Pervasive and Mobile Computing 7 (2011) 344–358

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

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

Cross-domain activity recognition via transfer learning☆

Derek Hao Hu *, Vincent Wenchen Zheng, Qiang Yang [1]

Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong

ARTICLE INFO

Article history:
Available online 13 December 2010

Keywords:
Activity recognition
Transfer learning
Web mining

ABSTRACT

In activity recognition, one major challenge is how to reduce the labeling effort one needs to make when recognizing a new set of activities. In this paper, we analyze the possibility of transferring knowledge from the available labeled data on a set of existing activities in one domain to help recognize the activities in another different but related domain. We found that such a knowledge transfer process is possible, provided that the recognized activities from the two domains are related in some way. We develop a bridge between the activities in two domains by learning a similarity function via Web search, under the condition that the sensor readings are from the same feature space. Based on the learned similarity measure, our algorithm interprets the data from the source domain as "pseudo training data" in the target domain with different confidence levels, which are in turn fed into supervised learning algorithms for training the classifier. We show that after using this transfer learning approach, the performance of activity recognition in the new domain is increased several fold as compared to when no knowledge transfer is done. Our algorithm is evaluated on several real-world datasets to demonstrate its effectiveness. In the experiments, our algorithm could achieve a 60% accuracy most of the time with no or very few training data in the target domain, which easily outperforms the supervised learning methods.

1. Introduction

With the proliferation of sensor technologies, the task of recognizing human activities from a series of low-level sensor readings has drawn much interest in AI, user modeling and ubiquitous computing areas [1,2]. Early activity recognition algorithms are based on logical reasoning, in which conclusions are deducted from the observations and "closed world" assumptions [3]. As the sensor data became available, recent activity recognition research focuses more on reasoning under uncertainty. For example, Bui introduced abstract Hidden Markov Models to represent the user's activity hierarchy [4]. Liao et al. applied a hierarchical Markov model to estimate a user's locations and transportation modes [5]. Yin et al. applied a dynamic Bayesian network to infer a user's actions from raw WiFi signals, and an N-gram model to infer the users' high-level goals from actions [6]. Similarly, Patterson et al. also applied a dynamic Bayesian network to recognize fine-grained activities by aggregating the abstract object's usage [7]. Hu and Yang used a skip-chain Conditional Random Field and an activity correlation graph to model the concurrent and interleaving activities [8,9]. In a recent work [10], Hidden Markov Models with an infinite state space were also used to detect abnormal activities from sensor readings.

☆ This paper is an extension of the paper which was published in Ubicomp 2009 with the title "Cross-Domain Activity Recognition".

* Corresponding author.
E-mail addresses: derekhh@cse.ust.hk (D.H. Hu), vincentz@cse.ust.hk (V.W. Zheng), qyang@cse.ust.hk (Q. Yang).
[1] Tel.: +852 23588768; fax: +852 23581477.

1574-1192/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.pmcj.2010.11.005

Fig. 1. An example of cross-domain activity recognition.

Although much work has been done in activity recognition, a major challenge still remains: given a new domain of activities: it usually requires a lot of human effort to label the sensor data for training a recognition model. Data labels are usually meaningful activity terms associated with the data vectors, such as the sensor readings. If the labeled data for the target activities are few, then the trained activity recognizer may not perform well. Such an issue has become a critical challenge for applying activity recognition systems in practice. In the real world, users may only have a small amount of time and effort to set up an activity recognition system; otherwise, they are quite likely to quit using such a recognition system. Furthermore, users usually do not have the expertise in activity recognition research. Therefore when we try to design some activity recognition algorithm, we should make the algorithm as simple as possible. In terms of data labeling effort, we wish to ensure that the users need not label all activities; if they have done some labeling, it is best for our system to automatically *transfer* the labeled knowledge to help recognize other different, but related activities. In this paper, we show how to transfer the available labeled data from a set of activities to help train a recognizer for another set of different, but related activities.

Consider an example in Fig. 1. The taxonomy is an activity taxonomy extracted from the MIT PLIA1 dataset [11,8], representing the common daily activities. Suppose that some user wants to set up an activity recognition system at his/her home to recognize the activities in the taxonomy. However, the user only wants to spend a very little amount of time and effort to label the sensor data, where labels correspond to activity names such as "washing dishes". A user may not be able to label the sensor data associated with all activities described in the taxonomy. For example, the user may only label the sensor data from the activities in the "Cleaning Indoors" category, and leave unlabeled the sensor data from the other categories' activities (e.g. in "Laundry", "Dishwashing"). So in our problem, we have a *source domain* of activities that has the labeled sensor data from activities in the "Cleaning Indoors" category. We also have some *target domain* that has the unlabeled sensor data from the activities in some other category such as "Laundry" (denoted as "Target Domain 1") or "Dishwashing" (denoted as "Target Domain 2"). Then, we ask the following fundamental questions:

1. *Is it possible for us to use the labeled data in the source domain to help train an activity recognizer in the target domain?* For example, can we use the sensor data from the "Cleaning Indoors" category in training an activity recognizer for the "Laundry" category?
2. *Under what conditions can domain transfer work for activity recognition?*

In this paper, we answer the above questions by presenting a novel algorithm for cross-domain activity recognition (CDAR), which can transfer the labeled data from a source domain to a target domain under the condition that (1) the

activities in the source and target domains are related by some Web pages and thus we can build a mapping between them using the Web, and (2) the underlying feature spaces are identical between the source and target domains (they use the same set of sensors, although activity labels can be different). We observe that although the activities in the source domain and the target domain are different, some of them are similar in semantics as well as the corresponding sensor data. For example, in the above Fig. 1, one may transfer useful information from activity "Washing-laundry" to "Hand-washing dishes", considering that, although these two activities are different, the underlying physical actions one performs for these two activities are similar, i.e. hand-washing. Therefore, if we have sensors attached to the body arms or hands, the accelerometer values or other motion values detected should be similar for us to transfer the knowledge from the source domain to the target domain. Intuitively, we use similar activities in the source domain to help enrich the labeled data for the target domain. Specifically, we first learn a similarity function between activities in both domains by exploiting Web search and applying information retrieval techniques. We then train a (multi-class) weighted Support Vector Machine (SVM) model with different probabilistic confidence weights learned from the similarity function. Experiments on real world data sets show that the transferred activity recognizer can indeed improve performance by using the auxiliary data and outperform some other state-of-the-art algorithms.

2. Related work

In this section we review some related research work. We first briefly review some papers in activity recognition. Since our problem is related to transfer learning, we also review some transfer learning papers as well as some recent papers in activity recognition which also exploit transfer learning ideas.

2.1. Activity recognition

Activity recognition aims to infer a user's behavior from observations such as sensor data, and has various applications including medical care [12], logistics service [13], robot soccer [14], plan recognition [15], etc. However, most of the proposed activity recognition algorithms are focused only on sensor readings from only one domain, and usually require lots of annotated data to train the activity recognition model.

There has been lots of progress in recognizing activities in the past twenty to thirty years. As early as 1987, Kautz developed a formal theory to track the logical consistency of observed activities for plan recognition [3]. Many graphical models have now been adopted to model the action sequences. For example, [16] built a hierarchical Dynamic Bayesian Network to model the connection between inter/intra time slices activities. [14] focused on using Conditional Random Fields and its variant to model the activity recognition problem.

Activity recognition is also a very important research topic in the context of computer vision, in which researchers aim to recognize human actions from vision-based sensors, like video cameras. For example, in [17] and [18], the problem of detecting abnormal activity patterns based on object motion patterns is studied. Many vision-based activity recognition research also provide directions for ubiquitous sensor-based activity recognition research. The problems may seem similar at the first glance, however, the differences between vision-based sensors and other sensors like accelerometers or RFID sensors is that the knowledge and information one can gain from vision data is significantly larger than those of ubiquitous sensors. Besides, vision-based sensory input has developed to a level where many kinds of useful features are already well studied and used in recognition, for example, [19] proposed the fusion of multiple features including a quantized vocabulary of local spatiotemporal (ST) volumes (or cuboids) and a quantized vocabulary of spin-images. However, for ubiquitous sensor-based activity recognition, it is often the case that such a feature set does not even exist. In this paper, our proposed approach is only applied to datasets accumulated from ubiquitous sensors and we omit the discussion about whether our algorithm could be extended to vision-based sensory input.

Our work exploits the Web to connect two domains. In the past, some previous research works had considered learning common sense knowledge from the Web (such as ehow and KnowItAll [20,21]) to assist model training. For example, Perkowitz et al. proposed to mine the natural language descriptions of activities (e.g. "making-tea") from ehow.com as labeled data, and translated them into probabilistic collections of object terms (e.g. "teacup", "teabag", etc.) [22]. Then, they use these probabilistic collections as input data to train a dynamic Bayesian network model for prediction. Wyatt et al. also proposed to mine recipes of the activities from the Web as the labeled data, but they only use this knowledge as a prior and trained a Hidden Markov Model from the unlabeled RFID sensor data [23]. Wang et al. further improved this work by utilizing personal activity data from wearable sensors [24]. They first extracted the actions from the wearable sensors, and then incorporated the actions with the object's usage to finally predict the activities. Most previous approaches either exploited only object-usage information or required explicit action modeling. Few of them exploited auxiliary source domains for activity recognition.

2.2. Transfer learning

As we aim to transfer the labeled data across domains, our work is also related to *transfer learning*, which is a state-of-art learning paradigm in machine learning [25,26].

The idea of transfer learning is motivated by the fact that humans can intelligently apply knowledge learned previously to solve new problems faster or with better solutions. For example, we may find that having knowledge of C++ can help in studying Java or other programming languages and playing the electronic organ can help learning the piano. Transfer learning aims at transferring knowledge from some source domains to a target domain. In general, the data from the both domains may follow different distributions or be represented in different feature spaces. In [26], the definition of transfer learning is given in the following form: we are given a source domain D_s and learning task T_s, a target domain D_T and learning task T_T, transfer learning aims to help improve the learning of the target predictive function f_T in D_T using the knowledge in D_S and T_S, where $D_S \neq D_T$, or $T_S \neq T_T$.

There have been several main approaches to transfer learning in the past. Such approaches can be summarized into four cases based on "what to transfer". The first approach can be referred to as the instance-transfer [27] approach, which assumes that certain parts of the data in the source domain can be reused for learning in the target domain by re-weighting.

The second approach can be referred to as the feature-representation-transfer [28], which finds a "good" feature representation in the target domain. In such an approach, the knowledge used to transfer across domains is encoded into the learned feature representation. With such a new feature representation, the performance of the target task is expected to improve significantly.

The third approach is parameter-transfer [29], which makes the assumption that the source and target tasks share some parameters or prior distributions of the hyperparameters of the models. The transferred knowledge is then encoded into the shared parameters or priors. Thus, by discovering the shared parameters, knowledge can be transferred across tasks.

The fourth approach is the relational-knowledge-transfer, which deals with transfer learning for relational domains. The basic assumption of such an approach is that some relationship among the data in the source and target domains are similar. Thus, the knowledge being transferred is the relationship among the data. For example, [30] builds the mapping of relational knowledge between both domains.

Our work can be seen as an instance-transfer approach. However, our work is different from most previous works, because they usually assume that the domains can have different feature spaces but share the same label space. In our work, we consider that the domains can share the same feature space (e.g. a same set of sensors at a home), but have different label spaces (i.e. different activities). Most research works on transfer learning have not explicitly targeted this problem. One relevant example, to the best of our knowledge, is in [31]. In [31], the authors proposed an algorithm to perform Web query classification, where a bridging classifier is proposed to map user's queries to the target categories via an intermediate taxonomy. Their algorithm could support the need of real-world search engines where the target label set for queries are often changing among different domains. However, using the intermediate taxonomy, they could transfer useful knowledge from the intermediate taxonomy to the target domain. Our research work shares some merits with this work, in that we also aim to use similar information retrieval techniques based on Web knowledge to model the similarities between actions (words) in two domains.

2.3. Transfer learning in activity recognition

With more and more research work available in the area of transfer learning, there also have been some papers which aim to solve some activity recognition related problems that also fall into the category of transfer learning. In this subsection, we provide a brief overview of these papers and also provide a coarse categorization of these papers.

- Transferring Across Devices: One important component of activity recognition is localization, where locations are used as key factors which could imply the subject's activities. In [32], the authors propose a multi-task learning algorithm to solve the multi-device indoor localization problem. One major drawback of previous localization systems lies in the assumption that the collected signal distribution remains the same across different devices. However, empirical studies in [32] show that this is often not the case and that the localization accuracy would be greatly affected. In [32], the multi-device indoor localization problem was formulated as an optimization problem and an alternating optimization approach was employed to solve the problem.
- Transferring Across Time: Another important dimension by which transfer learning can be carried out is the time dimension. Similarly, signal data distributions also vary from time to time and if we directly apply a model learned in the previous month to try to perform recognition tasks in the current month, variations in signal distribution would degrade the algorithm performance. To solve this problem, [33] proposed a semi-supervised Hidden Markov Model to transfer the learned model from one time period to another.
- Transferring Across Space: Space is another dimension where transfer learning can be used. Since most localization models are supervised learning approaches and would require us to collect labeled data across an entire building if we want to do indoor localization in that building. Therefore, transfer learning would greatly alleviate the calibration efforts one needs to perform before a learning model is trained. In [34], the authors propose a novel approach to learn localization models across space, where a mapping function between the signal space and the location space is learned by solving an optimization problem based on manifold learning techniques.
- Transferring Across Sensor Networks: In activity recognition, one possible scenario when one tries to apply transfer learning is to transfer knowledge between different sensor networks where the layout and the functionality of the sensors may not necessarily be the same. Earlier research work [35] on this topic attempted to transfer the knowledge between

two houses which have different sensor network layouts. However, the correspondence between different sensors was defined manually, which greatly reduced the learning difficulty but limited the applicability of such systems. One recent paper [36] tries to solve the activity recognition also via a transfer learning approach. In this paper, the authors can perform activity recognition across different sensor networks. However, their algorithm is based on the usage of a *meta-feature space*, which are features that describe the properties of the actual features. Each sensor is described by one or more meta features, for example, a sensor on the microwave might have one meta feature describing the sensor is located in the kitchen, and another that the sensor is attached to a heating device. The limitations of the approach described in [36] is that the meta-feature space needs to be manually constructed, and it has to be the same kind of sensors in order to have the common meta-feature space. Such limitations also avoid the transfer between different kinds of sensors or in very different room layouts.

- Transferring Across Activity Types: This category refers to our work in this paper since we are aiming to transfer knowledge between different activity label spaces but make the assumption that the feature space (sensor networks) remain the same between different domains. Based on the above categorization, we can see that our work sets a different dimension from all previous research work on activity recognition that exploit some transfer learning ideas.

3. Problem formulation

We consider two domains that have the same set of sensors spanning a feature space but have different activity (label) spaces. Specifically, we have a source domain with a set of activities $\mathcal{A}_{src} = \{a_1, \ldots, a_m\}$, and a target domain with another set of activities $\mathcal{A}_{tar} = \{a_{m+1}, \ldots, a_n\}$. $\mathcal{A}_{src}$ and $\mathcal{A}_{tar}$ do not overlap, i.e. $\mathcal{A}_{src} \bigcap \mathcal{A}_{tar} = \emptyset$. In other words, we have two sets of activities, one of which is called the source domain and the other of which is called the target domain. In the source domain, the sensor readings are all labeled with activity names, and they will be used as training data. In the target domain, we do not have any labeled data for training, but we know how many activities are in the target domain and what their names are. Our aim is that, given some sensor readings (i.e. test data) from the target domain, we can use the labeled training data from the source domain to learn a recognizer and thus recognize their activity labels.

Notice that we are constraining our training data in the source domain whereas the test data is in the target domain. We will not test the sensor readings drawn from the source domain in the testing phase. Such a setting is reasonable due to two reasons. First, as the source domain has plentiful labeled data, using traditional learning approaches such as Naive Bayes, decision tree or support vector machines would be enough to recognize the source domain's activities. So in this paper, we are more focused on recognizing those activities from the target domain without any labeled data. Second, it is not hard for us to identify whether a sensor reading is drawn from the source domain or the target domain, by using some external sensor information. For example, we can have a location sensor in the house to identify where the user is, so we can know a sensor reading related to the user belongs to the source domain's activities (say, "making-the-bed" in the bedroom) or the target domain's activities (say, "laundry" in the laundry closet).

We also make an underlying assumption that the source domain's activities and the target domain's activities do have some kind of relationship. For example, "laundry" and "cleaning indoors" are related because they both involve some kind of "cleaning". However, "laundry" and "watching TV/movies" may only be weakly related, so we may not be able to transfer that much useful knowledge from "laundry" to "'watching TV/movies" and hence the algorithm may not perform well. We studied the impact of such domain differences in the experiment section. Two other assumptions are also made due to the nature of our method. First, we assume that the webpages describing an activity could mostly generalize the most important objects and procedures one needs to perform an activity. Second, we also assume that there are sensors monitoring object interaction, so that the sensor information contains enough knowledge about how the subject interacts with the objects of interest.

To be more precise, let $x \in \mathbb{R}^k$ be a k-dimensional sensor reading (i.e. feature) vector at some time slice, and y be a random variable whose value represents an activity (i.e. label). In the source domain, we have plentiful labeled training data $\mathcal{D}_{src}^{trn} = \{(x_{src}^{(i)}, y_{src}^{(i)})\}_{i=1}^{T_1}$, where $y_{src}^{(i)} \in \mathcal{A}_{src}$. In the target domain, we do not have any training labeled data; instead, we only have some test data $\mathcal{D}_{tar}^{tst} = \{x_{tar}^{(j)}, y_{tar}^{(j)}\}_{j=1}^{T_2}$, where $y_{tar}^{(j)} \in \mathcal{A}_{tar}$ are used as ground truth for testing only. We would note that the source domain's sensor data and the target domain's sensor data share the same feature space in $\mathbb{R}^k$, but the two domains have different label spaces.

In this paper, we make a simplification; that is, we break the sequences into time slices each, and omit the possible sequential information we could take advantage of in dealing with the problem of cross-domain activity recognition. There are several reasons for this. The first reason is that, when we constrain ourselves to using training data from only one subset of activities from the original training data (source domain), we are already "choosing" the activities in the sequences and have damaged the sequential information contained in the original data. The second reason is that, our paper, being the first paper to tackle the problem of cross-domain activity recognition formally, would put more emphasis on how to transfer useful knowledge between different activity sets; or, more loosely speaking, how to calculate similarities between different activities and demonstrate that such a simple method is indeed effective in cross-domain activity recognition tasks. Therefore, in this paper, we omit the sequential information and treat each time sequence of sensor readings with length T as T instances in the training or testing datasets.

4. Proposed approach

4.1. Algorithm overview

Our work belongs to the instance-transfer category in the transfer learning framework. In general, the instance-transfer algorithms are motivated by data instance importance sampling [27]. That is, the training data from a source domain are weighted to train a model for the target domain, and the weights can be generally seen as the similarities between the source domain's data and the target domain's data. The more similar some source domain's data are to the target domain's data, the higher weights the source domain's data will be assigned in the learning procedure. Different from the previous work on instance-transfer which measures the similarities from the data (features), we show that in our cross-domain activity recognition problem, we need to measure the similarities from the label information.

We first present an overview of our cross-domain activity recognition (referred to as CDAR below) algorithm to provide the readers with a high-level overview of our algorithm. Our CDAR algorithm can be generalized into three steps.

In the first step, we aim to learn a similarity function between different activities by mining knowledge from the Web. In particular, we will use Web search to extract related Web pages for the activities, and then apply information retrieval techniques to further process the extracted Web pages. After that, we use some similarity measure, such as Maximum Mean Discrepancy in Eq. (2), to calculate the similarities between any pair of activities from the source domain and target domain. Such similarities will be used later to calculate confidence for pseudo training data for domain transfer.

In the second step, given the assumption that we only have labeled training data in the source domain but no labeled training data in the target domain, it is impossible to follow supervised learning methods to train a recognizer for the target domain's activities. Therefore, by using the similarity values we have learned in the first step, we aim to generate some pseudo training data for the target domain with some confidence values. Here "pseudo training data" are the training data with the same feature values as in the source domain, but relabeled with the activity labels in the target domain. Such data relabelings will be assigned with some confidence, whose values equal the similarities we calculated in the first step; and this confidence will measure how "strong" a particular training data instance in the source domain can be explained as the data instances in the target domain.

Following the second step, if we have N activities in the source domain and M activities in the target domain, we will create pseudo training data in the target domain with a label space with size NM. Each pseudo training instance is supported with a confidence value. Therefore, by using the pseudo training data, we can apply a weighted Support Vector Machine method [37] to train a classifier, so that we can use it to recognize the activities in the target domain.

4.2. Learning the similarity function

In this section, we will show how to learn a similarity function for any pair of activities from the source domain and the target domain. To achieve this, we will novelly exploit the Web data.

4.2.1. Calculate similarity from web data

With the proliferation of the Web services, there are emerging Web pages that describe the daily activities. For example, we can easily find many web pages introducing how to make coffee. Such web pages encode the human understanding of the activity semantics, such as what kind of activity it is, what kind of objects it uses, etc. Such semantics can greatly help measure the similarities between the activities.

In practice, as the activity names are known, we can employ Web search to extract Web pages related to the activities. For example, for an activity "Vacuuming" defined in the taxonomy of Fig. 1, we can search on Google with the query "Vacuuming" as shown in Fig. 2. Then we can get a list of search results on the page. By clicking all search results, we can get a set of Web pages. Although the Web pages contain a lot of information, only a small amount of such information is related to the semantics of the queried activity. So we apply classic information retrieval techniques to retrieve the useful information for each Web page.

In particular, for each Web page, we first extract all the words in it, and then treat the extracted words as a document d_i with a bag of words. Such a document d_i can be further processed as a vector x_i^w, each dimension of which is the term frequency-inverse document frequency value (tf-idf) [38] of a word t:

$$tf\text{-}idf_{i,t} = \frac{n_{i,t}}{\sum_l n_{i,l}} \cdot \log \frac{|\{d_i\}|}{|\{d_i : t \in d_i\}|},$$

where $n_{i,t}$ is the number of occurrences of word t in document d_i. Besides, $|\{d_i\}|$ is the total number of collected documents, and $|\{d_i : t \in d_i\}|$ is the number of documents where word t appears. The terms in the tf-idf equation are explained as follows:

- The first term $\frac{n_{i,t}}{\sum_l n_{i,l}}$ denotes the frequency of the word t that appears in the document d_i. If the word t appears more frequently in the document d_i, then $|\{d_i : t \in d_i\}|$ is larger, and thus the whole term is larger. For example, in the returned Web page of "Vacuuming", the word "clean" may appear many times, so its term frequency is high. It means that such a word encodes some semantics of "Vacuuming", and it has higher weights in the Web data's feature vector.

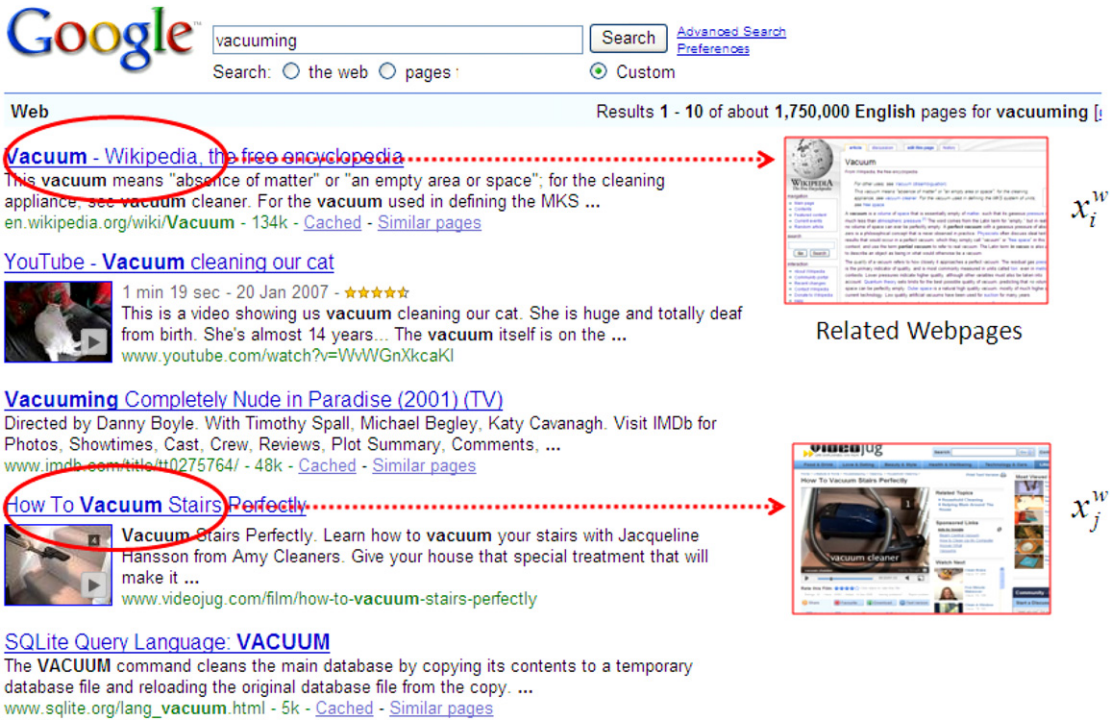

Fig. 2. Extract Web data for activities.

- The second term $\log \frac{|\{d_i\}|}{|\{d_i : t \in d_i\}|}$ denotes the inverse document frequency for the word t. If the word t appears in more documents of the corpus, then $|\{d_i : t \in d_i\}|$ is larger, and thus the whole term is smaller. For example, for the word "the", it is used in almost all the documents, but it is a stop word without any meaning. Hence, its inverse document frequency will vanish to zero, thus the whole tf-idf value of the word is zero. It means that such a word does not encode any semantics of the searched activity, so it can be removed from the Web data's feature vector.

Therefore, for an activity u (e.g. "Vacuuming"), we can get a set of documents $\mathfrak{D}_u^w = \{x_i^w | i = 1, \ldots, m_u\}$, with each x_i^w as a tf-idf vector. Similarly, for another activity v (e.g. "Washing-laundry"), we can also Google it and get another set of documents $\mathfrak{D}_v^w = \{z_i^w | i = 1, \ldots, m_v\}$, with each z_i^w as a tf-idf vector.

After having the extracted Web data $\mathfrak{D}_u^w$ and $\mathfrak{D}_v^w$, now we will show how to measure the similarity between the activity u and the activity v. Note that a possible choice to calculate the similarity between two (Web) data distributions is using the Kullback–Leibler (KL) divergence as [23] did. However, generally the Web text data are high-dimensional and it is hard to model the distributions over the two different data sets. Hence, we propose to use the Maximum Mean Discrepancy (MMD) [39], which can directly measure the distribution distance without density estimation, to calculate the similarity.

Definition 1. Let $\mathcal{F}$ be a class of functions $f : \mathcal{X} \to \mathbb{R}$. Let p and q be Borel probability distributions, and let $\mathcal{X} = (x_1, \ldots, x_m)$ and $\mathcal{Z} = (z_1, \ldots, z_n)$ be i.i.d. samples drawn from distributions p and q, respectively. Then, the Maximum Mean Discrepancy (empirical estimation) is

$$MMD[\mathcal{F}, \mathcal{X}, \mathcal{Z}] = \sup_{f \in \mathcal{F}} \left(\frac{1}{m} \sum_{i=1}^{m} f(x_i) - \frac{1}{n} \sum_{i=1}^{n} f(z_i) \right).$$

Considering the universal reproducing kernel Hilbert spaces (RKHS), we can interpret the function f as the feature mapping function $\phi(\cdot)$ of a Gaussian kernel [39].

Given the Web data $\mathfrak{D}_u^w = \{x_i^w | i = 1, \ldots, m_u\}$ for activity u and the Web data $\mathfrak{D}_v^w = \{z_i^w | i = 1, \ldots, m_v\}$ for activity v, we can finally have the similarity between u and v as

$$\text{sim}(u, v) = MMD^2[\mathfrak{D}_u^w, \mathfrak{D}_v^w], \tag{1}$$

where $MMD^2[\mathfrak{D}_u^w, \mathfrak{D}_v^w]$ is the maximum mean discrepancy defined as:

$$MMD^2[\mathfrak{D}_u^w, \mathfrak{D}_v^w] = \left\| \frac{1}{m_u} \sum_{i=1}^{m_u} \phi(x_i^w) - \frac{1}{m_v} \sum_{i=1}^{m_n} \phi(z_i^w) \right\|_{\mathcal{H}}^2$$

$$= \frac{1}{m_u^2}\left\|K_{uu}^w\right\|_1 - \frac{2}{m_u m_v}\left\|K_{uv}^w\right\|_1 + \frac{1}{m_v^2}\left\|K_{vv}^w\right\|_1, \tag{2}$$

where K_{uv}^w is the Gaussian kernel defined over the data $\mathfrak{D}_u^w$ and $\mathfrak{D}_v^w$. Specifically, K_{uv}^w is a $m_u \times m_v$ matrix, with its entry at row i and column j defined as

$$K_{uv}^w(x_i^w, z_j^w) = \exp\left(-\frac{\left\|x_i^w - z_j^w\right\|^2}{2\sigma^2}\right),$$

where σ is the kernel width for the Gaussian kernel function. In Eq. (2), $\|\cdot\|_1$ is an entry-wise norm which sums up all the entries in the matrix.

4.3. Generating pseudo training data

Now we have the similarity value $\mathrm{sim}(u, v)$ for each pair of activities $u \in \mathcal{A}_{src}$ and $v \in \mathcal{A}_{tar}$. How can we generate a new training data set defined over the label space of the target domain? Recall that, in the source domain, we have the training labeled data $\mathcal{D}_{src}^{trn} = \{(x_{src}^{(i)}, y_{src}^{(i)})\}_{i=1}^{T_1}$, where $y_{src}^{(i)} \in \mathcal{A}_{src}$. For each training instance $(x_{src}^{(i)}, y_{src}^{(i)})$ with $y_{src}^{(i)} = u$ where $u \in \mathcal{A}_{src}$, we will relabel it to get a set of pseudo training data as $\{(x_{src}^{(i)}, v_j, sim(u, v_j))|v_j \in \mathcal{A}_{tar}\}_{j=1}^{|\mathcal{A}_{tar}|}$. Here, the similarity $sim(u, v_j)$ between activity u and v_j is used as the confidence of such a relabeling. In other words, we duplicate each training instance $|\mathcal{A}_{tar}|$ times; and each duplication will be relabeled using one activity category in the target domain with some confidence. Finally, these relabeled training data duplications, which we call "pseudo" training data, are then used for training classifiers to classify activities in the target domain.

4.4. Weighted SVM method

Now we have a pseudo training data set on the target domain where each data instance in the dataset contains not only a category label but also a confidence value. The confidence value is defined as the similarity value we calculate between two activities. Therefore, the larger the value is, the more similar the two activities are, and the more confident we are when interpreting such a training data instance to this activity in the target domain.

However, training support vector machines with confidence values attached to training instances is a non-trivial task and we apply the method proposed in [37] to accomplish our goal. Interested readers can follow the original paper for technical details. Here we will briefly introduce the weighted SVM model for multi-class classification.

In [37], a "one-against-one" approach is employed for multi-class classification. Given the N classes (in our case, each activity in the target domain is a class, so $N = |\mathcal{A}_{tar}|$), this approach constructs $N(N-1)/2$ classifiers, each of which trains the data from two different classes. For training data from the ith and the jth classes, the weighted SVM model solves the following two-class classification problem:

$$\begin{aligned} \min_{\mathbf{w}^{ij}, b^{ij}, \xi^{ij}} \ & \frac{1}{2}(\mathbf{w}^{ij})^T\mathbf{w}^{ij} + C_t^i \sum_{y_t=i}(\xi^{ij})_t + C_t^j \sum_{y_t=j}(\xi^{ij})_t \\ \text{s.t.} \quad & (\mathbf{w}^{ij})^T\phi(x_t) + b^{ij} \geq 1 - \xi_t^{ij}, \quad \text{if } y_t = i, \\ & (\mathbf{w}^{ij})^T\phi(x_t) + b^{ij} \leq -1 + \xi_t^{ij}, \quad \text{if } y_t = j, \\ & \xi_t^{ij} \geq 0. \end{aligned} \tag{3}$$

Here, x_t is the tth data instance, y_t is its class label. $\phi(x_t)$ is a feature mapping to x_t. $\mathbf{w}^{ij}$ is the model parameter, b^{ij} is the bias term, and ξ^{ij} is the slack variable denoting the classification error. C_t^i and C_t^j are the weights for the tth instance of ith and jth classes respectively. C_i^t and C_t^j are derived using the similarity function learned from the previous step; and they reflect the confidence values of the data instances x_t interpreted as being from the ith class (i.e. activity) in the target domain. In other words, the pseudo training data point x_t is from the ith class with confidence value of C_t^i. Therefore, in Eq. (3), the first term makes sure that in training the support vector machine the margin is maximized; the second and third terms controls the weighted classification errors for both classes. Intuitively, if the weight C_t^i is higher, the pseudo training data instances x_t from the ith class are more trusted in training the SVM model.

After the optimization in Eq. (3) is solved, [37] uses a voting strategy for multi-class classification. In particular, each binary classification for the ith and the jth classes is considered to be a vote. Then, the votes can be cast for all data instances x_t, and in the end each x_t is designated to be in a class with maximum number of votes. In the case that two classes have identical votes, the one with the smallest class index is simply chosen.

4.5. Cross-domain activity recognition (CDAR) algorithm

Finally, we summarize our CDAR method in Algorithm 1. As shown in Algorithm 1, at the first 3 steps, we extract the Web pages for each activity from both domains, and apply the information retrieval technique to transform the Web pages into

tf-idf vectors. At step 4, after having the Web data (i.e. a set of tf-idf vectors) for each activity, we compute a similarity matrix for each pair of activities between the source domain and the target domain. At step 5, based on the learned similarities, we generate the pseudo training data by relabeling each training instance with the activity labels from the target domain. Each relabeled training instance is assigned with some confidence (weight), which equals the similarity between its original activity label (from the source domain) and the newly given activity label (from the target domain). At step 6, we train the CDAR model using a weighted Support Vector Machine. At step 7, we use the trained weighted SVM classifier to performing testing on the target domain's data.

Algorithm 1 Algorithm for CDAR

Input: Source domain has T_1 labeled training data $\mathcal{D}_{src} = \{(x_{src}^{(i)}, y_{src}^{(i)})\}_{i=1}^{T_1}$, where $y_{src}^{(i)} \in \mathcal{A}_{src}$. Target domain does not have any labeled training data; instead it has T_2 test data $\mathcal{D}_{tar} = \{(x_{tar}^{(j)}, y_{tar}^{(j)})\}_{j=1}^{T_2}$, where $y_{tar}^{(j)} \in \mathcal{A}_{tar}$ are the ground truth labels for testing only.
Output: Predicted labels on the test data in target domain.
begin

1: For each activity $u \in \mathcal{A}_{src}$, extract a list of Web pages from some search engine (such as Google);
2: For each u's Web pages, apply information retrieval technique and transform each Web page to a tf-idf vector, and form a Web data set $\mathfrak{D}_u^w$;
3: Similar to the above two steps, extract a Web data set $\mathfrak{D}_v^w$ for each activity $v \in \mathcal{A}_{tar}$;
4: For any two activities $u \in \mathcal{A}_{src}$ and $v \in \mathcal{A}_{tar}$, calculate the similarity $sim(u, v) = MMD^2(\mathfrak{D}_u^w, \mathfrak{D}_v^w)$ using the maximum mean discrepancy in Eq.(2);
5: Generate pseudo training data as follows: each training instance $(x_{src}^{(i)}, y_{src}^{(i)})$ with $y_{src}^{(i)} = u$ where $u \in \mathcal{A}_{src}$, is relabeled to get a set of pseudo training data as $\{(x_{src}^{(i)}, v_j, sim(u, v_j))|v_j \in \mathcal{A}_{tar}\}_{j=1}^{|\mathcal{A}_{tar}|}$ with $sim(u, v_j)$ as the confidences.
6: Train the model CDAR with a weighted SVM [37] on the generated pseudo training data.
7: Testing by the trained weighted-SVM classifier.

end

5. Experimental results

In this section, we plan to validate the effectiveness of our algorithm through experiments on several real-world datasets. We plan to answer the following questions we had posed in previous sections to provide a systematic view of the performance of our algorithm.

- First, is it possible for us to transfer useful knowledge between the source domain and the target domain using our proposed approach?
- Second, how accurate can it be when we create pseudo training data in the target domain using Web knowledge?
- Third, how would the choice of activities in the source domain affect the algorithm performance when we attempt to transfer knowledge to the same set of activities in the target domain?
- And finally, how would our algorithm parameters, such as the similarity function we use and the number of top ranked pages we pick out in the searching step, affect our algorithm's performance?

5.1. Datasets, evaluation criteria and implementation details

In this section, we briefly describe the datasets we use in our experiments, the evaluation criteria we adopt, and some other detailed information about our algorithm implementation.

We use three datasets in our experiments. All the datasets we use are publicly available on the Web. Our first dataset (UvA in short)[2] is from [40] where a dataset is recorded in the house of a 26-year-old man, living alone in a three-room apartment where 14 state-change sensors are installed. Locations of sensors include doors, cupboards, refrigerators and a toilet flush sensor. Sensors were left unattended and collected data for a period of 28 days, resulting in 2120 sensor events and 245 activity instances. Activities were annotated by the subject himself using a bluetooth headset. Seven different activities were annotated: "Leave house", "Toileting", "Showering", "Sleeping", "Preparing breakfast", "Preparing Dinner", "Preparing a beverage".

The second dataset we use is the MIT PLIA1 dataset[3] [11], which was recorded on March 4, 2005 from 9 AM to 1 PM in the MIT PlaceLab. The subject was asked to perform a set of common household activities during the four-hour period. The dataset contains 89 different activities and was manually classified into several categories such as "Cleaning", "Yardwork", etc.

2 http://staff.science.uva.nl/~tlmkaste/research/software.php.
3 http://architecture.mit.edu/house_n/data/PlaceLab/PLIA1.htm.

Table 1
Algorithm performance on UvA and intel dataset.

K	UvA Acc (Std)	Intel Acc (Std)
5	40.2% (21.7%)	39.8% (19.7%)
10	53.7% (22.8%)	47.3% (20.2%)
20	65.8% (22.1%)	58.1% (20.7%)
50	**66.7**% (**21.2**%)	**63.2**% (**23.5**%)
100	66.0% (22.4%)	63.1% (20.7%)
Supervised	72.3% (20.7%)	78.3% (17.6%)

The third dataset is from [7][4] and is provided by Intel Research Lab (Intel in short), which aims to recognize 11 routine morning activities including using the bathroom, making oatmeal, making soft-boiled eggs, preparing orange juice, making coffee, making tea, making or answering a phone call, taking out the trash, setting the table, eating breakfast and clearing the table. Sensor readings were recorded by the subject simultaneously wearing two RFID gloves outfitted with antennae.

The evaluation criteria we use in this paper is rather standard and simple. Since we are omitting sequential information in the dataset, therefore we just calculate the accuracy we achieve on the test data set; in other words, the number of correctly predicted activities over the total number of activities in all time slices.

We also briefly describe how we handle the Web pages we trawled from the search engine. Using the activity names (e.g. preparing breakfast, cleaning misc, etc.) as queries, we submit these queries to Google and then the top K results are retrieved from the search engine, where K is a parameter we will tune in the following subsections to show the relationship between algorithm performance and the number of Web pages we retrieve for each query. Next, data preprocessing has been applied to the raw data where all letters in the text are converted to lower case and the words are stemmed using the Porter stemmer [41]. Furthermore, stop words are removed using a stop word list from the Web.[5] The SVM implementation we used is the LIBSVM package [37].

5.2. Algorithm performance

We report the performance of our algorithm on three datasets with different activities in both the source domain and the target domain. We also describe how we choose the activities in the source domain and the target domain in detail.

Since the number of activities in the UvA dataset and the Intel dataset are significantly fewer than that of the MIT PLIA1 dataset, we will use a relatively simpler strategy to choose the source and target domain activities for the UvA dataset and the Intel dataset. For the MIT PLIA1 dataset, we will analyze the algorithm's performance with different pairs of source and target domain activities, which can help us understand that the "closeness" of the activities could affect the algorithm's performance a lot.

In the UvA dataset, there are only 7 activities. Therefore, we use training data from 3 activities for training (source domain), and use the remaining 4 activities for testing (target domain). All sensor readings with their activities in the source domain are used as training data and the rest used as testing data. The process of selecting different activities is repeated ten times and we report the average accuracy and standard deviation we get under different parameters K. Here K is the number of top ranked Web pages we extract from the search engine results.

We use a similar way of choosing activities in the source domain and the target domain in the Intel dataset, while we use 5 activities in the source domain and the remaining 6 activities in the target domain. Again, the algorithm is repeated ten times to report a mean accuracy and standard derivation. Detailed results are shown in Table 1. The row "Supervised" indicates the accuracy we achieve using a SVM classifier with normal 10-fold cross validation on the target domain when we could acquire labeled data on them, in other words, the performance of the SVM classifier under the traditional supervised learning setting on the target domain. Such a result could be used as a baseline and an upper-bound to understand how good the performance of our cross-domain activity recognition system is.

In the MIT PLIA1 dataset, since there are many activities included in this dataset and a taxonomy could be built to describe these activities [8,11]. Therefore, in the MIT PLIA1 dataset, we analyze how the performance will be affected when we use the activities under the same category as activities in the source domain and construct training data, and then do the testing on another set of activities under the same category in the target domain.

Examples are shown in Fig. 1, when we use activities under the node of "Cleaning Indoors" as activities in the source domain and activities under the node of "Laundry" or "Dishwashing" as activities in the target domain. In other words, we are using all sensor readings with activities "Sweeping, Swiffering, Mopping, Vacuuming, Dusting, Making the bed, Putting things away, Disposing Garbage, Taking out trash, Cleaning a surface, Scrubbing, Cleaning misc, Cleaning background" as training data and the testing data contain all sensor readings with activities "Washing laundry, Drying laundry, Washing laundry background, Drying laundry background, Folding laundry, Putting away laundry, Ironing, Laundry misc".[6]

4 http://www.cs.rochester.edu/~kautz/Courses/577autumn2007/a5data.zip.

5 http://armandbrahaj.blog.al/2009/04/14/list-of-english-stop-words/.

6 Some activities, although defined in the activity taxonomy, do not appear in the MIT PLIA1 dataset, e.g. "Mopping".

Table 2
Algorithm performance on MIT PLIA1 dataset.

Source	Target	$K = 10$	$K = 20$	$K = 50$
Cleaning	Laundry	53.4% (19.2%)	57.3% (18.7%)	58.9% (20.5%)
Cleaning	Dishwashing	43.2% (18.7%)	49.3% (23.0%)	53.2% (20.7%)
Cleaning	Hygiene	48.3% (22.8%)	52.4% (17.6%)	58.3% (20.7%)
Cleaning	Leisure	45.7% (17.9%)	52.8% (20.7%)	54.9% (22.8%)
Laundry	Cleaning	52.8% (20.5%)	53.2% (20.7%)	60.2% (20.0%)
Laundry	Dishwashing	53.2% (20.7%)	58.3% (20.5%)	61.2% (21.2%)
Laundry	Hygiene	46.3% (19.2%)	49.5% (23.0%)	58.3% (21.4%)
Laundry	Leisure	40.2% (20.7%)	48.3% (20.5%)	49.2% (22.6%)
Dishwashing	Cleaning	48.3% (20.5%)	53.2% (21.7%)	59.2% (19.2%)
Dishwashing	Laundry	50.1% (21.7%)	54.8% (21.9%)	60.8% (22.6%)
Dishwashing	Hygiene	52.7% (22.1%)	57.3% (20.7%)	59.2% (16.7%)
Dishwashing	Leisure	43.7% (21.7%)	48.3% (18.7%)	50.3% (19.2%)
Hygiene	Cleaning	45.8% (20.5%)	49.7% (20.7%)	52.9% (19.5%)
Hygiene	Laundry	43.7% (20.2%)	53.8% (23.0%)	53.7% (17.0%)
Hygiene	Dishwashing	42.8% (19.2%)	47.1% (22.8%)	51.2% (18.7%)
Hygiene	Leisure	33.8% (20.5%)	38.3% (20.2%)	42.3% (21.2%)

The reason for us to choose the source domain's activity set and the target domain's activity set in such a way is as follows: we would like to analyze the performance of our algorithm to transfer from a more "categorized" set of activities to another set of activities, which is more similar rather than chosen at random, as we had done in the previous two datasets. We report the performance of our algorithm tested on different pairs of source activity sets and target activity sets, with mean accuracies and standard deviations calculated over ten independent runs. Results are reported below in Table 2 with various settings of parameter K.

From Tables 1 and 2, we can observe that our cross domain activity recognition (CDAR) algorithm could achieve a comparable performance to supervised learning classifiers when evaluated on the target domain's activities and trained on the source domain's activities. Especially, we find that when we evaluate a SVM classifier under a supervised learning scenario on the UvA dataset, the accuracy is around 72%, whereas we could achieve a performance of 66% using our cross-domain activity recognition algorithm, which is very close to the upper bound.

We also observe that with the increasing value of K being introduced in the algorithm, the performance generally increases. Such an observation follows our intuition, which means more information is being extracted from the top Web pages and generally the MMD value calculated is more accurate, thereby improving the performance of the algorithm overall. Here we make another special note of how the choice of value K would affect our algorithm's performance. From Tables 1 and 2, we could observe that a good value of K, around 20 or 50, would give us the optimal results. When K is larger than the threshold of 50, the performance will degrade instead of continue improving. The reason is that, when K is too small, the Web page data is sparse and hence we could not get much useful information for calculating the similarity function and when K is too large, the web pages we had crawled may not still be relevant to our querying activities and it may probably add noise into the Web pages we had crawled and therefore it may degrade the overall performance of our algorithm.

Besides, another important observation in our experimental results in Tables 1 and 2 is that the standard deviation we present is quite high. This result is understandable since in each round, the random procedure will select probably different sets of activities in the source domain and/or the target domain. Therefore, the fact that the standard deviation is high probably means that for some specific sets of source domain activities and target domain activities, the performance can be high, whereas for some other sets of source and target domain activities, the performance will be poor. However, the mean accuracy can still give us a general idea about the overall performance level of the algorithm.

Therefore, our experimental results on three datasets validates the effectiveness of our algorithm.

5.3. Choice of similarity functions

The Maximum Mean Discrepancy (MMD) function in our proposed approach seems to be an ad hoc choice, and it is natural for one to ask the question about whether it is chosen deliberately to improve the performance of our algorithm or whether other similarity functions would also achieve a comparative performance, compared to MMD? To answer this question, we also evaluated our algorithm using another similarity function that is also popular in calculating the similarity in information retrieval, i.e. the cosine similarity [38].

In short, the cosine similarity is defined as the cosine of the angle between two vectors, defined as:

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{|A||B|}, \tag{4}$$

where A and B are two vectors. Such a value is easy to calculate based on the Webpages we crawled. In our case, we generate A and B by merging the term-frequency inverse-document frequency (tf-idf) vectors of the documents for the two candidate activities. For example, given the K Web pages extracted for some activity a_A, we will add up all the corresponding K tf-idf values to get such a vector A. Similarly, we can get the tf-idf vector B for some activity a_B, so that we can compute the Eq. (4).

Table 3
Mined similarity values between activities.

	Leave	Toilet	Shower	Bed	Breakfast	Dinner	Drink
Leave	1	**0.29**	0.21	0.19	0.15	0.16	0.25
Toilet	0.29	1	**0.30**	0.25	0.20	0.19	0.26
Shower	0.21	**0.30**	1	0.28	0.19	0.19	0.26
Bed	0.20	0.25	0.28	1.	0.18	0.17	**0.29**
Breakfast	0.15	0.20	0.19	0.18	1	**0.23**	0.18
Dinner	0.16	0.19	0.19	0.17	**0.23**	1	0.21
Drink	0.25	0.26	0.26	**0.29**	0.18	0.21	1

Table 4
Algorithm performance on UvA and intel dataset with cosine similarities and MMD similarities.

K	Cosine similarity		MMD similarity	
	UvA Acc (Std)	Intel Acc (Std)	UvA Acc (Std)	Intel Acc (Std)
5	48.7% (19.5%)	42.4% (20.5%)	40.2% (21.7%)	39.8% (19.7%)
10	53.2% (20.7%)	45.3% (20.2%)	53.7% (22.8%)	47.3% (20.2%)
20	58.3% (20.2%)	52.1% (20.7%)	65.8% (22.1%)	58.1% (20.7%)
50	62.1% (19.2%)	57.3% (19.0%)	66.7% (21.2%)	63.2% (23.5%)
100	**65.3**% (**16.7**%)	**62.3**% (**21.7**%)	**66.0**% (**22.4**%)	**63.1**% (**20.7**%)

We provide an example of the mined similarities with a set of seven activities: leave house, use toilet, take shower, go to bed, prepare breakfast, prepare dinner and get drink. The similarity values are shown in Table 3.

From the values in Table 3, we could see that the cosine similarity also provides a reasonable estimate about the "closeness" of activities. For example, "Use toilet" probably should be closest to "Take shower" since the location of the two activities are often close together. Another example is that "Prepare breakfast" and "Prepare dinner" should also be close. Such examples of "closeness" are all correctly reflected in the values shown in Table 3. However, there are some other erroneous cases such as "Get drink" and "Go to bed" having a very high similarity value, which cannot be explained using our intuition.

Using the same algorithm except the choice of similarity function being changed, we report our results for the UvA dataset [40] and the Intel dataset [7] in Table 4.

In Table 4, the left two columns are results using cosine similarity functions and the right two columns are results using MMD functions, which is the same as the results reported in Table 1. We could see that although the results using cosine similarity functions are slightly worse than the results we report using MMD functions, it still achieves comparable performance to MMD. Therefore, we could make the conclusion that by incorporating other "meaningful" and "reasonable" similarity functions, our cross-domain activity recognition algorithm could still achieve reasonable performance.

5.4. Using similarities rated by humans

Besides the cosine similarity function chosen above, one might also ask whether a manually-specified similarity function, i.e. similarity functions generated by human knowledge, can also achieve similar results? To answer this question, we further conducted user studies to check whether a similarity function generated (or voted) by a human, could also achieve similar results. We have approached five students ranging from different departments to fill a table which encodes the similarity values of different activities, in a similar form as the above Table 3. The only difference lies in that this time the values of the tables are filled with their own beliefs, with the ratings ranging from 1–5.[7] To avoid potential drawbacks in our user studies, all the students we have approached are not related to our research and also have no background knowledge about activity recognition. The students are all Ph.D. students, but not necessarily majoring in computer science. Of the five of them, two of them are females.

More specifically, the ratings state the degree of belief of whether the two activities are correlated or linked. For example, score 1 indicates that these two activities "are not correlated at all and I cannot imagine these two activities have any links or relationships"; score 2 indicates that these two activities "may have some weak correlations, but the probability of these two actions co-occurring or sharing similar features are not high". And finally a score of 5 indicates that these two activities "are definitely correlated and they share many features, like the items used, the view of the actions, etc". We asked five people to fill in their own similarity beliefs to get a more balanced view of different people with different background knowledge and understandings of human actions. Finally, we averaged the similarity ratings across all the similarity tables they filled and divide the rating by 5 in order to normalize the rating in the range [0, 1]. The average ratings of all 5 users across all activity pairs are shown in Table 5 as a reference. Such a similarity ratings is also fed into our proposed algorithm to check whether it can also provide reasonable results.

[7] We use the dataset from University of Amsterdam for this user study, rather than the Intel Research Lab dataset or the MIT PlaceLab dataset. The main reason is that the number of activity pairs in the other two datasets are simply too large for humans to label them.

Table 5
Similarity values between activities based on user studies.

	Leave	Toilet	Shower	Bed	Breakfast	Dinner	Drink
Leave	1	0.48	0.48	0.24	0.4	0.4	0.48
Toilet	0.48	1	0.72	0.48	0.36	0.32	0.44
Shower	0.48	0.72	1	0.6	0.36	0.36	0.28
Bed	0.24	0.48	0.6	1	0.36	0.28	0.32
Breakfast	0.4	0.36	0.36	0.36	1	0.76	0.72
Dinner	0.4	0.32	0.36	0.28	0.76	1	0.72
Drink	0.48	0.44	0.28	0.32	0.72	0.72	1

The above-mentioned similarity matrix could achieve an average accuracy of 59.5%, which is worse than the accuracy achieved using mined similarities when K is larger than 50 but better than the accuracy achieved when K is smaller than 50, but such an accuracy is still acceptable. Since we just replace the similarities mined from the Web knowledge with this similarity matrix based on user studies, therefore the result does not depend on any parameter. Hence, it can be shown that our algorithm could achieve a reasonable result with any choice of similarity function as long as such a similarity function is reasonable as well.

Here we provide some informal analysis on why similarities mined from the Web performs better than the similarities we got from such a user study. Different people have different views about what "similarity" between two activities means, for example, in our user studies, some students rate the similarity between "breakfast" and "dinner" as 5 whereas others only rated 2. When asking them for the specific reason to do so, the response from the one rated 5 was "they are both having meals" and the response from the one rated 2 was "how can these two activities co-occur"? Therefore, different people have their own standards and ways of measuring similarities and therefore the similarities of a specific pair of activities is not coherent between different people. However, mined similarities do not have such a problem. Since the similarity metric we use are based on term occurrence overlap from the documents mined from the Web, different aspects of similarities between activities are considered including object-usage overlap (nouns), action similarities (verbs), location overlap (places), etc. Therefore, such a similarity mined from the Web takes multiple aspects of similarities between activities into consideration, and is better than manually constructed similarity.

5.5. Discussion of experiments

Here we briefly discuss and summarize some characteristics of the results we report from our experiments.

1. Our algorithm CDAR, could successfully transfer knowledge between the source domain and the target domain and thus solve the cross-domain activity recognition task. From Tables 1, 2 and 4, we could see that in most cases, our algorithm could achieve an accuracy of more than 60% when evaluating on the target domain activity set. Such a performance is rather promising since (i) we did not use any sequential information, which had proved to be of great help in traditional activity recognition tasks; (ii) we did not use any labeled data in the target domain, such a feature of our algorithm is especially effective in real world use, since the activity recognition system will be trained on a predefined set of activities and then when users use such an activity recognition system, he may perform activities outside of the predefined activity set. Therefore, such an algorithm that could perform cross-domain activity recognition and does not acquire training data in the target domain would be especially useful.
2. The number of K Web pages we extract from the search engine results would affect the algorithm's performance but would be rather close to the optimal results or converge when K is larger than 50. This means that we could approximate the underlying similarity score between different activities by using around 50 related Web documents. Another advantage is that the cost of performing such a Web search would not be heavy to put into real world usage.
3. The choice of similarity functions would not affect our algorithm's performance that much as long as a reasonable similarity function that approximates the underlying similarity would be used. In our experiments we have already validated this conclusion through the usage of both MMD and cosine similarity functions and from the comparison between these two functions, we arrive at such a conclusion.
4. Whether the cross-domain activity recognition performance would be successful or not depends not only on the algorithm but on some underlying characteristics of the activity set. In Table 2, we could note that some activity pairs perform worse than others, for example, when we are transferring knowledge from activities under the subcategory of "Hygiene" (Source Domain) to activities under the subcategory of "Information/Leisure" (Target Domain), we could only achieve an accuracy of around 42% at the best. (Corresponding to the last row in Table 2.) Such a result might be of the reason that there is a relatively larger distance between "Hygiene" activities and "Information/Leisure" activities. In contrast, we could also find that when we use activities in the "Dishwashing" subcategory as the source domain and activities in the "Cleaning" or "Laundry" subcategory as the target domain, the accuracy we achieve is much higher, suggesting that these two kind of activities have closer relationships, which also follows our intuition. Thus, one interesting topic to study is to determine when we could successfully "transfer the knowledge" between the source domain and the target domain, or, if we had known the activity set in the target domain in advance, what source domain would be best for us to use to ensure that the algorithm performance can be guaranteed? One possible choice can be

using some activity ontology and shrinkage based approach to combine data from similar activities for training the classifier [21].

6. Conclusion and future work

In this paper, we have proposed a simple yet effective approach to solve the cross-domain activity recognition problem. The basic setting of our problem is that we have labeled training data in the source domain but no labeled training data in the target domain; and our goal is to recognize the activities in the target domain. Furthermore, the activities in the source and target domains do not overlap, which means that traditional supervised learning approaches cannot be applied in this scenario. Our proposed approach makes use of top ranked Web pages returned by search engines to mine the similarities between the activities in the two domains. By doing so, we aim to create some "pseudo training data" in the target domain by making use of the training data in the source domain as well as the similarities between activities in the two domains. We validate our approach in several real-world sensor-based activity recognition datasets, and achieve results comparable to those of traditional activity recognition algorithms based on supervised learning.

In the future, we aim to extend our work in several directions.

First, we aim to investigate into cases where not only the activity label spaces, but also the feature spaces in two domains are different. Our proposed approach will not be able to solve this problem since the same supervised learning algorithm is applied to train the classifier in the target domain, which clearly requires the equality of feature spaces in the two domains. In the related work section, we have mentioned a recent paper [36] which attempts to solve the activity recognition problem under such a transfer learning setting. However, we have also mentioned that such an algorithm still needs much human effort in creating the meta feature space.

Second, in this paper, our proposed approach does not take the sequential nature of sensor readings into account. Furthermore, as we mentioned in the Related Work section, state-based models like Hidden Markov Models or Conditional Random Fields are easy to model such a sequential nature. Therefore, another possible direction is to propose new approaches for cross-domain activity recognition which could make better use of sequential information.

Third, in our experiments, we had observed that if we fix the activities in the target domain, applying our algorithm on some particular set of activities in the source domain would achieve better performance than using some other set of activities as the source domain. Such a phenomenon suggests that the "relatedness" of activities in the source domain and the target domain would affect the algorithm's performance greatly. In transfer learning literature, such a problem is also referred to as "negative transfer" [26] and some empirical studies also show that if two tasks are too dissimilar, brute-force transfer may hurt the performance of the target task [42]. In our cross-domain activity recognition task, when we aim to recognize the activities from a target domain, if we can calculate the "relatedness" between activities in the two domains, it could help us choose appropriate activities in the source domain to carry out the transfer procedure.

Acknowledgement

We thank the support from Microsoft Research Asia and the support of project Hong Kong CERG/China-NSFC Grant N_HKUST624/09.

References

[1] E.M. Tapia, S.S. Intille, K. Larson, Activity recognition in the home using simple and ubiquitous sensors, in: Pervasive Computing, 2004, pp. 158–175.
[2] M.R. Hodges, M.E. Pollack, An 'object-use fingerprint': the use of electronic sensors for human identification, in: Proceedings of the Ninth International Conference on Ubiquitous Computing, UbiComp 2007, 2007, pp. 289–303.
[3] H. Kautz, A formal theory of plan recognition, Ph.D. Thesis, University of Rochester, 1987.
[4] H.H. Bui, A general model for online probabilistic plan recognition, in: Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, IJCAI 2003, 2003, pp. 1309–1318.
[5] L. Liao, D. Fox, H.A. Kautz, Learning and inferring transportation routines, in: Proceedings of the Nineteenth National Conference on Artificial Intelligence, AAAI, 2004, pp. 348–353.
[6] J. Yin, X. Chai, Q. Yang, High-level goal recognition in a wireless lan, in: Proceedings of the Nineteenth National Conference on Artificial Intelligence, AAAI, 2004, pp. 578–584.
[7] D.J. Patterson, D. Fox, H.A. Kautz, M. Philipose, Fine-grained activity recognition by aggregating abstract object usage, in: Proceedings of the Ninth IEEE International Symposium on Wearable Computers, ISWC 2005, 2005, pp. 44–51.
[8] D.H. Hu, S.J. Pan, V.W. Zheng, N.N. Liu, Q. Yang, Real world activity recognition with multiple goals, in: Proceedings of the Tenth International Conference on Ubiquitous Computing, UbiComp 2008, 2008, pp. 30–39.
[9] D.H. Hu, Q. Yang, Cigar: concurrent and interleaving goal and activity recognition, in: Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI, 2008, pp. 1363–1368.
[10] D.H. Hu, X.-X. Zhang, J. Yin, V.W. Zheng, Q. Yang, Abnormal activity recognition based on hdp-hmm models, in: IJCAI, 2009, pp. 1715–1720.
[11] S.S. Intille, K. Larson, E.M. Tapia, J. Beaudin, P. Kaushik, J. Nawyn, R. Rockinson, Using a live-in laboratory for ubiquitous computing research, in: Proceedings of the Fourth International Conference on Pervasive Computing, Pervasive 2006, 2006, pp. 349–365.
[12] M.E. Pollack, L.E. Brown, D. Colbry, C.E. McCarthy, C. Orosz, B. Peintner, S. Ramakrishnan, I. Tsamardinos, Autominder: an intelligent cognitive orthotic system for people with memory impairment, Robotics and Autonomous Systems (RAS) 44 (3–4) (2003) 273–282.
[13] L. Liao, Location-based activity recognition, Ph.D. Thesis, University of Washington, 2006.
[14] D.L. Vail, M.M. Veloso, J.D. Lafferty, Conditional random fields for activity recognition, in: Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS 2007, 2007, pp. 1–8.

[15] C.W. Geib, J. Maraist, R.P. Goldman, A new probabilistic plan recognition algorithm based on string rewriting, in: Proceedings of the Eighteenth International Conference on Automated Planning and Scheduling, ICAPS 2008, 2008, pp. 91–98.
[16] D.J. Patterson, L. Liao, D. Fox, H.A. Kautz, Inferring high-level behavior from low-level sensors, in: Proceedings of the Fifth International Conference on Ubiquitous Computing, UbiComp 2003, 2003, pp. 73–89.
[17] A. Basharat, A. Gritai, M. Shah, Learning object motion patterns for anomaly detection and improved object detection, in: CVPR, 2008.
[18] X.-X. Zhang, H. Liu, Y. Gao, D.H. Hu, Detecting abnormal events via hierarchical dirichlet processes, in: PAKDD, 2009, pp. 278–289.
[19] J. Liu, S. Ali, M. Shah, Recognizing human actions using multiple features, in: CVPR, 2008.
[20] O. Etzioni, M.J. Cafarella, D. Downey, A.-M. Popescu, T. Shaked, S. Soderland, D.S. Weld, A. Yates, Methods for domain-independent information extraction from the web: an experimental comparison, in: Proceedings of the Nineteenth National Conference on Artificial Intelligence, AAAI, 2004, pp. 391–398.
[21] E.M. Tapia, T. Choudhury, M. Philipose, Building reliable activity models using hierarchical shrinkage and mined ontology, in: Proceedings of the Fourth International Conference on Pervasive Computing, Pervasive 2006, 2006, pp. 17–32.
[22] M. Perkowitz, M. Philipose, K. Fishkin, D.J. Patterson, Mining models of human activities from the web, in: Proc. of the 13th International Conference on World Wide Web, WWW, 2004, pp. 573–582.
[23] D. Wyatt, M. Philipose, T. Choudhury, Unsupervised activity recognition using automatically mined common sense, in: Proceedings, The Twentieth National Conference on Artificial Intelligence, AAAI 2005, 2005, pp. 21–27.
[24] S. Wang, W. Pentney, A. maria Popescu, Common sense based joint training of human activity recognizers, in: Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI, 2007, pp. 2237–2242.
[25] R. Caruana, Multitask learning, Machine Learning 28 (1) (1997) 41–75.
[26] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Transactions on Knowledge and Data Engineering 99 (PrePrints).
[27] M. Sugiyama, S. Nakajima, H. Kashima, P. von Bünau, M. Kawanabe, Direct importance estimation with model selection and its application to covariate shift adaptation, in: NIPS, 2007.
[28] R. Raina, A. Battle, H. Lee, B. Packer, A.Y. Ng, Self-taught learning: transfer learning from unlabeled data, in: ICML, 2007, pp. 759–766.
[29] T. Evgeniou, M. Pontil, Regularized multi-task learning, in: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, Seattle, Washington, USA, 2004, pp. 109–117.
[30] L. Mihalkova, T. Huynh, R.J. Mooney, Mapping and revising markov logic networks for transfer learning, in: Proceedings of the 22nd AAAI Conference on Artificial Intelligence, 2007, pp. 608–614.
[31] D. Shen, J.-T. Sun, Q. Yang, Z. Chen, Building bridges for web query classification, in: SIGIR, 2006, pp. 131–138.
[32] V.W. Zheng, S.J. Pan, Q. Yang, J.J. Pan, Transferring multi-device localization models using latent multi-task learning, in: AAAI, 2008, pp. 1427–1432.
[33] V.W. Zheng, E.W. Xiang, Q. Yang, D. Shen, Transferring localization models over time, in: AAAI, 2008, pp. 1421–1426.
[34] S.J. Pan, D. Shen, Q. Yang, J.T. Kwok, Transferring localization models across space, in: AAAI, 2008, pp. 1383–1388.
[35] T. van Kasteren, G. Englebienne, B. Kröse, Recognizing activities in multiple contexts using transfer learning, in: AAAI Fall 2008 Symposium: AI in Eldercare, Washington DC, USA, 2008.
[36] T. van Kasteren, G. Englebienne, B.J.A. Kröse, Transferring knowledge of activity recognition across sensor networks, in: Pervasive, 2010.
[37] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.
[38] C.D. Manning, P. Raghavan, H. Schutze, Introduction to Information Retrieval, Cambridge University Press, 2008.
[39] K. Borgwardt, A. Gretton, M. Rasch, H. Kriegel, B. Schölkopf, A. Smola, Integrating stuctured biological data by kernel maximum mean discrepancy, in: Proc. of the 14th International Conference on Intelligent Systems for Molecular Biology, 2006, pp. 49–57.
[40] T. van Kasteren, A.K. Noulas, G. Englebienne, B.J.A. Kröse, Accurate activity recognition in a home setting, in: UbiComp, 2008, pp. 1–9.
[41] M. Porter, An algorithm for suffix stripping, Program 14 (3) (1980) 130–137.
[42] M. Rosenstein, Z. Marx, L.P. Kaelbling, To transfer or not to transfer, in: NIPS-05 Workshop on Inductive Transfer: 10 Years Later, 2005.