future, we plan to extend STL in the following research directions:

Deep STL. The biggest difference between this paper and the deep version is we can exploit deep networks to automatically extract features and then perform domain selection and activity transfer. All these steps can be operated in one neural network.

Heterogeneous STL. In heterogeneous HAR, the features for different domains may be different. For instance, if the source domain contains activity information of the accelerometer and the target domain has information of the gyroscope, we need to develop heterogeneous algorithms accordingly.

Acknowledgments

This work is supported in part by National Key R & D Plan of China (No. 2017YFB1002802), NSFC (No. 61572471, 61472399, 61702520), and Beijing Municipal Science & Technology Commission (No. Z171100000117017).

References

[1] N.Y. Hammerla, J. Fisher, P. Andras, L. Rochester, R. Walker, T. Plötz, Pd disease state assessment in naturalistic environments using deep learning, in: AAAI, 2015, pp. 1742–1748.
[2] T. Plötz, N.Y. Hammerla, P. Olivier, Feature learning for activity recognition in ubiquitous computing, in: IJCAI, Vol. 22, 2011, p. 1729.
[3] J.M. Wardlaw, E.E. Smith, et al., Neuroimaging standards for research into small vessel disease and its contribution to ageing and neurodegeneration, Lancet Neurol. 12 (8) (2013) 822–838.
[4] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359.
[5] D. Cook, K.D. Feuz, N.C. Krishnan, Transfer learning for activity recognition: A survey, Knowl. Inf. Syst. 36 (3) (2013) 537–556.
[6] J. Blitzer, R. McDonald, F. Pereira, Domain adaptation with structural correspondence learning, in: EMNLP, 2006, pp. 120–128.
[7] W.M. Kouw, L.J. van der Maaten, J.H. Krijthe, M. Loog, Feature-level domain adaptation, J. Mach. Learn. Res. 17 (171) (2016) 1–32.
[8] S.J. Pan, I.W. Tsang, J.T. Kwok, Q. Yang, Domain adaptation via transfer component analysis, IEEE Trans. Neural Netw. 22 (2) (2011) 199–210.
[9] B. Gong, Y. Shi, F. Sha, K. Grauman, Geodesic flow kernel for unsupervised domain adaptation, in: CVPR, 2012, pp. 2066–2073.
[10] M. Long, J. Wang, J. Sun, S.Y. Philip, Domain invariant transfer kernel learning, IEEE Trans. Knowl. Data Eng. 27 (6) (2015) 1519–1532.
[11] Y. Lin, J. Chen, Y. Cao, Y. Zhou, L. Zhang, Y.Y. Tang, S. Wang, Cross-domain recognition by identifying joint subspaces of source domain and target domain, IEEE Trans. Cybern. (2016).
[12] E. Elhamifar, R. Vidal, Sparse subspace clustering: Algorithm, theory, and applications, IEEE Trans. Pattern Anal. Mach. Intell. 35 (11) (2013) 2765–2781.
[13] J. Wang, Y. Chen, L. Hu, X. Peng, P.S. Yu, Stratified transfer learning for cross-domain activity recognition, in: PerCom, 2018.
[14] A. Bulling, U. Blanke, B. Schiele, A tutorial on human activity recognition using body-worn inertial sensors, ACM Comput. Surv. 46 (3) (2014) 33.
[15] V.W. Zheng, Q. Yang, User-dependent aspect model for collaborative activity recognition, in: IJCAI, Vol. 22(3), 2011, pp. 2085–2090.
[16] Y. Chen, Y. Gu, X. Jiang, J. Wang, Ocean: A new opportunistic computing model for wearable activity recognition, in: UbiComp Adjunct, ACM, 2016, pp. 33–36.
[17] H.S. Hossain, N. Roy, M.A.A.H. Khan, Active learning enabled activity recognition, in: IEEE PerCom Conference, 2016, pp. 1–9.
[18] W.S. Lasecki, Y.C. Song, H. Kautz, J.P. Bigham, Real-time crowd labeling for deployable activity recognition, in: CSCW, ACM, 2013, pp. 1203–1212.
[19] L.T. Nguyen, M. Zeng, P. Tague, J. Zhang, I did not smoke 100 cigarettes today!: avoiding false positives in real-world activity recognition, in: UbiComp, ACM, 2015, pp. 1053–1063.
[20] L. Hu, Y. Chen, S. Wang, J. Wang, J. Shen, X. Jiang, Z. Shen, Less annotation on personalized activity recognition using context data, in: UIC, 2016, pp. 327–332.
[21] S.J. Pan, J.T. Kwok, Q. Yang, Transfer learning via dimensionality reduction, AAAI, Vol. 8, 2008, pp. 677–682.
[22] F. Dorri, A. Ghodsi, Adapting component analysis, in: ICDM, IEEE, 2012, pp. 846–851.
[23] C.-W. Seah, I.W.-H. Tsang, Y.-S. Ong, Q. Mao, Learning target predictive function without target labels, in: ICDM, IEEE, 2012, pp. 1098–1103.
[24] X. Glorot, A. Bordes, Y. Bengio, Domain adaptation for large-scale sentiment classification: A deep learning approach, in: ICML, 2011, pp. 513–520.
[25] M. Gong, K. Zhang, T. Liu, D. Tao, C. Glymour, B. Schölkopf, Domain adaptation with conditional transferable components, in: ICML, 2016, pp. 2839–2848.
[26] Z. Zhao, Y. Chen, J. Liu, Z. Shen, M. Liu, Cross-people mobile-phone based activity recognition, in: IJCAI, Vol. 11, 2011, pp. 2545–2550.
[27] M.A.A.H. Khan, N. Roy, Transact: Transfer learning enabled activity recognition, in: PerCom Workshops, IEEE, 2017, pp. 545–550.
[28] K.D. Feuz, D.J. Cook, Collegial activity learning between heterogeneous sensors, Knowl. Inf. Syst. (2017) 1–28.
[29] J. Wang, V.W. Zheng, Y. Chen, M. Huang, Deep transfer learning for cross-domain activity recognition, in: ICCSE, 2018, p. 16.
[30] J. Wang, Y. Chen, S. Hao, W. Feng, Z. Shen, Balanced distribution adaptation for transfer learning, 2017, in: ICDM, pp. 1129–1134.
[31] M. Long, J. Wang, G. Ding, J. Sun, P.S. Yu, Transfer feature learning with joint distribution adaptation, in: CVPR, 2013, pp. 2200–2207.
[32] D. Prelec, H.S. Seung, J. McCoy, A solution to the single-question crowd wisdom problem, Nature 541 (7638) (2017) 532–535.
[33] A. Gretton, K.M. Borgwardt, M.J. Rasch, B. Schölkopf, A. Smola, A kernel two-sample test, J. Mach. Learn. Res. 13 (Mar) (2012) 723–773.
[34] J. Zhang, W. Li, P. Ogunbona, Joint geometrical and statistical alignment for visual domain adaptation, 2017, arXiv preprint arXiv:1705.05498.
[35] R. Chavarriaga, H. Sagha, A. Calatroni, S.T. Digumarti, G. Tröster, J.d.R. Millán, D. Roggen, The opportunity challenge: A benchmark database for on-body sensor-based activity recognition, Pattern Recognit. Lett. 34 (15) (2013) 2033–2042.
[36] A. Reiss, D. Stricker, Introducing a new benchmarked dataset for activity monitoring, in: ISWC, IEEE, 2012, pp. 108–109.
[37] B. Barshan, M.C. Yüksek, Recognizing daily and sports activities in two open source machine learning environments using body-worn sensor units, Comput. J. 57 (11) (2014) 1649–1667.
[38] O. Banos, R. Garcia, J.A. Holgado-Terriza, M. Damas, H. Pomares, I. Rojas, A. Saez, C. Villalonga, mHealthDroid: a novel framework for agile development of mobile health applications, in: International Workshop on Ambient Assisted Living, Springer, 2014, pp. 91–98.
[39] L. Hu, Y. Chen, J. Wang, et al., Okrelm: online kernelized and regularized extreme learning machine for wearable-based activity recognition, Int. J. Mach. Learn. Cybern. (2017) 1–14.
[40] I.K. Fodor, A Survey of Dimension Reduction Techniques, Vol. 9, Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, 2002, pp. 1–18.