Improving Self-Adaptation For Multi-Sensor Activity Recognition with Active Learning

Tuan Pham Minh, Daniel Kottke, Anna Tsarenko, Christian Gruhl, Bernhard Sick
Intelligent Embedded Systems, University of Kassel, Germany
{tuan.pham | daniel.kottke | atsarenko | cgruhl | bsick}@uni-kassel.de

***Abstract*—Heterogeneous domain adaptation adapts a machine learning model, here classification model, from a source domain to a target domain to leverage data from both domains. Thereby, supervised heterogeneous domain adaptation expects labeled data from the target domain, while unsupervised heterogeneous domain adaptation does not. In this article, we study the inclusion of active learning to bridge unsupervised and supervised domain adaptation. The active learning approach iteratively queries the most useful instances from the target domain, which are then labeled and used to improve the classification model. Using active learning, the selection of training instances can focus on areas where ambiguity in the source domain resolves in the target domain. Hence, we achieve the same performance with fewer labels. Experiments on real activity recognition data confirm our claims.**

***Index Terms*—Active Learning, Activity Recognition, Heterogeneous Domain Adaptation, Sensor Adaptation**

I. Introduction

Assume the following example for illustration purposes: Your smartphone includes an activity recognition system to switch between different profiles depending on the current situation, e.g., muting the phone during meetings and hiding work-related notifications when relaxing after work. The activity recognition system of your smartphone is not able to acquire data about your physiological state, i.e., it cannot detect if your body is relaxing. Buying a new smartwatch to use it with your phone can provide such data. This scenario is sketched in Fig. 1. The user buys a new smartwatch with the expectation of a better activity prediction. Hence, case (1), ignoring the smartwatch's data and continuing to use the old prediction model, is not acceptable. Case 2 provides excellent performance. However, training a new classifier from scratch is very expensive, as a new potentially large dataset has to be labeled. Case (3) sketches the setting we study in this article. We use the unsupervised heterogeneous domain adaptation (HDA) method proposed by Jänicke et al. [1] to learn from labeled data in the source domain, as well as unlabeled data from the target domain. Their idea is to train an unsupervised model using the freely available unlabeled data collected after installing the new sensor. They use the *old* labeled data to infer a classification model using the expensively labeled data from the past. So far, this method is not able to incorporate labeled data from the new sensors. In this article, we overcome this deficit and use active learning (AL) to further increase the performance. Thus, we can maintain lower annotation costs induced by experts. It ensures a well-performing model with reasonable cost by prioritizing instances from the pool of newly gathered unlabeled data that are likely to improve the classification result.

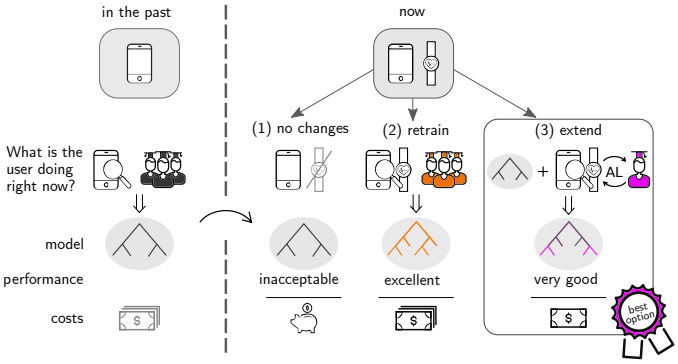

Fig. 1. An exemplary application of our approach: In the past, we expensively trained a personalized activity recognition system on our smartphone. Adding a new sensor (e.g., a smartwatch), we (1) do not want to use the existing model as our system should improve, (2) we do not want to repeat the expensive complete training procedure, instead (3) we would like to only ask for the necessary information, which is expected to improve the system which is used for adaptation.

The contribution of this article is to use AL to outperform a model adapted from a source domain to a target domain with unsupervised HDA. Using a real-world activity recognition dataset, we show that the proposed method achieves similar performances using less labeled data compared to traditional training with fully labeled data.

The following section discusses related work. Section III gives a short introduction to the fundamental methods of this article. In Section IV, we propose our method to further train a Classifier based on mixture models (CMM) and show how active learning can be applied to do so efficiently. We present our design of experiments and discusses the obtained results in Section V. The last section provides a summary and sketches directions for possible future work.

This work was funded by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) within the project "Organic Computing Techniques For Runtime Self-Adaptation Of Multi-Modal Activity Recognition System" (SI674/12-1,LU 1574/2-1)

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

II. Related Work

Activity recognition methods aim to distinguish between various activities the user may be doing currently. Chen et al. [2] give an overview over activity recognition methods. They categorize activity recognition in knowledge and data-driven approaches. Knowledge-driven approaches use prior knowledge of experts from the specific domain where the inference model will be used. This knowledge is then used to infer classification models, e.g., a rule-based system. These models have the advantage that they are often easier to be interpreted as their origin stems from human knowledge. However, a disadvantage of those methods is the need for an expert who can provide such knowledge [2]. Data-driven approaches use a high amount of data to construct an activity recognition model using the underlying knowledge within the data. However, they often rely on a large dataset and may exploit individual characteristics shown in the data, which limits the application across different persons. An advantage of these approaches is that only labels are needed, instead of the formalization of the expert's knowledge [2].

The task of domain adaptation is to learn a machine learning model using data from a source domain in some other domain, often referred to as the target domain. In this article, we focus on heterogeneous domain adaptation, which assumes that number and type of features in the source and target domain may differ [3]. This is especially the case, when new sensors are added to a classification system. Jiang [4] and Daumé III [5] distinguish between supervised and unsupervised domain adaptation. In both scenarios, we assume that labeled data from the source domain is available. The difference between supervised and unsupervised domain adaptation is that the former uses labeled data from the target domain, while the latter does not. Our proposed method extends the approach introduced by Jänicke et al. [1], an unsupervised HDA method. Additionally to the unlabeled data in the target domain, we have a budget to label this dataset. Thus, we are bridging unsupervised and supervised domain adaptation. Our method assumes that the source domain is a subspace of the target domain, as this is the assumption of the method proposed by Jänicke et al. [1]. However, other methods do not necessarily rely on that assumption and may only require a common subspace between source and target domain [3].

The goal of active learning is to reduce costs for data annotation. Instances are selected depending on their usefulness for the classifier [6]. There are two relevant active learning scenarios, namely pool-based active learning and stream-based active learning. The main difference between both scenarios is the source of unlabeled data. Pool-based active learning queries its instances from a static pool, while stream-based active learning only sees a single instance coming from the data stream. The selection strategy can then decide to query this instance or discard it completely. A thorough explanation for these scenarios can be found in [7]. In this work, we focus on pool-based active learning, as unlabeled data is abundantly available. The usefulness is often derived from the classifier's prediction. A common approach is uncertainty sampling [8]. Uncertainty sampling selects the instance which is closest to the model's decision boundary and thus has the highest uncertainty [8]. However, uncertainty sampling only handles binary classification problems. Thus, we use multi-class variants of uncertainty sampling for our experiments [7], [9]. Those selection strategies use different uncertainty measures such as confidence-based uncertainty, where the probability of the most probable class is used as a selection criterion. Margin sampling selects the classes according to the difference between the confidences of the two most likely classes, while using the entropy as an uncertainty measure takes the prediction for all classes into account.

Another approach, which is often used is expected error reduction [10]. This approach simulates label acquisitions and evaluates the performance improvement using a representative evaluation set . However, the simulation of a label acquisition requires retraining of the classifier.

Krempl et al. [6] proposed probabilistic active learning and identified shortcomings in assessing the usefulness solely based on the classifier's probabilistic prediction. This shortcoming is that uncertainty may arise due to lack of knowledge, i.e., epistemic uncertainty, or uncertainty intrinsic to the data, i.e., aleatoric uncertainty. While adding labeled data reduces the epistemic uncertainty, the aleatoric uncertainty remains unchanged. Thus, observing high uncertainty in a large amount of data within a small region may indicate high aleatoric uncertainty. Hence, Krempl et al. incorporate the number of nearby labeled instances as a proxy for the classifier's reliability [11]. Here we use the multi-class variant of probabilistic active learning, namely Multi-class probabilistic active learning (McPAL) [11].

Active learning has been used successfully in various settings to improve the detection rate of activities with fewer labels [12]–[14]. Zhao et al. [13] combine active learning with crowdsourcing for motion data as well as a synchronized video recording of a cooking scenario. The instances to be labeled (in this case, single frames of video recordings) were selected by using various active learning strategies based on the motion data. Next to the activities, various objects in the video were labeled to help improve the quality of activity labels. Liu et al. [12] and Stikic et al. [14] used active learning with multiple classifiers to improve the training of their activity recognition system with various sensors. Liu et al. used one classifier per motion sensor, while Stikic et al. used one classifier for the motion data and another classifier for infra-red sensors. All works above show that performances comparable to the passively trained classifiers are achievable with active learning using fewer labeled instances.

III. Methodological Foundation

This section provides the relevant information on Gaussian mixture models (GMM), the Classifier based on mixture models (CMM), and structure-based sensor adaptation with CMM, and Multi-class probabilistic active learning which is needed to understand our approach.

A. Gaussian Mixture Models

Gaussian mixture models [15] use a linear combination, cf. Eq. (1), of multiple (here J) normal distributions (Gaussians) called *components*. Each *component* has its own set of parameters. These are a mean vector $\boldsymbol{\mu_j}$ that describes its location and a covariance matrix $\boldsymbol{\Sigma}_j$ that describes its shape. To ensure that a GMM retains the properties of a density (i.e., $\int P(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = 1$), *mixture coefficients* π_j, with the constraints given in Eq. (2), are introduced for each *component*. GMMs are a popular choice to approximate arbitrary continuous probability distributions.

$$P(\boldsymbol{x}) = \sum_{j=1}^{J} \pi_j \cdot \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu_j}, \boldsymbol{\Sigma}_j) \tag{1}$$

$$0 \leq \pi_j \leq 1 \qquad\qquad 1 = \sum_{j=1}^{J} \pi_j \tag{2}$$

The parameters for a GMM are estimated with unsupervised machine learning algorithms applied to training data, such as Variational Bayesian Inference (VI) or Expectation-Maximization (EM) (cf. [15] for an in-depth discussion on GMMs, EM, and VI).

B. Classifier Based On Mixture Models

A fitted GMM can be used for decision-making, i.e., classification in a supervised manner, which is called Classifier based on mixture models (CMM) [16]. The class posteriors $P(c|\boldsymbol{x})$ are estimated from labeled training data $\mathcal{X}$:

$$P(c|\boldsymbol{x}) = \sum_{j=1}^{J} P(c|j) \cdot P(j|\boldsymbol{x}) = \sum_{j=1}^{J} \xi_{j,c} \cdot \gamma_{\boldsymbol{x},j}. \tag{3}$$

The probabilities $\gamma_{\boldsymbol{x},j}$ in Eq. (4) are often called *responsibilities* [15] and specify the probability that a given sample $\boldsymbol{x} \in \mathcal{X}$ originates from the component j of the given GMM. The other term $\xi_{j,c}$ describes the probability that a certain component j belongs to class c. These are estimated from the subset $\mathcal{X}_c \subset \mathcal{X}$, cf. Eq. (5), which contains all samples that belong to a class c.

$$\gamma_{\boldsymbol{x},j} = \frac{\pi_j \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu_j}, \boldsymbol{\Sigma}_j)}{\sum_{j'=1}^{J} \pi_{j'} \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_{j'}, \boldsymbol{\Sigma}_{j'})} \tag{4}$$

$$\xi_{j,c} = \frac{\sum_{\boldsymbol{x} \in \mathcal{X}_c} \gamma_{\boldsymbol{x},j}}{\sum_{\boldsymbol{x} \in \mathcal{X}} \gamma_{\boldsymbol{x},j}} \tag{5}$$

The actual decision function $h(\boldsymbol{x})$ is given by the maximum a-posteriori of the class probabilities:

$$h(\boldsymbol{x}) = \underset{c \in \{1,\ldots,C\}}{\arg\max} \left(P(c|\boldsymbol{x})\right). \tag{6}$$

C. Structure-Based Sensor Adaptation With CMM

The data-based adaption method for activity recognition tasks proposed by Jänicke et al. [1] and is sketched in Fig. 2. This approach trains the data's structure using unlabeled high dimensional data and infers the class distributions using

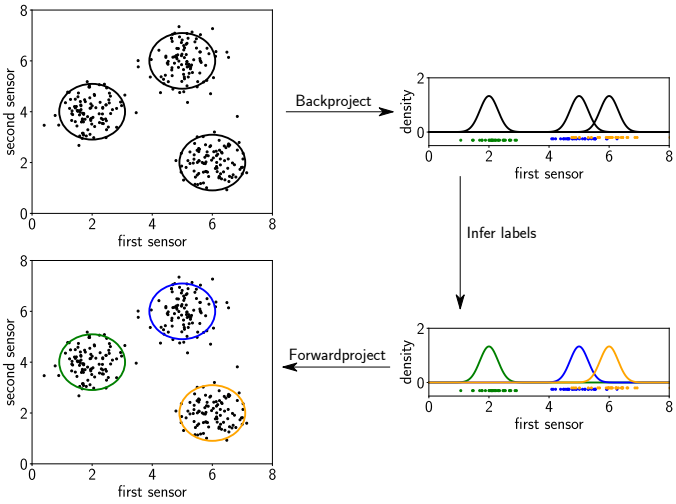

Fig. 2. Adaptation from source domain to target domain. [1]

labeled low dimensional data. The approach assumes that the set of features from the source domain is a strict subset of the features in the target domain. We denote the unlabeled high dimensional data from the target domain as $\boldsymbol{x}^t \in \mathcal{X}^t$ and the labeled data from source domain as $(\boldsymbol{x}^s, y_{\boldsymbol{x}^s})$. The main idea is to train a GMM (GMM^t) in the target domain using $\mathcal{X}^t$. The GMM^t is then projected into the source domain discarding the dimensions not present in the source domain from the mean vectors $\boldsymbol{\mu_j}$ and the covariance matrices $\boldsymbol{\Sigma}_j$ for each component. The resulting GMM (GMM^s) can then be used to infer the class probabilities for each component, summarized in $\boldsymbol{\xi}$, as shown in Eq. (5). These class probability estimates can be used in conjunction with GMM^t as a CMM to classify data in the target domain. Thus, the more detailed knowledge about the data's structure in the target domain is combined with the available labels in the source domain.

D. Multi-Class Probabilistic Active Learning

Multi-class probabilistic active learning (McPAL) [11] is an active learning method extended from Optimised probabilistic active learning [6]. The main idea of McPAL is the acquisition of labels for instances from an unlabeled pool of data (X) with the highest usefulness, which in case of McPAL is the density weighted performance gain, when acquiring the label for a given classifier. To calculate this, we use the density estimate ($P(\boldsymbol{x}|X)$) and the frequency estimate for each class ($\boldsymbol{k_x}$) provided by the classifier.

$$\text{usefulness}(\boldsymbol{x}) = P(\boldsymbol{x}|X) \cdot \text{gain}(\boldsymbol{k_x}) \tag{7}$$

The performance gain is the difference between the current performance and the performance when M additional labels are acquired at $\boldsymbol{x}$.

$$\text{gain}(\boldsymbol{k_x}) = \max_{m \in \{1,\ldots,M\}} \frac{\text{expPerf}(\boldsymbol{k_x}, m) - \text{expCurPerf}(\boldsymbol{k_x})}{m} \tag{8}$$

The performances for calculating gain($\boldsymbol{k_x}$) are estimated by modeling the accuracy as a random variable. This depends on

the frequency estimate for each class $\boldsymbol{k_x}$ and the true posterior $\boldsymbol{p_x}$. As $\boldsymbol{p_x}$ is unknown, the authors show that $\boldsymbol{p_x}$ can be estimated with the dirichlet distribution with the parameter $\boldsymbol{k_x}$+1:

$$P(\boldsymbol{p_x} \mid \boldsymbol{k_x}) = \text{Dir}(\boldsymbol{p_x} \mid \boldsymbol{k_x} + 1). \tag{9}$$

The current performance (Eq. (10)) and the performance after m label acquisitions (Eq. (11)), with $1 < m \leq M$, can be estimated as expected value of the performance measure, in this case accuracy.

$$\text{expCurPerf}(\boldsymbol{k_x}) = \mathop{\mathbb{E}}_{\boldsymbol{p_x}} [\text{perf}(\boldsymbol{k_x} \mid \boldsymbol{p_x})] \tag{10}$$

$$\text{expPerf}(\boldsymbol{k}, m) = \mathop{\mathbb{E}}_{\boldsymbol{p_x}} \left[\mathop{\mathbb{E}}_{\boldsymbol{l}} [\text{perf}(\boldsymbol{k_x} + \boldsymbol{l} \mid \boldsymbol{p_x})] \right] \tag{11}$$

To simulate the label acquisitions, the authors denote the frequency estimates per class for m label acquisitions as $\boldsymbol{l}$, which is multinomially distributed given $\boldsymbol{p_x}$, with:

$$\sum_{c=1}^{C} l_c = m. \tag{12}$$

The accuracy is the true posterior for the class $\hat{y}$ where $\boldsymbol{k}$ has its highest value (Eq. (14)), as the classifier predicts the most likely class according to the labeled data:

$$\text{perf}(\boldsymbol{k_x} \mid \boldsymbol{p_x}) = p_{\boldsymbol{x},\hat{y}}, \tag{13}$$

$$\hat{y} = \mathop{\arg\max}_{y \in \{1,\ldots,C\}} (k_{\boldsymbol{x},y}). \tag{14}$$

IV. Improving Domain Adapted CMMs Using AL

This section shows how the prediction quality of a CMM can be further improved after it has been adapted from the source domain to the target domain. We propose a method to combine labeled data from both domains. Following that, we show how the frequency estimate $\boldsymbol{k_x}$ required for McPAL can be calculated to use the probabilistic gain for selecting the most useful instances

A. Learning From New Labels After Structure-Based Sensor Adaptation

In this section, we propose methods that build upon this approach but allows to further improve the classification performance using additional labeled data from the target domain. While the structure is learned from the target domain, the class distribution is derived from labels in the source domain. As overlapping classes in the source domain may be separable in the target domain, we need additional labeled data from the target domain, as shown in Fig. 3. Using only 2 label acquisitions allows the CMM to distinguish the 3 classes, while this was not possible with the unsupervised HDA alone.

The setting differs slightly from the setting in Section III-C. Now, we have labeled and unlabeled data from target domain. To simplify the notation in this section, $\gamma^s_{\boldsymbol{x},j}$ and $\gamma^t_{\boldsymbol{x},j}$ denote the responsibilities using GMM^{s}and GMM^t, respectively. Additionally, we use $\mathcal{X}^s_c$ and $\mathcal{X}^t_c$ to denote the labeled instances

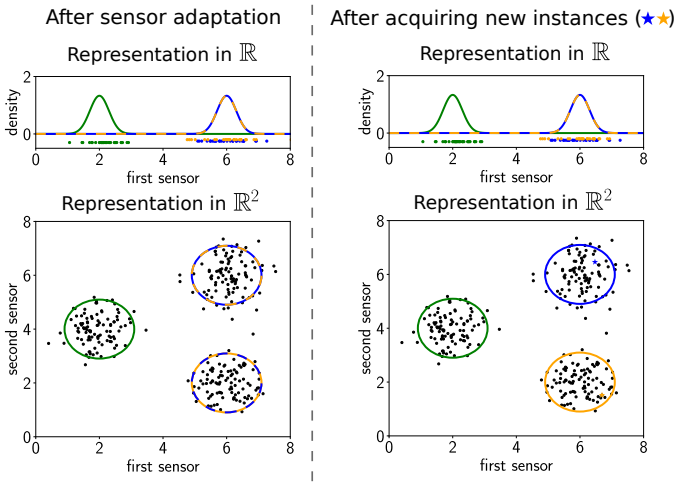

Fig. 3. Using active learning to improve a CMM after HDA.

from class $c \in \{1, \ldots, C\}$ coming from the source and target domain, respectively.

A naive way to integrate the newly labeled data is shown in Eq. (15), which combines the responsibilities from both domains:

$$\xi^{naive}_{j,c} = \frac{\sum\limits_{\boldsymbol{x} \in \mathcal{X}^s_c} \gamma^s_{\boldsymbol{x},j} + \sum\limits_{\boldsymbol{x} \in \mathcal{X}^t_c} \gamma^t_{\boldsymbol{x},j}}{\sum\limits_{\boldsymbol{x} \in \mathcal{X}^s} \gamma^s_{\boldsymbol{x},j} + \sum\limits_{\boldsymbol{x} \in \mathcal{X}^t} \gamma^t_{\boldsymbol{x},j}}. \tag{15}$$

While this may be sufficient to incorporate new knowledge into the model, this method lacks efficiency. As we assume to have only few labeled instances from target domain, the sum of the responsibilities for instances from the target domain is a lot lower than from the source domain. The higher the number of labeled samples in the source domain compared to those in the target domain, the lower the model adaptation. Therefore, decreasing the influence of $\mathcal{X}^s$ hopefully leads to a better performance. Scaling down the responsibilities solves this problem. Hence, we normalize the responsibilities from the source domain by dividing through the sum of all responsibilities from there. Additionally, we introduce a new parameter $\alpha \in \mathbb{R}$, which expresses the average weight of each class for instances from the source domain. Thus, $\alpha \cdot C$ expresses the weight of the data from the source domain, which we use to rescale the normalized responsibilities. To simplify the equations we denote the scalar value used to rescale the responsibilities from the source domain as $\beta \in \mathbb{R}$ as shown in Eq. (16).

$$\beta = \frac{\alpha \cdot C}{\sum\limits_{j=1}^{J} \sum\limits_{\boldsymbol{x} \in \mathcal{X}^s} \gamma^s_{\boldsymbol{x},j}} \tag{16}$$

Hence, the equation to calculate the class probabilities is as follows:

$$\xi^{final}_{j,c} = \frac{\beta \sum\limits_{\boldsymbol{x} \in \mathcal{X}^s_c} \gamma^s_{\boldsymbol{x},j} + \sum\limits_{\boldsymbol{x} \in \mathcal{X}^t_c} \gamma^t_{\boldsymbol{x},j}}{\beta \sum\limits_{\boldsymbol{x} \in \mathcal{X}^s} \gamma^s_{\boldsymbol{x},j} + \sum\limits_{\boldsymbol{x} \in \mathcal{X}^t} \gamma^t_{\boldsymbol{x},j}}. \tag{17}$$

Instead of storing responsibilities, we store a matrix $\boldsymbol{\phi}$, which contains the sum of responsibilities per component j and class c.

$$\phi_{j,c} = \sum_{\boldsymbol{x} \in \mathcal{X}_c} \gamma_{\boldsymbol{x},j}. \tag{18}$$

The matrix $\boldsymbol{\phi}$ is the matrix $\boldsymbol{\xi}$ (Eq. (5)) without the normalization to obtain valid probabilities. This allows us to simplify Eq. (16) and Eq. (17) as follows:

$$\beta = \frac{\alpha \cdot C}{\sum_{j=1}^{J} \sum_{c=1}^{C} \phi_{j,c}}, \tag{19}$$

$$\xi_{j,c}^{final} = \frac{\beta \phi_{j,c}^{s} + \phi_{j,c}^{t}}{\sum_{c=1}^{C} \left(\beta \phi_{j,c}^{s} + \phi_{j,c}^{t} \right)}. \tag{20}$$

B. Optimizing Label Acquisition With Active Learning

This section presents how we can select the necessary labeled data efficiently using McPAL. To be able to apply this selection strategy, we need $\boldsymbol{k_x}$, the vector of frequency estimates, for each instance $\boldsymbol{x} \in \mathcal{X}^u$. We can calculate $\boldsymbol{k_x}$ based on the matrix $\boldsymbol{\phi}$ (Eq. (18)). For this, we calculate the number of labels in the neighborhood for each components' center, denoted as $n_{\boldsymbol{\mu_j}}$:

$$n_{\boldsymbol{\mu_j}} = \sum_{c=1}^{C} \phi_{j,c}^{final}. \tag{21}$$

Using these, we can calculate $\boldsymbol{n_x}$, the number of labels in the neighborhood of x, as sum of $n_{\boldsymbol{\mu_j}}$ weighted with the corresponding responsibilities:

$$n_{\boldsymbol{x}} = \sum_{j=1}^{J} n_{\boldsymbol{\mu_j}} \gamma_{\boldsymbol{x},j}. \tag{22}$$

The observed class posteriors $\hat{\boldsymbol{p}}_{\boldsymbol{x}} = P(c \mid \boldsymbol{x})$ (Eq. 3) of the CMM are used to get the frequency estimate $\boldsymbol{k_x}$:

$$\boldsymbol{k_x} = n_{\boldsymbol{x}} \cdot \hat{\boldsymbol{p}}_{\boldsymbol{x}}. \tag{23}$$

Using the frequency estimate from Eq. (23) and the density estimate from GMM^t, we can calculate the density weighted performance gain (Eq. (7)), and use McPAL to successively select the most beneficial instances for labeling to train the adapted CMM.

V. Experimental Evaluation

In this section we present and discuss our experimental results. The goal of the experiments is to assess whether the proposed approach is able to achieve good performances with fewer labels compared to other active learning strategies. We show the design of our experiments and present the obtained results.

A. Activity Recognition Data

For our experimental evaluation we use the PAMAP2 dataset [17]. This dataset provides data collected in a 45-minute experiment with nine subjects, which were asked to do various activities each for 1 to 3 minutes. During those activities, the subjects were monitored using a heart rate sensor and three inertial measurement units (IMUs). Those IMUs record data with a sampling rate of 100Hz, such as temperature, 3D-acceleration, and 3D-gyroscope data, with the latter two being relevant for our experiments. The IMUs were placed on the chest, the ankle, and wrist on the subject's dominant side. Additionally, a heart rate sensor was placed on the chest but only had a sampling rate of 9Hz and provides the heartbeats per minute. In total, there are 12 monitored activities such as lying, ironing, ascending as well as descending stairs, and running.

B. Data Preparation

One of the subjects (subject 9) was discarded due to having a lot fewer classes than the other subjects. Due to the different sampling frequencies of heart rate monitor and the IMUs, we resampled them to 32 Hz. We use the mean and variance from a sliding window of 4 seconds without overlap as features for the classifier. The data processed this way are the heart rate, acceleration, and gyroscope data. Thus, the data of each IMU consists of 12 dimensions, while the data for the heart rate sensor only consists of 2 dimensions.

C. Design of Experiments

The experiments are done using 5-fold cross-validation with 40 repetitions to compare the proposed method against different selection strategies, namely the proposed algorithm (*McPAL*) to random selection (*Random*) and uncertainty sampling with various uncertainty measures, i.e., best vs. second best [7] (*1vs2*), density weighted uncertainty sampling [9] (*DWUS*), entropy [7] (*Entropy*), and confidence [7] (*Confidence*). The uncertainty measures used for uncertainty sampling are given in the equations below. To simplify the equations, we denote the most probable class as well as the second most probable class according to the classifier's prediction for $\boldsymbol{x}$ as c_1 and c_2, respectively. Additionally, we denote the density estimate of the GMM^t trained on the target domain data as $P_{GMM^t}(\boldsymbol{x})$

$$\mathrm{US}_{\mathrm{1vs2}}(\boldsymbol{x}) = p_{\boldsymbol{x},c_1} - p_{\boldsymbol{x},c_2} \tag{24}$$

$$\mathrm{US}_{\mathrm{DWUS}}(\boldsymbol{x}) = P_{GMM^t}(\boldsymbol{x}) \cdot \mathrm{US}_{\mathrm{1vs2}}(\boldsymbol{x}) \tag{25}$$

$$\mathrm{US}_{\mathrm{Entropy}}(\boldsymbol{x}) = \sum_{c=1}^{C} (p_{\boldsymbol{x},c} \cdot \ln(p_{\boldsymbol{x},c})) \tag{26}$$

$$\mathrm{US}_{\mathrm{Confidence}}(\boldsymbol{x}) = 1 - p_{\boldsymbol{x},c_1} \tag{27}$$

The classifier used for the experiments is a CMM with 30 components. The CMM is extended from the BayesianGaussianMixture in sklearn [18]. For each fold, the training data is permutated randomly and split evenly into two separate training datasets. These represent the labeled data in the source domain and the unlabeled data in the target domain.

Additionally, the training dataset of the source domain is modified so that it only contains the features available in the source domain. The task is to acquire 60 labels for unlabeled instances to improve the CMM. After each label acquisition, we assess the performance using the classifier's accuracy and F1 score, more specifically the macro F1 score, on the test dataset, which is often used in evaluating active learning [19]. The difficulty of datasets may vary greatly between subjects, as each of them is different. Therefore, we normalize the obtained learning curves to be able to aggregate them across subjects. For this purpose, in addition to the selection strategies, we train another classifier, which we denote as *Full*. Here, we use a CMM trained on all labels. Simply dividing the achieved performance, i.e. accuracy and F1 score, of the selection strategy by the performance of *Full* allows us to express the performance in relation to the performance of *Full*. The averaged learning curve of those performance values is then multiplied by the mean performance of *Full* to obtain normalized performance values. To reduce the number of tested sensor combinations, we fix the order in which sensors can be added. This order depends on the performance a trained CMM could achieve using a sensor alone. We assessed this by splitting the sensor data 40 times into training and test data. These are used to evaluate the performance a CMM could reach. The performance across subjects is then aggregated in the same fashion as we did for the real experiments (explained in more detail in Section V-C). The CMM performs the worst on the heart rate sensor (HR) data, while the order of IMUs is as follows: ankle (AI), chest (CI), and hand (HI). For our experiments, we sorted the sensors according to this ranking (HR, AI, CI, HI) for one half of our experiments and sorted them in the reverse order of the ranking (HI, CI, AI, HR) for the other half. Thus, for half of our experiments, we add increasingly better-performing sensors to a set of worse-performing sensors. In contrast, for the other half, we add worse sensors to sensors that perform better individually compared to the new sensor. To denote the various sensor constellations, we separate the sensors, which span the source domain, with a "+" from the sensors, which are additionally used to spawn the target domain. Hence, for "HR + AI, CI" the source domain is spanned by HR while HR, AI, and CI span the target domain. Additionally, we set α (Eq. (16)) to 1, 10, 20 and 50 to evaluate the sensitivity of the parameter.

D. Experimental Results

In this section, we present the results for our experiments. A good active learning algorithm achieves high performances with few labels. Hence, a learning curve for such an approach can be identified by a steep performance increase. In this article we can only show a subset of the results for our experiments. To see our testing framework, all results and all learning curves confer to our appendix[1].

[1]https://p.ies.uni-kassel.de/hdaal

TABLE I
THE NUMBER OF DATASETS WHERE THE CHOSEN α RESULTS IN THE HIGHEST PERFORMANCES FOR *Full*. THE NUMBER OF DATASETS IS 20.

	α=1	α=10	α=20	α=50
Datasets, where α leads to the best accuracy	3	6	8	3
Datasets, where α leads to the best F1 score	3	5	10	2

In Table I, we compare the four tested α-values in terms of achievable performance. An α of 20 results in higher performances more often than 1, 10 and 50. Even though the performance differences are often small, the performance difference during the first 60 label acquisitions varies greatly. This is shown in Fig. 4.

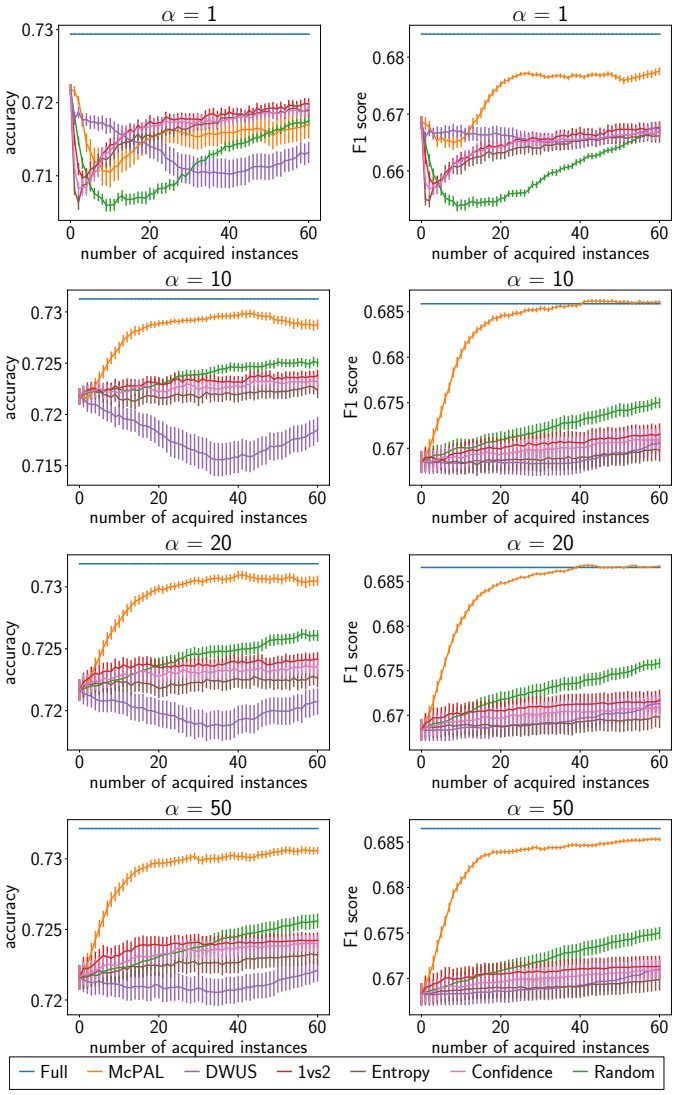

Fig. 4. Accuracy and F1 score for $\alpha \in \{1, 10, 20, 50\}$. The dataset used in these plots is HI,CI,AI + HR.

There, we show the learning curves for a single sensor setting (HI,CI,AI + HR) with all tested α-values ($\alpha \in \{1, 10, 20, 50\}$). In the case of $\alpha = 1$, we observe a sudden decrease in performance during the first label acquisitions

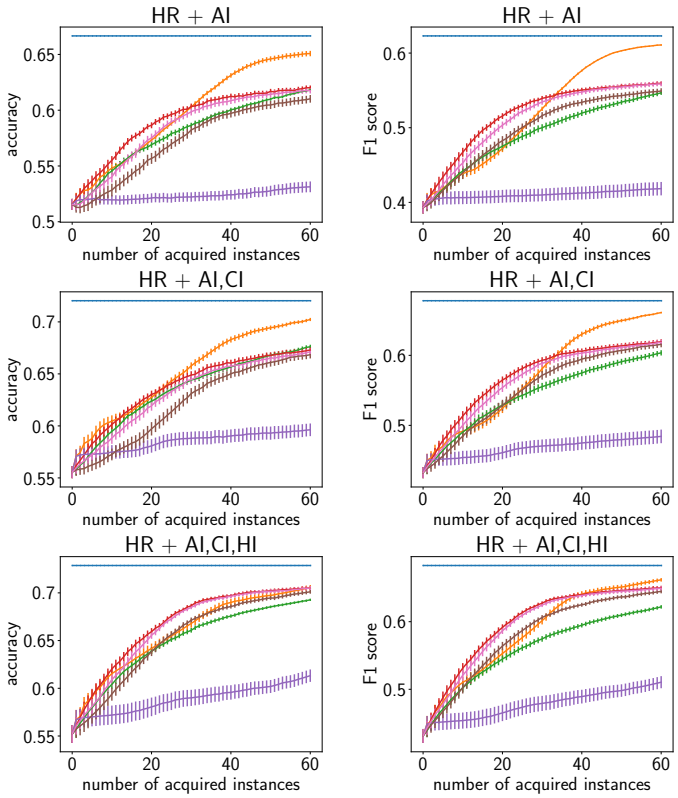

Fig. 5. Accuracy and F1 score for $\alpha = 1$ where there is no sudden decrease in classification performance.

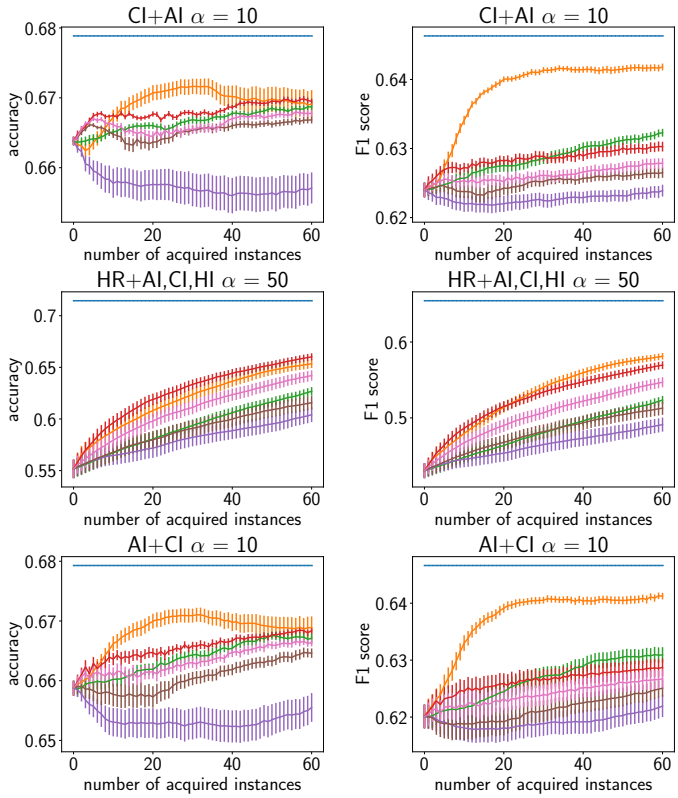

Fig. 6. Accuracy and F1 score for experiments with $\alpha \in \{10, 20, 50\}$ where *McPAL* performs worse or only negligible better, in terms of accuracy, than its competitors.

for almost all selection strategies. This phenomenon is not present for the other 3 cases. The reason for this phenomenon is that the knowledge gained from the source domain is almost ignored with such a small α. Hence, prior knowledge learned from the source domain has to be relearned. Thus, we see a sudden decrease in performance for the first acquisitions which leads to a worse performance after 60 label acquisitions than after the initial adaptation. There are only three cases where the performance of the classifiers do not decrease after acquiring new labels for $\alpha = 1$. Those are the configurations where the source domain is spanned by HR as shown in Fig. 5.

This indicates that separating the classes is easier using the IMU sensors than HR. Hence, the trade-off between forgetting the knowledge from HR and gaining knowledge from the other sensors is still beneficial in terms of classification performance. For $\alpha \in \{10, 20, 50\}$, there is no such sudden decrease, with the exception of *DWUS*, which performs poorly regardless of the chosen α. The approaches *1vs2*, *Entropy* and *Confidence* perform better for $\alpha = 1$. However, *McPAL* outperforms all its competitors in the remaining three cases, where even *Random* achieves a higher performance than the uncertainty sampling strategies.

Out of all 60 experiments for $\alpha \in \{10, 20, 50\}$, there are only two experiments where *McPAL* does not outperform its competitors after 60 label acquisitions in terms of accuracy. However, in terms of F1 score, *McPAL* outperforms the other selection strategies with $\alpha \in \{1, 10, 20, 50\}$ in each experiment. The experiments where *McPAL* performs worse, are CI + AI with $\alpha = 10$, and HR + AI,CI,HI with $\alpha = 50$. Additionally, for AI + CI with $\alpha = 10$ *McPAL* performs only negligible better than *1vs2*. Those cases are shown in Fig. 6.

In Fig. 7, we present learning curves for different sensor constellations but fixed $\alpha = 20$. The experiments show that *1vs2* performs best out of all tested uncertainty measures, but is still outperformed by *Random* in some experiments. *Confidence* performs similarly to *1vs2* while *Entropy* and *DWUS* achieve poor results. In some cases, *DWUS* even leads to worse performance than prior to acquiring new labels. *McPAL* achieves the highest performances after 60 label acquisitions for all plots shown in Fig. 7. For HR + AI,CI,HI, the performance increases for *1vs2* during the first label acquisitions is higher than for *McPAL*, but plateaus quickly.

E. Discussion

Our experiments show that the usage of labeled data from the target domain can help to improve a model adapted from the source domain to the target domain using unsupervised HDA. This is especially noticeable as the performance of *Full* is always higher than the performance of the adapted classifier without any label acquisitions from the target domain. Additionally, the usage of active learning in this scenario reduces the number of required labeled data. *McPAL* outperforms its competitors in this case as its performance

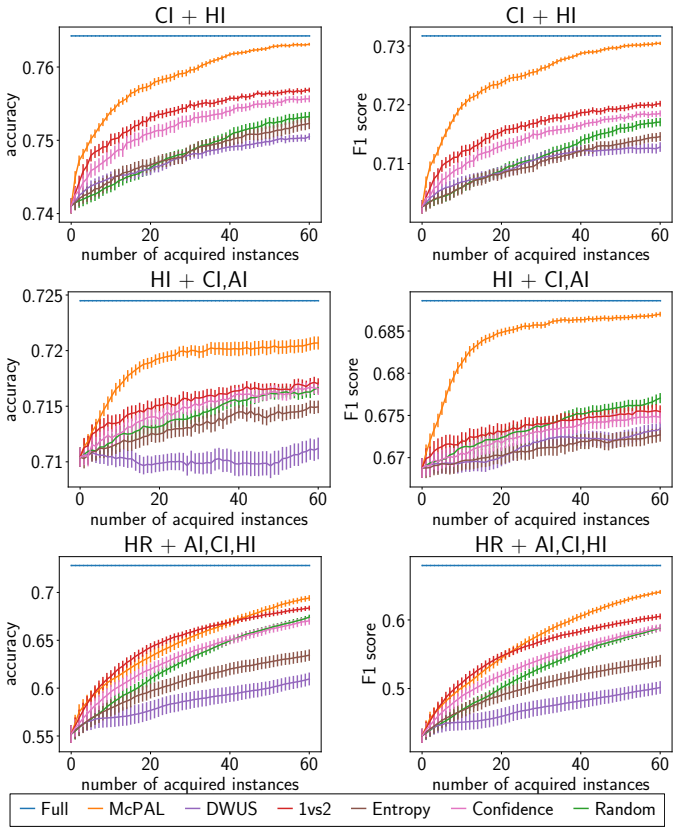

Fig. 7. Accuracy and F1 score for various sensor constellations with $\alpha = 20$.

improvement occurs after fewer label acquisitions than with the tested strategies. However, the experiments highlight the importance of using an appropriate value for α. Given an α which is too small, in our case $\alpha = 1$, the performance decreases as the model has to relearn knowledge from the original input source. Additionally, our experiments show that using uncertainty sampling may result in worse performances compared to random sampling.

VI. Conclusion

This article shows that active learning is beneficial in HDA. The setting we investigate starts with a model adapted using unsupervised HDA. Using active learning, we incorporate labeled data iteratively to improve this model. To do that, we extended the HDA method proposed in [1].

Our extension allows for further tuning of the classifier, which improves the classification performance in areas where the classes overlap in the source domain, but are separated in the target domain. Furthermore, we use the underlying CMM to obtain density and frequency estimates, which allows us to efficiently query instances using McPAL. Hence, we are now able to handle unsupervised as well as supervised HDA.

We conducted experiments using the PAMAP2 dataset, which consists of real data for activity recognition using different sensors simultaneously. The results show that the model can be improved, using only a few label acquisitions (in our case 60) compared to using only the unsupervised HDA.

The proposed method can be extended in various ways. This method can still be optimized by automatically estimating a suitable α. Another research direction is streaming data. This impacts the adaptation as we expect to know the data's distribution beforehand to fit the CMM's components as well as the incorporation of knowledge in the target domain. Currently, the method expects all sensors available in the source domain to be available in the target domain. This is not necessarily the case and should be investigated in the future as well. Finally, the proposed algorithms may be applied to other suitable areas of machine learning.

References

[1] M. Jänicke, B. Sick, and S. Tomforde, "Self-Adaptive Multi-Sensor Activity Recognition Systems Based on Gaussian Mixture Models," *Informatics*, vol. 5, no. 3, p. 38, 2018.

[2] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu, "Sensor-based activity recognition," *IEEE Trans. Syst., Man, Cybern. C*, vol. 42, no. 6, pp. 790–808, Nov 2012.

[3] L. Duan, D. Xu, and I. W. Tsang, "Learning with Augmented Features for Heterogeneous Domain Adaptation," Tech. Rep., 2012.

[4] J. J. Jiang, "A literature survey on domain adaptation of statistical classifiers," 2007.

[5] H. Daumé III, "Frustratingly Easy Domain Adaptation," Tech. Rep., 2009.

[6] G. Krempl, D. Kottke, and V. Lemaire, "Optimised probabilistic active learning (opal)," *Machine Learning*, vol. 100, no. 2, pp. 449–476, 2015.

[7] B. Settles, "Active learning literature survey," *University of Wisconsin, Madison*, vol. 52, 2010.

[8] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '94. Springer-Verlag New York, Inc., 1994, pp. 3–12.

[9] H. T. Nguyen and A. W. M. Smeulders, "Active learning using pre-clustering," in *ICML*, 2004.

[10] N. Roy and A. McCallum, "Toward optimal active learning through sampling estimation of error reduction," in *Proceedings of the Eighteenth International Conference on Machine Learning*, ser. ICML '01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 441–448.

[11] D. Kottke, G. Krempl, D. Lang, J. Teschner, and M. Spiliopoulou, "Multi-class probabilistic active learning," *Frontiers in Artificial Intelligence and Applications*, vol. 285, pp. 586–594, 2016.

[12] R. Liu, T. Chen, and L. Huang, "Research on human activity recognition based on active learning," in *2010 International Conference on Machine Learning and Cybernetics*, vol. 1, July 2010, pp. 285–290.

[13] L. Zhao, G. R. Sukthankar, and R. Sukthankar, "Robust active learning using crowdsourced annotations for activity recognition," in *Human Computation*, 2011.

[14] M. Stikic, K. Van Laerhoven, and B. Schiele, "Exploring semi-supervised and active learning for activity recognition," in *2008 12th IEEE International Symposium on Wearable Computers*. IEEE, 2008, pp. 81–88.

[15] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2006.

[16] T. Reitmaier, A. Calma, and B. Sick, "Transductive active learning – a new semi-supervised learning approach based on iteratively refined generative models to capture structure in data," *Information Sciences*, vol. 293, pp. 275 – 298, 2015.

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

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[19] D. Kottke, A. Calma, D. Huseljic, G. Krempl, and B. Sick, "Challenges of reliable, realistic and comparable active learning evaluation," in *IAL@PKDD/ECML*, 2017.