# Autonomous Training of Activity Recognition Algorithms in Mobile Sensors: A Transfer Learning Approach in Context-Invariant Views

Seyed Ali Rokni, *Student Member, IEEE* and Hassan Ghasemzadeh, *Senior Member, IEEE*

**Abstract**—Wearable technologies play a central role in human-centered Internet-of-Things applications. Wearables leverage machine learning algorithms to detect events of interest such as physical activities and medical complications. A major obstacle in large-scale utilization of current wearables is that their computational algorithms need to be re-built from scratch upon any changes in the configuration. Retraining of these algorithms requires significant amount of labeled training data, a process that is labor-intensive and time-consuming. We propose an approach for automatic retraining of the machine learning algorithms in real-time without need for any labeled training data. We measure the inherent correlation between observations made by an old sensor view for which trained algorithms exist and the new sensor view for which an algorithm needs to be developed. Our multi-view learning approach can be used in both online and batch modes. By applying the autonomous multi-view learning in the batch mode, we achieve an accuracy of 83.7 percent in activity recognition which is an improvement of 9.3 percent due to the automatic labeling of the data in the new sensor node. In addition to gain the less computation advantage of incremental training, the online learning algorithm results in an accuracy of 82.2 percent in activity recognition.

**Index Terms**—Wearable sensors, machine learning, transfer learning, computational autonomy, optimization

## 1 Introduction

Internet of Things (IoT) is emerging as a promising paradigm for a large number of application domains such as environmental and medical monitoring [1], [2], home automation [3] as well as smart grids [4]. Many of these applications involve humans, where humans and things will operate synergistically [5]. At the heart of these systems is human monitoring where wearable sensors are utilized for sensing, processing, and transmission of physiological and contextual data. Smartphones, wrist-band sensors, smartwatches, necklaces, sports shoes, smart-insoles, and sensors embedded in clothing are only few examples of these sensors. Typically, sensors acquire physical measurements, use computational models such as machine learning algorithms for local data processing, and communicate the results to a gateway or through the Internet [6].

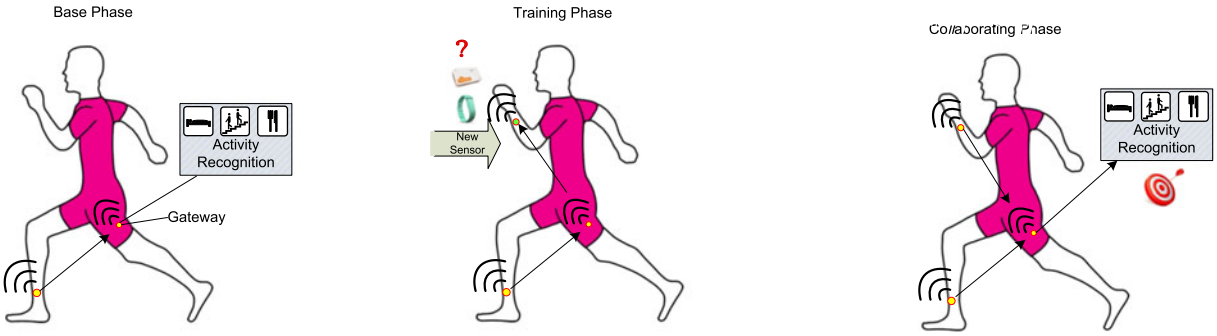
The computational algorithms offer core intelligence of these systems by allowing for continuous and real-time extraction of clinically important information from sensing data. Currently, an implicit assumption in development of machine learning algorithms for human-centered IoT applications is that the training and future data are in the same feature space and have the same distribution [7]. These algorithms, however, need to be reconfigured (i.e., retrained) upon any changes in configuration of the system, such as addition/removal of a sensor to/from the network, displacement/misplacement/mis-orientation of the sensors, replacement/upgrade of the sensors, adoption of the system by new users, and changes in physical/behavioral status of the user. Retraining of the computational algorithms requires collecting sufficient amount of labeled training data, a time consuming, labor-intensive, and expensive process that has been identified as a major barrier to personalized medicine [8].

When used in uncontrolled environments, wearables face many uncertainties that dramatically impact performance of the computational algorithms. For example, it is shown that the accuracy of an activity recognition algorithm drops from 98 to 33 percent due to sensor displacements [9], [10]. Furthermore, prior research shows that sensor displacement increases the error of regression-based Metabolic Equivalent of Task (MET) estimation [11], [12] by a factor of 2.3 times. The reason for such a sudden performance degradation is that computational algorithms learn a model based on a set of training data that is collected in a controlled environment. In reality, however, wearables are deployed in highly dynamic and uncontrolled environments, due to their direct and continuous exposure to end-users and their living environments.

In this paper, we take first steps in developing autonomous and real-time retraining of machine learning algorithms without labeled training data. Specifically, we focus on cases where a new sensor is added to the system. Addressing the problem of expanding computational capabilities from single setting embedded software with predefined configuration to

• *The authors are with the School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA 99164. E-mail: {alirokni, hassan}@eecs.wsu.edu.*

*Manuscript received 28 Oct. 2016; revised 3 July 2017; accepted 26 Dec. 2017. Date of publication 5 Jan. 2018; date of current version 29 June 2018. (Corresponding author: Seyed Ali Rokni.)*
*For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.*
*Digital Object Identifier no. 10.1109/TMC.2018.2789890*




(a) Base setting with two sensors

(b) Training newly added sensor

(c) Collaboration

Fig. 1. Evolution of a wearable network with plug-n-learn feature. Initially, the system consists of two sensors (a). Existing sensors, in the source view, contribute to training the new sensor, in the target view (b). When the training phase is completed, new sensor contributes to the system for Activity Recognition (AR), as shown in (c).

a dynamic setting where sensors can be added dynamically is challenging. In such cases, successful knowledge transfer is needed to improve the learning performance by avoiding expensive data collection, segmentation, and labeling efforts.

Our pilot application in this paper is activity recognition where the goal is to detect physical movements of the user based on wearable sensor measurements. Several studies leveraged transfer learning in instance level [13], [14], [15] or signal level [16] to develop a system capable of performing reliable activity recognition in a dynamic sensor environment. However, the accuracy of target model is bounded by the accuracy of source model. We address this problem by proposing the *plug-n-learn* framework which transfers activity recognition capabilities of one sensor to another where the collective network of the two sensors achieves much higher accuracy performance. Our approach allows to transfer machine learning knowledge from an existing sensor, called *source view*, to a new sensor, called *target view*, and these capabilities are augmented through the sensing observations made by the target view with different view point. We call this approach a multi-view learning and formulate this problem using Linear Programming (LP). We propose a greedy heuristic to solve this optimization problem. First, we devise a clustering approach which runs in the combine feature space of *source* and *target* with higher separability. Then, the problem is modeled as a weighted perfect matching of clusters and labels. Our multi-view learning method runs in both online and batch modes; while in online mode we update the matching weights and consequently the model as new observations are made in *target* over time, in the batch method, the clustering and matching perform after sufficient observations are made by both *source* and *target*. We evaluate the performance of the proposed autonomous learning approach using real data collected with wearable motion sensors.

## 2 Plug-N-Learn Framework

Fig. 1 shows the *plug-n-learn* framework as a new sensor is added to the system and a machine learning model (i.e., activity recognition classifier) is trained using unlabeled data captured by the new sensor. Initially, as shown in Fig. 1a, the network consists of a fixed number of sensors (e.g., two sensors worn on 'Thigh' and 'Leg') with trained machine learning models for activity recognition. We refer to the collection of the existing sensors as *'source view'*. A new sensor (e.g., a 'wrist-band') is added to the system as shown in Fig. 1b. We refer to this newly added sensor as *'target view'*. Our multi-view learning approach captures sensor data in both views simultaneously, labels instances captured by the target based on the collective knowledge of source and target, and constructs a new classification algorithm for activity recognition. The system can now detect activities with higher accuracy in a collaborative fashion as shown in Fig. 1c.

### 2.1 Activity Recognition

A sensing device typically has several sensors for capturing different states of the user (e.g., body acceleration), an embedded software module to perform signal processing, machine learning and information extraction, and a radio for data transmissions. When a decision is made on the collected data, the results can be used locally or forwarded to a back-end storage in the cloud for further processing, and to provide decision support. In this paper, however, we limit our focus to the network around the user.

Early research that formulated activity recognition from accelerometer sensor data as a classification problem was conducted over a decade ago [17]. In activity recognition, readings from motion sensors such as accelerometers, magnetometers, and gyroscopes undergo signal processing and machine learning techniques to detect human movements such as 'walking', 'running', or 'sitting'. Each sensing node processes sensor readings through a chain of embedded software for signal processing and machine learning. The signals that are sampled by each sensor node are first passed through a filter to reduce high frequency noise. The next phase is segmentation which is intended to identify 'start' and 'end' points of the movements being classified. The next module is 'feature extraction' which is responsible for computing statistical and/or morphological characteristics of the signal segment. These features are typically extracted in time domain and/or frequency domain [18]. Features represent different attributes of the signal such as 'peak-to-peak amplitude', 'standard deviation', and 'mean value' [19]. Each feature could have a single value such 'mean' of the signal or could be a vector of values such as 'binned distribution' [20]. Finally, a trained classification

model uses extracted features to determine the current activity of the user. In this study, we focus on the classification module when a new sensor without a trained classifier is added to the system.

### 2.2 Problem Definition

An observation $X_t$ made by a wearable sensor at time '$t$' can be represented as $D$-dimensional feature vector, $X_t = \{f_{t1}, f_{t2}, \ldots, f_{tD}\}$. Each feature is computed from a given time window and a marginal probability distribution over all possible feature values. The activity recognition task is composed of a label space $\mathcal{A} = \{a_1, a_2, \ldots, a_m\}$ consisting of the set of labels for activities of interest, and a conditional probability distribution $P(\mathcal{A}|X_t)$ which is the probability of inferring label $a_j \in \mathcal{A}$ given an observed instance $X_t$. Although a trained sensor can detect some activities, its observations are limited to the body segment on which the sensor is worn. Therefore, some activities are not recognizable in one sensor's view point. For example, a sensor worn on 'Leg', although expert in detecting lower-body activities such as *walking*, cannot detect activities such as 'eating' that involves upper-body motions. To increase the accuracy of activity recognition, it is desirable to add more sensors to the network to cover a larger set of physical activities. We note that advances in embedded sensor design and wearable electronics allow end-users to utilize new wearables such as smart watches and smart shoes. In such a realistic scenario, the newly added sensor has not been trained to collaborate with existing sensors in detecting physical activities. Particularly, in our proposed method, sensors observations generated by the target view, which are presented in a feature space, are combined with labels that are received from the source views. These source-generated labels are then calibrated through a label refinement process that examines consensus among target observations that reside in each cluster. Our autonomous transfer learning algorithm expands activity recognition capabilities of the system beyond a set of activities that are recognizable by existing sensor by allowing the system to autonomously learn new activities that were not recognizable previously. Without loss of generality, in our formulation of this multi-view learning problem, we assume that the source view consists of a single sensor.

**Problem 1 (Synchronous Multi-View Learning).** *Let $S = \{s_1, \ldots, s_k\}$ be a set of $k$ sensors. Let $s_r \in S$ be an existing sensor with a trained classification model. Furthermore, let $s_t \in S$ be a sensor newly added to the network. The sensors $s_r$ and $s_t$ are also referred to as* source *and* target *respectively. Moreover, let $\mathcal{A}$=$\{a_1, a_2, \ldots, a_m\}$ be a set of $m$ activities/labels that the system aims to recognize, and $X$=$\{X_1, X_2, \ldots, X_N\}$ the set of $N$ observations made by $s_t$. The Synchronous Multi-View Learning (SML) problem is to accurately label observations made by $s_t$ with the assistance of $s_r$ such that the mislabeling error is minimized. Once these instances are labeled, it is straightforward to develop a new activity recognition classifier using the labeled data.*

We note that transferring training data or classifier from 'source' to 'target' is ineffective for the purpose of activity recognition [21]. The main reason is that the feature spaces of the two sensors/views are completely different. Moreover, in practical applications, the training data is not available or the sensor is commercially trained and functions as a black-box. Thus, our goal is to devise an approach that accurately labels observations made by the new sensor with assistance from the source view.

### 2.3 Problem Formulation

The SML problem described in Problem 1 can be formulated as follows:

$$\text{Minimize} \quad \sum_{i=1}^{N}\sum_{j=1}^{m} x_{ij}\epsilon_{ij} \tag{1}$$

$$\text{Subject to:} \quad \sum_{i=1}^{N} x_{ij} \leq \Delta_i, \qquad \forall j \in \{1, \ldots, m\} \tag{2}$$

$$\sum_{j=1}^{m} x_{ij} = 1, \qquad \forall i \in \{1, \ldots, N\} \tag{3}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i, j, \tag{4}$$

where the optimization is done over binary variable $x_{ij}$ which indicates whether or not instance $X_i$ is assigned activity label $a_j$. Furthermore, $\epsilon_{ij}$ denotes error due to such a labeling. The constraint in (2) guarantees that $a_j$ is assigned to at most $\Delta_i$ instances, where $\Delta_i$ is the upper bound on the number of instances with $a_j$ as their possible labels. This constraint can be used to encode strong prior contextual knowledge about the occurrence of various activities. Since each instance must be assigned to exactly one actual activity label, the constraint in (3) guarantees that each instance $X_i$ receives one and only one label. If $\epsilon_{ij}$ is known, this problem can be easily reduced to Generalized Assignment Problem. The main challenge in solving the SML problem is that the ground truth labels are not available in the target view and therefore $\epsilon_{ij}$ are unknown *a priori*. In Section 3.4, we propose a method to estimate $\epsilon_{ij}$.

Each sensor has a limited ability in detecting activities depending on the physical placement of the sensor on the body. That is, each sensor is expert in detecting some activities and is incapable of accurately detecting other activities. We define an 'ambiguity relation' to represent this concept.

**Definition 1 (Ambiguity Relation).** *Let $s_p$ refer to a sensor on body location $p \in P$. An* ambiguity relation *$AR$ is defined as a mapping $P \times \mathcal{A} \rightarrow \mathcal{A}$ such that for two activities $a_i, a_j \in \mathcal{A}$, $((p, a_i), a_j) \in AR$* if and only if, *there exists an observation instance with predicted label $a_i$ made by $s_p$ which corresponds to the actual activity $a_j$.*

The ambiguity relation represents confusion in detecting a particular activity given an instance of the sensor data. We use the concept of ambiguity relation to determine deficiency of a sensor in detecting particular activities given a collection of training instances in the source view. To quantify this deficiency of 'source' with a given location $p$ and each $((p, a_i), a_j) \in AR$, we define $L_p(a_i, a_j)$ as the *likelihood deficiency* for sensor $s_p$ to incorrectly detect activity $a_i$ as $a_j$.

**Definition 2 (Likelihood Deficiency).** *Given a sensor $s_p$ and a pair of activities $(a_i, a_j) \in \mathcal{A}$, where $((p, a_i), a_j) \in AR$. Let $n_p(a_i, a_j)$ denote the number of instances of activity $a_j$ incorrectly detected by $s_p$ as $a_i$. The likelihood deficiency $L_p(a_i, a_j)$ is given by*

$$L_p(a_i, a_j) = \frac{n_p(a_i, a_j)}{\sum_{a_r \in \mathcal{A}} n_p(a_i, a_r)}. \quad (5)$$

We note that $((p, a_i), a_j) \notin AR$ implies that $L_p(a_i, a_j)$=0. As soon as the likelihood deficiencies are computed, the source view, instead of sending the predicted label, will transmit its predicted label combined with likelihood of the actual label. We call this complex label a *semi-label* generated by 'source' and transmitted to 'target' for the purpose of autonomous learning.

**Definition 3 (Semi-Label).** *A* semi-label $SL_i$ *generated by sensor* $s_p$ *worn on location* $p$ *is a 2-tuple* $(a_i, Q)$ *where* $|Q| = |\mathcal{A}|$ *and* $Q_j = L_p(a_i, a_j)$. *In other words,* $Q$ *is a likelihood vector of all possible actual labels when predicted label is* $a_i$.

For example, assume $\mathcal{A}$ = {'standing', 'eating', 'writing'}. By computing the sensor deficiency measurements, we will know that when the sensor detects the current activity as 'eating', 50 percent of the times, the actual activity is 'eating', 40 percent of the times it is 'writing', and 10 percent of the times, the actual activity is 'standing'. Therefore, when the sensor detects an activity as 'eating' it generates the semi-label $(0.1, 0.5, 0.4)$ associated with the current observation. In this paper, we use confusion matrix of the classification algorithm in the source view to retrieve values of the likelihood deficiency function.

### 2.4 Synchronous Multi-View Learning

In this section, we introduce *synchronous multi-view learning*, shown in Algorithm 1, as the core element of our knowledge transfer module in our *plug-n-learn* framework.

**Algorithm 1.** Synchronous Multi-View Learning

**while** (more learning is required) **do**
  ***Source view* at time** $t$
    Observes $X^t$ and performs activity recognition.
    Detects activity $a^t$.
    Computes *semi-label* $SL^t$ for activity $a^t$.
    Sends $(X^t, SL^t)$ to the target view in
  ***Target view* at time** $t$
    Observes $X'^t$ from its local sensors
    Receives semi-label $(X^t, SL^t)$ from source
    Combines $X^t$ and $X'^t$ to obtain $XX'^t$
    Assigns $SL^t$ to $XX'^t$.
    **if** Online Mode **then**
      Run Online Minimum-Error Exact Labeling (O-MEEL)
**if** Batch Mode **then**
  Run Batch Minimum-Error Exact Labeling (B-MEEL)

We assume that the sensors within the two views are worn on the body at the same time while the user performs physical activities. Because the source and target sensors may have different clocks, the target sensor needs to know to which observation each semi-label corresponds. We resolve the problem of different clocks by first synchronizing source and target sensors.

Fig. 2 illustrates the synchronous multi-view method. At time $t$, source sensor has it own observation $X^t$ of the current activity and its classifier detects the activity as $a^t$. However, prediction of the source sensor is not completely accurate and is represented as a semi-label $SL^t$. Therefore,

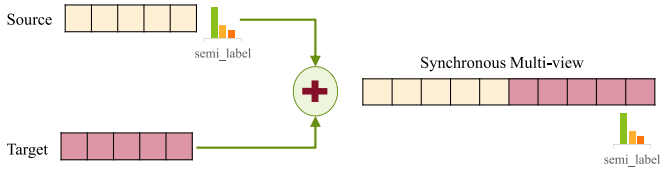



Fig. 2. Synchronous multi-view: Observations of both views combine with the semi-label of source.

source sensor transmits $(X^t, SL^t)$, its own observation and predicted semi-label, to the target sensor. Simultaneously, target sensor observes $X'^t$, the current activity in target feature space. In synchronous multi-view model, target sensor, combines its own observation with observation and semi-label sent by 'source'.

We present two modes of the SML algorithm those include batch and online. The online mode requires less memory usage and distributes the computation over time while its accuracy might be lower than batch mode. On the other hand, using batch mode, gaining the higher accuracy is at the cost of more memory usage and one-time intense computation. We will show that the online mode could reach a high accuracy after training for a sufficient long time. When SML runs in the online mode, upon availability of a new observation, it performs Online Minimum-Error Exact Labeling (O-MEEL), labels new instance and updates the already assigned labels. On the other hand, in the batch mode, the SML waits until sufficient amount of instances for each activity are accumulated. It then applies the B-MEEL problem on the entire data and assigns optimal labels to all instances in the target view.

## 3 Minimum Error Exact Labeling

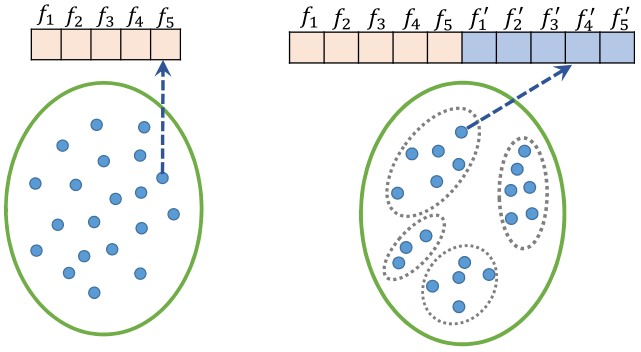
In our multi-view learning approach, the source sensor acts as a 'Teacher' and sends a semi-label of the current activity to the target sensor in real-time. The target sensor, also called 'Learner', learns from the information provided by 'Teacher' and boosts the labeling accuracy by combining its local observations with the received semi-labels. Thus, the challenging task is how to convert semi-labels to exact-labels such that the overall labeling error is minimized.

**Problem 2 (Minimum-Error Exact Labeling).** Given a set of (instance, semi-label) pairs $\{(X_1, SL_1), (X_2, SL_2), \ldots, (X_n, SL_n)\}$ provided by a 'source' $s_r$, Minimum-Error Exact Labeling (MEEL) problem is to assign a label $a_j \in A$ to each instance $X_i$ for all such instances such that the amount of mislabeling is minimized.

We aim to use semi-labels provided by 'source' and develop a solution to the MEEL problem in order to enhance the labeling accuracy. To address this problem, we not only rely on the knowledge of 'source' (i.e., semi-labels) but also combine that knowledge with the observations made by 'target' to develop an accurate activity recognition model. Since the number of activities of interest is naturally much less than the number of instances, we enhance the efficiency of our approach by breaking the solution into two steps: *instance clustering* and *cluster labeling*.

### 3.1 Instance Clustering

As shown in Algorithm 1, the source view acts as 'Teacher' and transfers knowledge to 'Learner'. However, 'Teacher' is




(a) Source feature space (b) Multi-view feature space

Fig. 3. Single-view (source) versus multi-view feature space. (a) Instances in one view may not be separable, however by combining them into (b) multi-view feature space, they become separable.

not completely accurate because all observations of different activities are not entirely separable in the feature space 'Teacher. By combining the observation of 'Teacher' and 'Learner', analogous to kernel method [7], we represent observations in a higher dimensional space with higher likelihood of separability. Therefore, multi-view feature space enables distinguishing more observations and leads to more effective clustering. An example of this process is shown in Fig. 3. While in Fig. 3a instances in feature space of 'Teacher' are not easily separable, representing them in a higher dimensional space, Fig. 3b, enables us to partition the data.

Consider a symmetric distance function $d : X \times X \rightarrow R^+$ on feature space $X$ such that the function satisfies $d(x, x) = 0$ and is being invoked by the clustering algorithm. In particular, we assume that $d$ satisfies triangle inequality.

**Definition 4 (Clustering).** *A* clustering *of $\mathcal{X}$ returns a set of clusters $C$ ={$C_1$, $C_2$, ..., $C_k$} such that $\forall i \neq j, C_i \cap C_j = \emptyset$, $X = \cup_{i=1}^{k} X_i$*

The sensor in the 'target' view, based on its placement on the body, has some discriminating features to separate instances of ambiguous classes of the source sensor represented by semi-labels. We use $K$-Means [7] to cluster instances of 'target' into $k$ groups where $k$ denotes the number of activities in $\mathcal{A}$.

### 3.2 Optimal Labeling

Even though, instances can be clustered with a high quality in the multi-view feature space, there is not trivial mapping between clustered instances and their corresponding activities. Assume that a clustering algorithm groups instances gathered by the target view into $k$ clusters $C$ ={$C_1$, $C_2$, ..., $C_k$}, $k \ll N$. We find optimal label for each cluster such that if the label of the cluster propagates to the instances within that cluster, the overall misclassification error is minimized.

By clustering instances in the multi-view feature space, the optimization problem in Section 2.3 can be re-written as follows:

$$\text{Minimize} \quad \sum_{i=1}^{k}\sum_{j=1}^{m} y_{ij}\epsilon_{ij} \tag{6}$$

$$\text{Subject to:} \quad \sum_{i=1}^{k} y_{ij} = 1, \quad \forall j \in \{1, \ldots, m\} \tag{7}$$

$$\sum_{j=1}^{m} y_{ij} = 1, \quad \forall i \in \{1, \ldots, k\} \tag{8}$$

$$y_{ij} \in \{0, 1\}, \quad \forall i, j \tag{9}$$

$$m = k, \tag{10}$$

where the optimization is over binary variable $y_{ij}$ indicating whether or not cluster $C_i$ is assigned label $a_j$, and $\epsilon_{ij}$ denotes the assignment error. With the clustering, we can estimate $\epsilon_{ij}$ by measuring the amount of disagreement in labels among individual instances within each clusters. Note that instances within each cluster already carry semi-labeling information provided by 'source'. Any conflict between the semi-label of an instance and its cluster label results in an assignment error for that instance. It is reasonable to assume that, with sufficient accumulated data, the number of clusters is equal to the number of activities of interest. Therefore, we assume $k = m$ in the revised formulation. The constraints in (7) and (8) guarantee that each activity label is assigned to exactly one cluster and each cluster is assigned one and only one label.

The aforementioned problem is in slake form and can be solved using *Simplex* algorithm whose time complexity is not polynomial [22]. However, constraints in (9) and (10) can transform the problem into a classic assignment problem. Because the nature of the problem is an assignment problem, it can be modeled using a bipartite graph and solved as a matching problem.

### 3.3 Weighted Labeling Graph

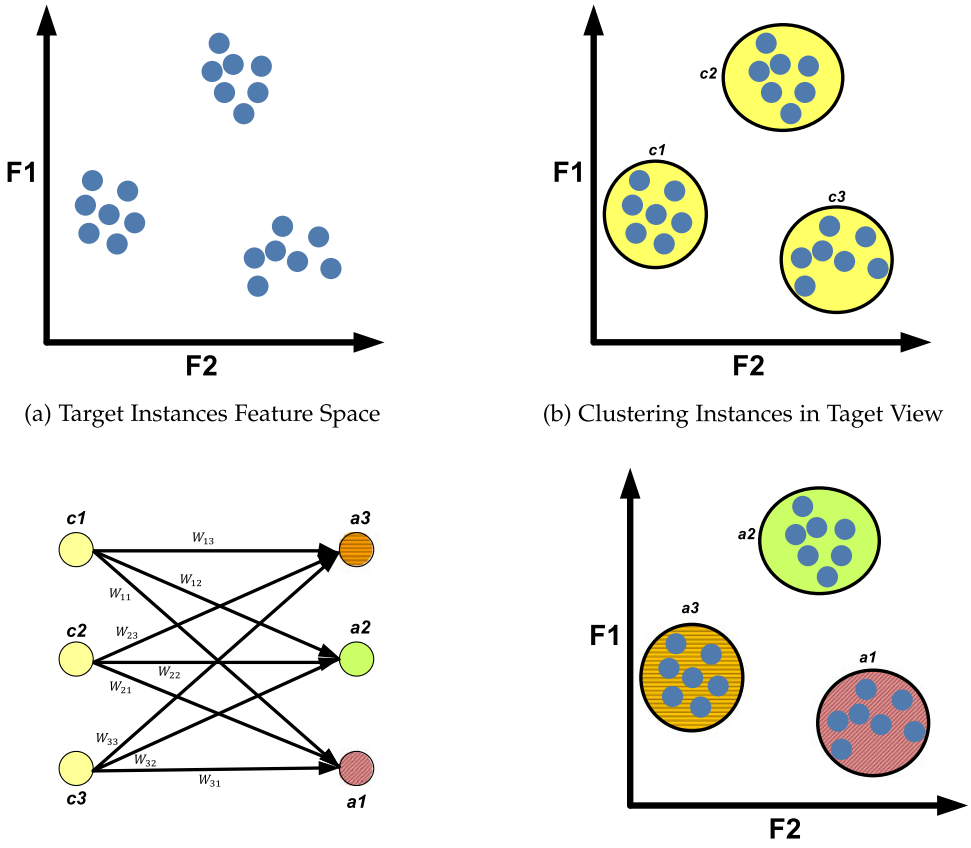
We construct a weighted complete bipartite graph to model the MEEL problem based on the clustered instances of the target view.

**Definition 5 (Weighted Labeling Graph (WLG)).** *Let $\mathcal{C}$ be a set of clusters, each containing a set of instances for target sensor and $\mathcal{A}$ set of activities of interest. A weighted labeling graph $G(V, E, W)$ is a weighted complete bipartite graph such that $V$ = {$\mathcal{C} \cup \mathcal{A}$} is the set of nodes in $G$ such that $|\mathcal{C}| = |\mathcal{A}|$. $E$ refers to the set of edges such that an edge connecting cluster $C_i \in C$ to activity $a_j \in \mathcal{A}$ has a weight of $W_{ij}$ which is equivalent to the assignment error calculated by measuring the amount of disagreement between each instance's semi-label and the label of the cluster that instance belongs to. For impossible assignments $W_{ij} = \infty$.*

In the simple example shown in Fig. 4, the semi-labels provided by 'source' are used to label the 'target' instances shown in Fig. 4a. First, the data instances are clustered as shown in Fig. 4b. The clustering results are then used to construct a labeling graph shown in Fig. 4c. Finally, the assigned labels propagate to the instances of each cluster as shown in Fig. 4d.

### 3.4 Estimating Mis-Classification Error

In practice, we do not have access to actual labels of activities being performed in real-time. Therefore, we cannot directly compute edge weights in $G$. In order to estimate the mis-classification error, we utilize the semi-labels. Recall that semi-labels consist of likelihood vector of all possible actual labels due to a prediction made by *source*. Therefore,




(a) Target Instances Feature Space

(b) Clustering Instances in Taget View

(c) Weighted Labeling Graph

(d) Label Propagation

Fig. 4. Synchronous Multi-view Learning starts consists of (a) labeling target instances by semi-labels and continues with (b) clustering target instances. (c) A weighted bipartite graph is built on clusters and activities and after finding the best assignment, and (d) cluster labels are propagated to their instances.

a straightforward estimator for the error of assigning label $a_j$ to cluster $C_i$ is the $j$th element of $W_i$ vector computed by averaging over likelihood vectors of all instances of cluster $C_i$. In other words

$$W_i = \frac{\sum_{(a,Q)\in C_i} Q}{|C_i|}, \tag{11}$$

where $(a, Q)$ are semi-label pairs of instances in cluster $C_i$.

### 3.5 Labeling Algorithm

We propose a simple polynomial time algorithm to solve the MEEL problem. Our goal is to assign exactly one class label to each cluster such that the total cost of the assignment is minimized. This problem can be viewed as a weighted matching problem. Since $G(V, E, W)$ is a complete bipratite graph and $|C| = |\mathcal{A}|$, we can find a perfect matching in $G$ to minimize the assignment costs (Fig. 4c) using the well-known *Hungarian* algorithm [23] which runs in $O(n^3)$ where $n = |C| = |\mathcal{A}|$. The algorithm starts with an empty matched list of nodes and at each stage it increases the number of matched pairs by one. After $n$ stages, all graph nodes are matched and the algorithm terminates. Using an appropriate data structures [24], [25], *Hungarian* algorithm at each stage requires $O(n^2)$ arithmetic operations; and consequently, the computational complexity of matching all $n$ pairs is $O(n^3)$. In has been shown that the Hungarian algorithm is complete and optimal [24], [26]. Finally, we propagate the assigned label of the clusters to the instances within the cluster (Fig. 4d) in $O(n)$.

**Algorithm 2.** Greedy Algorithm for B-MEEL

**INPUT:** Set $(X, SL)$
**OUTPUT:** finalSet
  $PA \leftarrow$ getPossibleLabels($SL$)
  $C \leftarrow$ cluster($X$)
  $G \leftarrow$ buildBipartiteGraph($C$, $PA$)
  $(C, A) \leftarrow$ findBestAssignment($G$)
  finalSet $\leftarrow \emptyset$
  **for** each $(C_i, a_j) \in (C, A)$ **do**
    **for** each $X_k \in C_i$ **do**
      finalSet $\leftarrow$ finalSet $\cup\, (X_k, a_j)$

The input to the batch labeling algorithm, shown in Algorithm 2, is $(X, SL)$, a set of all target instances and their corresponding semi-labels. Given that each semi-label associated with an instance is a set of potential labels for that instance, we define $PA$ as the union of all such label sets. The algorithm proceeds by clustering all instances based on the feature space of the target sensor. Next, it constructs a weighted labeling graph, as discussed in Definition 5, from all clusters and all labels. In the next step, *findBestAssignment* executes the *Hungarian* algorithm on the constructed graph and generates a set of assignments, one for each cluster. Finally, we propagate the cluster labels to their corresponding instances and generate the final set $(X, TL)$ of instances and their transferred labels.

## 4 Online Minimum Error Exact Labeling

To be able to run our autonomous algorithm in an online mode, we propose an online version of *instance clustering* and *optimal labeling* in this section.

### 4.1 Online Instance Clustering

Thus far, we have used clustering as a tool for batch clustering of all accumulated instances in the multi-view space. We introduce the online version of instance clustering algorithm which incrementally updates the model upon availability of new instances.

An *Online Clustering* algorithm could be considered as a selection of one clustering $\mathcal{C}^t$ at time $t$ from $\mathcal{U}$ which is a collection of possible clusterings of $\mathcal{X}$. In this paper, we use $\mathcal{U} \in \mathcal{X}^k$ to express that each model is a collection of $k$ data points $(u_1, \ldots, u_k)$ as cluster centroids where $\forall x \in \mathcal{X}$ and $\forall i, j \in \{1, \ldots, k\}$

$$x \rightarrow C_i \Leftrightarrow d(x, u_i) \leq d(x, u_j). \tag{12}$$

Here $x \rightarrow C_i$ means that $x$ is assigned to cluster $C_i$.

There are different methods of performing online clustering. One of the most well-known algorithms is Lloyd [27], [28]. This algorithm, which has several variants, is also known as *sequential k-means*.

**Algorithm 3.** Lloyd Online Clustering

```
Initialize U = (u_1, ..., u_k) to the first k data points
Initialize n_1, n_2, ..., n_k to 1
for every next element x do
  if i ∈ {1, ..., k} and d(x, u_i) is minimum then
    Increment n_i
    u_i = u_i + (1/n_i)(x − u_i)
```

This algorithm assumes that $k$ is known in advance. However, the study in [29] shows when the number of clusters is not pre-defined, using extra clusters helps to detect more accurate clusters.

Clustering algorithms aim to group similar instances without access to class labels. However, in this paper, we are using the clustering method as the first step toward final classification. Therefore, knowing the instance labels, we can define another cost function based on the accuracy of the contribution of clustering to the final classification task. Subsequently, for each cluster $C_i$ at time $t$, we define a representative label $R^t_{C_i}$ using majority vote over the instances of each cluster. The representative label is given by

$$R^t_{C_i} = \underset{a_j}{\operatorname{argmax}} \mathcal{N}^t_i(a_j), \tag{13}$$

where $\mathcal{N}^t_i(a_j)$ represents the number of instances in cluster $C_i$ with label $a_j$. We represent the new cost function for clustering $\mathcal{C}^t$ as

$$cost(\mathcal{C}^t) = \sum_{C_i \in \mathcal{C}^t} \frac{\sum_{a_j \in \mathcal{A}} \mathcal{N}^t_i(a_j)(a_j \neq R^t_{C_i})}{|C_i|}. \tag{14}$$

In other words, for each cluster, the cost is defined as the fraction of instances with a label different than that of the cluster representative, $R^t_{C_i}$.

Algorithm 3, aims to approximate K-Means using the online clustering paradigm. In the context of online learning, a *regret loss* intends to compare the performance of an online algorithm with a competing hypothesis that is chosen in retrospect, after observing all examples [30]. In this paper, the competing algorithm is the original K-Means algorithm which works on the entire example set. To show how Algorithm 3 converges to its competing algorithm (e.g, K-Means), we define a relative loss measure. Assuming $CK$ and $CO$ are clustering results of K-means and Algorithm 3 on instance sequence $\mathcal{X}$ respectively, we define relative loss $CO$ compared to $CK$ as shown

$$RL(CO, CK) = cost(CO) - cost(CK). \tag{15}$$

### 4.2 Online Optimal Labeling

The algorithm proposed in Section 3.5 works on the final clustering result, builds bipartite graph and executes the *Hungarian* algorithm to find the optimal assignment. However, the clustering changes in the online mode. Subsequently, the estimators change and the cost of the assignments are subject to change. Furthermore, in online the mode, we aim to have the prediction model based on the already observed instances. Therefore, in each step in Algorithm 4, we aim to assign the best labels to instances based on the current estimator values. Obviously, upon any changes in the clustering after an assignment is made, the graph weights change and the new problem, similar to the original problem, could be solved from scratch using the *Hungarian* algorithm. However, this solution is inefficient. We introduce an online labeling algorithm in this section.

**Algorithm 4.** Greedy Algorithm for O-MEEL

```
INPUT: Stream of (X, SL)
OUTPUT: finalLabeling
C^0 = ∅
while there is more pair (X^t, SL^t) do
  C^t ← onlineCluster(C^{t−1}, (X^t, SL^t))
  W ← reEstimateError(C^t)
  G ← changeBipartiteGraph(C^t, W, G)
  (C^t, A) ← updateAssignment(G)
  finalSet ← ∅
  for each (C_i, a_j) ∈ (C^t, A) do
    for each X_k ∈ C_i do
      finalSet ← finalSet ∪ (X_k, a_j)
```

Using an online clustering algorithm, any new observation makes a change in clusters and consequently in the value of the estimator. However, compensating for this change and the adjusting estimator's value could be done in $O(|\mathcal{A}|)$. Particularly, using LIoyd clustering algorithm, a new observation is merged with the closest cluster. Therefore, other clusters remains unchanged and consequently their mis-labeling estimator remains unchanged. For the only cluster that included the new observation, modifying the previous estimator, without re-computing from scratch, could provide us with the updated estimator. Let, the new instance $X^{t+1}$ with corresponding semi-label $(a, P)$ is merged to cluster $C^t_i$. New estimator could be obtained with a simple weighted average using following equation:

TABLE 1
Sensor Locations

| No. | On Body Location | Abbr. |
|---|---|---|
| 1 | Torso | TO |
| 2 | Right Arm | RA |
| 3 | Left Arm | LA |
| 4 | Right Leg | RL |
| 5 | Left Leg | LL |

$$W_{ij}^{(t+1)} = \frac{|C_i^t| \times W_{ij}^t + P_j}{|C_i^{(t+1)}|}. \quad (16)$$

In order to address changes in the assignment cost, we use a dynamic version of *Hungarian* algorithm [31]. This approach effectively solves the new problem by just repairing the previous assignment. This algorithm, performs a single stage of the original *Hungarian* algorithm in $O(|\mathcal{A}|^2)$ and returns the new optimal assignment. Since each new observation contributes a weight change in the bi-partite graph, the overall complexity of the assignment is $O(N \times |\mathcal{A}|^2)$ where $N$ denotes the number of new observations.

## 5 Validation

In this section, we demonstrate the effectiveness of our multi-view learning algorithms using real data collected from eight subjects (4 female, 4 male, between the ages 20 and 30) performing 19 different physical activities for 5 minutes [32], [33], [34]. The dataset used for our validation is publicly available through the UCI Repository.

### 5.1 Data Collection

Subjects performed the activities at Bilkent University Sports Hall while wearing 5 motion sensor nodes on different body locations. Table 1 shows the details of sensor placements. Each sensor node is a Xsens MTx [35] inertial sensor unit with a 3-axis accelerometer, a 3-axis gyroscope and a 3-axis magnetometer. The sampling frequency is set to 25 Hz and the 5 min signals are divided into 60 segments of 5 sec so that 480 (= 60 × 8) signal segments are obtained for each activity from 8 subjects. The collected dataset contains over 9,120,000 samples of acceleration, angular velocity and magnetic field. From the 19 activities, some activities are very similar such as lying on back or on right side which we picked one of them for our experiments.

The detail of the final 15 physical activities has been shown in Table 2. In this dataset, there are obvious inter-subject variations in the speeds and amplitudes of the same activity because the subjects were asked to perform the activities in their own style and were not restricted on how the activities should be performed.

### 5.2 Data Analysis

From each 5-second segment of the individual sensor streams, we extracted 9 statistical features. Potentially, there are many different features that can be extracted from human activity signals. However, as shows in Table 3, these features aimed to capture both shape and amplitude of the signals. For example, features such as AMP and MNVALUE are useful to capture intensity of the signal while STD or S2E intend to capture morphology of the signal.

TABLE 2
Experimental Activities

| No. | Description | Label |
|---|---|---|
| 1 | sitting | SIT |
| 2 | standing | STN |
| 3 | lying | LYN |
| 4 | ascending stairs | AST |
| 5 | descending stairs | DST |
| 6 | moving around in an elevator | MAE |
| 7 | walking in a parking lot | WPL |
| 8 | running on a treadmill with a speed of 8 km/h | RUN |
| 9 | exercising on a stepper | EST |
| 10 | exercising on a cross trainer | ECT |
| 11 | cycling on an exercise bike in horizontal position | CYH |
| 12 | cycling on an exercise bike in vertical position | CYV |
| 13 | rowing | ROW |
| 14 | jumping | JMP |
| 15 | playing basketball | BSK |

TABLE 3
Activity List

| Label | Description |
|---|---|
| AMP | Amplitude of the signal |
| MED | Median of the signal |
| MNVALUE | Mean of the signal |
| MAX | Maximum of the signal |
| MIN | Minimum of the signal |
| P2P | Peak to Peak Amplitude |
| STD | Standard Deviation of the signal |
| RMS | Root Mean Square Power |
| S2E | Stand to End Value |

We chose two sensor nodes as source and target respectively. To assure that the results are not biased by the order of observations, we randomly shuffled the observations associated with both sensors. To maintain the sensors synchronized, we used the same random sequence for both source and target. We divided the obtained sequence of the observations into three categories including train source, train target, and test. The first category of the observations was used for training the source sensor. After constructing a trained model on the source sensor, we are ready to start our SML approach on the second category of the data. At this point, we assume that the target sensor was added to the system. We followed the *plug-n-learn* framework to train the target sensor using semi-labels provided by the source sensor. After training target, we tested the trained classifier of the target model on the third category of the sequence. We repeated this procedure 1,000 times and averaged the results to ensure that the online algorithm works reliably regardless of the observations' sequence. All the analyses were performed in Matlab.

### 5.3 Single-Node Performance

Our initial analysis focused on assessing performance of the system in a single-node (i.e., torso (T), right arm (RA), left arm (LA), right leg (RL), left leg (LL)) scenario. We used each one of these sensors to detect all activities using four machine learning classifiers, namely Decision Tree (DT), Discriminant Analysis (DA), $k$-Neareset-Neighbor ($k$NN)

TABLE 4
Accuracy (%) of Single-Node Activity Recognition

| Node/ Classifier | kNN | DT | DA | NB |
|---|---|---|---|---|
| T | 70.13 | 69.06 | 69.48 | 69.40 |
| RA | 72.97 | 63.72 | 68.85 | 58.18 |
| LA | 80.81 | 73.41 | 74.35 | 65.55 |
| RL | 81.72 | 78.78 | 87.03 | 76.93 |
| LL | 80.76 | 75.81 | 87.60 | 79.58 |
| Average | 77.28 | 72.16 | 77.46 | 69.93 |

and Naive Bayes (NB). Table 4 shows the activity recognition accuracy for this analysis. In each experiment, we used the data of one subject for testing and sensor readings of other subjects were used as training. We performed the entire experiment for each subject separately. The reported results of each sensor node are averaged over experiments of all subjects.

The average accuracy of all classifiers over all the five sensor nodes ranged from 69.93 to 77.46 percent. Since the subjects were not restricted on how the activities should be performed, we observed much variation among subjects. Even when we trained on seven subjects, the final trained classifier was not enough powerful in detecting personalized activities. Therefore, it motivates our attempt to develop a personalized model. In addition, these results suggest that we cannot achieve a high recognition accuracy using only one sensor. This finding confirms our hypothesis that, in a single sensor view, some activities are not reliably distinguishable. Furthermore, the ability of different sensors in detecting different activities varies. For example, as shown in Table 4, 'Torso' is weaker than both legs in detecting all 15 activities regardless of the selected classification algorithm. For the rest of experiments, we use $k$-NN classifier because it is a lazy algorithm and doesn't require extra training phase after obtaining the labels.

### 5.4 Comparative Evaluation Method

Research in the area of transfer learning for wearables is new. To the best of our knowledge, there exist only two of such algorithms, namely Naive and System-Supervised, suggested in [15], which are applicable to the synchronous teacher/ learner approach studied in this paper in both online and batch modes. We compare our online and batch version of SML algorithm with these two algorithms as well as the experimental upper bound obtained using ground truth labels gathered during our data collection. Calatroni et al. [15] proposed the Naive approach as reusing the 'source' classifier in 'target'. In this case, the parameters of the classifier (e.g., support vectors for SVM) are transferred from source to the target. Then the target sensor uses the same model to classify its own observations. They emphasized that this method only works when 'source' and 'target' are highly similar in terms of modality and context (e.g., sensors are homogeneous and co-located on the body). The *System-Supervised* method refers to the case where 'target' labels its observations based on labels predicted by 'source'. In other words, both source and target sensor are synchronized and the target sensor collects its own training data using its own sensor observation and the predicted labels provided by 'source' without performing any further refinement. Our analysis compares the accuracy of provided labels in 'target' using B-MEEL and O-MEEL with that of Naive, System-Supervised, and Upper-Bound. We also report the accuracy of the target classifier built on the computed labels using all the algorithms.

### 5.5 Comparative Analysis

We studied 20 different scenarios where 'source' and 'target' were a selection of two different sensors from the set of torso, right arm, left arm, right leg, left leg sensors. By studying the confusion matrix of different classifiers, we noticed that failing to detect an activity is more associated with the position of the sensor rather than the type of algorithm or classifier used for activity recognition. Once semi-labels are extracted, we followed the rest of our SML procedure described in Algorithm 1 to transfer semi-labels and to compute exact labels in the target view. The algorithm clusters target instances based on the statistical features extracted from signals in the target view and continues with applying B-MEEL in the case of batch mode and O-MEEL for the online version.

As shown in Fig. 5, $x$-axis represents different scenarios and $y$-axis the accuracy of each method. The graph shows the accuracy of labeling using both B-MEEL and O-MEEL in all twenty scenarios is higher than 'system-supervised'. The overall accuracy of labeling using B-MEEL ranged from 78.7 to 88.5 percent. The accuracy for O-MEEL ranged from 78 to 87.8 percent with the average accuracy of $84.4\%$ and $82.8\%$ percent respectively. On average, this accuracy is 10.4 and 8.8 percent higher than the accuracy of 'system-supervised'. Whereas 'system-supervised' only relies on the source provided labels, for every scenario with the same source node and regardless of the target sensor, the accuracy of the labels remains the same. We note that the accuracy of the ground truth labeling is naturally 100 percent. In this experiment, we did not include the Naive method because it only uses already trained model of the source node and provides no training data.

In the next step, the obtained labels in the target view were used to construct a kNN classifier for activity recognition. As illustrated in Fig. 6, the classification accuracy of B-MEEL ranged from 78.4 to 88.6 percent and the O-MEEL accuracy varied from 78.4 to 87.7 percent. Compared to the 'system-supervised', B-MEEL improves the recognition accuracy by 9.3 percent, on average, while O-MEEL results in 7.9 percent enhancement in recognition accuracy. Furthermore, B-MEEL performs 61.8 percent better than 'Naive' method in terms of activity recognition accuracy whereas online version of the algorithm achieves an improvement of 60.4 percent. The interesting observation about the performance of the 'Naive' method is that its accuracy is much higher when the source and target are both from upper-body or from lower-body. It confirm the Calatroni's claim that when source and target are similar, the 'Naive' method is effective [15]. In addition to 'Naive' and 'system-supervised', we computed the experimental upper bound of the two-sensor configuration where we used the ground truth labels to train a classifier using both source and target sensors. The overall experimental upper bound was 96.6 percent averaged over all configurations. Therefore, the recognition accuracy of B-MEEL approach is 12.9 percent lower than the experimental upper bound.

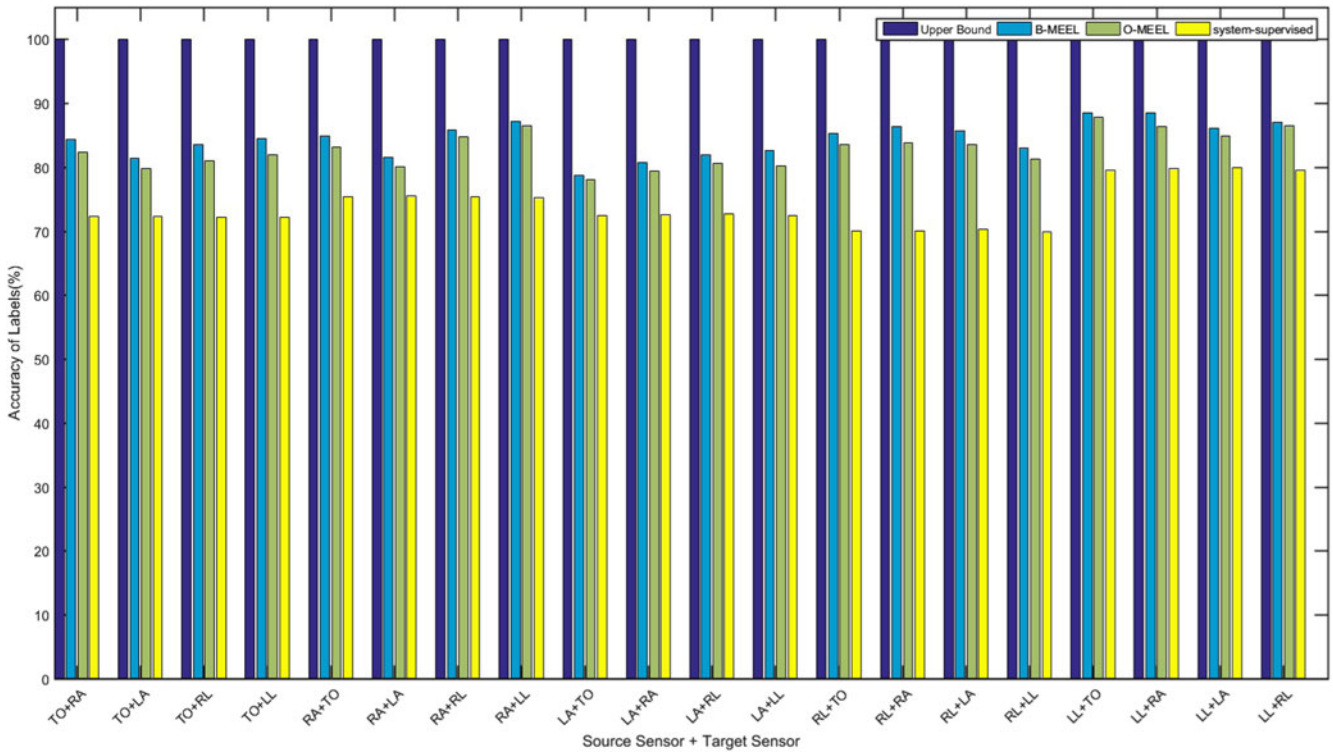



Fig. 5. Accuracy of provided label using four approaches under comparison.

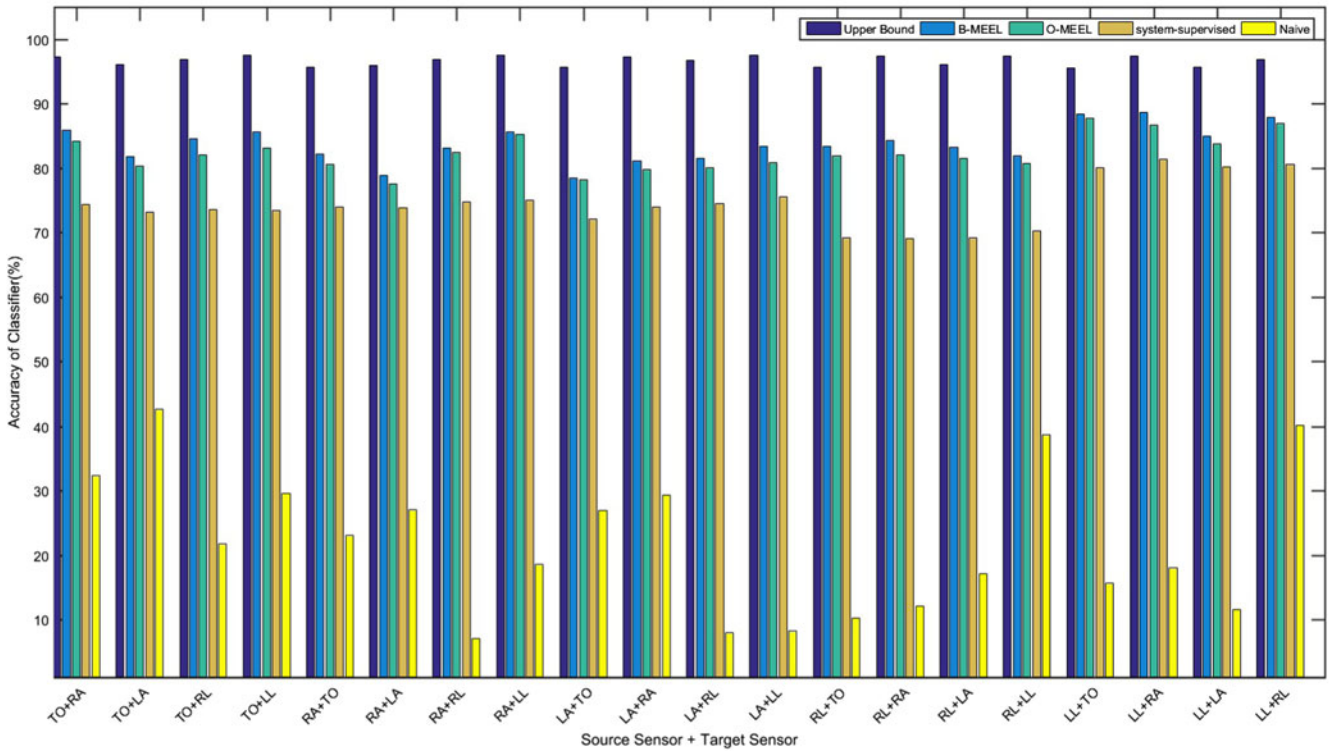
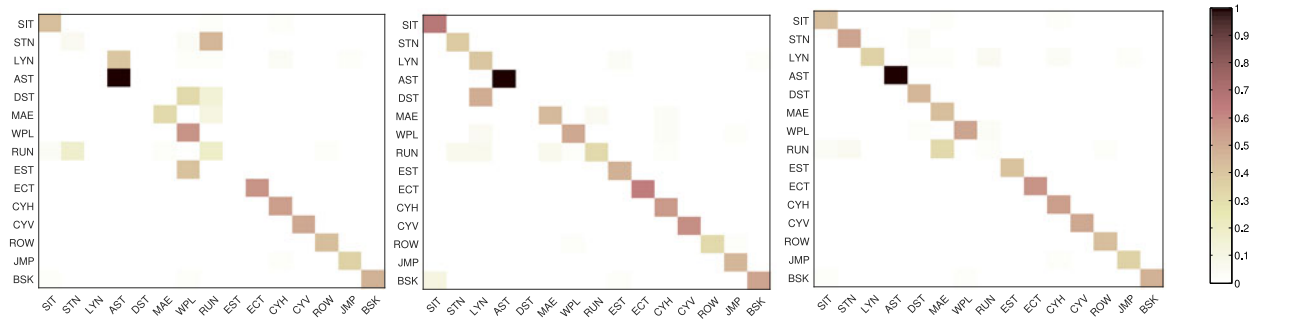



Fig. 6. Accuracy of five activity recognition classifiers under comparison.

### 5.6 Evolution of Labels

As shown in Fig. 6, the highest improvement in labeling is achieved for the scenarios where 'Right Leg' operates as 'source'.

Fig. 7 demonstrates how SML improves the provided labels in 'target' by comparing confusion matrix of the provided labels by 'source' (system-supervised) and the confusion matrix of labels after applying SML in two different scenarios. The confusion matrix is used to visualize the performance of a classification model. Each column of the matrix shows the instances in a predicted class while each row represents the instances in an actual class (or vice-versa). Here, the darker cells specify more density of instances in that cell. As shown in Fig. 7b, after applying SML, more instances are concentrated




(a) Confusion matrix of source sensor (RL) (b) After adding sensor node (TO) and applying SML (c) After adding sensor node (RA) and applying SML (d) Color scaling

Fig. 7. Comparison of confusion matrices before and after applying SML, when 'source' is (a) 'Right Leg (RL)' and 'target' is (b) 'Torso (TO)' and (c) 'Right Arm (RA)'.

on the main diagonal of the confusion matrix. In addition, by comparing Figs. 7b and 7c, it is clear that even with the same source sensor, different target sensors contribute differently to the classification accuracy.

### 5.7 Convergence of Online Algorithm

In the previous section, we showed that the final result of O-MEEL does not outperform B-MEEL. However, it improves the accuracy when compared with the 'system-supervised'. In both Figs. 5 and 6, we reported the final result of O-MEEL. The online clustering algorithm introduced in Section 4 attempts to refine the cluster centroids based on the current observation. It would be interesting to study how the online method improves by observing more instances. Fig. 8 illustrates the effect of number of observations on the relative loss defined in (15). It shows that by observing more instances the number of mistakes made by Lloyd clustering becomes closer to the number of mistakes made by original K-Mean. Since the Algorithm 3, analogous to the other online algorithms, is sensitive to the sequence of data, we repeated the experiment 100 times and averaged the relative loss. Because Algorithm 3 changes its model based on the current observation, there are sharp fluctuations between consecutive instances. However, the overall trend of relative loss shows that the online algorithm converges to a point close to K-Means.

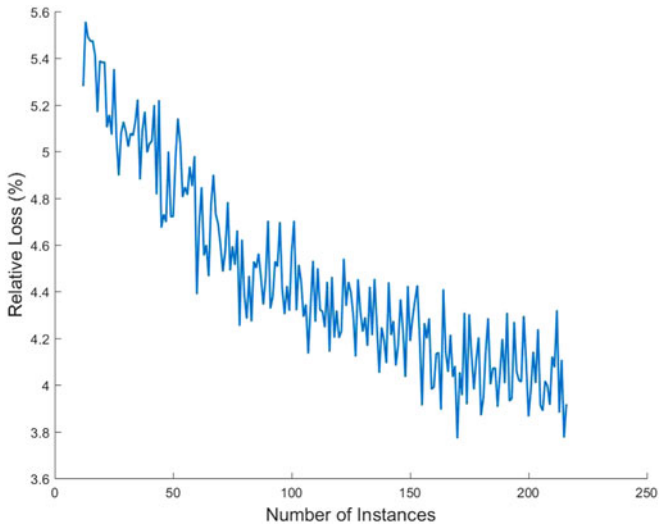



Fig. 8. Relative loss of online clustering algorithm comparing to K-means. It shows that with more observations the result of online clustering algorithm is converging to off-line K-means.

## 6 Related Work

To enable a newly added sensor for activity recognition, one approach is to first detect the location of the sensor and then use appropriate classifier from a repository of pre-trained models. For example, authors in [9], [10] proposed a sensor localization based on computationally simple features. A limitation of these approaches is that they require a set of labeled training data to train the localization model. Furthermore, the set of possible sensor locations (on-body wearing sites) is limited by the training data. [36] proposed an autonomous sensor context detection where the sensor can automatically choose the best model among a set of classifiers each of which are trained using sensor data from one body location. Although this approach does not require collecting labeled training data, its capability is limited by the number of models available in the repository. Our approach to realizing self-configurable system is based on the concept of transfer learning [37]. Transfer learning applies the learned knowledge in one problem domain, the source (e.g., existing sensors in this case), to a new but related problem, the target (e.g., a new sensor). However, in machine learning literature, it has been studied under various names such as learning to learn, knowledge transfer, inductive transfer, and meta-learning [38], [39], [40], [41]. According to [37], transfer learning approaches could be classified into *instance-transfer*, *feature-representation transfer*, *parameter-transfer* and *Relational knowledge transfer* which our approach is in the category of instance transfer.

In order to transfer instances, TrAdaBoost [42], an extension of AdaBoost [43], proposed a boosting algorithm to enable transferring knowledge from one domain to another using training data from the source domain to incrementally build up the training data for the target domain. However, in the dynamic environment of human-centered IoT, the assumption that both source and target systems have the same feature space and operate in the very similar domains is unrealistic.

In domain of transfer learning, when source and target domains work on the same data points (instances) but have different feature spaces, Multi-view learning approaches [44] could be used to address transfer learning problems.

Several algorithms such as Manifold Alignment [45] try to align feature spaces using un-informed methods. Uninformed learning assumes labeled data is available in the source view but not the target view. In contrast, informed learning approaches such as Co-Training [46] and Co-EM [47] assumes that labeled training data is available in the target domain and try to use multi-view learning techniques by using a small amount of labeled data in each view to train two classifiers [48]. The important underlying assumption in methods like Co-Training is that data points of different labels are distinguishable in both views, however, there are not enough labeled data available in each view. Contrary to our problem which each sensor based on its position has limited ability to separate data points.

In smart home application, [49], [50] showed that it is possible to avoid data collection phase by transferring classifier models of activity recognition from one home to another with similar activity recognition system. To resolve the need for a common feature space, they proposed "meta-features" which are the common ground on which the transfer of knowledge can happen. However, defining meta-features needs a good understanding of both source and target spaces which might change by changing the target space and therefore, cannot happen in real time. In addition, their proposed method is not generalized and was only applied to binary sensors.

Besides informed and uninformed methods, Teacher/Learner (TL) transfer learning has being used when there is no direct access to the training data. Several studies [13], [14], [15] applied the teacher/ learner model to develop an opportunistic system capable of performing reliable activity recognition in a dynamic sensor environment. [15] showed that by synchronizing current sensor and new sensor, the existing sensor can provide labels of future activities. Although noticeably less studies have been done using teacher/learner approaches, studies of using these approaches could increase the performance of transfer learning. One limitation of TL approaches is that the accuracy of learner's training is bounded by the accuracy of teacher. However, in this paper, we introduce a method for transferring and refining labels in order to outperform the accuracy of the initial system.

Authors in [51] proposed a synchronous dynamic view learning method to boost the accuracy of transferred labels when the on-body location of the target sensor changes. They proposed an approach to construct a similarity graph followed by an iterative label propagation technique to refine the transferred labels. The algorithm, however, does not operate in an online mode and the computing complexity of the approach is quadratic in the number of sensor observations.

In addition to instance-transfer methods, other studies which try to learn mapping between teacher and learner in the signal level. Authors of [16] represent the mapping between teacher and learner as a linear multiple-input-multiple-output (MIMO) system with different channels of the teacher and learner signals. Although, their experiments on five different gestures show a successful mapping of signals between two similar body parts such as left and right hand, their approach does not provide acceptable results when transfer is between two different body segments.

In order to transfer complex activities, [52] suggested a partonomy-based approach to transfer knowledge which requires less amounts of training data for new complex activities. However, defining the required hierarchical model for new problems is not always applicable in real-time.

In contrast to traditional supervised learning methods which assume the training data is available in advance, there are few approaches such as [53] which use online learning [30] paradigm. Authors of [53], designed a novel online learning algorithm which just uses a binary response (correct/wrong) on its current prediction and tries to adapt its classifier. However, they showed this method has a faster convergence comparing to traditional methods, their accuracy is still far from the teacher. However, in our online approach, in addition to incrementally learning from the instances as soon as they become available, we gradually refine provided labels of teacher using combined observation of teacher and learner which eventually leads to more accurate classifier than teacher.

## 7 Discussion and Future Work

Our study in this paper is a first step toward designing next generation wearables that are computationally autonomous and can automatically learn machine learning algorithms. Dynamic attributes of wearables are not limited to real-time sensor addition. A sensor can be be removed, misplaced, displaced, upgraded, or replaced, sensors can produce noisy signals, and the system can be adopted by new users. Our ongoing research involves development of transfer learning algorithms that address dynamically evolving behavior of human-centered monitoring applications.

Plug-n-learn enables us to improve the activity recognition capability by adding a new un-trained sensor. However, when the prediction of the source sensor, due to its placement site or inaccuracy of its model, on a new observation is not better than random guess, the MEEL process cannot refine the provided label. Particularly, in these cases where the learnability of our approach is limited, we may consider other sources of knowledge such as other wearable sensors, ambient sensors or even input from the end-user to label sensor observations in the target view. We also note that our work in this paper assumes that activities that occur in the target view are a subset of those that are previously observed in the source view. Expansion of activity recognition capability to new activities unseen in 'source' is an interesting future work where approaches such as active learning could minimize the number queries to other knowledge sources.

In this study, we considered simple activities which can fit into a fixed-time signal segment. However, the segment length can be arbitrary long in complex, co-occurred, and interwoven activities. In general, structured prediction can help in detection of such a long and complex activities. This is an area of research that we plan to purse in the future.

There is a trade-off between the final performance of the classifier, its memory usage and computation power required for the training process. Using online learning method, our technique consumes low memory and incrementally updates the trained model during the training phase while using batch version, we gain higher performance model at the cost of higher memory and computation. However, we showed that the online model improves due to more collaboration with the source sensor while observing new instances. Generally, our algorithms are

computationally simple that can run on wearable devices with stringent constrained resources. Furthermore, we need to store only cluster centers and their corresponding estimators in addition to the model parameters.

## 8 Conclusion

As wearable sensors are becoming more prevalent, their function becomes more complex and they operate in highly dynamic environments. Machine learning algorithms for these sensors cannot be designed only for one specific setting. To address the dynamic nature of wearable sensors, we proposed a multi-view learning approach that uses the knowledge of existing sensors to adapt with sensor addition. We used activity recognition task as our pilot application. We introduced a multi-view learning approach to learn computational algorithms in new settings without any need for labeled training data in the target view. Our experiments showed that we can combine knowledge of existing system with the observation of a new sensor to boost the accuracy of the system without labeled training data. The results of batch and online variants of our multi-view approach demonstrate promising performance with an overall accuracy of 83.7 and 82.2 percent for both batch and online mode, respectively.

## Acknowledgments

This work was supported in part by the US National Science Foundation, under grants CNS-1566359. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding organizations.

## References

[1] R. Fallahzadeh, S. Aminikhanghahi, A. N. Gibson, and D. J. Cook, "Toward personalized and context-aware prompting for smartphone-based intervention," in *Proc. IEEE 38th Annu. Int. Conf. Eng. Med. Biol. Soc.*, 2016, pp. 6010–6013.

[2] S. A. Rokni, H. Ghasemzadeh, and N. Hezarjaribi, "Smart medication management, current technologies, and future directions," *Handbook Res. Healthcare Admin. Manage.*, IGI Global, pp.188–204, 2017, Art. no. 188.

[3] D. J. Cook and S. K. Das, "How smart are our environments? An updated look at the state of the art," *Pervasive Mobile Comput.*, vol. 3, no. 2, pp. 53–73, 2007.

[4] S. D. Manshadi and M. E. Khodayar, "Expansion of autonomous microgrids in active distribution networks," *IEEE Trans. Smart Grid*, to be published, doi:10.1109/TSG.2016.2601448.

[5] J. Stankovic, "Research directions for the internet of things," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 3–9, Feb. 2014.

[6] H. Ghasemzadeh, N. Amini, R. Saeedi, and M. Sarrafzadeh, "Power-aware computing in wearable sensor networks: An optimal feature selection," *IEEE Trans. Mobile Comput.*, vol. 14, no. 4, pp. 800–812, Apr. 2015.

[7] C. M. Bishop, *Pattern Recognition and Machine Learning*, vol. 4. New York, NY, USA: Springer, 2006.

[8] D. Zakim and M. Schwab, "Data collection as a barrier to personalized medicine," *Trends Pharmacological Sci.*, vol. 36, no. 2, pp. 68–71, 2015.

[9] R. Saeedi, J. Purath, K. Venkatasubramanian, and H. Ghasemzadeh, "Toward seamless wearable sensing: Automatic on-body sensor localization for physical activity monitoring," in *Proc. 36th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, 2014, pp. 5385–5388.

[10] R. Saeedi, B. Schimert, and H. Ghasemzadeh, "Cost-sensitive feature selection for on-body sensor localization," in *Proc. ACM Conf. Ubiquitous Comput.*, 2014, pp. 833–842.

[11] P. Alinia, R. Saeedi, A. Rokni, B. Mortazavi, and H. Ghasemzadeh, "Impact of on-body sensor localization on met calculation using wearables," in *Proc. IEEE Int. Conf. Body Sensor Netw.*, Jun. 2015, pp.1–6.

[12] P. Alinia, R. Saeedi, R. Fallahzadeh, A. Rokni, and H. Ghasemzadeh, "A reliable and reconfigurable signal processing framework for estimation of metabolic equivalent of task in wearable sensors," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 5, pp. 842–853, Aug. 2016.

[13] D. Roggen, K. Frster, A. Calatroni, and G. Trster, "The adARC pattern analysis architecture for adaptive human activity recognition systems," *J. Ambient Intell. Humanized Comput.*, vol. 4, no. 2, pp. 169–186, 2013.

[14] M. Kurz, et al., "The opportunity framework and data processing ecosystem for opportunistic activity and context recognition," *Int. J. Sensors Wireless Commun. Control*, vol. 1, no. 2, pp. 102–125, 2011.

[15] A. Calatroni, D. Roggen, and G. Tröster, "Automatic transfer of activity recognition capabilities between body-worn motion sensors: Training newcomers to recognize locomotion," in *Proc. 8th Int. Conf. Netw. Sens. Syst.*, 2011.

[16] O. Banos, et al., "Kinect= IMU? Learning MIMO signal mappings to automatically translate activity recognition systems across sensor modalities," in *Proc. 16th Int. Symp. Wearable Comput.*, 2012, pp. 92–99.

[17] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, "Activity recognition from accelerometer data," in *Proc. 17th Conf. Innovative Appl. Artif. Intell.*, 2005, pp. 1541–1546.

[18] O. D. Lara and M. A. Labrador, "A survey on human activity recognition using wearable sensors," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1192–1209, Jul.–Sep. 2013.

[19] H. Ghasemzadeh, N. Amini, and M. Sarrafzadeh, "Energy-efficient signal processing in wearable embedded systems: An optimal feature selection approach," in *Proc. ACM/IEEE Int. Symp. Low Power Electron. Des.*, 2012, pp. 357–362.

[20] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *ACM SIGKDD Explorations Newslett.*, vol. 12, no. 2, pp. 74–82, 2011.

[21] S. A. Rokni and H. Ghasemzadeh, "Plug-n-learn: Automatic learning of computational algorithms in human-centered internet-of-things applications," in *Proc. 53rd Annu. Des. Autom. Conf.*, 2016, Art. no. 139.

[22] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 2001.

[23] R. Jonker and T. Volgenant, "Improving the hungarian assignment algorithm," *Operations Res. Lett.*, vol. 5, no. 4, pp. 171–175, 1986.

[24] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. North Chelmsford, MA, USA: Courier Corporation, 1982.

[25] E. L. Lawler, *Combinatorial Optimization: Networks and Matroids*. North Chelmsford, MA, USA: Courier Corporation, 2001.

[26] J. A. Bondy and U. S. R. Murty, *Graph Theory with Applications*. London: Macmillan, vol. 290, 1976.

[27] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. 28, no. 2, pp. 129–137, Mar. 1982.

[28] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probability*, 1967, pp. 281–297.

[29] M. Ackerman and S. Dasgupta, "Incremental clustering: The case for extra clusters," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 307–315.

[30] S. Shalev-Shwartz and Y. Singer, "Online learning: Theory, algorithms, and applications," PhD Thesis, the Hebrew university, 2007.

[31] G. A. Mills-Tettey, A. Stentz, and M. B. Dias, "The dynamic hungarian algorithm for the assignment problem with changing costs," Robotics Institute, Tech. Rep. CMU-RI-TR-07-23, 2007.

[32] K. Altun, B. Barshan, and O. Tunçel, "Comparative study on classifying human activities with miniature inertial and magnetic sensors," *Pattern Recognit.*, vol. 43, no. 10, pp. 3605–3620, 2010.

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

[34] K. Altun and B. Barshan, "Human activity recognition using inertial/magnetic sensor units," in *Proc. Int. Workshop Human Behavior Understanding*, 2010, pp. 38–51.

[35] Manual, Xsens Moven Studio User. "Xsens technologies BV," 2009.

[36] S. A. Rokni and H. Ghasemzadeh, "Autonomous sensor-context learning in dynamic human-centered internet-of-things environments," in *Proc. 35th Int. Conf. Comput.-Aided Des.*, 2016, Art. no. 75.

[37] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[38] A. Arnold, R. Nallapati, and W. W. Cohen, "A comparative study of methods for transductive transfer learning," in *Proc. 7th IEEE Int. Conf. Data Mining Workshops*, 2007, pp. 77–82.
[39] C. Elkan, "The foundations of cost-sensitive learning," in *Proc. Int. Joint Conf. Artif. Intell.*, 2001, pp. 973–978.
[40] S. Thrun and L. Pratt, *Learning to Learn*. Berlin, Germany: Springer, 2012.
[41] R. Vilalta and Y. Drissi, "A perspective view and survey of meta-learning," *Artif. Intell. Rev.*, vol. 18, no. 2, pp. 77–95, 2002.
[42] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Boosting for transfer learning," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 193–200.
[43] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.
[44] S. Sun, "A survey of multi-view machine learning," *Neural Comput. Appl.*, vol. 23, no. 7/8, pp. 2031–2038, 2013.
[45] C. Wang and S. Mahadevan, "Manifold alignment using procrustes analysis," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 1120–1127.
[46] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proc. 11th Annu. Conf. Comput. Learn. Theory*, 1998, pp. 92–100.
[47] K. Nigam and R. Ghani, "Analyzing the effectiveness and applicability of co-training," in *Proc. 9th Int. Conf. Inf. Knowl. Manage.*, 2000, pp. 86–93.
[48] D. J. Cook, K. D. Feuz, and N. C. Krishnan, "Transfer learning for activity recognition: A survey," *Knowl. Inf. Syst.*, vol. 36, no. 3, pp. 537–556, 2013.
[49] T. van Kasteren, G. Englebienne, and B. J. Kröse, "Transferring knowledge of activity recognition across sensor networks," in *Pervasive Computing*. Berlin, Germany: Springer, 2010, pp. 283–300.
[50] K. D. Feuz and D. J. Cook, "Transfer learning across feature-rich heterogeneous feature spaces via feature-space remapping (FSR)," *ACM Trans. Intell. Syst. Technol.*, vol. 6, no. 1, 2015, Art. no. 3.
[51] S. A. Rokni and H. Ghasemzadeh, "Synchronous dynamic view learning: A framework for autonomous training of activity recognition models using wearable sensors," in *Proc. 16th ACM/IEEE Int. Conf. Inf. Process. Sensor Netw.*, 2017, pp. 79–90.
[52] U. Blanke and B. Schiele, "Remember and transfer what you have learned-recognizing composite activities based on activity spotting," in *Proc. Int. Symp. Wearable Comput.*, 2010, pp. 1–8.
[53] K. Forster, S. Monteleone, A. Calatroni, D. Roggen, and G. Troster, "Incremental kNN classifier exploiting correct-error teacher for activity recognition," in *Proc. 9th Int. Conf. Mach. Learn. Appl.*, 2010, pp. 445–450.

**Seyed Ali Rokni** received the BS degree in computer engineering from the Amirkabir University of Technology, Tehran, Iran, in 2005, and the MSc degree in computer science from the Sharif University of Technology, Tehran, Iran, in 2008. From 2008 to 2014, he was involved in research and development in software engineering and machine learning in industry. Currently, he is working toward the PhD degree in computer science with Washington State University. His research focus is on multi-view learning algorithm development for embedded and pervasive systems. He spent summer 2016 as a research intern at Samsung Research America, Richardson, Texas. He is a student member of the IEEE.

**Hassan Ghasemzadeh** received the BSc degree from the Sharif University of Technology, Tehran, Iran, the MSc degree from the University of Tehran, Tehran, Iran, and the PhD degree from the University of Texas at Dallas, Richardson, Texas, in 1998, 2001, and 2010, respectively, all in computer engineering. He was on the faculty of Azad University from 2003 to 2006 where he served as founding chair of the Computer Science and Engineering Department at Damavand branch, Tehran, Iran. He spent the academic year 2010-2011 as a postdoctoral fellow in the West Wireless Health Institute, La Jolla, California. He was a research manager in the UCLA Wireless Health Institute 2011-2013. Currently, he is assistant professor of computer science in the School of Electrical Engineering and Computer Science, Washington State University, Pullman, Washington. The focus of his research is on algorithm design and system level optimization of embedded and pervasive systems with applications in healthcare and wellness. He is a senior member of the IEEE.