Knowl Inf Syst (2017) 53:337–364
DOI 10.1007/s10115-017-1043-3

REGULAR PAPER

Collegial activity learning between heterogeneous sensors

Kyle D. Feuz[1] · **Diane J. Cook**[2]

Received: 2 December 2014 / Accepted: 17 March 2017 / Published online: 27 March 2017

Abstract Activity recognition algorithms have matured and become more ubiquitous in recent years. However, these algorithms are typically customized for a particular sensor platform. In this paper, we introduce PECO, a Personalized activity ECOsystem, that transfers learned activity information seamlessly between sensor platforms in real time so that any available sensor can continue to track activities without requiring its own extensive labeled training data. We introduce a multi-view transfer learning algorithm that facilitates this information handoff between sensor platforms and provide theoretical performance bounds for the algorithm. In addition, we empirically evaluate PECO using datasets that utilize heterogeneous sensor platforms to perform activity recognition. These results indicate that not only can activity recognition algorithms transfer important information to new sensor platforms, but any number of platforms can work together as colleagues to boost performance.

Keywords Activity recognition · Machine learning · Transfer learning · Pervasive computing

1 Introduction

Activity recognition and monitoring lie at the center of many fields of study. An individual's activities affect that individual, the people nearby, society, and the environment. In the past, theories about behavior and activity were formed based on self-report and limited in-person

✉ Diane J. Cook
cook@eecs.wsu.edu

Kyle D. Feuz
kylefeuz@weber.edu

1 Department of Computer Science, Weber State University, Ogden, UT 84408, USA

2 School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA 99164, USA

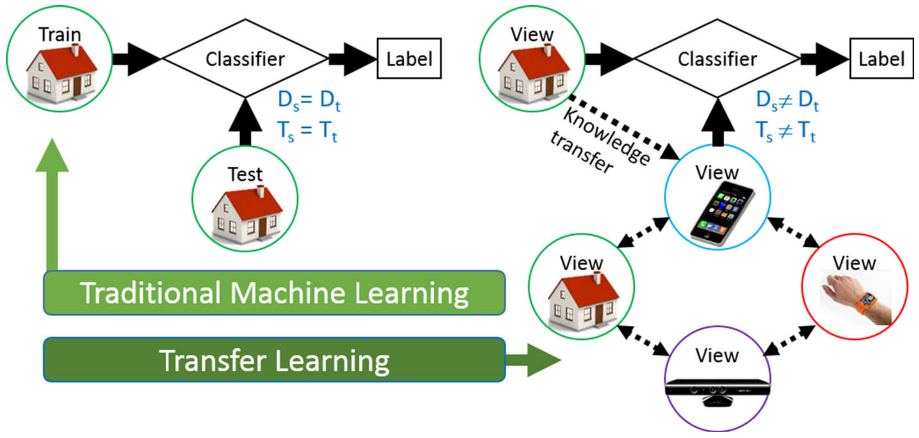


Fig. 1 In traditional machine learning, training and testing data come from the same domain and have similar distributions. In contrast, transfer learning uses knowledge from a different, related domain to improve learning for a new domain. In a personalized activity ecosystem, the home, phone, wearable, and camera use transfer learning to act as colleagues despite their diversity

observations. More recently, the maturing of sensors, wireless networks, and machine learning have made it possibly to automatically learn and recognize activities from sensor data. Now, activity recognition is becoming an integral component of technologies for health care, security surveillance, and other pervasive computing applications.

As the number and diversity of sensing devices increase, a personalized activity monitoring ecosystem can emerge. Instead of confining activity recognition to a single setting, any available device can “pick up the gauntlet” and provide both activity monitoring and activity-aware services. The sensors in a person’s home, phone, vehicle, and office can work individually or in combination to provide robust activity models.

One challenge we face in trying to create such a personalized ecosystem is that training data must be available for each activity based on each sensor platform. Gathering a sufficient amount of labeled training data is labor intensive for the user.

Transfer learning techniques have been proposed to handle these types of situations where training data are not available for a particular setting. Transfer learning algorithms apply knowledge learned from one problem domain, the *source*, to a new but related problem, the *target* (see Fig. 1). While these algorithms typically rely on shared feature spaces or other common links between the problems, in this paper we focus on the ability to transfer knowledge between heterogeneous activity learning systems where the domains, the tasks, the data distributions, and even the feature spaces can all differ between the source and the target.

As an example, consider a scenario where training data were provided to train a smart home to recognize activities based on motion and door sensors. If the user wants to start using a phone-based recognizer, the label-and-train process must be repeated. To avoid this step, we design an omni-directional transfer learning approach, or collegial learning, that allows the smart home to act as a teacher to the phone and allows the phone in turn to boost the performance of the smart home’s model.

Our collegial activity learning techniques can be applied to even broader scenarios. Smart Health is a recently proposed concept involving collaboration between a user’s mobile device and the city-wide sensing infrastructure to provide better healthcare. Solanas et al. [1] outline several of the challenges and opportunities presented by a Smart Health system, many of which are addressed through collegial learning. Collegial learning could be used to introduce new sensor platforms locally around the user and globally within the smart city infrastructure. Collegial learning could also play a vital role in integrating these heterogeneous sensing platforms with minimal human input.

2 Activity recognition

Our proposed Personalized activity ECOsystem, PECO, builds on the notion of activity recognition or labeling activities based on a sensor-based perception of the user and the environment. Let e represent a sensor reading and x be a sequence of such sensor readings, $e_1, \ldots e_n$. Y is a set of possible activity labels, and y is the activity label associated with a particular sequence of sensor events such as x. The problem of activity recognition is to map features describing a sequence of sensor readings (sensor events), $x = < e_1\, e_2 \ldots e_n >$, onto a value from a set of predefined activity labels, $y \in Y$. This is typically accomplished by using a supervised machine learning algorithm that learns the mapping based on a set of sample data points in which the correct label is provided.

Traditional activity recognition, as shown in Fig. 2, consists of collecting sensor data, preprocessing the data and partitioning it into subsequences, extracting a high-level set of features from the data subsequences, and providing the feature vector to a supervised learning algorithm [2–7]. If we want to extend traditional activity recognition to create a personalized activity ecosystem, we need to consider that raw sensor data and corresponding feature vectors change dramatically between sensor platforms. Different sensor types excel at representing different classes of activities. Not surprisingly, most activity recognition research thus focuses on a single sensor modality. Common activity learning sensor modalities are ambient sensors [8–11], wearable [12–16], object [17–19], phone [20,21], microphone [22], and video [23–26].

Many different machine learning methods have been developed for activity recognition. These include Bayesian approaches [8,27,28], hidden Markov models [29–32], conditional random fields [11,28,33], support vector machines [15], decision trees [27], and ensemble methods [28,34,35]. Each of these approaches offers advantages in terms of amount of training that is required, model robustness, and computational cost.

The focus of this paper is not on improving the underlying activity classification methodology but rather on transferring learned information between substantially different sensor platforms. As a result, the only modification to activity recognition itself we made that differentiates this work from some of the others is to perform recognition in real time from streaming data [36].

Collegial learning applies to other application domains when the target concept can be represented from different viewpoints. To illustrate this point, we describe a toy domain which has two different view points. The first viewpoint is a two-dimensional feature space with real-valued features A and B. The second viewpoint is three-dimensional feature space also with real-valued features C, D, and E. The features A and B are unrelated to the features C, D, or E. Both viewpoints describe the same two-class target concept, and we use 0, 1 as the label space. Figure 3 shows some sample data from the two viewpoints. We will use this toy domain as an additional example when describing the collegial learning process throughout the rest of this paper.

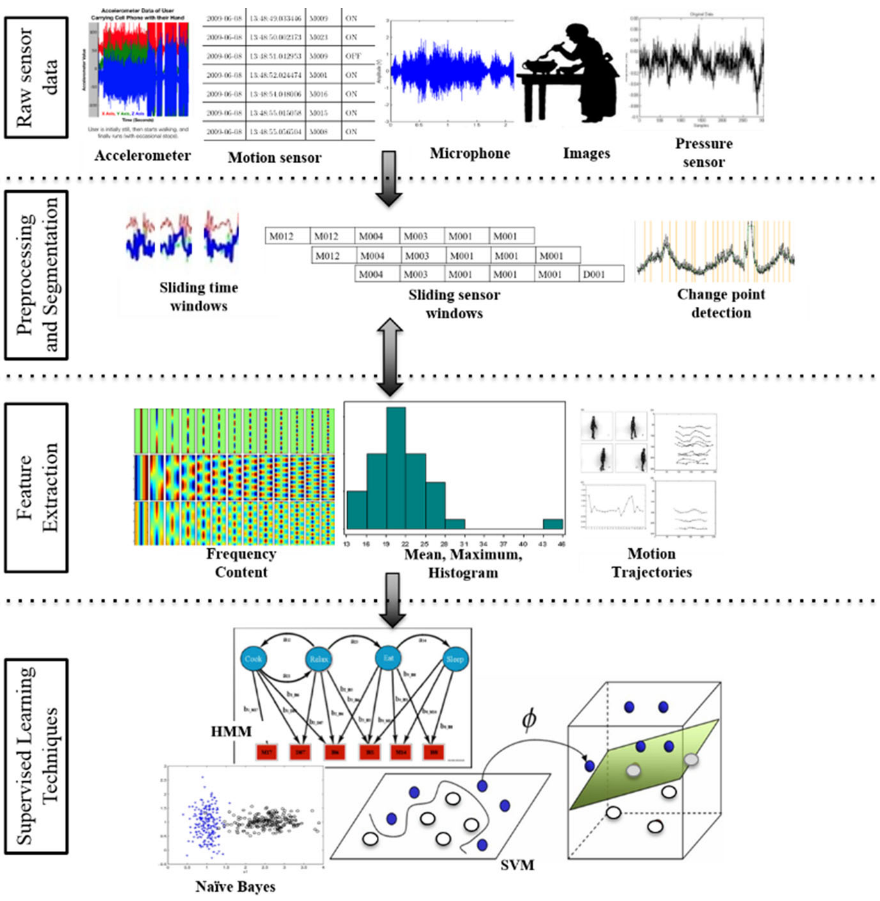


Fig. 2 Activity recognition includes stages of raw sensor data collection, preprocessing and segmentation, feature extraction and selection, classifier training and data classification

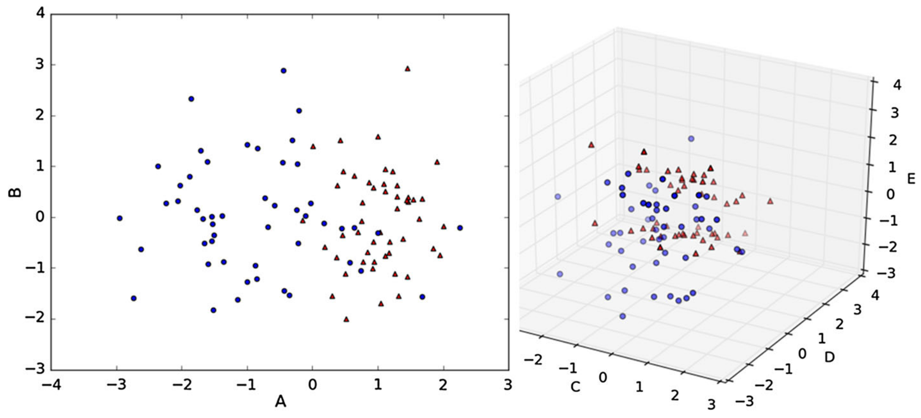


Fig. 3 Sample data from the toy dataset. Viewpoint 1 (*left*) has 2 features A and B while viewpoint 2 (*right*) has 3 features C, D, and E. Class 1 is shown as *red triangles*, and class 2 is shown as *blue circles* (color figure online)

3 Transfer learning for activity recognition

In order to share learned activity information between sensor platforms, we need to design heterogeneous transfer learning approaches. In the field of machine learning, transfer learning refers to transferring learned knowledge to a different but related problem. This idea is studied under a variety of pseudonyms such as learning to learn, life-long learning, knowledge transfer, inductive transfer, context-sensitive learning, and meta-learning [37–41]. It is also closely related to self-taught learning, multi-task learning, domain adaptation, and co-variate shift [42,43]. Because of the many terms that are used to describe transfer learning, we provide a formal definition of the terms which we will use throughout this paper, starting with definitions for domain and task, based on Pan and Yang [44]:

Definition 1 *(Domain)* A domain D is a two-tuple $(\chi, P(X))$. Here, χ represents the feature space of D and $P(X)$ is the probability distribution of $X = \{x_1, \ldots, x_m\} \in \chi$ where m is the number of features of X.

Definition 2 *(Task)* A task T is a two-tuple $(Y, f())$ for a given domain D. Y is the label space of D, and f() is a predictive function for D. $f()$ is sometimes written as a conditional probability distribution $P(y_i|x)$ where $y_i \in Y$ and $x \in \chi$. $f()$ is not given but can be learned from the training data.

In the case of activity recognition, the domain is defined by the feature space based on the most recent sensor readings and a probability distribution over all possible feature values. In the activity recognition example given earlier, the set of sensor readings x is one instance of x$\in \chi$. The task is composed of a label space Y which contains the set of labels for activities of interest together with a conditional probability distribution representing the probability of assigning label $y_i \in Y$ given the observed data point $x \in \chi$.

In the toy dataset, our two viewpoints each represent a domain. In viewpoint one, χ is the two-dimensional feature space. P(X) is the probability distribution of data instances in the viewpoint. Y is the label space {0, 1}, and $f()$ is the function which assigns and instance in the viewpoint a class label 0 or 1. We can now provide a definition of transfer learning.

Definition 3 *(Transfer Learning)* Given a set of source domains $DS = \{D_{s1}, \ldots, D_{sn}\}$, $n > 0$, a target domain D_t, a set of source tasks $TS = \{T_{s1}, \ldots, T_{sn}\}$ where $T_{si} \in TS$ corresponds with $D_{si} \in DS$, and a target task T_t which corresponds with D_t, transfer learning improves the learning of the target predictive function $f()$ in D_t, where $D_t \notin DS$ and/or $T_t \notin TS$.

Definition 3 encompasses many transfer learning scenarios. The source domains can differ from the target by having a different feature space, a different distribution of data points, or both. The source tasks can also differ from the target task by having a different label space, a different predictive function, or both. In addition, the source data can differ from the target data by having a different domain, a different task, or both. However, all transfer learning problems rely on the assumption that there exists some relationship between the source and target which allows for successful transfer of knowledge from source to target.

Previous work on transfer learning for activity recognition has focused primarily on transfer between users, activities, or settings. While most of these methods are constrained to one set of sensors [45–51], a few efforts have focused on transfer between sensor types. In addition to the teacher–learner model, we will discuss later [52], Hu and Yang [53] introduced a between-modality transfer technique that requires externally provided information about the relationship between the source and domain spaces.

Other transfer learning approaches have been developed outside activity learning that can be valuable for sharing information between heterogeneous sensor platforms. For example, domain adaptation allows different source and target domains, although typically the only difference is in the data distributions [54]. Differences in data distributions have been considered when the source and target domain feature spaces are identical, using explicit alignment techniques [55–57]. In contrast, we focus on transfer learning problems where the source and target domains have different feature spaces. This is commonly referred to as heterogeneous transfer learning, defined below.

Definition 4 *(Heterogeneous Transfer Learning)* Given domains *DS*, domain D_t, tasks *TS*, and task T_t as defined in Definition 3, heterogeneous transfer learning improves the learning of the target predictive function $f_t()$ in D_t, where $\chi_t \cap (\chi_{s1} \cup \cdots \cup \chi_{sn}) = \emptyset$.

Our toy dataset clearly represents a heterogeneous transfer learning scenario. The two viewpoints have the same label space but have different feature spaces and probability distributions on those feature spaces.

Heterogeneous transfer learning methods have not been attempted for activity recognition, although they have been explored for other applications. Some previous research has yielded feature space translators [58,59] as well as approaches in which both feature spaces are mapped to a common lower-dimensional space [60,61]. Additionally, previous multi-view techniques utilize co-occurrence data, or data points that are represented in both source and target feature spaces [62–64].

There remain many open challenges in transfer learning. One such challenge is performing transfer-based activity recognition when the source data are not labeled. Researchers have leveraged unlabeled source data to improve transfer to the target domain [42,65], but such techniques have not been applied to activity recognition nor used in the context of multiple source/target differences. We address both of these challenges in this paper by introducing techniques for transferring knowledge between heterogeneous feature spaces, with or without labeled data in the target domain.

Note that this work differs from multi-sensor fusion for activity learning [66–68]. Sensor fusion techniques combine data derived from diverse sensory data. However, they typically require that training data be available for all of the sensor types. In contrast, we are interested in providing a "cold start" for new sensor platforms that allow them to perform activity recognition with minimal training data of their own. The unique contributions of this work thus center on both two new approaches to multi-view learning with theoretical bounds and on development of a real-time personalized ecosystem that transfers activity knowledge in real time between heterogeneous sensor platforms.

Co-Training methods have been available for years as a semi-supervised machine learning technique, but have rarely if ever been applied as a transfer learning technique between heterogeneous systems. Furthermore, extending them to work without labeled data in the target learning space is critical to deploying new learning systems with minimal user effort required. Lastly, the learning bounds are useful both from a theoretical viewpoint and also from a practical viewpoint. Previously, predicting the accuracy of a learning system required labeled data in the target space, but now we can both train a learning system and predict its accuracy without requiring any user-labeled data in the target domain.

4 Personalized ecosystem

Every day brings new advances in sensing and data processing. Given the increasing prevalence of diverse sensors, we need to be able to introduce new activity sensory devices without requiring additional training time and effort. In response, we are designing multi-view techniques to transfer knowledge between activity recognition systems. The goal is to increase the accuracy of the collaborative ecosystem while decreasing the need for new labeled data.

To illustrate our approach, recall the earlier example in which a home (source view) is equipped with ambient sensors to monitor motion, lighting, temperature, and door use. The resident now wants to train smart phone sensors (target view) to recognize the same activities. Whenever the phone is located inside the home, both sensing platforms collect data while activities are performed, resulting in a multi-view learning opportunity where the ambient sensors represent one view and the phone sensors represent a second view. If the phone can be trained, it can also monitor activities outside of the home and can update the home's model when the resident returns. The phone may converge upon a stronger model than the home either because it receives training data of its own or because it has a more expressive feature space. In these cases, the target view can be used to actually improve the source's model. We will consider both informed and uninformed approaches to multi-view learning, distinguished by whether labeled data are available in the target domain (informed) or not (uninformed).

4.1 Informed multi-view learning

Two techniques have been extensively used by the community for multi-view learning when labeled training data are available in the target domain. In Co-Training [63], a small amount of labeled data in each view are used to train a separate classifier for each view (for example, one for the home and one for the phone). Each classifier then assigns labels to a subset of the unlabeled data, which can be used to supplement the training data for both views. We first adapt Co-Training for activity recognition, as summarized in Algorithm 1. While Algorithm 1 is designed for binary classification, it can handle activity recognition with more than two classes by allowing each view to label positive examples for each activity class.

Algorithm 1: Co-Training

```
Input: a set $L$ of labeled activity data points
       a set $U$ of unlabeled activity data points
Create set $U'$ of $u$ examples, $U' \subseteq U$
while $U' \neq \emptyset$ do
    Use $L$ to train activity recognizer $h_1$ for view 1
    ...
    Use $L$ to train activity recognizer $h_k$ for view $k$
    Label most confident $p$ positive data points and
        $n$ negative data points from $U'$ using $h_1$
    ...
    Label most confident $p$ positive data points and
        $n$ negative data points from $U'$ using $h_k$
    Add the newly-labeled data points to $L$
    Add $k(p + n)$ data points from $U$ to $U'$
```

```
ALGORITHM 2: CO-EM
  Input: a set $L$ of labeled activity data points
         a set $U$ of unlabeled activity data points
  Use $L$ to train classifier $h_1$ on view 1
  Create set $U_1$ by using $h_1$ to label $U$
  for $i = 0$ to $m$ do
      Use $L \cup U_1$ to train classifier $h_2$ on view 2
      Create set $U_2$ by using $h_2$ to label $U$
      ...
      Use $L \cup U_{k-1}$ to train classifier $h_k$ on view k
      Create set $U_1$ by using $h_k$ to label $U$
```

The second informed technique, Co-EM, is a variant of Co-Training that has been shown to perform better in some situations [69]. Unlike Co-Training, Co-EM labels the entire set of unlabeled data points every iteration. Training continues until convergence is reached (measured as the number of labels that change each iteration) or a fixed number of iterations *m* are performed, as in Algorithm 2. We can introduce additional classifiers as needed, one for each view. Each view can also employ a different type of classifier, as is best suited for the corresponding feature space.

4.2 Uninformed multi-view learning

In a personalized ecosystem, labeled data are not always available in the target domain. This would happen, for example, when a new sensor platform is first integrated into the ecosystem. In this situation, we need to use an uninformed multi-view technique. Uninformed multi-view algorithms have been proposed and tested for applications such as text mining [70]. One such algorithm is Manifold Alignment [65], which assumes that the data from both views share a common latent manifold which exists in a lower-dimensional subspace. If this is true, then the two feature spaces can be projected onto a common lower-dimensional subspace using principal component analysis [71]. The subsequent pairing between views can be used to optimally align the subspace projections onto the latent manifold using a technique such as Procrustes analysis. A classifier can then be trained using projected data from the source view and tested on projected data from the target view. This is summarized in Algorithm 3.

```
Algorithm 3: Manifold Alignment Algorithm (two views)

Input: a set $L$ of labeled activity data points in view 1
       sets $U_1$, $U_2$ of paired unlabeled data points
                (one set for each view)
$X, EV_1 = PCA(U_1)$   // map $U_1$ onto lower-dimensional space
$Y = PCA(U_2)$         // map $U_2$ onto same space
// Apply Procrustes Analysis to align X and Y
$U\Sigma V^T \leftarrow SVD(Y^T X)$
$Q \leftarrow U V^T$
$k \leftarrow$ Trace($\Sigma$) / Trace($Y^T Y$)
$Y' \leftarrow kYQ$
Project $L$ onto low-dimensional embedding using $EV$
Train classifier on projected $L$
Test classifier on $Y'$
```

Finally, we consider a teacher–learner model that was introduced by Kurz et al. [52] to train new views that have no labeled training data. This approach, summarized in Algorithm 4, can be applied in settings where labeled activity data are available in the source view but not the target view.

We note that the teacher–learner algorithm is equivalent to a single-iteration version of Co-EM when no labeled data are available in the target view. This observation allows us to provide a theoretical foundation for the technique. Valiant [72] introduced the notion of probably approximately correct (PAC) learning to determine the probability that a selected classification function will yield a low generalization error. Blum and Mitchell [63] show that multi-view learning has PAC bounds when three assumptions hold: (1) the two views are conditionally independent given the class label, (2) either view is sufficient to correctly classify the data points, and (3) the accuracy of the first view is at least weakly useful.

In this paper, we enhance these baseline multi-view learning approaches for application to activity learning, specifically for multiple activity classes. We also introduce new approaches to multi-view learning for this problem and provide a PAC analysis for the proposed PECO approach.

Algorithm 4: Teacher-Learner Algorithm

```
Input: a set $L$ of labeled activity data points in view 1
       a set $U$ of unlabeled data points
Use $L$ to train activity recognizer $h_1$ for view 1
Create set $U_1$ by using $h_1$ to label $U$
Use $U_1$ to train classifier $h_2$ on view 2
…
Use $U_1$ to train classifier $h_k$ on view $k$
```

Algorithm 5: PECO Algorithm

```
Input: a set $L$ of labeled activity data points in view 1
       a set $U$ of unlabeled data points
Use $L$ to train activity recognizer $h_1$ for view 1
Create set $U_1$ by using $h_1$ to label $U' \subset U$
$L = L \cup U_1$
$U = U - U_1$
Apply Co-Training or Co-EM
```

4.3 Personalized ECOsystem (PECO) algorithm

Our Personalized ECOsystem (PECO) approach to multi-view learning combines the benefits of iterative informed strategies such as Co-Training and Co-EM with the benefits of using teacher-provided labels for new uninformed sensor platforms that have no labeled activity data for training. The PECO approach is summarized in Algorithm 5. Not only can PECO facilitate activity recognition in a new view with no labeled data, but in some cases the accuracy of view 1 actually improves when view 2, the target view, subsequently takes on the role of teacher and updates view 1's model. This illustrates our notion of *collegial learning*.

In the PECO algorithm, the teacher (source view, view 1) initially provides labels for a few data points that both the teacher and the learner (target view, view 2) observe with their different sensors and represent with their different feature vectors. Next, PECO transitions to an iterative strategy by applying Co-Training or Co-EM. In this way, the learner can continue to benefit from the teacher's expertise while at the same time contributing its own knowledge back to the teacher, as a colleague.

In our home–phone transfer scenario, the smart home may initially act as a teacher because it has labeled activity data available and the phone does not. When the home and phone are co-located, the home can opportunistically "call out" activity labels that the phone can use to label the data captured in its own sensor space. Eventually the resident may grab their phone and leave the home. While out, the phone will observe new activity situations and may even receive labels from the user for those situations. When the individual returns home, the home and phone act as colleagues, transferring expertise from each independent classifier and feature representation to improve the robustness of both activity models for both sensor platforms. This scenario can be extended for any number of types and diversity of sensor platforms.

In addition to the new PECO algorithm, we introduce a second multi-view approach that handles more than two views, which we refer to as Personalized ECOsystem with Ensembles, or PECO-E. PECO-E first combines multiple teacher views into a single view using a weighted voting ensemble where each vote is weighted by the classifier's confidence in the activity label. This newly formed teacher model can then transfer its knowledge to the learner view using one of the existing binary multi-view techniques described in Sects. 4.2 and 4.3.

4.4 Online transfer learning

In order to demonstrate the ability to transfer activity knowledge between sensor platforms in real time, we implemented PECO as part of the CASAS smart home system [73]. The CASAS software architecture is shown in Fig. 4. The CASAS physical layer includes sensors and actuators. The middleware layer is driven by a publish/subscribe manager with named broadcast channels that allow bridges to publish and receive text messages. The middleware also provides services such as adding time stamps to sensor readings, assigning unique identifiers to devices, and maintaining the state of the system. Every CASAS component communicates via a customized XMPP bridge to this manager. These bridges include the ZigBee bridge to provide network services, the Scribe bridge to archive data, and bridges for each of the software components in the application layer including PECO. When the sensors generate readings, they are sent as text messages to the CASAS middleware. PECO's activity recognition algorithm generates activity labels for sensor readings in real time as they are received from one sensor platform. A new sensor platform can be announced to CASAS and integrated without any other changes to the system, simply by announcing its presence and function. Activity labels can then be sent from the middleware to PECO's new sensor view in real time in order to bring the new sensor platform up to speed. Implementing a personalized ecosystem within CASAS provides the ability to seamlessly introduce new sensor capabilities and transfer knowledge from one platform to another and back. Transferring this information will typically boost the activity recognition performance for both platforms.

4.5 PECO accuracy bounds

Without labeled activity data in the target view, we cannot compute performance measures such as model accuracy. We can, however, still derive theoretical bounds for worst-case, best-case, and expected-case learner performance. We make the assumption that the previously observed teacher accuracy on labeled data is a good indicator of the teacher's accuracy on

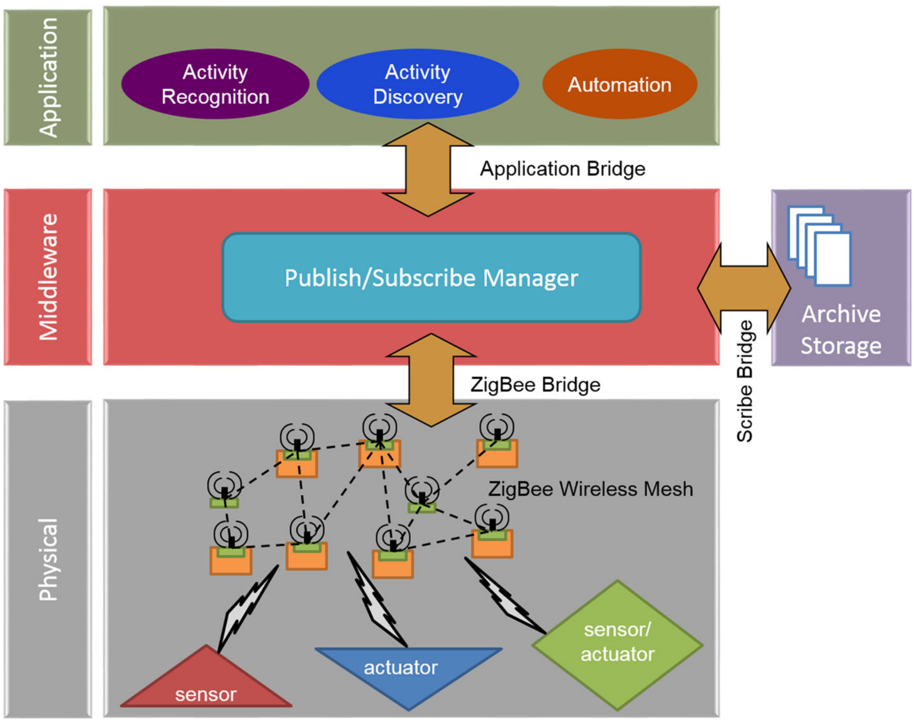


Fig. 4 CASAS software architecture to support real-time transfer learning between heterogeneous platforms

unlabeled data. Our accuracy bound also relies on the average level of agreement, q, between the teacher and learner for unlabeled data points in U. We define q in Eq. 1, where $h_1(x)$ is the teacher's activity label for data point x and $h_2(x)$ is the learner's label for x.

$$q = \frac{1}{|U|} \sum_{x \in U} \begin{cases} 1 & \text{if} \quad \mathrm{h}_1(\mathrm{x}) = \mathrm{h}_2(\mathrm{x}) \\ 0 & \text{if} \quad \mathrm{h}_1(\mathrm{x}) \neq \mathrm{h}_2(\mathrm{x}) \end{cases} \tag{1}$$

In the case of binary classification tasks, Eq. 2 calculates the expected accuracy of the learner, r. This calculation assumes that the teacher–learner agreement is independent of the teacher's classification accuracy, p.

$$\begin{aligned} r &= pq + (1-p)(1-q) \\ &= 2pq + (1-p-q) \end{aligned} \tag{2}$$

The first term, pq, represents the expected accuracy of the learner given that the teacher correctly classifies the data point. The second term represents the expected learner accuracy given that the teacher incorrectly classifies the data point. Note that the learner can only be accurate in the second case when it disagrees with the teacher.

To generate upper and lower learner accuracy bounds, we must consider both teacher–learner agreement when the teacher is correct (q_1) and the agreement when the teacher is incorrect (q_2). Substituting these variables into Eq. 2 results in Eq. 3. Accuracy is then optimized by maximizing q_1 and minimizing q_2.

$$r = pq_1 + (1-p)(1-q_2) \tag{3}$$

Here q_1 and q_2 are subject to the following constraints:

- $pq_1 + (1-p)q_2 = q$,
- $0 \leq q_1 \leq 1$, and
- $0 \leq q_2 \leq 1$.

These constraints ensure that the original level of agreement q is preserved and that q_1 and q_2 are valid levels of agreement. Note that in the first constraint, minimizing q_2 maximizes q_1 and vice versa. If $p \geq q$, then q_2 has a minimum value of 0 and all the constraints are satisfied. This implies that $q_1 = q/p$ is the maximum value of q_1. Substituting this expression into Eq. 3 results in Eq. 4.

$$r = q + 1 - p \tag{4}$$

If $p < q$, then q_1 has a maximal value of 1 and all of the constraints are satisfied. This implies that $q_2 = (q-p)/(1-p)$ is the minimum value of q_2. Substituting into Eq. 3 results in Eq. 5.

$$r = p + 1 - q \tag{5}$$

Equations 4 and 5 can then be combined into Eq. 6, which represents the upper bound on learner accuracy.

$$r = 1 - |p - q| \tag{6}$$

Next, the lower bound on learner accuracy can be derived by minimizing q_1 and maximizing q_2 in Eq. 3, subject to the same constraints on q_1 and q_2. If $(1-p) \geq, q$ then q_1 has a minimum value of 0 and all the constraints are satisfied. This implies that $q_2 = q/(1-p)$ is the maximum value of q_2. Substituting into Eq. 3 results in Eq. 7.

$$r = 1 - p - q \tag{7}$$

If $(1-p) < q$, then q_2 has a maximum value of 1 and all of the constraints are satisfied. This implies that $q_1 = (q - 1 + p)/p$ is the minimum value of q_1. Substituting in Eq. 3 results in Eq. 8.

$$r = q - 1 + p \tag{8}$$

Finally, Eqs. 7 and 8 are combined into Eq. 9, which calculates the lower bound on learner accuracy.

$$r = |1 - p - q| \tag{9}$$

We can now extend these bounds for the k-ary classification problem. We first note that we can add an additional term, z, to Eq. 3 which represents the probability that the learner correctly classifies a data point given that the teacher misclassified it and the teacher and learner disagree. The result is shown in Eq. 10.

$$r = pq_1 + (1-p)(1-q_2)z \tag{10}$$

In binary classification, $z = 1$ and can be ignored. Similarly, the upper bound for k-ary classification will also have $z = 1$ so our upper bound does not change. The lower bound for k-ary classification will have $z = 0$ which leads to a lower bound of 0 if $(1-p) \geq q$. If $(1-p) < q$, then the lower bound does not change from Eq. 9. For the expected bounds, in the k-ary case, $z \leq 1$. We propose two estimates for z. The first estimate considers the number of classes but not the distribution of the classes, $z = 1/(k-1)$. The second estimate factors in the distribution of class labels and is shown in Eq. 11. In this equation, $P(y)$ represents the probability that a data point has a label of y, where $y \in Y$. $P(y)$ can be estimated using the observed frequency of each class.

$$
\begin{aligned}
z &= \sum_{x \in Y} P(x) \sum_{y \neq x \in Y} \frac{P(y)P(y)}{(1 - P(x))(1 - P(x))} \\
&= \sum_{x \in Y} \frac{P(x)}{(1 - P(x))^2} \sum_{y \neq x \in Y} P(y)^2
\end{aligned} \tag{11}
$$

Intuitively, z represents the product of the probability that the teacher assigns a class label of x, the probability that y is the true class label, and the probability that the learner selects the correct class label of y, the result of which is summed over each possible class value and is normalized by the remaining probabilities given that x is not the class label.

Note that the above analysis assumes that teacher–learner agreement is uniform over all classes. We can extend this analysis to consider $\alpha = P(h_1() = x | f_1() = y)$, the probability that the teacher classifies a data point as x given that it has a true label of y. Similarly, we consider $\beta = P(h_2() = y | h_1() = x)$, the probability that the learner classifies the data point as y given that the teacher classified it as x. Both of these probabilities can be estimated without using any labeled data in the learner view. The expected bound is then calculated in Eq. 12. In this case, we no longer explicitly distinguish between the teacher being right and wrong. Instead, these cases are handled implicitly by $y = x$ and $y \neq x$.

$$
r = \sum_{y \in Y} P(y) \sum_{x \in Y} \alpha\beta \tag{12}
$$

In addition to these upper and lower bound learner accuracy estimates, we can also estimate average performance. A simple estimation can be performed by averaging upper and lower bounds, which avoids calculating class distributions and conditional probabilities. This also avoids making explicit assumptions about the probability of the teacher and learner agreeing. Interestingly, when $p \geq q$ and $(1 - p) < q$, then the average of the upper and lower bounds is just q.

Note that the expected accuracy of the learner can be simplified to the underestimate of $r = pq$. All of the above bounds provide insight into the expected learner performance based on characteristics of the teacher, without requiring labeled data in the learner view. To compute these bounds in practice, teacher accuracy can be estimated using its labeled data. Similarly, teacher–learner agreement can be estimated using unlabeled data in both views. Later, we empirically compute performance and compare it with these theoretical bounds.

5 Experimental evaluation

The goal of PECO is to transfer activity knowledge from one sensor platform's trained view to another sensor platform's untrained view. Here we evaluate whether a new sensor platform can learn these activity models without having any labeled activity data in its own view. We observe influence based on the choice of teacher view and consider more than two views in combination. In addition, we compare theoretical bounds with observed performance. Finally, we observe the beneficial effects to the teacher from the transfer, which allows sensor platforms to act as colleagues.

For our experiments, we make use of three activity recognition datasets and the toy dataset. All of the activity recognition datasets contain data from multiple heterogeneous sensor types and all include data from multiple participants. In each experiment, we report results using tenfold cross-validation averaged over all participants.

The toy dataset has two views. Each view is generated independently of the other view and data instances from each view with the same class label are randomly paired together as being the same instance from different views. The views are generated using the Scikit-Learn framework [74]. For the first view, samples are generated using two normally distributed clusters (one for each class) with one informative attribute and 1 non-informative attribute. For the second view, samples are generated using four normally distributed clusters (two per class) with two informative attributes and one non-informative attribute. Each view generates 100 samples (50 per class).

In the Opportunity dataset [75], 4 participants perform 5 repetitions of the following scripted activities: groom, relax, make/drink coffee, make/drink sandwich, clean, and execute a simple-movement drill. Twelve 3-axis wearable accelerometers represent one sensor platform (view 1), and seven wearable inertial measurement units represent the second platform (view 2). We use the same features as Sagha et al. [76] which consist of raw sensor values sampled every 500 ms and averaged over a 5-s window.

In the CASAS PUCK dataset [77] (ailab.wsu.edu/casas/ datasets.html), 10 participants perform 3 repetitions of 6 scripted activities: sweep, take medicine, cook oatmeal, water plants, wash hands, and clean countertops. Activities are performed in a smart home that is equipped with infrared motion sensors and magnetic door sensors (view 1). Each participant wears two 6-axis accelerometers (view 2). Finally, object vibration sensors (view 3) are attached to the broom, dustpan, duster, pitcher, bowl, measuring cup, glass, fork, watering can, hand soap dispenser, dish soap dispenser, medicine dispenser, and medicine bottles. For consistency, we employ the same wearable sensor features as in the Opportunity dataset. For views 1 and 3, the feature vector consists of the number of activations for each sensor during the sampling period.

In the CASAS Parkinson's dataset, 6 participants perform 3 repetitions of the same activities as in the PUCK dataset. In addition to the ambient sensor view (view 1), wearable accelerometer view (view 2), and object sensor view (view 3), a new view is introduced corresponding to Kinect depth cameras (view 4) that were placed in the smart home. We utilize a Kinect API that processes the video data into the 20 (x, y, z) joint positions found in the video.

All of the existing and new algorithms described in this paper can partner with virtually any classifier. We experimented with logistic regression, k-nearest neighbors, support vector machines, and decision trees. No classifier consistently performed best or significantly outperformed the others on average. We report all of our results here based on a decision tree classifier, which performed as well or better than other approaches on average. For completeness, we also report results of the different classifiers on the toy dataset using the bootstrapped method with ten percent of the labels being bootstrapped.

Figure 5 summarizes the size of each view's feature space, and Fig. 6 illustrates the sensor types and locations used in these datasets.

We are ultimately interested in seeing if one sensor platform can successfully transfer activity knowledge to a new platform. First, we consider the baseline performance of each sensor view without transfer learning. Figure 7 plots the performance of the sensor views in decreasing order of recognition performance when the view has all of the available labeled data for training and testing. Performance varies between the platforms. The varied strengths of each view will be utilized in later experiments when we analyze effects of the choice of teacher views on performance. For each experiment, we measure performance as activity classification accuracy (see Eq. 13). Because the datasets exhibit a skewed class distribution,

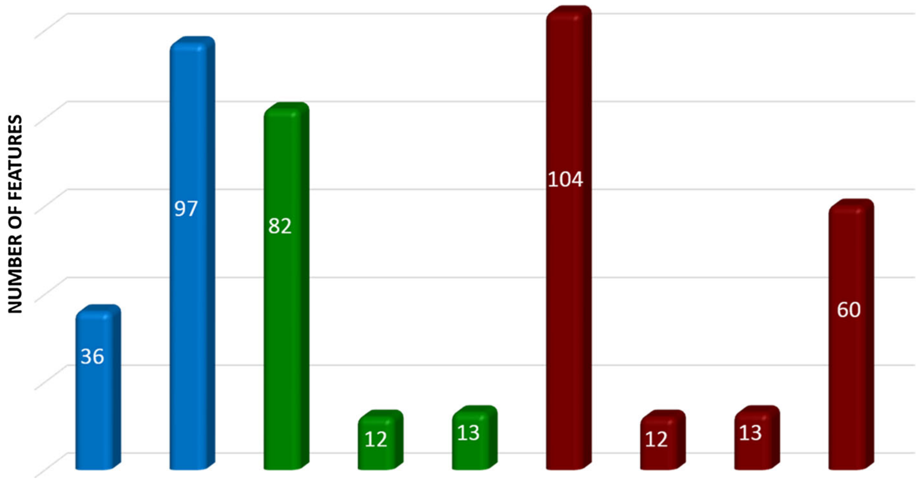


Fig. 5 Feature space sizes for each dataset and view

for some experiments we also report macro-averaged recall scores (see Eq. 14). In both of these equations, N is the total number of instances, K is the number of labels, and A is the confusion matrix where Aij is the number of instances of class i classified as class j.

$$\text{Accuracy} = \frac{1}{N}\sum_{i=1}^{K} A_{ii} \tag{13}$$

$$\text{Avg. Recall} = \frac{1}{K}\sum_{i=1}^{K} A_{ii} \sum_{j=1}^{K} A_{ij} \tag{14}$$

5.1 Informed learning with two views

We initially consider scenarios in which two sensor platforms are used. With informed methods, both platforms have a limited amount of training data and act as colleagues to boost each other's performance based on the different perspectives of the data. For each of the 10 cross-validation folds, the dataset D is split into three pieces: a labeled subset, an unlabeled subset, and a validation subset. The size of the validation subset is always $|D|/10$. We then vary the size of the labeled subset to show how each algorithm performs with different amount of labeled data. To see how the informed multi-view learning algorithms perform in this scenario, we plot classification accuracy as a function of the fraction of the available training data that is provided to both views. We evaluate these algorithms on the Opportunity and PUCK datasets.

In order to provide a basis for comparison, we provide three different baseline approaches. The first baseline, *Oracle*, uses an expert (the ground truth labels) to provide the correct labels for the unlabeled data. The second baseline, *None*, trains a classifier using only the target's labeled subset. The third baseline, *Random*, randomly assigns an activity label weighted by the class distribution observed in the labeled subset. For the Opportunity dataset, we specify (p = 10) examples to label for the Co-Training algorithm and (m = 3) iterations for Co-EM. For the PUCK dataset, we specify (p = 10) for Co-Training and (m = 10) for Co-EM. We

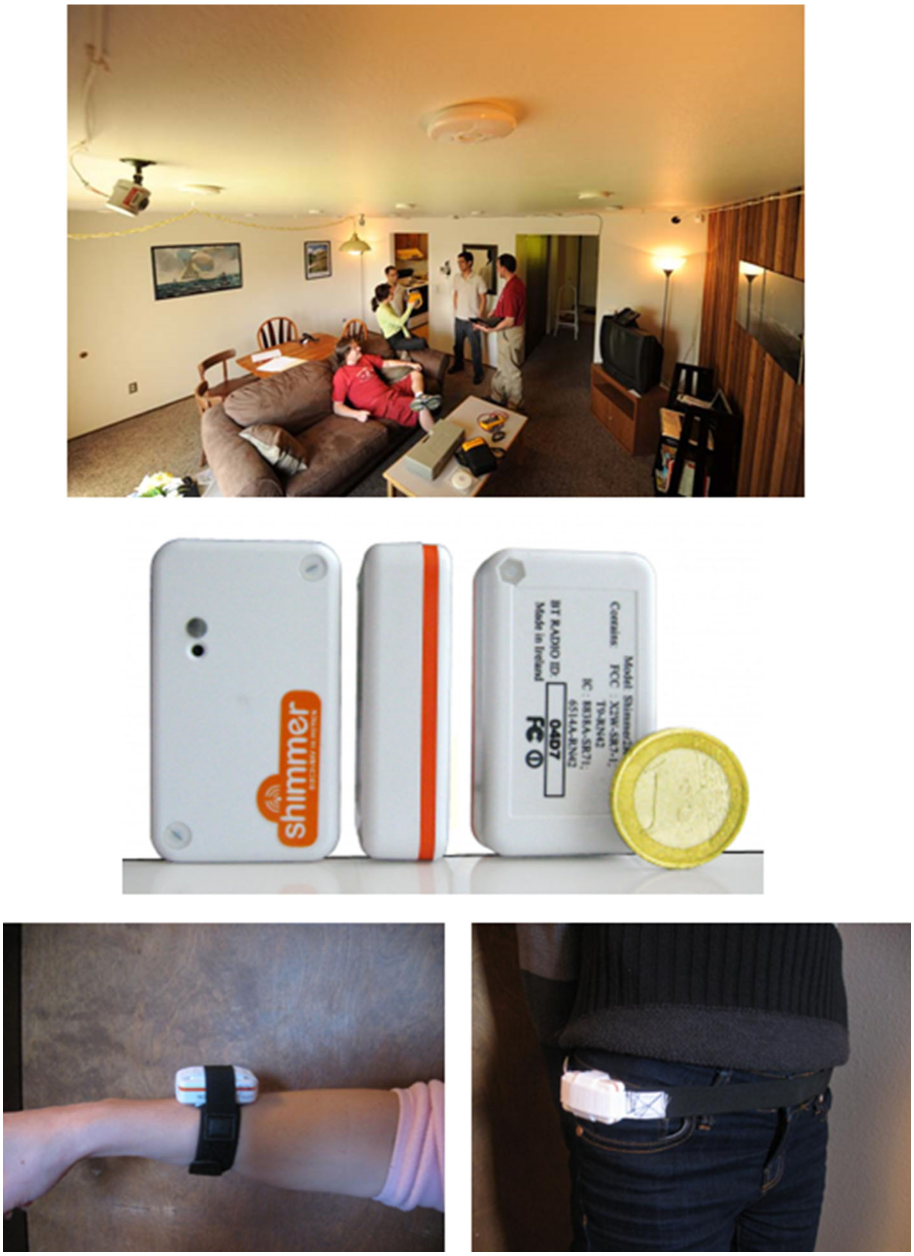


Fig. 6 Motion (*top*), object (*middle*), and accelerometer (*bottom*) sensors in the home

experimented with alternative values for both datasets and algorithms but observed little variation in the resulting accuracies. Figure 8 plots the resulting accuracies, and Fig. 9 plots the average recall scores.

As expected, the multi-view algorithms start at the same accuracy as Random but converge near the same accuracy as Oracle as the amount of labeled data in each view increases. The

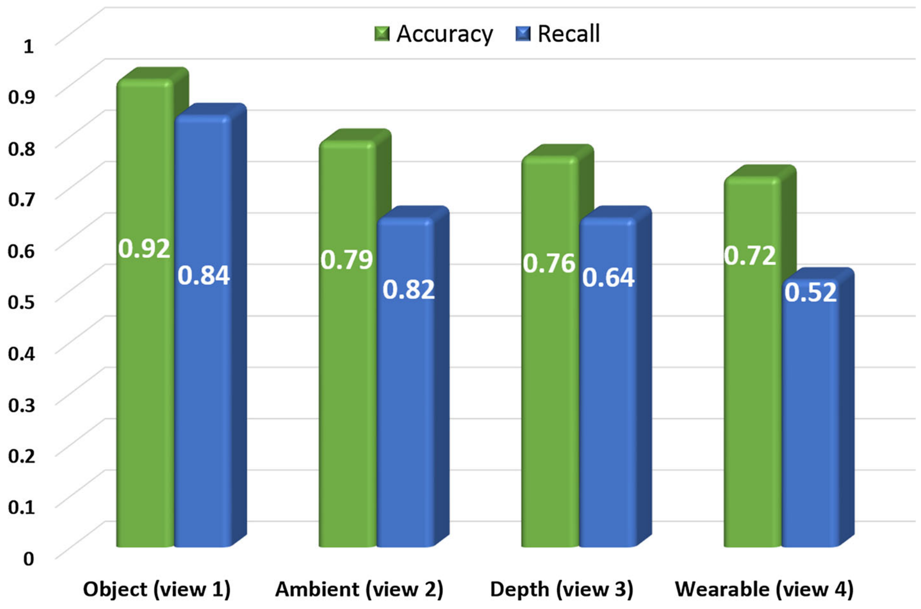
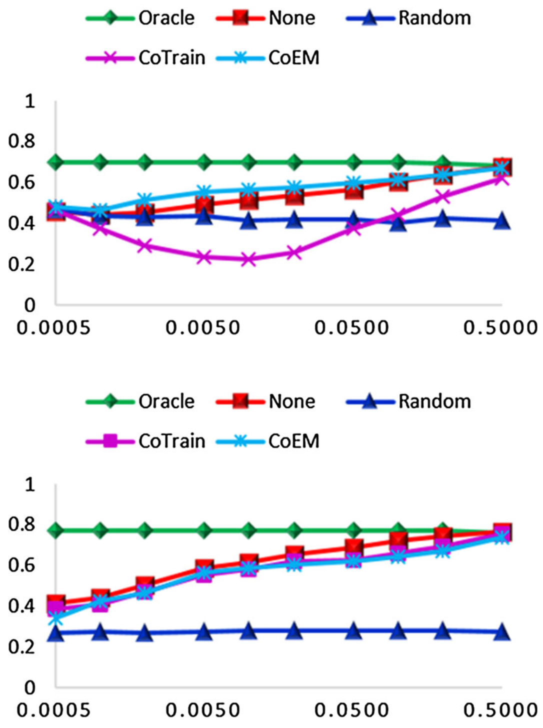


Fig. 7 Accuracy and recall scores for each sensor view using all of the labeled data for the PUCK data

Fig. 8 Classification accuracy of informed multi-view approaches for the PUCK (*top*) and Opportunity (*bottom*) datasets as a function of the fraction of training data that is used by both target and source views (on a log scale)



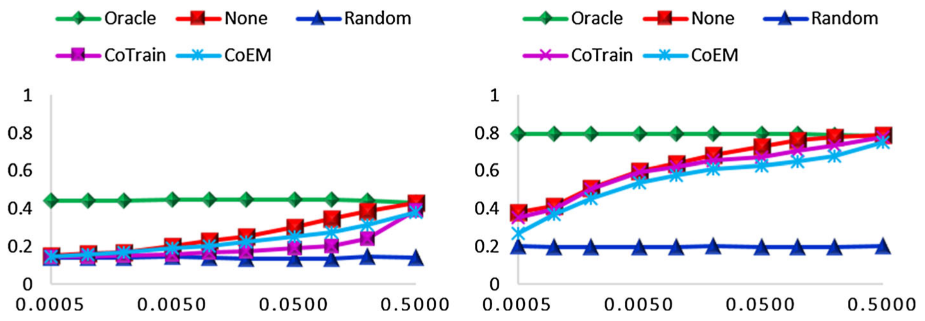


Fig. 9 Average recall of informed multi-view approaches for the PUCK (*left*) and Opportunity (*right*) datasets as a function of the fraction of training data that is used, on a log scale

effect of transfer learning can be seen when comparing the Co-Train and Co-EM curves with the None baseline. The results are mixed (differences between approaches are significant as determined by a one-way ANOVA, $p < 0.05$). In the PUCK dataset, only Co-EM outperforms None, and in the Opportunity dataset None outperforms both approaches. This may be due to the fact that the two views not only violate the conditional independence assumption but in the case of the Opportunity dataset the sensors are quite similar and are therefore highly correlated.

5.2 Uninformed learning with two views

We next repeat the previous experiment using uninformed techniques. This means that the labeled data are only available to the source view. A second sensor platform is later brought online (the target view) but has no training data and is therefore completely reliant on transferred information from the source.

For both the PUCK and Opportunity datasets, we select d for the Manifold Alignment algorithm to be set to the minimum number of dimensions found in the source and target views, which therefore maximizes the information that is retained by the dimensionality reduction step. Figures 10 and 11 show the results. Again, a one-way ANOVA indicates that the differences between the means of the techniques are significant, $p < 0.05$.

As shown in these graphs, Manifold Alignment does not perform well, although it does improve as more data become available in the Opportunity dataset. This is likely due to the invalid assumption that data from both source and target views can be projected onto a shared manifold in a lower-dimensional space. This is particularly problematic for the PUCK dataset because the sensor platforms are very different. In contrast, the teacher–learner method does clearly improve as the amount of labeled source data increases. In fact, it approaches the ideal accuracy achieved by the Oracle baseline.

This leads us to the evaluation of our PECO algorithm on the PUCK dataset. In this case, we divide the original data into four parts: A labeled subset, a bootstrap subset, an unlabeled subset, and a validation subset. As before, the validation subset size is $|D|/10$. The labeled subset is 40% of the remaining data. The bootstrap subset size varies and the unlabeled subset contains the remaining data. Only the source view (view 1) is trained on the labeled data. After training, the source view acts like a teacher and bootstraps the target view (view 2) by labeling the bootstrapped portion of the data. The target view then uses this bootstrapped data to create an initial model. Now PECO can employ an informed technique like Co-Training or Co-EM to refine the model. This simulates the situation in which a well-trained

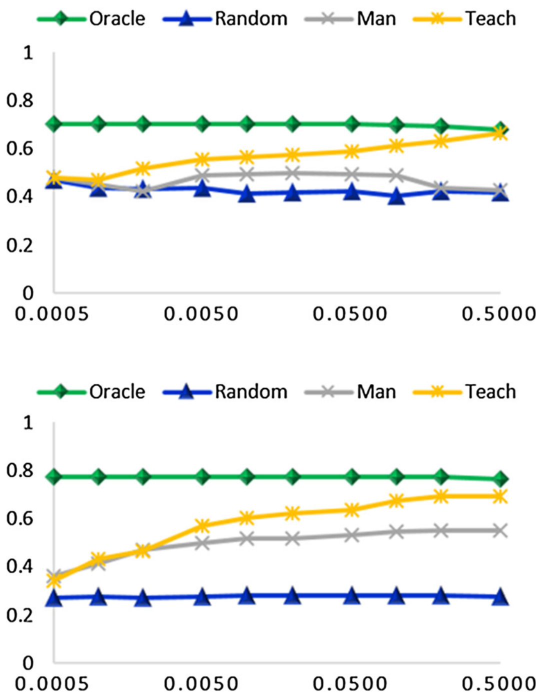


Fig. 10 Classification accuracy of informed multi-view approaches for the PUCK (*top*) and Opportunity (*bottom*) datasets as a function of the fraction of training data that is used by both target and source views (on a log scale)

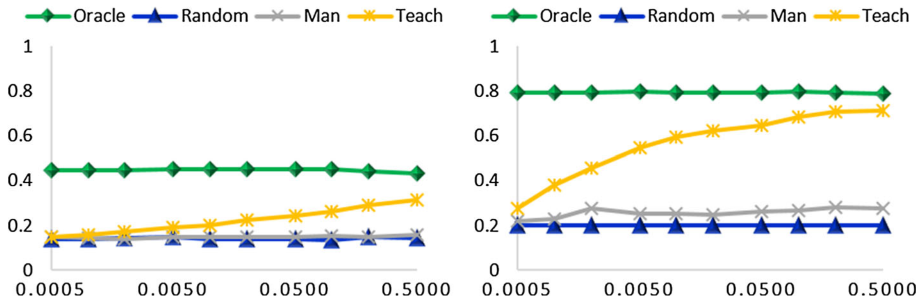


Fig. 11 Recall of uninformed multi-view approaches for the PUCK (*left*) and Opportunity (*right*) datasets as a function of the fraction of training data used by both views on a log scale

activity recognition algorithm is in place for one sensor platform, and we introduce a second sensor platform without providing it any labeled data. We train it using the source view and subsequently allow both views to improve each other's models.

Figure 12 shows the results of this experiment as a function of the amount of labeled data in the source view. In this and the remaining experiments, the weighted recall results are very similar to the accuracy results so these graphs are omitted. As shown here, PECO outperforms teacher–learner when using Co-EM. The performance in fact converges at a level close to that of the informed approaches. Unlike the informed approaches in Fig. 8, however, the PECO method achieved these results without relying on any training data for the target view. This capability will be valuable when it is coupled with a real-time activity recognition system like CASAS (Sect. 4.4) and used to smoothly transition between data sources such as environment sensors, wearable or phone sensors, video data, object sensors,

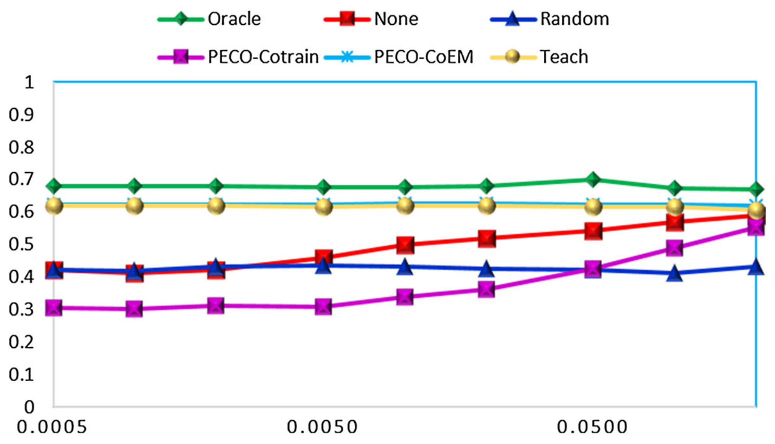


Fig. 12 Comparison of PECO to other methods on PUCK data as a function of the amount of labeled source view data

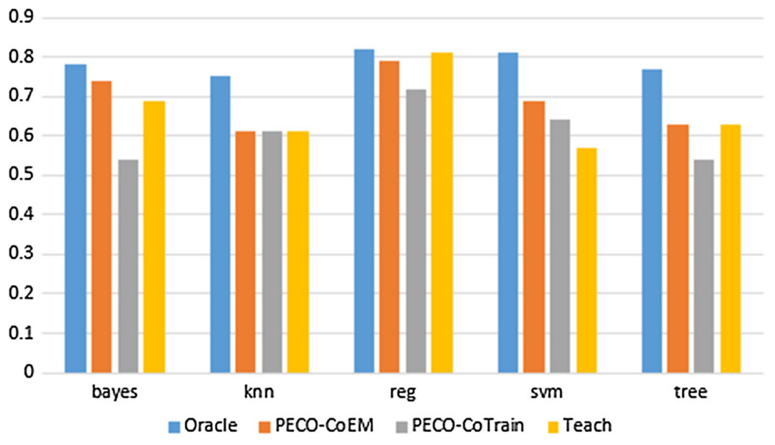


Fig. 13 Comparison of PECO to other methods on the toy dataset with different learning algorithms

or even external information sources such as web pages and social media, without the need for expert guidance or labeled data in the new view.

We also evaluated the PECO methods on the toy dataset using a variety of different machine learning algorithms. The same setup as the previous experiment is applied, with the validation subset size of $|D|/10$, a labeled subset of 40% of the data, a bootstrapped subset of 10% of the data, and the remaining data are unlabeled. Figure 13 shows the results with Naïve Bayes, k-nearest neighbors, linear regression, SVM, and a decision tree.

5.3 Comparing teachers

The earlier experiments provide evidence that real-time activity transfer can be effective at training a new sensor platform without gathering and labeling additional data. However, in the previous experiments the choice of teacher and learner views was fixed. We are interested in seeing how performance fluctuates based on the choice of teacher (source). Intuitively, we

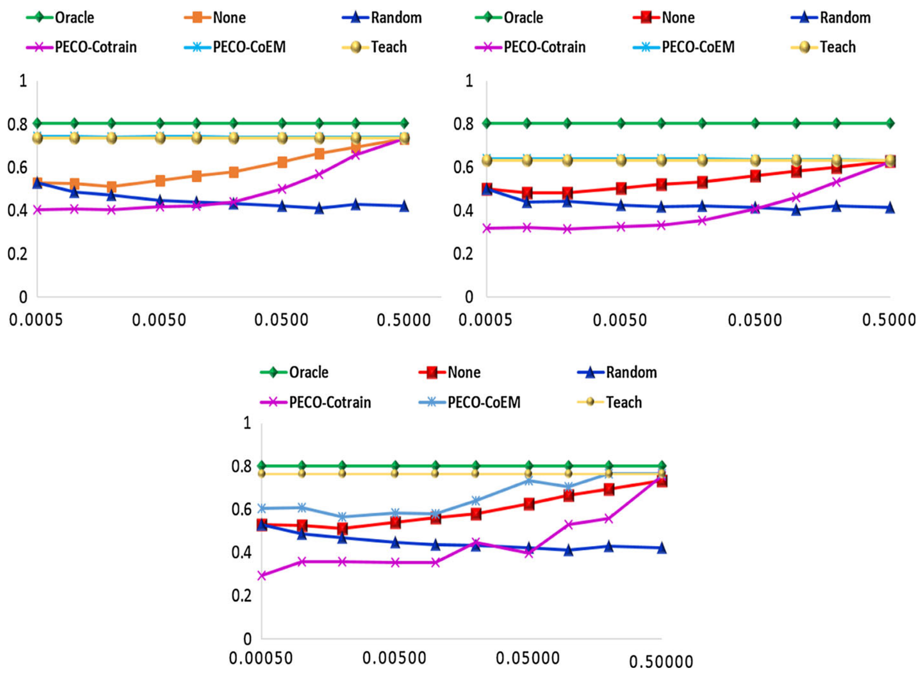


Fig. 14 Classification accuracy as a function of the amount of labeled data for the source view. The target view is ambient sensors, and the source view is object sensors (*top left*), wearable sensors (*top right*), and depth camera sensors (*bottom*)

anticipate that the performance of the learner view will be influenced by the strength of the teacher view as well as other factors such as the similarity of the views.

To investigate these issues, we consider the three views offered in the PUCK dataset and four views offered in the Parkinson's dataset. We fix the target view to be the ambient sensor view. We also note that the accuracy of each view on its own is listed in Fig. 8.

Figure 14 plots the accuracy of the ambient sensor target view using each of the other three sensor platforms as the source view. As before, PECO combined with Co-EM and the teacher–learner algorithm outperform PECO combined with Co-Training, and all reach the performance of the Oracle method. There are differences in performance, however, based on which view acts as the teacher. The depth camera and object views are the highest-performing teachers. This is consistent with the fact that they were the top two performers when acting on their own (see Fig. 7). All three views were effective teachers, which is interesting given the tremendous diversity of their data representations, particularly noting that dense video data successfully transfer activity knowledge to coarse-granularity smart home sensors.

5.4 Adding more views

We now investigate what happens when we add more than two views to the collegial learning environment. In this case, the different sensor views "pass their knowledge forward" by acting as a teacher to the next view in the chain. Alternatively, views with training data can be combined into an ensemble with PECO-E and the ensemble is used to train a student view. These approaches can benefit from utilizing the diversity of data representations. However, introducing extra views may also propagate error down the chain of views.

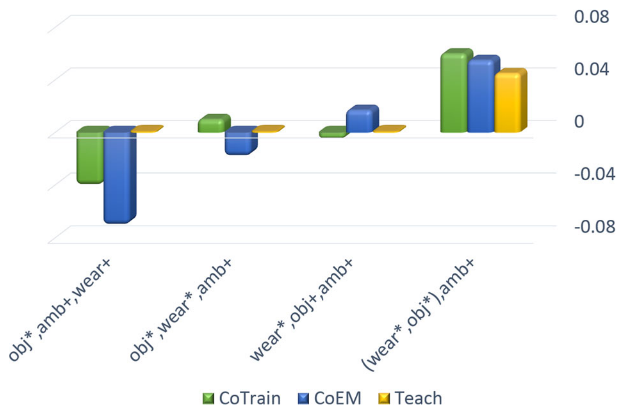


Fig. 15 Differences in accuracy when a third view is added (listed in order of adding). Views marked with *asterisks* have their own labeled data, and views marked with *plus* receive labels from the teacher(s). A PECO-E view ensemble is denoted by (x, y). Results are compared with the two-view case for the wearable target (*left*) and the ambient target (others). The first view is the original teacher in the two-view case. The second view is the extra view being added. The *third view* is the original learner in the two-view case

To explore these effects, we consider different ways of utilizing multiple views and applying them to the PUCK dataset. First, we consider the case where two views act together as teachers for the third target view by both providing labels to data for the target view. Second, we let one view act as a teacher for the second view and then the second view takes over the role of teacher to jumpstart the third view. Third, we let PECO-E create an ensemble of multiple source views and the ensemble acts as a teacher for the target view.

The performance differences between the two-view cases (see Fig. 14) and the three-view case are plotted in Fig. 15. Positive values indicate that the target view benefitted from the additional view and negative values indicate the additional view was harmful. We note that the teacher–learner algorithm is largely unaffected by the additional views unless they are combined into an ensemble classifier. In contrast, PECO combined with Co-Training and Co-EM does experience a noticeable negative or positive effect, depending on the order in which the views are applied. In particular, whenever the view containing wearable sensors (the lowest-accuracy view according to Fig. 7) is added as a teacher, the result lowers the accuracy for the target view (ambient sensors). When the object sensors (the highest-accuracy view) are added, the accuracy is increased.

These results shed some light on the multi-view approaches. PECO/Co-Training treats all views equally. This makes the performance more invariant to view order but also limits accuracy by not giving preference to stronger views. The teacher–learner algorithm is highly dependent on the selection of a good teacher and does not utilize extra views unless they are combined in an ensemble. PECO/Co-EM falls somewhere in the middle, although it too is affected by view order. We observe that ordering views by decreasing accuracy yields the best results. Furthermore, combining source views in an ensemble can mitigate the adverse effects of a poor view order if the relative performances are unknown.

One interesting feature of PECO is that in addition to jump-starting activity recognition on a new sensor platform, it can also improve activity recognition for the source (teacher) platform. This effect is highlighted in Fig. 16 by plotting the recognition accuracy of the source view instead of the target views, as was done in earlier experiments. This process demonstrates the transition from transfer learning to collegial learning between heterogeneous sensor platforms.

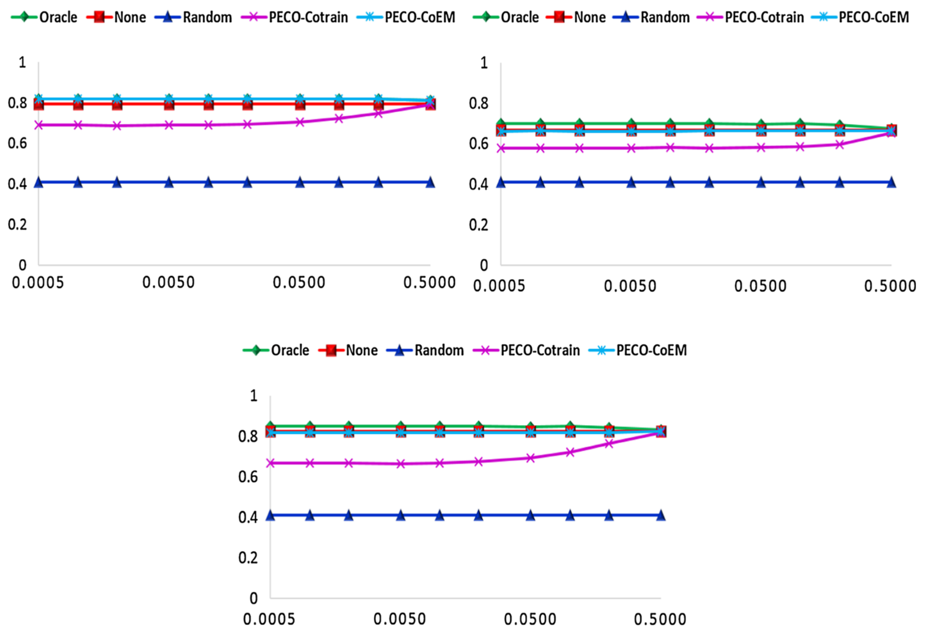


Fig. 16 Source (teacher) view accuracy as a function of the amount of data for source = ambient sensors and target = wearable sensors (*top left*), source = wearable sensors and target = ambient sensors (*top right*), and source = object sensors and target = ambient sensors (*bottom*)

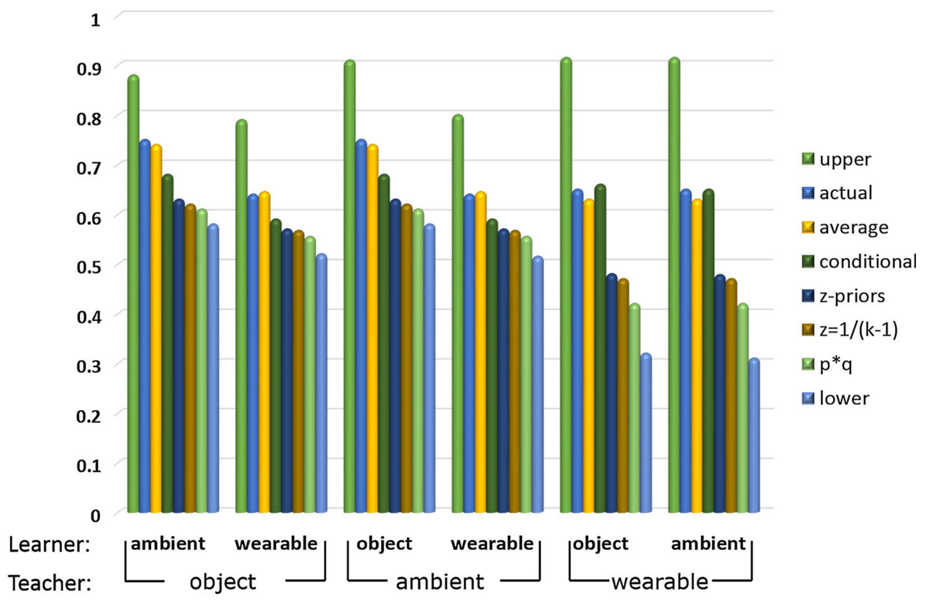


Fig. 17 Target view accuracy bounds using a teacher–learner method. The theoretical bounds are consistent with observed empirical bounds. The conditional estimate, z-priors estimate, $z = 1/(k-1)$ estimate, and $p * q$ estimate all underestimate actual recognition accuracy. The closest estimate is provided by the average of the theoretical upper and lower bounds

5.5 Accuracy

In our final analysis, we validate our theoretical accuracy bounds based on empirical performance. For the expected bounds, we consider the $z = 1/(k - 1)$ bound proposed in Eq. 10, the z-priors bound in Eq. 11, the conditional probability bound in Eq. 12, the average of the upper and lower bounds, and the underestimated expected bound of $p * q$. We evaluate these bounds using tenfold cross-validation on the PUCK data. The values used for teacher accuracy and level of agreement are based on observed performance for the validation set.

Figure 17 shows the results for each teacher–learner view combination. The theoretical upper and lower bounds bound the observed accuracies as well. The simplest estimation $p*q$ of the expected accuracy is the least accurate. As expected, including the $(1 - p)(1 - q)z$ term improves this estimate. The conditional expected bounds provide a closer estimate to the observed accuracy but still underestimate the actual learner accuracy. In addition to providing the simplest estimate, the average of the upper and lower bounds also provides the most accurate estimate of learner accuracy in practice.

6 Conclusion

In this paper, we introduce PECO, a technique to transfer activity knowledge between heterogeneous sensor platforms. From our experiments, we observe that we can reduce or eliminate the need to provide expert-labeled data for each new sensor view. This can significantly lower the barrier to deploying activity learning systems with new types of sensors and information sources.

In addition, we observe that transferring activity knowledge from source to target views with PECO can actually boost the performance of the source view as well. This is useful in situations where the new sensor platform may have more sensitive or denser information than the previous platforms. For example, a set of smart home sensors may transfer knowledge to a data-rich video platform such as the Kinect. In these cases, the target view, or student, is able to construct a more detailed model that benefits the teacher as well and transforms the relationship to that of colleagues.

We have successfully integrated PECO into the CASAS smart home system, which allows multi-view learning to operate in real time as diverse sensor platforms are introduced. In the future, we also want to consider integrating external information sources as well, such as web information, social media, or human guidance. By including a greater number and more diverse sources of information, we contribute to the goal of transforming single activity-aware environments into personalized ecosystems.

Acknowledgements The authors would like to thank Aaron Crandall and Biswaranjan Das for their help in collecting and processing Kinect data. Funding for research was provided by the National Science Foundation (DGE-0900781, IIS-1064628) and by the National Institutes of Health (R01EB015853).

References

1. Solanas A, Patsakis C, Conti M, Vlachos I, Ramos V, Falcone F, Postolache O, Perez-Martinez PA, Di Pietro R, Perrea DN, Martinez-Balleste A (2014) Smart health: a context-aware health paradigm within smart cities. IEEE Commun Mag 52(8):74–81
2. Aggarwal JK, Ryoo MS (2011) Human activity analysis: a review. ACM Comput Surv 43(3):1–47

3. Chen L, Hoey J, Nugent CD, Cook DJ, Yu Z (2012) Sensor-based activity recognition. IEEE Trans Syst Man Cybern Part C Appl Rev 42(6):790–808
4. Ke S-R, Thuc HLU, Lee Y-J, Hwang J-N, Yoo J-H, Choi K-H (2013) A review on video-based human activity recognition. Computers 2(2):88–131
5. Bulling A, Blanke U, Schiele B (2014) A tutorial on human activity recognition using body-worn inertial sensors. ACM Comput Surv 46(3):107–140
6. Reiss A, Stricker D, Hendeby G (2013) Towards robust activity recognition for everyday life: methods and evaluation. In: Pervasive Computing Technologies for Healthcare, pp. 25–32
7. Vishwakarma S, Agrawal A (2013) A survey on activity recognition and behavior understanding in video surveillance. Vis Comput 29(10):983–1009
8. Cook D (2012) Learning setting-generalized activity models for smart spaces. IEEE Intell Syst 27(1):32–38
9. Hagras H, Doctor F, Lopez A, Callaghan V (2007) An incremental adaptive life long learning approach for type-2 fuzzy embedded agents in ambient intelligent environments. IEEE Trans Fuzzy Syst 15(1):41–55
10. Munguia-Tapia E, Intille SS, Larson K (2004) Activity recognition in the home using simple and ubiquitous sensors. In: Pervasive, pp. 158–175
11. van Kasteren T, Noulas A, Englebienne G, Krose B (2008) Accurate activity recognition in a home setting. In: ACM conference on ubiquitous computing, pp. 1–9
12. He W, Guo Y, Gao C, Li X (2012) Recognition of human activities with wearable sensors. J Adv Signal Process 108:1–13
13. Junker H, Amft O, Lukowicz P, Groster G (2008) Gesture spotting with body-worn inertial sensors to detect user activities. Pattern Recognit 41:2010–2024
14. Maurer U, Smailagic A, Siewiorek D, Deisher M (2006) Activity recognition and monitoring using multiple sensors on different body positions. In: Proceedings of the international workshop on wearable and implantable body sensor networks, pp. 113–116
15. Bulling A, Ward JA, Gellersen H (2012) Multimodal recognition of reading activity in transit using body-worn sensors. ACM Trans Appl Percept 9(1):2:1–2:21
16. Lara O, Labrador MA (2013) A survey on human activity recognition using wearable sensors. IEEE Commun Surv Tutor. 15(3):1192–1209
17. Gu T, Chen S, Tao X, Lu J (2010) An unsupervised approach to activity recognition and segmentation based on object-use fingerprints. Data Knowl Eng 69(6):533–544
18. Philipose M, Fishkin KP, Perkowitz M, Patterson DJ, Hahnel D, Fox D, Kautz H (2004) Inferring activities from interactions with objects. IEEE Pervasive Comput 3(4):50–57
19. Buettner M, Prasad R, Philipose M, Wetherall D (2009) Recognizing daily activities with RFID-based sensors. In: International conference on ubiquitous computing, pp. 51–60
20. Kwapisz J, Weiss G, Moore S (2010) Activity recognition using cell phone accelerometers. In: International workshop on knowledge discovery from sensor data, pp. 10–18
21. Zhixian Y, Vigneshwaran S, Chakraborty D, Misra A, Aberer K (2012) Energy-efficient continuous activity recognition on mobile phones: an activity-adaptive approach. In: International symposium on wearable computers, pp. 17–24
22. Moncrieff S, Venkatesh S, West G, Greenhill S (2007) Multi-modal emotive computing in a smart house environment. Pervasive Mob Comput 3(2):79–94
23. Candamo J, Shreve M, Goldgof D, Sapper D, Kasturi R (2010) Understanding transit scenes: a survey on human behavior recognition algorithms. IEEE Trans Intell Transp Syst 11(1):206–224
24. Gill T, Keller JM, Anderson DT, Luke RH (2011) A system for change detection and human recognition in voxel space using the Microsoft Kinect sensor. In: IEEE applied imagery pattern recognition workshop, pp. 1–8
25. Malgireddy MR, Nwogu I, Govindaraju V (2013) Language-motivated approaches to action recognition. J Mach Learn Res 14:2189–2212
26. Wang J, Liu Z, Wu Y (2013) Learning actionlet ensemble for 3D human action recognition. IEEE Trans Pattern Anal Mach Intell 36(5):914–927
27. Bao L, Intille S (2004) Activity recognition from user annotated acceleration data. In: Pervasive, pp. 1–17
28. Ravi N, Dandekar N, Mysore P, Littman ML (2005) Activity recognition from accelerometer data. In: Innovative applications of artificial intelligence, pp. 1541–1546
29. Ward JA, Lukowicz P, Troster G, Starner TE (2006) Activity recognition of assembly tasks using body-worn microphones and accelerometers. IEEE Trans Pattern Anal Mach Intell 28(10):1553–1567
30. Singla G, Cook DJ, Schmitter-Edgecombe M (2010) Recognizing independent and joint activities among multiple residents in smart environments. Ambient Intell Humaniz Comput J 1(1):57–63

31. Lester J, Choudhury T, Kern N, Borriello G, Hannaford B (2005) A hybrid discriminative/generative approach for modeling human activities. In: International joint conference on artificial intelligence, pp. 766–772
32. Amft O, Troster G (2009) On-body sensing solutions for automatic dietary monitoring. IEEE Pervasive Comput 8:62–70
33. Blanke U, Schiele B, Kreil M, Lukowicz P, Sick B, Gruber T (2010) All for one or one for all? Combining heterogeneous features for activity spotting. In: IEEE international conference on pervasive computing and communications workshops, pp. 18–24
34. Wang S, Pentney W, Popescu AM, Choudhury T, Philipose M (2007) Common sense based joint training of human activity recognizers. In: International joint conference on artificial intelligence, pp. 2237–2242
35. Lester J, Choudhury T, Borriello G (2006) A practical approach to recognizing physical activities. In: International conference on pervasive computing, pp. 1–16
36. Krishnan N, Cook DJ (2014) Activity recognition on streaming sensor data. Pervasive Mob Comput 10:138–154
37. Arnold A, Nallapati R, Cohen WW (2007) A comparative study of methods for transductive transfer learning. In: International conference on data mining
38. Elkan C (2011) The foundations of cost-sensitive learning. In: International joint conference on artificial intelligence, pp. 973–978
39. Thrun S (1996) Explanation-based neural network learning: a lifelong learning approach. Kluwer Academic Publishers, Bostom, MA
40. Thrun S, Pratt L (1998) Learning to learn. Kluwer Academic Publishers, Bostom, MA
41. Vilalta R, Drissi Y (2002) A prospective view and survey of meta-learning. Artif Intell Rev 18:77–95
42. Raina R, Battle A, Lee H, Packer B, Ng AY (2007) Self-taught learning: transfer learning from unlabeled data. In: International conference on machine learning, pp. 759–766
43. Hachiya H, Sugiyama M, Ueda N (2012) Importance-weighted least-squares probabilistic classifier for covariate shift adaptation with application to human activity recognition. Neurocomputing 80:93–101
44. Pan SJ, Yang Q (2010) A survey on transfer learning. IEEE Trans Knowl Data Eng 22(10):1345–1359
45. Duan L, Xu D, Tsang I, Luo J (2012) Visual event recognition in videos by learning from web data. IEEE Trans Pattern Anal Mach Intell 34(9):1667–1680
46. Wei B, Pal C (2011) Heterogeneous transfer learning with RBMs. In: AAAI conference on artificial intelligence, pp. 531–536
47. Yang W, Wang Y, Mori G (2011) Learning transferable distance functions for human action recognition. In: Machine learning for vision-based motion analysis, pp. 349–370
48. Feuz K, Cook DJ (2015) Transfer learning across feature-rich heterogeneous feature spaces via feature-space remapping. ACM Trans Intell Syst Technol 6(1):3
49. Zhao Z, Chen Y, Liu J, Shen Z, Liu M (2011) Cross-people mobile-phone based activity recognition. In: International joint conference on artificial intelligence, pp. 2545–2550
50. Cook DJ, Feuz K, Krishnan N (2013) Transfer learning for activity recognition: a survey. Knowl Inf Syst 36(3):537–556
51. Ren Y, Wu Y, Ge Y (2014) A co-training algorithm for EEG classification with biomimetic pattern recognition and sparse representation. Neurocomputing 2(137):212–222
52. Kurz M, Holzl G, Ferscha A, Calatroni A, Roggen D, Troster G (2011) Real-time transfer and evaluation of activity recognition capabilities in an opportunistic system. In: International conference on adaptive and self-adaptive systems and applications, pp. 73–78
53. Hu DH, Yang Q (2011) Transfer learning for activity recognition via sensor mapping. In: International joint conference on artificial intelligence, pp. 1962–1967
54. Daume H, Marcu D (2006) Domain adaptation for statistical classifiers. J Artif Intell Res 26(1):101–126
55. Daume H, Kumar A, Saha A (2010) Co-regularization based semi-supervised domain adaptation. In: Advances in neural information processing systems, pp. 478–486
56. Blitzer J, Dredze M, Pereira F (2007) Biographies, bollywood, boom-boxes and blenders: domain adaptation for sentiment classification. In: Annual meeting of the association for computational linguistics
57. Zhong E, Fan W, Peng J, Zhang K, Ren J, Turaga D, Verscheure O (2009) Cross domain distribution adaptation via kernel mapping. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1027–1036
58. Prettenhofer P, Stein B (2011) Cross-lingual adaptation using structural correspondence learning. ACM Trans Intell Syst Technol 3(1):13
59. Zhou JT, Pan SJ, Tsang IW, Yan Y (2014) Hybrid heterogeneous transfer learning through deep learning. In: AAAI conference on artificial intelligence, pp. 2213–2219
60. Shi X, Yu P (2012) Dimensionality reduction on heterogeneous feature space. In: International conference on data mining, pp. 635–644

61. Yang Q, Chen Y, Xue G-R, Dai W, Yu Y (2009) Heterogeneous transfer learning for image clustering via the social web. In: Joint conference of the annual meeting of the ACL, pp. 1–9
62. Kira Z (2010) Inter-robot transfer learning for perceptual classification. In: International conference on autonomous agents and multiagent systems, pp. 13–20
63. Blum A, Mitchell T (1998) Combining labeled and unlabeled data with co-training. In: Annual conference on computational learning theory, pp. 92–100
64. Sun S (2013) A survey of multi-view machine learning. Neural Comput Appl 23(7–8):2031–2038
65. Wang C, Mahadevan S (2008) Manifold alignment using procrustes analysis. In: International conference on machine learning, pp. 1120–1127
66. Pansiot J, Stoyanov D, McIlwraith D, Lo BPL, Yang GZ (2007) Ambient and wearable sensor fusion for activity recognition in healthcare monitoring systems. In: International workshop on wearable and implantable body sensor networks, pp. 208–212
67. Ugolotti R, Sassi F, Mordonini M, Cagnoni S (2013) Multi-sensor system for detection and classification of human activities. J Ambient Intell Humaniz Comput 1(4):27–41
68. Banos O, Damas M, Pomares H, Rojas I (2013) Activity recognition based on a multi-sensor meta-classifier. In: Advances in computational intelligence: international work conference on artificial neural networks, pp. 208–215
69. Nigam K, Ghani R (2000) Analyzing the effectiveness and applicability of co-training. In: International conference on information and knowledge management, pp. 86–93
70. Kakade SM, Foster DP (2007) Multi-view regression via canonical correlation analysis. In: Learning theory, Springer, pp. 82–96
71. Wall ME, Rechtsteiner A, Rocha LM (2003) Singular value decomposition and principal component analysis. In: A practical approach to microarray data analysis, pp. 91–109
72. Valiant LG (1984) A theory of the learnable. CACM 27(11):1134–1142
73. Cook DJ, Crandall A, Thomas B, Krishnan N (2012) CASAS: a smart home in a box. IEEE Comput 46(7):62–69
74. Pedregosa F, Varoquaux G, Framfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in Python. J Mach Learn Res 12:2825–2830
75. Roggen D, Frster K, Calatroni A, Trster G (2013) The adARC pattern analysis architecture for adaptive human activity recognition systems. J Ambient Intell Humaniz Comput 4:169–186
76. Sagha H, Digumarti ST, del R. Millan J, Chavarriaga R, Calatroni A, Roggen D, Troster G (2011) Benchmarking classification techniques using the opportunity human activity dataset. In: IEEE international conference on systems, man, and cybernetics, pp. 36–40
77. Sahaf Y, Krishnan N, Cook DJ (2011) Defining the complexity of an activity. In: Workshop AAAI on techniques and languages, activity context representation

Kyle D. Feuz received the B.S. and M.S. degrees from Utah State Univesity, Logan, UT, USA, in 2010 and 2011, respectively, and the Ph.D. degree from Washington State University, Pullman, WA, USA, in 2014. He is an Assistant Professor with the Department of Computer Science, Weber State University, Ogden, UT, USA. His research interests include artificial intelligence, multiagent systems, smart environments, and computer security.

Diane J. Cook received the B.S. degree from Wheaton College, Wheaton, IL, USA, in 1985, and the M.S. and Ph.D. degrees from University of Illinois, Champain, IL, USA, in 1987 and 1990, respectively. She is a Huie-Rogers Chair Professor with the School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA, USA. Her research interests include artificial intelligence, machine learning, activity recognition, and smart environments.