

DAPPER: Label-Free Performance Estimation after Personalization for Heterogeneous Mobile Sensing

TAESIK GONG, School of Electrical Engineering, KAIST, Republic of Korea and Nokia Bell Labs, UK

YEWON KIM, School of Electrical Engineering, KAIST, Republic of Korea

ADIBA ORZIKULOVA, School of Electrical Engineering, KAIST, Republic of Korea

YUNXIN LIU, Institute for AI Industry Research (AIR), Tsinghua University and Shanghai Artificial Intelligence Laboratory, China

SUNG JU HWANG, Graduate School of AI, KAIST, Republic of Korea

JINWOO SHIN, Graduate School of AI, KAIST, Republic of Korea

SUNG-JU LEE, School of Electrical Engineering, KAIST, Republic of Korea

Many applications utilize sensors in mobile devices and machine learning to provide novel services. However, various factors such as different users, devices, and environments impact the performance of such applications, thus making the domain shift (i.e., distributional shift between the training domain and the target domain) a critical issue in mobile sensing. Despite attempts in domain adaptation to solve this challenging problem, their performance is unreliable due to the complex interplay among diverse factors. In principle, the performance uncertainty can be identified and redeemed by performance validation with ground-truth labels. However, it is infeasible for every user to collect high-quality, sufficient labeled data. To address the issue, we present DAPPER (Domain AdaPtation Performance Estimator) that estimates the adaptation performance in a target domain with only *unlabeled* target data. Our key idea is to approximate the model performance based on the mutual information between the model inputs and corresponding outputs. Our evaluation with four real-world sensing datasets compared against six baselines shows that on average, DAPPER outperforms the state-of-the-art baseline by 39.8% in estimation accuracy. Moreover, our on-device experiment shows that DAPPER achieves up to 396× less computation overhead compared with the baselines.

CCS Concepts: • **Human-centered computing** → **Ubiquitous and mobile computing systems and tools**.

Additional Key Words and Phrases: Mobile sensing; Deep learning; Domain adaptation; Performance estimation

ACM Reference Format:

Taesik Gong, Yewon Kim, Adiba Orzikulova, Yunxin Liu, Sung Ju Hwang, Jinwoo Shin, and Sung-Ju Lee. 2023. DAPPER: Label-Free Performance Estimation after Personalization for Heterogeneous Mobile Sensing. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 7, 2, Article 55 (June 2023), 27 pages. <https://doi.org/10.1145/3596256>

Authors' addresses: **Taesik Gong**, School of Electrical Engineering, KAIST, Republic of Korea, taesik.gong@kaist.ac.kr; **Yewon Kim**, School of Electrical Engineering, KAIST, Republic of Korea, yewon.e.kim@kaist.ac.kr; **Adiba Orzikulova**, School of Electrical Engineering, KAIST, Republic of Korea, adiorz@kaist.ac.kr; **Yunxin Liu**, Institute for AI Industry Research (AIR), Tsinghua University and Shanghai Artificial Intelligence Laboratory, China, liuyunxin@air.tsinghua.edu.cn; **Sung Ju Hwang**, Graduate School of AI, KAIST, Republic of Korea, sjhwang82@kaist.ac.kr; **Jinwoo Shin**, Graduate School of AI, KAIST, Republic of Korea, jinwoos@kaist.ac.kr; **Sung-Ju Lee**, School of Electrical Engineering, KAIST, Republic of Korea, profsj@kaist.ac.kr.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.

2474-9567/2023/6-ART55 \$15.00

<https://doi.org/10.1145/3596256>

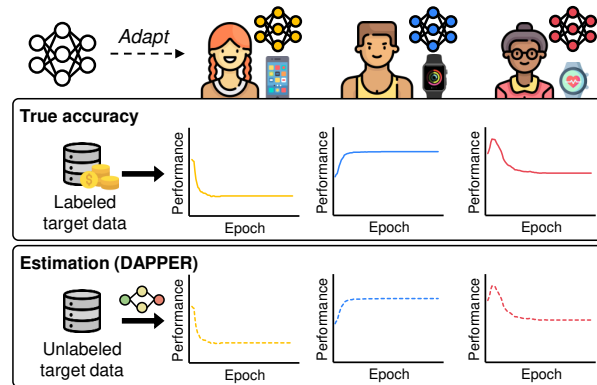


Fig. 1. DAPPER estimates the adaptation performance with only unlabeled data from the target user.

1 INTRODUCTION

Mobile sensing utilizes the sensors from mobile devices (e.g., smartphones and wearables) to infer user contexts and provide appropriate services accordingly. Integrated with deep learning, which enables understanding of multi-dimensional sensory data, mobile sensing has broad applications ranging from human activity recognition [60, 62, 68] to context recognition [46, 83], authentication [7, 44, 93], emotion recognition [45, 49], and healthcare [89–91]. While a body of research in this area has demonstrated its potential, existing methods have been limited in real-world deployments due to the *domain shift* issue, i.e., distribution shift of target data with respect to training (or source) data [21, 70, 76]. Specifically, users have different physical conditions and behaviors, and their mobile devices have various specifications, making each user’s sensing distribution different from others. Therefore, a pre-trained model typically suffers from poor generalization to an unseen user or device. A popular approach in addressing this problem is *domain adaptation* that utilizes labeled/unlabeled samples [6, 15, 21, 30, 48, 80, 94] to adapt to the target condition.

Domain adaptation for mobile sensing is particularly challenging as countless combinations between users and devices, the quality and quantity of the target samples, numerous hyperparameters, and even software/hardware are known to affect the model’s performance [47, 52]. As such, the performance gain after adaptation is usually uncertain under such heterogeneous mobile sensing environments.

Two common practices used for model evaluation in domain adaptation have critical limitations to solve the aforementioned problem. First, many studies trained until a fixed number of training epochs (e.g., 100) and reported the last epoch’s performance [5, 11, 15, 17, 25, 34, 38, 43, 67, 71, 78, 81, 94]. However, simply using the last epoch’s model often leads to sub-optimal (or even worse) performance with unnecessary computational overhead. To avoid such undesirable situations, performance validation is critical regardless of adaptation algorithms. Second, existing studies used labeled test data from each target domain and reported the best accuracy [10, 13, 21, 29, 41, 73, 74, 79, 80, 82]. However, it is extremely costly and unrealistic to require every user to collect data and manually label them for validation. Our research question arises here: *can we validate the model performance on the target domain without user effort?*

We propose *DAPPER* (Domain AdaPtation Performance Estimator), a label-free performance estimation for domain adaptation in mobile sensing (Figure 1). Once deployed to users, DAPPER leverages only unlabeled target data for performance estimation. As unlabeled time-series sensory data could be easily collected in mobile sensing applications, DAPPER does not require any labeling effort. As DAPPER needs training only once before deployment, it does not incur extra training overhead on target devices, which is beneficial in resource-constrained mobile devices. To the best of our knowledge, DAPPER is the first proposal of performance estimation with

unlabeled data in mobile sensing. Knowing the adapted performance has many benefits for the deployment of sensing models. One can avoid the aforementioned undesirable accuracy degradation or computational overhead. It could also help model maintenance; model developers could monitor whether their sensing applications provide desired performance for their customers and conduct additional actions when needed. Moreover, recent studies found that demonstrating the model’s performance improves user trust in AI systems [35, 85, 87].

DAPPER’s performance estimation with unlabeled data is based on information theory [66]. Specifically, we focus on the information gain in the model predictions given unlabeled target data, and then we demonstrate the correlation between the information gain and the actual performance on the target domain. Based on the analysis, we design a lightweight label-free performance estimation network that leverages domain knowledge, such as dataset-specific characteristics and the nature of the adaptation process. Note that training neural networks requires a large number of training instances for generalization. Without additional data collection, we generate virtual training data by simulating various adaptation scenarios only with source data.

We evaluate our proposed method with four real-world datasets: Heterogeneity Human Activity Recognition (HHAR) [70], Wearable Stress and Affect Detection (WESAD) [65], Individual-Condition Human Activity Recognition (ICHAR) [21], and Individual-Condition Speech Recognition (ICSR) [21]. We demonstrate the effectiveness of our method compared with six baselines, including the state-of-the-art performance estimation algorithm for domain adaptation [8]. Our results show that DAPPER is the most accurate performance estimator, outperforming the state-of-the-art baseline by 39.8% on average in estimation accuracy. In fact, we discovered that the state-of-the-art baselines do not perform well due to their assumptions that do not hold in mobile sensing scenarios. We further evaluate the computational overhead by implementing DAPPER and the baselines on mobile devices with the Mobile Neural Network (MNN) framework [28]. The results show that DAPPER requires 180~396× fewer computations than the state-of-the-art training-based algorithms and only 0.2% extra computational overhead compared with validation from the labeled test data.

We summarize our contributions as follows:

- We highlight the importance of performance validation in domain adaptation for mobile sensing by uncovering the performance dynamics under domain shifts and suggesting useful applications of performance validation.
- We present DAPPER, the first domain adaptation performance estimation in mobile sensing with *unlabeled* target data. DAPPER combats the uncertainty of the adapted performance by modeling the relationship between model outputs and the performance.
- We conduct extensive experiments with four real-world datasets against six baselines including the state-of-the-art performance estimation algorithm. Our evaluation indicates that DAPPER not only outperforms the baselines in estimation accuracy but also requires significantly less computational overhead.

2 BACKGROUND AND MOTIVATION

2.1 Performance Dynamics in Mobile Sensing

In mobile sensing, users have different behaviors and physical characteristics. For instance, an elderly’s jogging might be slower than a young person’s jogging. Moreover, their mobile devices also have diverse hardware specifications, such as sensing sampling rates. In addition, the placement of mobile devices varies according to the type of device (smartphone vs. smartwatch) and user’s preference (hand vs. pocket). These differences collectively create a unique sensing condition for each individual [21, 70, 76]. Unlike other research fields where domains are relatively well-defined and categorized (e.g., real-world objects vs. object icons in computer vision), every target user becomes a unique domain in mobile sensing. Therefore, *domain shift*, i.e., distribution shift of the target data with respect to the source data, is inevitable in mobile sensing and is regarded as a major hurdle

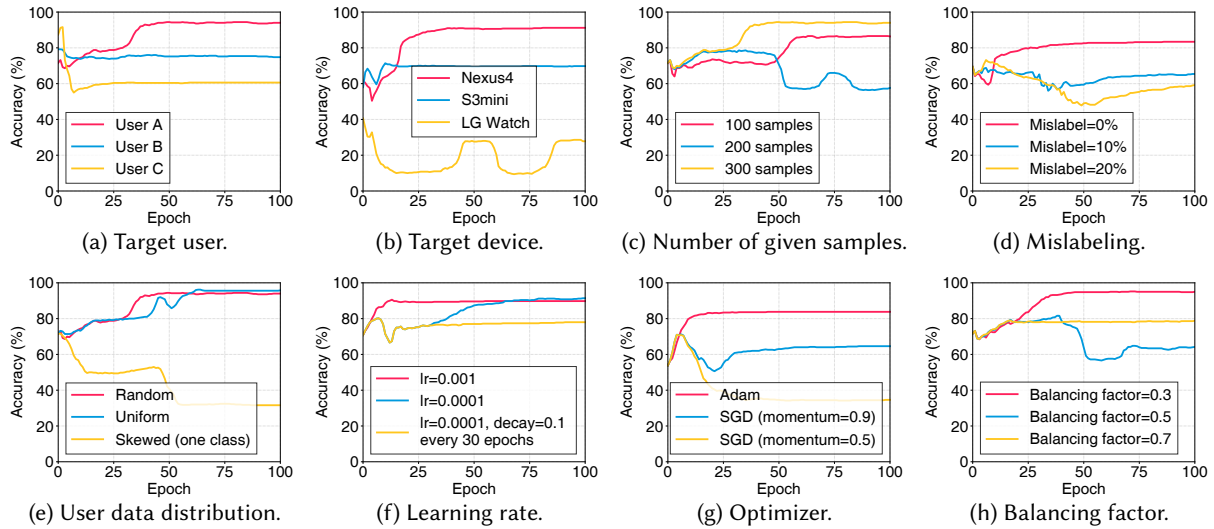


Fig. 2. Performance dynamics in mobile sensing domain adaptation. We varied the (i) user-related factors in mobile sensing (the target user, the target device, the number of given samples, the mislabeling rate, and the sample distribution) and (ii) varied machine-learning hyperparameters (the learning rate, the optimizer, and the algorithm-specific hyperparameter).

for realizing mobile sensing in the wild. To tackle this problem, various *domain adaptation* (DA) algorithms have been proposed in mobile sensing; given a model trained on one or more source domains, a domain adaptation algorithm utilizes few labeled and/or unlabeled target data to adapt to the target domain [6, 15, 21, 30, 48, 80, 94].

While previous research has focused on the algorithmic aspect of domain adaptation, we argue that the performance dynamics in the adaptation stage must be addressed. In addition to the user behaviors, numerous other factors, such as physical characteristics, the device’s sensor specifications, and device placements, also affect the adapted performance. The quantity and quality of the data from the target user are usually not guaranteed (e.g., mislabeling, incorrect time stamps, skewed data distribution, etc.) [3]. When the model is deployed, both software (OSes, library versions, etc.) and hardware (GPUs, etc.) specifications on which adaptation is applied also impact the performance [52]. These factors are dependent on every possible target user and difficult to control in the model development stage. Furthermore, traditional machine learning design choices, such as model architecture, set of hyperparameters, or training algorithms, lie in a huge space and are known to be extremely difficult to optimize for every single domain [47].

We conducted a preliminary empirical study to see how various factors affect adaptation performance. We used the Heterogeneity Human Activity Recognition (HHAR) dataset [70] and considered a common multi-source domain adaptation scenario: for each target domain, we pre-trained a base model with the remaining domains, and the base model was adapted to the target domain. We used a recent unsupervised domain adaptation algorithm (SHOT [38]) with 300 unlabeled samples except for the mislabeling rate experiment, where we used fine-tuning with 25 labeled data. We used 1D convolutional neural networks (CNN) and the Adam optimizer with a fixed learning rate of 0.001 as the default. Additional details about the dataset and the training procedure are in §4. We varied diverse user-related factors in mobile sensing: the user, the device, the number of given target samples, the mislabeling rate, and the distribution of samples with respect to classes. We also varied machine-learning hyperparameters: the learning rate, the optimizer, and the algorithm-specific hyperparameter (balancing factor). For each experiment, all other factors remained the same.

Figure 2 visualizes the accuracy on the target test dataset as the adaptation proceeded for 100 training epochs. The adapted accuracy has diverse patterns according to a variety of factors. As multiple factors interplay and

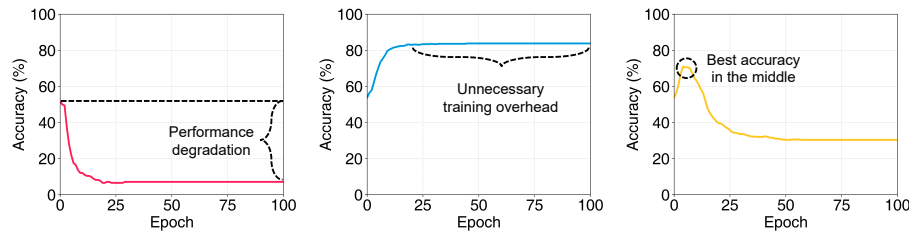


Fig. 3. Three observed problems in domain adaptation for a fixed epoch without validation: (i) performance degradation, (ii) unnecessary training overhead, and (iii) best accuracy in the middle.

some are beyond the developer’s control, it is difficult to predict the adapted performance. For instance, Figure 2c shows that “100 samples” converged to a higher accuracy than “200 samples,” which contradicts the common expectation that training with more samples would result in higher accuracy. We found that this often happens as a higher number of samples does not guarantee a higher quality of the samples with respect to the algorithm’s objective.

2.2 Limitations of Previous Approaches

We found that common approaches in performance validation for domain adaptation have critical limitations. Many studies used a fixed number of training epochs and reported the accuracy at the last epoch [5, 11, 15, 17, 25, 34, 38, 43, 67, 71, 78, 81, 94]. However, simply selecting the model at the last epoch suffers from several critical issues, as we illustrate in Figure 3. First, in some cases, adaptation results in undesirable performance degradation due to various factors, as discussed in §2.1. Second, if the model’s accuracy has rapidly saturated in the first few epochs, training for more epochs will bring in unnecessary computational overhead. Third, the model’s accuracy can fluctuate and reach its maximum in the middle of training. Thus, selecting the last epoch’s model without proper performance validation might lead to selecting a sub-optimal model. In summary, merely choosing the model at a certain point without validation would often be subject to performance issues, highlighting the importance of performance validation.

Note that the accuracies reported in Figures 2 and 3 are calculated from the labeled test data. Similarly, existing studies often used hold-out labeled test data to compute accuracy and reported the best model’s accuracy among epochs to validate the model’s performance precisely [10, 13, 21, 29, 41, 73, 74, 79, 80, 82]. However, it is extremely unrealistic for every user to collect an adequate amount of high-quality data and manually label them. Moreover, non-expert users could often mislabel data, which further harms the reliability of the labels [88].

2.3 Applications of Performance Validation in Mobile Sensing

Knowing the adapted performance has many benefits in the deployment of sensing models. We summarize the importance of performance validation in practical settings.

2.3.1 Avoiding Performance Degradation. With the user and device factor added to the complex hyperparameter space of deep learning, domain adaptation in mobile sensing does not necessarily guarantee performance improvement for an arbitrary user. Moreover, the performance often degrades after adaptation, as shown in Figure 3. If the accuracy after adaptation could be estimated, one can avoid such accuracy degradation by discarding the updated model.

2.3.2 Reducing Computational Overhead. Performance validation could reduce unnecessary computation overhead, which is crucial in resource-constrained devices. For instance, a model sometimes converges before it reaches the last training epoch, as illustrated in Figure 3. With the performance knowledge of the adapted model,

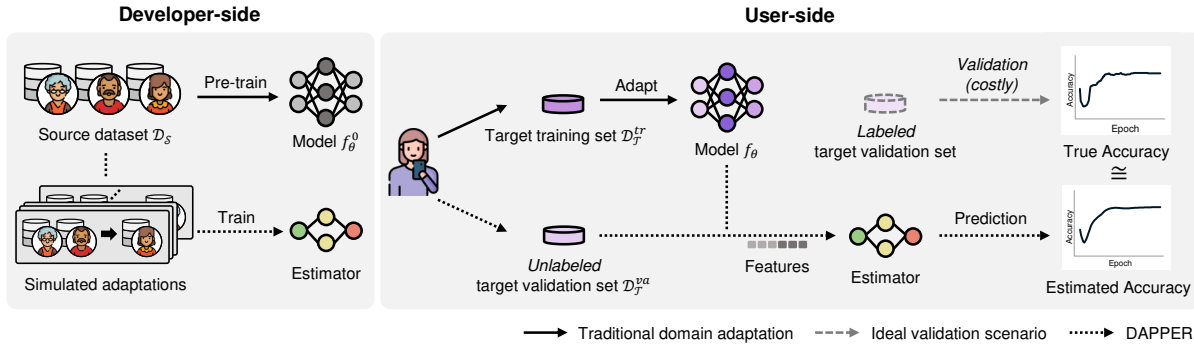


Fig. 4. An overview of DAPPER that estimates the performance of an adapted model with unlabeled target data.

the system could halt adaptation and avoid redundant computation when the model's performance does not improve for consecutive training iterations or reaches a satisfactory performance.

2.3.3 Model Development and Maintenance. Performance validation could help model maintenance. Model developers (or companies) could monitor whether their sensing applications are providing desired performance for their customers and conduct additional actions when needed. For instance, developers could group poorly-performing models and improve them by identifying the common characteristics of the users. Furthermore, developers could provide personalization such as selecting the best hyperparameters/models for a specific user to improve performance.

2.3.4 User-Interactive Adaptation. Recent studies have demonstrated that providing the model performance enhances users' trust and acceptance of ML systems, which leads to improved user experiences [35, 85, 87]. Knowing the performance of a model allows designing a user-interactive adaptation algorithm. For instance, the model could proactively ask the user to collect more (labeled) data when the current performance is poor. When the presented performance is not satisfactory, the user could accept the request to gather more data. After adaptation, the model provides the user with the performance gain from the help of the user's data collection effort. If the model performs well, there is no need to collect further data. This AI-assisted decision-making not only lets users understand the model's current behavior better but also effectively utilizes users' feedback for model improvement.

3 DAPPER: A LABEL-FREE PERFORMANCE ESTIMATOR FOR MOBILE SENSING

3.1 Overview

Our goal is to validate the performance of a model under heterogeneous mobile sensing scenarios without user effort. Figure 4 shows the overall workflow of our performance estimation framework, *DAPPER*. We follow a common domain adaptation setting, where a pre-trained model with source data is deployed as an initial model and is adapted to the target user with some training data from the target domain [21, 38]. *DAPPER* estimates the change in the performance as close as possible to the change in the performance calculated from the labeled data, utilizing only unlabeled data from the target domain. Note that it is extremely challenging, if not impossible, to compute the exact performance for a target domain without labeled data. We approach this problem with information theory [66]. By analyzing the information gain between the inputs and the outputs of the model, we demonstrate the correlation between the information gain and the actual performance on the target domain. Furthermore, we design a performance estimator to overcome the challenges in applying the theoretical analysis in practice.

Design principles of DAPPER for mobile sensing: DAPPER is designed by considering the characteristics and challenges of mobile sensing. First, collecting unlabeled time-series sensory data from a target user could be done in mobile sensing without the user’s manual effort [2, 72]. Taking human activity recognition as an example, while a user is walking, unlabeled sensory data of walking comes to the recognition app. DAPPER utilizes such labor-free unlabeled data to estimate the performance of the adapted model. Second, obtaining high-quality labeled data in mobile sensing is costly as it involves users in the data collection and labeling process. Moreover, lots of labeled data are necessary to train an accurate estimator. DAPPER mitigates this problem by training with *simulated* data in which diverse adaptation scenarios are generated from the already collected source data. Third, minimizing the computation overhead of an estimation model is critical in resource-constrained mobile/wearable devices. By design, DAPPER requires training only once from the developer side and no further training from the user side. Moreover, DAPPER’s lightweight features and architecture collectively make the inference overhead on the user side negligible.

3.2 Problem Formulation

Formally, let $\mathcal{D}_S = \{\mathbf{x}_S^{(i)}, y_S^{(i)}\}_{i=1}^{N_S}$ be the source data collected from one or more domains and $\mathcal{D}_T = \{\mathbf{x}_T^{(i)}, y_T^{(i)}\}_{i=1}^{N_T}$ be the target data to adapt to, where $x^{(i)}$ and $y^{(i)}$ denote data instances and their corresponding labels regardless of their subscripts. We follow the standard covariate shift assumption in domain adaptation, which is defined as $P_S(\mathbf{x}) \neq P_T(\mathbf{x})$ and $P_S(y|\mathbf{x}) = P_T(y|\mathbf{x})$ [59] where $P_S(\mathbf{x})$ and $P_T(\mathbf{x})$ refer to the probability distribution of the source and target domains, respectively. We consider a realistic scenario where $\mathcal{D}_T \not\subset \mathcal{D}_S$. Let \mathcal{D}_T^{tr} be training data for adaptation and \mathcal{D}_T^{va} be unlabeled validation data, where $\mathcal{D}_T^{tr}, \mathcal{D}_T^{va} \subset \mathcal{D}_T$, and $\mathcal{D}_T^{tr} \cap \mathcal{D}_T^{va} = \emptyset$. We want to estimate the adapted model’s performance with unlabeled target validation data \mathcal{D}_T^{va} . We are interested in the target *accuracy*,¹ which is the most representative metric of performance. We focus on estimating the *change in the target accuracy* as adaptation proceeds. We use the terms “performance” and “accuracy” interchangeably throughout the paper. Specifically, the target accuracy at epoch $e \in E = \{0, 1, 2, \dots, T\}$ is defined as

$$a^{(e)} = \frac{\sum_{(\mathbf{x}_T, y_T) \in \mathcal{D}_T^{va}} \mathbb{1}_{[\hat{y}_T^{(e)} = y_T]}}{N^T}, \quad (1)$$

where $\hat{y}_T^{(e)}$ is a predicted label with an adapted model at epoch e , i.e., $\hat{f}_\theta^{(e)}(\mathbf{x}_T)$, and $\mathbb{1}_{[\hat{y}_T^{(e)} = y_T]}$ is one if $\hat{y}_T^{(e)} = y_T$ and zero otherwise. Accordingly, the *change in the target accuracy* at epoch e is defined as follows:

$$\Delta a^{(e)} = a^{(e)} - a^{(0)}, \quad (2)$$

which represents how accuracy changed after adaptation compared against before adaptation. Our objective is to minimize the difference between the change in the ground-truth accuracy and the estimated change in accuracy computed from the unlabeled target validation data \mathcal{D}_T^{va} during the training epochs, which is defined as follows:

$$\text{minimize} \sum_{e=1}^T \left| \Delta a^{(e)} - \Delta \hat{a}^{(e)} \right|, \quad (3)$$

where $\Delta \hat{a}^{(e)}$ is the estimated accuracy change of $f_\theta^{(e)}$ through \mathcal{D}_T^{va} . We chose this objective as it is intuitive and one of the most widely used objectives (i.e., L1 distance).

3.3 Theoretical Insight

We tackle the problem of performance estimation with unlabeled data based on *mutual information* [66]. In information theory, mutual information is a measure that quantifies the amount of information gained about

¹We have also tried other metrics, such as the F1 score, and observed similar findings throughout the experiments.

a random variable after observing the other random variable. The *infomax principle* [40] explains that neural networks can be viewed as an information channel and proposes a principle to self-organize such networks by conveying as much information as possible of given data. This principle has been widely applied in deep learning, such as feature learning [1] and clustering [27]. Grounded on this, we explain the performance of the model in the aspect of the information gain between the inputs and the corresponding outputs of a model. We then demonstrate that it can be derived via Monte Carlo estimation. Finally, we investigate the relationship between mutual information and model performance.

Specifically, the mutual information (also known as information gain) between two random variables X (i.e., inputs) and Y (i.e., predictions) is defined as:

$$I(X, Y) = H(Y) - H(Y|X) \quad (4)$$

where $H(\cdot)$ is the entropy [66] of a probability distribution. Note that entropy is a quantitative measure of *uncertainty* about a random variable. Mutual information represents the *reduction of the uncertainty* about the variable Y after observing X . Mutual information becomes zero when X and Y are independent, i.e., when $H(Y|X) = H(Y)$. In the context of the domain shift problem, high mutual information is interpreted as a high reduction of the uncertainty of the model predictions Y , given inputs X from a target domain. On the other hand, low mutual information means a low reduction of the uncertainty of Y for inputs X from the target domain, which indicates that the model does not successfully correlate X with Y .

Let $p(y|x)$ denote the probability distribution of the output y from the model f_θ given an input x . Mutual information can be expressed with $p(y|x)$:

$$\begin{aligned} H(Y) - H(Y|X) &= \sum_y p(y) \log p(y) - \sum_x p(x) \left(\sum_y p(y|x) \log p(y|x) \right) \\ &= \sum_y \left(\sum_x p(x) p(y|x) \right) \log \left(\sum_x p(x) p(y|x) \right) - \mathbb{E}_x \left(\sum_y p(y|x) \log p(y|x) \right) \\ &= \sum_y \left(\mathbb{E}_x p(y|x) \right) \log \left(\mathbb{E}_x p(y|x) \right) - \mathbb{E}_x \left(\sum_y p(y|x) \log p(y|x) \right). \end{aligned} \quad (5)$$

In practice, $p(y|x)$ can be calculated by applying the softmax function to the output logits of neural networks.

We now show that mutual information can be driven from unlabeled target validation data \mathcal{D}_T^{va} via Monte Carlo approximation:

$$\begin{aligned} H(Y) - H(Y|X) &= \sum_y \left(\mathbb{E}_x p(y|x) \right) \log \left(\mathbb{E}_x p(y|x) \right) - \mathbb{E}_x \left(\sum_y p(y|x) \log p(y|x) \right) \\ &\approx \underbrace{\sum_y \left(\frac{1}{N_T} \sum_{x \in \mathcal{D}_T^{va}} p(y|x) \right) \log \left(\frac{1}{N_T} \sum_{x \in \mathcal{D}_T^{va}} p(y|x) \right)}_{\text{Global Diversity}} - \underbrace{\frac{1}{N_T} \sum_{x \in \mathcal{D}_T^{va}} \sum_y p(y|x) \log p(y|x)}_{\text{Individual Uncertainty}}. \end{aligned} \quad (6)$$

Note that the first term is the entropy of the averaged predictions of the model and the second term is the average of the entropy for each prediction. As a high-level interpretation, the first term becomes high if the predictions are globally diverse, and the second term becomes low when the uncertainties of individual predictions are low. Therefore, mutual information is high when the predictions on the target validation data are both globally diversified and individually certain. Intuitively, an accurate classifier should provide confident class predictions,

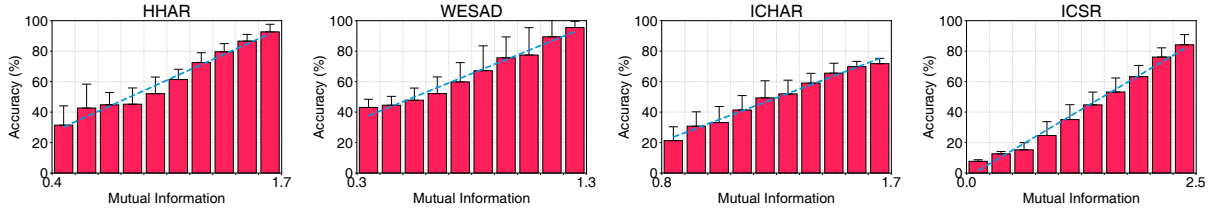


Fig. 5. Correlation between mutual information and ground-truth accuracy across four datasets.

while the output class distribution should be diverse for class-balanced inputs, and thus have high mutual information [4]. This also aligns with the *infomax principle* for neural networks, which prescribes that a classifier should maximize the mutual information between inputs and outputs [40].

With the aforementioned theoretical interpretation, we conducted a preliminary study to empirically understand the relationship between mutual information and accuracy. For this study, we used a total of 20k domain adaptation results where we logged both mutual information and ground-truth accuracy using four sensory datasets: Heterogeneity Human Activity Recognition (HHAR) [70], Wearable Stress and Affect Detection (WESAD) [65], Individual-Condition Human Activity Recognition (ICHR) [21], and Individual-Condition Speech Recognition (ICSR) [21] (details in §4). Figure 5 shows the correlation between mutual information and accuracy across four sensory datasets. As shown, mutual information has a positive correlation with performance demonstrating the feasibility of performance estimation via unlabeled target data. Still, the relationship between the accuracy and mutual information is not straightforward, and one might argue that model calibration [22] might be useful in this context. We discuss the relevance of our method to model calibration in §7.2. In the following section, we design a performance estimator by addressing the limitations of mutual information.

3.4 Performance Estimator

While mutual information provides theoretical insights for performance estimation with unlabeled data, using it directly as a proxy for performance has several practical limitations: (i) The unit of mutual information denotes the amount of information (e.g., bits with the log base 2). It does not represent the target performance metric of our interest, e.g., accuracy. Therefore, mutual information itself is not an interpretable metric to users. (ii) As illustrated in Figure 5, the relationship between mutual information and accuracy depends on the given unlabeled data from each dataset; the relationship is not a one-to-one function. Mutual information cannot decode the underlying characteristics of datasets, which makes it different among datasets. (iii) We are interested in accuracy changes for each iteration of domain adaptation. In such scenarios, previous information (e.g., previously estimated accuracy) could give a hint to performance estimation. Mutual information, however, does not utilize previous information.

To overcome the limitations of our theoretical analysis, we propose to train a domain adaptation performance estimator that gets the outputs of an adapted model on \mathcal{D}_T^{ad} and predicts the change in accuracy. The benefits of the proposed performance estimator are three-fold: (i) machine learning can capture the underlying optimal mapping from mutual information to the desired metric (e.g., accuracy) with sufficient training data and supervised objective, (ii) dataset-dependent characteristics can be learned by training dataset-specific estimators, and (iii) machine learning can leverage previous information for better prediction via sequential modeling.

Practical concerns for training such an estimator would be the cost of data collection and computation overhead. DAPPER resolves the concerns by generating lots of virtual training data using already collected source data and decoupling the training process from the user side. DAPPER utilizes the virtual training data to simulate heterogeneous domain adaptation scenarios and learn to estimate accuracy. Unlike state-of-the-art performance estimation algorithms [8, 9], the entire training process of DAPPER is conducted on the developer side. Therefore, once deployed to users, only inference is required for estimation.

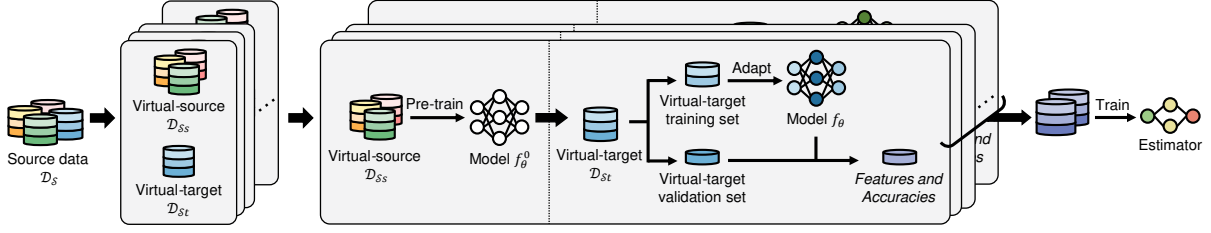


Fig. 6. An illustration of the training process of the performance estimator. We generate training data via simulation. The whole process runs on the developer side.

3.4.1 Virtual Training Data Generation. We propose a simulation-based virtual training data generation to train the estimator. Our data generation methodology is inspired by the principle of domain randomization, postulating that the disparities between source and target domains could be represented as variations within the source domain [55, 75]. Specifically, we generate training data with the following objectives. First, we create training data only from source data. Collecting target data is not required to train the estimator, which is beneficial in terms of cost and effort. Second, we train the estimator with as diverse data as possible to make the estimator learn various adaptation scenarios.

Figure 6 is an illustration of the training process. The training starts from virtual training data generation with the source data. Note that we do not have access to the target data that is ideal for training the estimator. We instead *simulate* adaptation with the source data to generate the training data for the estimator. We assume the model developer collected or obtained the source datasets \mathcal{D}_S from multiple domains, and we randomly split the source data into non-overlapping virtual-source domains \mathcal{D}_{S_s} and virtual-target domains \mathcal{D}_{S_t} . With the virtual-source and virtual-target, we generate the training data by *simulating* adaptation; the model is pre-trained with the virtual-source data, and the adaptation algorithm is applied to the pre-trained model to adapt to the virtual-target. We randomly sample the virtual-target training and validation data to simulate diverse adaptation scenarios. While adapting to the virtual-target data, features are calculated for every epoch from the unlabeled virtual-target validation data. We also log accuracy as the ground truth. This process is repeated 10,000 times to generate enough training data, which takes around one day with a server equipped with eight NVIDIA GeForce RTX 3090 GPUs. We then train the estimator with dataset-specific virtual training data. This way, we expect the estimator can be trained from myriad combinations of virtual-source and virtual-target domains, and, more importantly, learn the dataset-specific domain-invariant relationship between the features and ground-truth accuracy, which we explain in detail in the following section.

3.4.2 Training. We train the performance estimator g_ϕ of DAPPER with the data generated from the source data (Figure 6). To train an accurate and lightweight estimator, we (i) use computationally light features as the inputs for the estimator and (ii) utilize an LSTM architecture for sequential modeling.

For features, we use global diversity and individual uncertainty in Equation (6) to learn an optimal relationship between them. Given unlabeled target validation data \mathcal{D}_T^{va} , global diversity (GD) is defined as:

$$GD(\mathcal{D}_T^{va}) = \sum_y \left(\frac{1}{N_T} \sum_{x \in \mathcal{D}_T^{va}} p(y|x) \right) \log \left(\frac{1}{N_T} \sum_{x \in \mathcal{D}_T^{va}} p(y|x) \right). \quad (7)$$

Similarly, the definition of individual uncertainty (IU) is:

$$IU(\mathcal{D}_T^{va}) = \frac{1}{N_T} \sum_{x \in \mathcal{D}_T^{va}} \sum_y p(y|x) \log p(y|x). \quad (8)$$

In addition, we consider prediction distribution (PD) as a feature, which is defined as:

$$PD(\mathcal{D}_{\mathcal{T}}^{va}) = \frac{1}{N_{\mathcal{T}}} \sum_{x \in \mathcal{D}_{\mathcal{T}}^{va}} p(y|x). \quad (9)$$

Note that GD and IU are single values while PD contains K -dimensional information where K is the number of classes. As GD and IU are single values (i.e., encoded values) summed over all y s, relying on only those features might lose important data-specific characteristics that can be observed in class information y . PD reveals class-wise information (i.e., decoded values) that is beneficial in capturing data characteristics and thus improves performance estimation. Our features are extremely lightweight as opposed to using raw sensory inputs as features, e.g., one-second accelerometer input corresponding to 400×3 values under 400 Hz sampling rate.

In addition, we use the difference from the previous values as additional input features, i.e.,

$$\Delta Feat_e = Feat_e - Feat_{e-1}, \quad \text{where } Feat_e \in \mathcal{F}_e, \quad \text{and } \mathcal{F}_e = \{GD_e, IU_e, PD_e\}, \quad (10)$$

where e is the training epoch. By providing the value differences as additional features, not only can the estimator consider the current features, but it also leverages the changes in the features from the previous model. This explicit use of the absolute values (current) and the relative values (difference) of features helps the estimator better correlate the given information with the change in performance. We evaluate the effectiveness of our features via an ablative study in §5.3.

We notice that domain adaptation is a sequential process, i.e., the model is likely to be updated periodically with multiple training epochs. In such scenarios, previous information about the target model (e.g., previous features) could provide hints about the current accuracy estimation. To leverage such previous “states” of the model, we use Long Short-Term Memory (LSTM) [24] as the backbone network. LSTM is widely used for sequential modeling as it is simple and robust to the vanishing gradient problem in taking long sequences as inputs. Specifically, we use two stacks of LSTM layers with 200 hidden dimensions, followed by two fully connected layers with one ReLU activation. We use an L1 loss to minimize the difference between the predicted change in accuracy and the ground-truth change in accuracy (Equation (3)). For optimization, we adopt the Adam optimizer [31] with a learning rate of 10^{-5} . The resulting model size is affordable for mobile devices (1.9 MB) and incurs negligible computation overhead. We conduct a comprehensive analysis for the computation overhead in §5.6.

3.4.3 Inference. Given unlabeled target validation data $\mathcal{D}_{\mathcal{T}}^{va}$ and a target model $f_{\theta}^{(e)}$, DAPPER calculates GD, IU, PD, and the differences of each of them. Those features are fed into the performance estimator g_{ϕ} , and the estimator predicts the accuracy change from the initial model $f_{\theta}^{(0)}$, i.e.,

$$\Delta \hat{a}^{(e)} = g_{\phi}(GD_e, IU_e, PD_e, \Delta GD_e, \Delta IU_e, \Delta PD_e). \quad (11)$$

Note that the performance estimator g_{ϕ} is pre-trained by the developer, and only inferences are made from the estimator after deployment. We conduct a detailed computation overhead analysis of DAPPER and comparison with baselines in §5.6.

4 EXPERIMENT SETTINGS

4.1 Datasets and Preprocessing

We evaluate our method with the following four real-world mobile sensing datasets.

4.1.1 HHAR. HHAR [70] contains six activity classes of IMU data, obtained from nine users and 12 devices (eight smartphones and four smartwatches). Smartphones were kept in a tight pouch around their waist and smartwatches were on their arms. Sampling rates and sensor specifications vary by device. After removing two

users whose Samsung Galaxy Gear data were missing, we created domains with seven users and five devices. We randomly selected four users and three devices and grouped them as sources (12 domains), and used the remaining devices and users as targets (six domains).

4.1.2 WESAD. WESAD [65] is a multi-modal dataset for wearable stress and affect detection with four classes. It was collected from 15 users equipped with the same medical devices around their chest and wrist. WESAD includes diverse sensing modalities: three-axis acceleration, electrocardiogram, electrodermal activity, electromyogram, respiration, body temperature, and blood volume pulse. Following the previous study [20], we down-sampled each modality to 4 Hz, which is the lowest sampling rate. Among 15 domains, we randomly chose 10 domains as sources and the others as targets.

4.1.3 ICHAR. ICHAR [21] contains nine classes of IMU data collected from 10 pairs of different users and devices. Each participant was carrying a unique smartphone or smartwatch and there was no restriction on how to hold the device. This makes the domain adaptation task in ICHAR more challenging than HHAR which had clear instructions on how to carry devices. Among 10 individual conditions, i.e., domains, we randomly chose seven domains as sources and the remaining three as targets.

4.1.4 ICSR. ICSR [21] is a speech recognition dataset collected under the same condition as ICHAR. During data collection, each subject held one unique device in a preferred way and spoke out 14 different words with their own styles (loudness, pitch, etc.). Among 10 resulting domains, we randomly chose seven domains as sources and the remaining three as targets.

4.2 Adaptation Methods

We introduce domain adaptation methods upon which DAPPER estimates performance. We consider a common multi-source DA scenario: for each target domain, we pre-trained a base model with the source domains, and the base model was adapted to the target domain with a DA algorithm. We experimented with four common adaptation methods as follows.

4.2.1 Fine-tuning. Fine-tuning utilizes labeled data from a target domain for adaptation. We used a well-known fine-tuning method where a pre-trained model updated its parameters with a few labeled data from the target domain. Specifically, we randomly chose a target domain first and selected 1~50 random labeled data from the target with which we performed adaptation. Here, we used a separate unlabeled dataset that was not used for adaptation to estimate performance. This process was repeated 50 times for each dataset, and we reported the average.

4.2.2 DANN. Domain-Adversarial Training of Neural Network (DANN) [18] is a widely used unsupervised domain adaptation (UDA) algorithm that adapts to a target domain with labeled source and unlabeled target data. Similar to fine-tuning, we chose a target domain randomly, selected 50~500 random unlabeled data from the target, performed adaptation, and estimated the performance with separate unlabeled data, which was repeated 50 times for each dataset. We provided more samples in DANN than fine-tuning, considering that unlabeled data are easier to collect.

4.2.3 CDAN. Conditional Domain Adversarial Network (CDAN) [42] improves the adversarial training of DANN by jointly utilizing feature representations and classifier predictions with labeled source and unlabeled target data. We adopted the same experiment setting as DANN, where we randomly chose a target domain, selected 50~500 random unlabeled data from the target, performed adaptation, and estimated performance with separate unlabeled data, which was repeated 50 times for each dataset.

4.2.4 SHOT. Source HypOthesis Transfer (SHOT) [38] is a recent UDA algorithm that does not require source data in adaptation. We adopt SHOT as it has a practical setting since the source data might not be available to the target users due to privacy and intellectual property concerns. Same as DANN, we chose a target domain randomly, selected 50~500 random unlabeled data from the target, performed adaptation, and estimated performance with a separate unlabeled data, which was repeated 50 times for each dataset.

4.2.5 Implementation. For the backbone networks, we adopted 1D convolutional neural networks (CNN) [32] followed by fully-connected layers, which are commonly used in mobile sensing [6, 21, 30, 63]. We used Rectified Linear Unit (ReLU) [50] for activation function and Batch Normalization layer [26] after each CNN layer for regularization and fast convergence. We used L2-regularization [51] as a regularization technique. We trained the model with Adam optimizer [31]. We used a fixed learning rate of 0.001. We used a conventional standardization for preprocessing the datasets. We used the PyTorch framework [54] for implementation and trained the model with NVIDIA TITAN Xp GPUs. We ran each training for 100 epochs in both adaptation scenarios.

4.3 Baselines

We compare DAPPER with five baselines, including the state-of-the-art performance estimation algorithms [8, 9]. We note that in the context of mobile sensing, no previous work has proposed performance estimation utilizing unlabeled data. Most studies trained until a fixed number of epochs [15, 25, 94], while several studies selected the best models on the labeled target validation data [21, 80].

4.3.1 TgtLabel. TgtLabel computes accuracy with labeled target validation data. This performance validation is often used in domain adaptation research [10, 13, 21, 29, 41, 73, 74, 79, 80, 82]. We use TgtLabel as the ground-truth performance.

4.3.2 FixedEpoch. Training until a fixed number of epochs (FixedEpoch) is a widely used model selection method [5, 11, 15, 17, 25, 34, 38, 43, 67, 71, 78, 81, 94]. Note that as FixedEpoch always chooses the last epoch's model without any validation data, it does not have performance validation.

4.3.3 SrcLabel. Validating performance with labeled source data (SrcLabel) is also a popular method [19, 36, 37, 64] as source data are more likely labeled with a larger amount, compared with target data. SrcLabel computes accuracy from a hold-out labeled source data to estimate the target performance. In order not to affect the performance of the pre-trained model, the hold-out source data for validation were never used in the training process for other baselines as well.

4.3.4 SoftmaxScore. Using the confidence score from the last softmax layer [14] might be the simplest approach utilizing target unlabeled data for performance estimation. This method is hence commonly adopted as a baseline in related studies [8, 9]. Specifically, it estimates the accuracy of the target domain by averaging the softmax confidence scores computed from unlabeled target validation data.

4.3.5 DIR. A recent study utilizes Domain-Invariant Representations (DIR) for estimating generalization under domain shifts [9]. This study learns an accurate “check” model with domain-invariant representations as a proxy for estimating performance. Specifically, it pre-trains a check model with both labeled source data and unlabeled target data via DANN [18]. After each adaptation, to estimate performance, it updates the check model that maximizes disagreement with the adapted model. An estimated proxy risk is the maximum disagreement between the check model and the adapted model. Therefore, this algorithm requires updating the check model for every update of the adapted model. We used this proxy risk to calculate an estimated accuracy. For the check model, we used the same model as the adaptation model and used the unlabeled target validation data to train. We referred to the official implementation and the check model was updated for 20 epochs.

Table 1. Comparison of the baselines and DAPPER.

	Src data	Tgt data	Extra Training	Validation
TgtLabel (oracle)	No	Labeled	No	Yes
FixedEpoch	No	No	No	No
SrcLabel	Labeled	No	No	Yes
SoftmaxScore	No	Unlabeled	No	Yes
DIR	Labeled	Unlabeled	Yes	Yes
EnsRM	Labeled	Unlabeled	Yes	Yes
DAPPER (ours)	No	Unlabeled	No	Yes

4.3.6 *EnsRM*. Ensemble via Representation Matching (EnsRM) [8] is the state-of-the-art performance estimation algorithm that utilizes ensemble models and self-training to estimate target accuracy. Similar to DIR [9], EnsRM assumes that an accurate check model is pre-trained via DANN [18]. For performance estimation, EnsRM further fine-tunes the check model to the unlabeled target validation data for multiple epochs and stores the models after each epoch, which jointly become an ensemble of check models. These ensemble models are utilized to identify misclassified target points where the ensemble’s class predictions disagree with those of the adapted model, and the misclassified target points are added to training samples. This procedure is repeated for multiple rounds with self-training, and the disagreement ratio from the last round becomes an estimated error of the model. Following the authors’ implementation, we updated the check model for five epochs and five rounds.

4.3.7 *Comparison with Baselines*. Table 1 compares the baselines and DAPPER. “Src data” means whether it requires hold-out source data during performance estimation while “Tgt data” means whether it requires target data during estimation. “Labeled” means the requirement of labeled data, “Unlabeled” means the requirement of unlabeled data, and “No” means no requirement of data for that category. “Extra Training” is whether it requires *additional* training for every updated model. Unlike DIR and EnsRM, DAPPER requires training only once before deployment, without additional training for every adapted model. “Validation” is whether the method includes performance validation. FixedEpoch does not have performance validation.

4.4 Performance Estimation Setup

For performance validation (or estimation), we randomly selected 500 samples throughout experiments unless specified. The validation samples were consistent for each dataset to ensure a fair evaluation. For instance, all baselines that require target data used the same 500 samples from a target for validation, but those samples were labeled for TgtLabel and unlabeled for DIR, EnsRM, and DAPPER. SrcLabel used 500 samples from the hold-out source data. For each dataset and adaptation method pair, the result was averaged over 50 training episodes as explained in §4.2.

We use similarity calculated from an L1 distance as a metric to compare with baselines. L1 distance is a metric that calculates the distance between two N dimensional vectors. We regard the true accuracy (TgtLabel) as the oracle and calculate the L1 distance between the ground truth and the estimations. We subtract the averaged L1 distance from 1 to compute similarity. Specifically, we define similarity as follows:

$$Similarity = 1 - \frac{1}{|E|} \sum_{e \in E} \left| \Delta a^{(e)} - \Delta \hat{a}^{(e)} \right|, \quad (12)$$

which ranges from 0 to 1 (0~100%). Therefore, the higher the similarity, the better the estimation.

Table 2. Average similarities (%) of the baselines and DAPPER under four domain adaptation methods (fine-tuning, DANN, CDAN, and SHOT), across four datasets (HHAR, WESAD, ICHAR, and ICSR). D1~Dn represent target domains. Bold type numbers indicate those of the highest average similarity.

Method	HHAR						WESAD					ICHR			ICSR			Avg.
	D1	D2	D3	D4	D5	D6	D1	D2	D3	D4	D5	D1	D2	D3	D1	D2	D3	
<i>Adaptation algorithm: Fine-tuning</i>																		
TgtLabel	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
FixedEpoch	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SrcLabel	78.5	72.3	77.6	71.4	83.5	71.7	73.0	60.2	78.9	82.8	65.7	73.9	71.5	67.8	84.2	87.0	81.8	75.4
SoftmaxScore	85.2	89.5	88.8	88.7	91.9	87.4	82.7	86.7	82.1	74.9	90.9	82.4	75.1	76.3	70.2	78.9	85.4	83.4
DIR	70.1	69.6	62.4	64.7	70.4	66.8	68.7	28.7	52.0	45.4	57.3	64.0	61.5	64.0	71.7	69.5	71.6	62.3
EnsRM	44.2	41.3	39.8	38.3	39.8	43.1	59.4	48.5	47.0	57.8	47.8	53.6	59.4	55.6	62.8	44.8	40.8	48.5
DAPPER (ours)	94.3	95.9	93.8	97.0	96.3	95.8	94.5	94.9	90.6	77.0	89.1	94.5	95.9	96.2	89.4	93.1	89.8	92.8
<i>Adaptation algorithm: DANN</i>																		
TgtLabel	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
FixedEpoch	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SrcLabel	85.7	84.3	92.1	91.1	90.7	77.9	70.4	64.0	73.2	41.3	93.6	66.0	64.1	60.6	64.7	93.3	84.0	76.3
SoftmaxScore	89.2	84.1	94.8	93.4	96.1	85.1	65.4	49.8	83.7	42.6	86.3	76.1	76.1	73.6	75.8	95.4	88.3	79.8
DIR	82.9	87.4	54.1	66.4	63.6	76.9	93.9	82.9	70.8	81.2	70.2	92.8	83.1	87.7	74.5	63.9	70.6	76.6
EnsRM	42.6	44.3	35.6	37.3	36.3	49.4	87.8	84.8	61.8	85.4	54.9	66.7	67.3	74.2	56.4	27.8	36.9	55.9
DAPPER (ours)	93.4	94.8	95.1	89.5	93.5	96.7	92.8	92.6	84.9	78.1	90.0	95.7	89.6	93.4	95.2	91.7	97.5	92.0
<i>Adaptation algorithm: CDAN</i>																		
TgtLabel	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
FixedEpoch	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SrcLabel	86.8	86.8	92.8	90.2	87.3	81.1	86.6	68.1	95.0	57.9	94.9	77.3	71.3	75.8	61.3	93.8	86.8	82.0
SoftmaxScore	79.6	78.1	85.2	81.9	82.3	74.2	73.6	56.1	88.1	46.5	84.9	55.2	50.8	55.4	53.7	84.3	76.6	71.0
DIR	76.4	81.4	62.3	79.3	71.0	77.3	72.1	67.9	46.9	69.8	58.0	92.0	74.6	79.1	74.9	63.5	67.2	71.4
EnsRM	41.1	42.8	35.8	39.7	41.3	47.4	77.2	91.5	46.4	69.9	52.1	64.0	68.5	64.4	60.4	25.3	35.6	53.1
DAPPER (ours)	92.3	97.4	96.1	89.1	96.1	96.5	94.9	90.8	89.0	90.8	91.2	95.3	94.7	90.9	96.6	89.1	95.3	93.3
<i>Adaptation algorithm: SHOT</i>																		
TgtLabel	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
FixedEpoch	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SrcLabel	89.5	82.2	75.9	80.7	91.6	86.3	95.8	83.3	91.1	76.5	92.3	92.9	79.0	90.0	92.9	79.0	90.0	86.4
SoftmaxScore	75.1	82.9	91.4	74.9	74.0	76.8	47.7	33.0	50.4	34.6	65.9	54.5	89.5	74.7	54.5	89.5	74.7	67.3
DIR	75.5	79.4	58.3	67.6	78.9	72.9	79.2	77.5	72.5	75.5	68.0	79.9	65.0	75.5	79.9	65.0	75.5	73.3
EnsRM	48.6	41.6	28.6	43.1	50.1	47.8	70.0	82.5	72.8	86.3	60.8	71.9	29.6	47.3	71.9	29.6	47.3	54.7
DAPPER (ours)	95.6	90.2	96.9	90.7	91.2	96.2	91.9	92.3	92.4	91.4	90.4	96.2	96.2	88.6	96.2	96.2	88.6	93.0

5 EVALUATION

5.1 Overall Results

Table 2 is the overall result that shows how effective DAPPER is compared with the baselines. The best values among different methods are highlighted in bold, except for TgtLabel. The similarity in TgtLabel is 100, as it is the ground truth. Measuring similarity is not available in FixedEpoch, and thus marked as “N/A.” For each adaptation method, we specify similarities per each domain (D1~Dn) and the average of them.

While both TgtLabel and SrcLabel use labeled data for performance validation, the estimation by SrcLabel is often inaccurate as it assumes that the performance is generalizable via hold-out source data without any information from the target. Using the softmax score for performance estimation (SoftmaxScore) is not reliable due to the well-known incorrect calibration issue [22]. We also observe that the performance estimation by DIR is far below DAPPER. This is due to its assumption that an accurate check model is available [9]. We found that this assumption often does not hold in mobile sensing, where a high-quality check model is not always assured. Note that DIR uses DANN for training the check model for each target domain. Unlike traditional vision domains

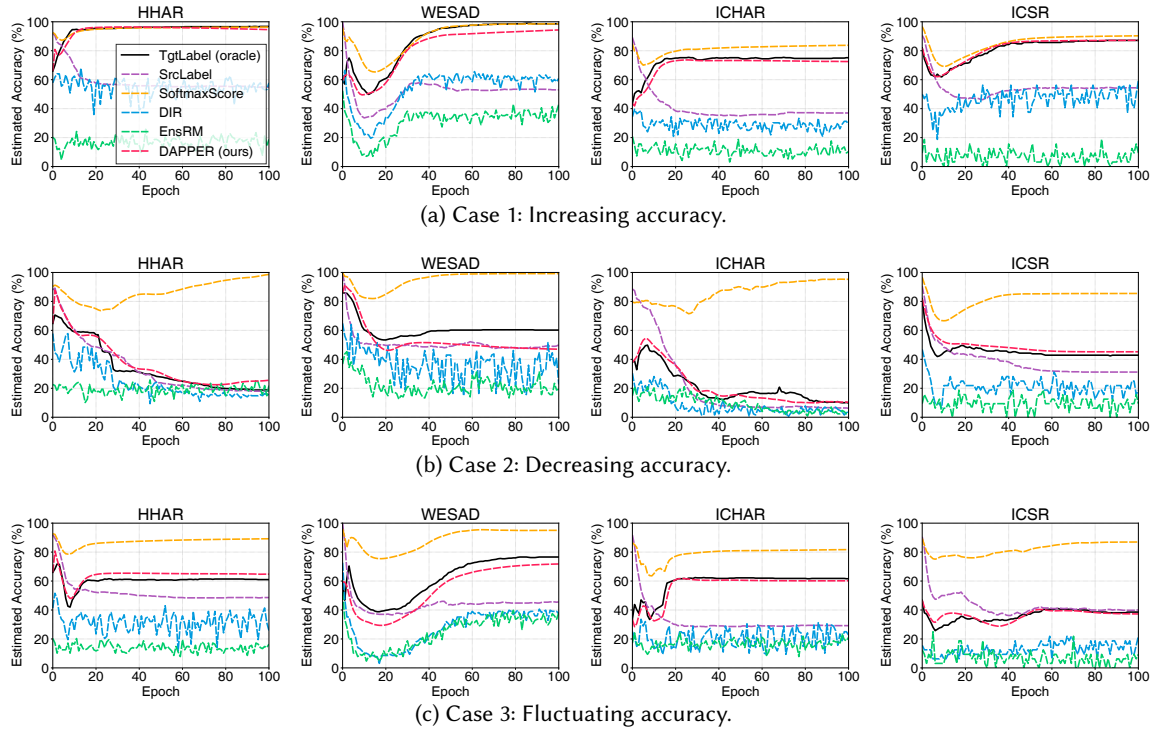


Fig. 7. Qualitative analysis with three categories: (a) increasing accuracy, (b) decreasing accuracy, and (c) fluctuating accuracy.

with thousands of target data and well-defined domains, we discovered that the check model’s errors were often high in mobile sensing data, which aligns with our observation of performance dynamics in Figure 2. Note that EnsRM performs generally worse than DIR in our experiments. We attribute this to its assumptions that (i) an ensemble of check models has small errors on points that the target model is correct and (ii) the ensemble models have a proper covariance with the target model, i.e., “the covariance should not be negative or too positive”, as noted by the authors [8]. These assumptions could fail when there is no guarantee for the performance of the ensemble models or the correlation between the ensemble models and the target model, e.g., due to high variability of performance in heterogeneous mobile sensing.

Overall, DAPPER outperformed the baselines consistently across four datasets. On average, DAPPER achieved 12.8% similarity improvement over SrcLabel, 17.5% improvement over SoftmaxScore, 21.9% improvement over DIR, and 39.8% improvement over the state-of-the-art method, EnsRM. This highlights the effectiveness of our proposed features coupled with our training strategies. We believe the results indicate that DAPPER is the most appropriate performance estimation method for heterogeneous mobile sensing. We however found some challenging cases where DAPPER showed low similarities (e.g., WESAD D4 in Fine-tuning). These cases have unique patterns that are not well generalized with our virtual training data, which means that domain randomization might not be very effective in those cases. There might be several ways to overcome this, such as generating more virtual training data and increasing the number of source data.

5.2 Qualitative Analysis

We conducted a qualitative analysis to understand further the behaviors of the baselines of DAPPER. Specifically, we investigated how similar the estimated performance is to the true performance. Figure 7 compares different

Table 3. The ablation study result of DAPPER. Average similarities (%) for four datasets under the four adaptation settings are reported. The numbers are averaged across target domains. Dataset names are abbreviated for better visibility (HHA:HHAR, WES: WESAD, ICH: ICHAR, and ICS:ICSR). Bold type numbers indicate those of the highest average similarity.

Method	Fine-tuning				DANN				CDAN				SHOT				Avg
	HHA	WES	ICH	ICS	HHA	WES	ICH	ICS	HHA	WES	ICH	ICS	HHA	WES	ICH	ICS	
GD	86.2	77.7	91.5	91.1	94.4	94.3	92.4	95.1	95.4	91.3	93.5	93.4	93.3	84.7	89.2	87.6	90.7
GD+IU	91.0	81.0	94.4	89.7	94.8	94.4	93.0	95.7	95.1	93.1	92.2	93.5	90.4	84.5	89.4	88.5	91.3
GD+IU+PD	95.5	89.2	95.5	90.8	93.9	87.7	92.9	94.8	94.6	91.3	93.6	93.7	93.5	91.7	88.8	93.6	92.6
Feat	77.2	77.4	54.5	73.8	40.5	60.1	78.5	71.6	58.6	58.9	83.3	72.6	73.1	70.6	87.2	75.2	69.6
Δ Feat	77.0	63.7	80.0	85.4	76.5	79.2	90.6	95.5	78.4	91.1	85.5	95.6	72.4	67.1	80.3	87.0	81.6
Feat+ Δ Feat	95.5	89.2	95.5	90.8	93.9	87.7	92.9	94.8	94.6	91.3	93.6	93.7	93.5	91.7	88.8	93.6	92.6
MLP	85.8	82.1	92.1	88.7	94.0	93.2	92.2	94.7	94.5	90.4	93.1	94.4	92.2	88.9	90.3	92.6	91.2
LSTM	95.5	89.2	95.5	90.8	93.9	87.7	92.9	94.8	94.6	91.3	93.6	93.7	93.5	91.7	88.8	93.6	92.6

methods (SrcLabel, SoftmaxScore, DIR, EnsRM and DAPPER) to TgtLabel (oracle) under three different performance patterns: (i) increasing accuracy, (ii) decreasing accuracy, and (iii) fluctuating accuracy. While the other baselines directly predict the accuracy, DAPPER predicts relative accuracy change. We added the true accuracy at epoch zero to the estimated accuracy by DAPPER for better visualization.

The estimated performance by SrcLabel generally decreased as training proceeded, regardless of whether the actual performance improved or not. This means validation with hold-out source data could not be applied to unseen targets due to domain shifts. Thus, the estimated performance decreased as the model adapted to the target data. SoftmaxScore often showed high estimated accuracy in the decreasing case as the model became overconfident for misclassifications due to the incorrect calibration [22]. The estimations by DIR and EnsRM were noisy as they require updating the check model(s) for every adaptation epoch. Since the learned representations change as the training continues, the estimations fluctuate accordingly. In addition, the estimated accuracy by DIR and EnsRM was generally lower than the other algorithms, indicating that the disagreement between the target model and the check model(s) was higher than it should be, probably due to the low accuracy of the check model(s).

DAPPER predicted accuracy better than other baselines in general. As DAPPER relies on the target data to estimate the performance, it does not have the domain shift issue of SrcLabel mentioned above. Moreover, DAPPER uses a pre-trained estimator for performance estimation and does not incur noisy estimations as DIR and EnsRM.

5.3 Ablation Study

We conducted ablative experiments on DAPPER to analyze the effectiveness of our design choices in training DAPPER (§3.4.2). Table 3 is the result. We first inspected the effect of the feature modalities used in training DAPPER. We sequentially added features, global diversity (GD), individual uncertainty (IU), and prediction distribution (PD), and compared the average similarity values. As our suggested features were added, the average similarity became higher in general. This shows the effectiveness of our features for training the estimator. Second, we compared the effectiveness of using both the original features (*Feat*) and the difference of them (Δ *Feat*) for performance estimation. We observe that using both the original and the difference features significantly improved the similarity further than using only one of them. This attests the importance of considering both the original value of the features and the difference of them from the previous epoch for accurate performance estimation. Finally, we compared our LSTM architecture with a simple multi-layer perceptron (MLP) architecture to understand the effectiveness of LSTM. Specifically, we used a two-layer MLP with a hidden dimension of 20

Table 4. Average similarities (%) of the baselines and DAPPER under two cross-dataset adaptation scenarios (from ICHAR to HHAR and from HHAR to ICHAR). Bold type numbers indicate those of the highest average similarity.

Method	ICHAR → HHAR						HHAR → ICHAR			Avg.
	D1	D2	D3	D4	D5	D6	D1	D2	D3	
<i>Adaptation algorithm: Fine-tuning</i>										
TgtLabel	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
FixedEpoch	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SrcLabel	73.2	75.3	59.4	51.3	66.2	62.4	69.2	63.1	74.4	66.1
SoftmaxScore	81.1	84.7	91.3	89.7	86.1	89.8	80.0	86.2	85.4	86.0
DIR	52.5	58.7	55.7	50.3	50.1	53.4	60.5	31.8	58.2	52.4
EnsRM	50.8	53.8	41.2	39.6	47.8	47.5	58.1	45.7	53.0	48.6
DAPPER (ours)	91.7	95.3	89.5	92.6	87.4	90.5	95.0	65.6	86.3	88.2
<i>Adaptation algorithm: DANN</i>										
TgtLabel	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
FixedEpoch	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SrcLabel	83.2	77.4	64.0	65.4	59.2	52.6	48.8	43.0	61.3	61.6
SoftmaxScore	87.6	81.6	67.2	72.7	69.1	62.7	47.1	51.7	67.4	67.5
DIR	69.4	69.4	90.0	77.7	83.1	90.7	73.2	77.5	86.2	79.7
EnsRM	52.4	54.9	67.9	64.9	75.3	79.8	85.6	82.0	79.6	71.4
DAPPER (ours)	73.1	91.5	75.7	59.6	87.9	85.6	93.6	78.5	95.4	82.3
<i>Adaptation algorithm: CDAN</i>										
TgtLabel	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
FixedEpoch	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SrcLabel	83.5	79.0	68.3	64.2	56.2	65.4	50.2	56.1	66.0	65.4
SoftmaxScore	71.1	68.1	55.3	53.9	46.8	55.1	36.8	45.0	55.8	54.2
DIR	67.1	67.2	78.5	90.0	87.8	85.6	76.9	74.6	91.0	79.8
EnsRM	53.0	56.1	68.5	67.9	84.1	69.3	95.4	79.7	75.7	72.2
DAPPER (ours)	74.5	91.2	87.3	66.2	87.4	89.2	95.3	67.9	97.3	84.0
<i>Adaptation algorithm: SHOT</i>										
TgtLabel	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
FixedEpoch	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SrcLabel	91.7	74.5	59.4	59.8	60.5	47.7	49.1	43.8	64.7	61.2
SoftmaxScore	91.8	79.0	59.7	66.2	67.1	58.0	35.9	32.6	54.7	60.6
DIR	49.2	66.8	86.1	85.9	91.2	91.2	74.9	80.1	91.1	79.6
EnsRM	39.3	55.1	71.7	69.2	72.6	86.4	94.9	85.9	73.6	72.1
DAPPER (ours)	72.8	93.2	71.2	78.1	85.2	87.6	95.3	73.1	96.6	83.7

and a ReLU activation, which is the same architecture used in our estimator, excluding the LSTM part. We found that leveraging LSTM shows more accurate predictions than MLP, demonstrating the effectiveness of utilizing sequential modeling in performance estimation.

5.4 Cross-dataset Adaptation

Cross-dataset adaptation is challenging but beneficial as it utilizes well-established existing data for a new related task. It has been investigated in mobile sensing scenarios such as human activity recognition [58]. We evaluate DAPPER under this challenging setting to further understand the estimated accuracy and possible limitations of DAPPER under such a setting. Specifically, we selected the five common classes between HHAR and ICHAR datasets (sit, stand, walk, stairs up, and stairs down). We consider two scenarios: adaptation from HHAR to ICHAR and adaptation from ICHAR to HHAR. We use the same source and target domains used in the previous experiment in Table 2.

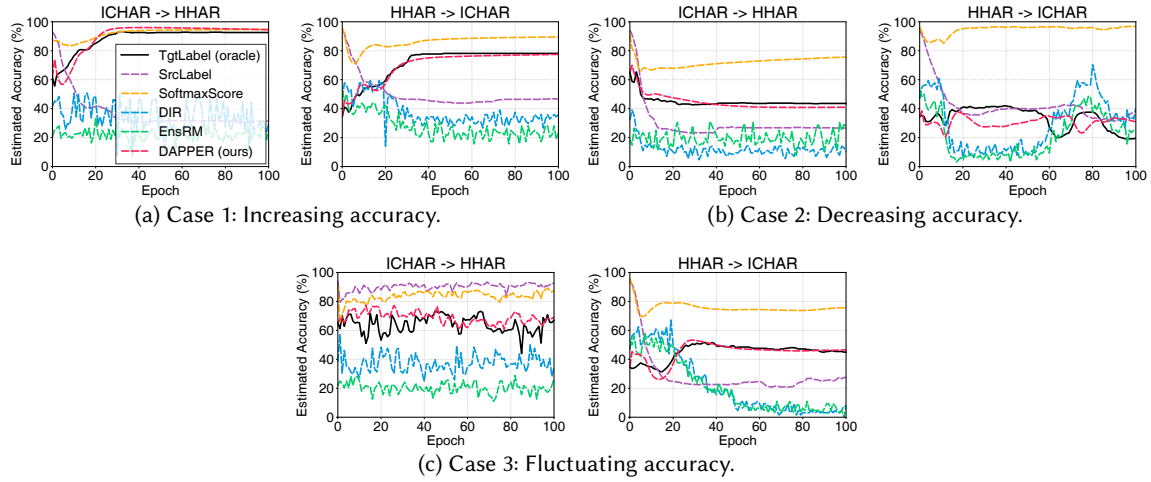


Fig. 8. Qualitative analysis for two cross-dataset adaptation scenarios (from ICHAR to HHAR and from HHAR to ICHAR) with three categories: (a) increasing accuracy, (b) decreasing accuracy, and (c) fluctuating accuracy.

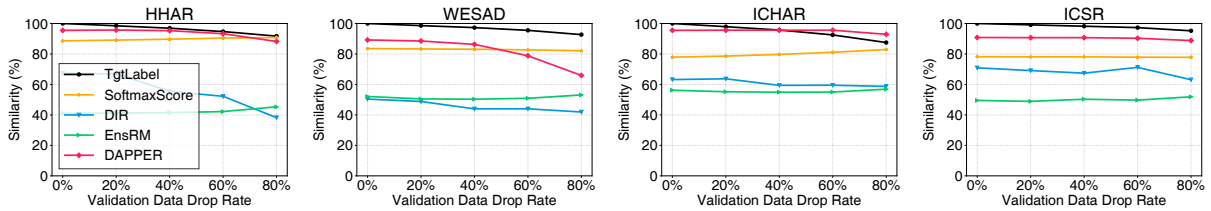


Fig. 9. Result with imbalanced target validation data.

Table 4 shows the result and Figure 8 illustrates the qualitative analysis. The result shows that the overall similarities decreased compared with the within-dataset experiments. This might be attributed to more significant domain gaps, demonstrating the fundamental difficulty of the task. Still, the trend is similar to the previous within-dataset experiments, and DAPPER overall shows better performances than the other baselines.

5.5 Impact of Validation Data Distribution

For the other experiments, we used 500 randomly sampled validation data, which naturally follows the target data distribution. In practice, however, it is possible for target validation data to be imbalanced. Taking human activity recognition for instance, one could stay home during sick leaves, which would contain more static activities (sleeping, sitting, etc.) than dynamic activities. In this case, performance estimation would be conducted under skewed validation data. We investigated how resilient DAPPER is on such an imbalanced distribution of target validation data. Specifically, for each dataset, we randomly selected half of the classes and dropped 20%, 40%, 60%, and 80% out of the 500 samples for the selected classes. The number of samples used for validation decreased accordingly.

Figure 9 shows the results with different distributions of the validation data in the fine-tuning scenario (similar trends were found in other adaptation algorithms). Here, we set the TgtLabel with balanced data (0%) as the oracle and calculated the average similarity with respect to the oracle. Note that TgtLabel is also affected by the target

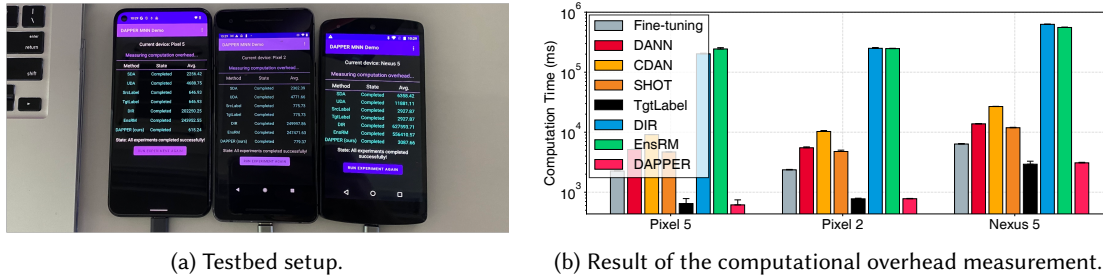


Fig. 10. On-device computational overhead measurement with three off-the-shelf smartphones (Pixel 5, Pixel 2, and Nexus 5). The y-axis in the result (b) is in the logarithm scale.

data distribution; even if we use the labeled target data, the estimation becomes gradually inaccurate as data distribution becomes imbalanced. To some extent of imbalance, DAPPER showed relatively stable performance. Interestingly, we found that the average similarity degradation in DAPPER from 0% to 80% drop (5.9%) was lower than that of the oracle, TgtLabel (7.6%). We conjecture that this robustness was attributed to the diverse simulated adaptation data (10k) used to train DAPPER. We note that the similarity in DIR deteriorated more with higher drop rates than that of EnsRM. We believe EnsRM is more robust to the dataset distribution owing to its ensemble models.

5.6 On-Device Computational Overhead

On-device deep learning has been increasingly important owing to its advantages over central training. A variety of private information, such as location, health states, emotions, and identifiable information, could be inferred by leaking personal sensing data [33, 77], which could be protected via on-device training. Additionally, on-device training can save communication bandwidth and cloud management cost [12, 92]. We inspect the computational overhead of DAPPER and the baselines in terms of on-device training and inference.

We implemented the adaptation methods, the performance validation baselines, and DAPPER with the Mobile Neural Network (MNN) framework by Alibaba [28]. MNN supports both training and inference and achieves state-of-the-art performance with optimized assembly code. We included adaptation methods (Fine-tuning, DANN, CDAN, and SHOT) to compare the adaption time of those algorithms with the computation time of performance validation. FixedEpoch was excluded as it does not have additional computation. SrcLabel and SoftmaxScore were excluded as they had almost the same computation as TgtLabel. We used three commodity smartphones (Google Pixel 5, Google Pixel 2, and LG Nexus 5) for this experiment. We reproduced the experiment with the HHAR dataset under those mobile devices to measure the computational overhead. Both training and inference were made through the CPU with four threads. We set the number of source and target data used for each method as 500 to isolate the impact of the number of data from the algorithmic overhead. The overall trend was kept similar with different numbers of data. For each case, we measured the computational overhead 20 times and reported the average and standard deviation.

Figure 10 shows that the computational overhead of the state-of-the-art baselines (DIR and EnsRM) was much larger than that of DAPPER (180~396× larger) and even larger than the adaptation algorithms. Their computational bottleneck was updating the check model(s) for every adapted model. This suggests that regardless of the estimation accuracy, requiring extra training by the user could harm the user experience with heavy computations. TgtLabel and DAPPER showed almost identical computation time, as the primary computational overhead is from forwarding the validation samples to the target model. The computational difference between TgtLabel and DAPPER is from DAPPER requiring (i) the calculation of the features (GD, IU, and PD) and (ii)

additional forward-pass in the estimation network. The calculation of the features took only 0.6~2.1 ms and additional forwarding took only 0.7~3.6 ms, depending on the device. Compared with the forward-pass in the target model (613~3081 ms), additional overhead by DAPPER was negligible, which took only around 0.2% of the total computation in estimation. In summary, not only DAPPER accurately estimates the performance, but it also generates affordable computation overhead similar to TgtLabel.

6 RELATED WORK

We summarize existing studies that utilize unlabeled data to validate performance. Several schemes utilize unlabeled data for performance estimation but require assumptions that might not hold in mobile sensing scenarios. Limberg et al. [39] estimated the accuracy of four shallow machine learning classifiers, such as k-Nearest Neighbors, with a semi-supervised approach for robot object recognition tasks. However, it was experimentally shown that it worked well only when the training and test conditions were the same, which might not be applicable in heterogeneous mobile sensing. Platonis et al. [56] estimated accuracy with unlabeled data by the agreement of multiple boolean functions. Their subsequent work [57] improved estimation accuracy given logical constraints among classes. However, both studies were limited to binary classifiers and might be difficult to extend to general multi-class classifiers. Steinhardt et al. [69] estimated unsupervised risk (error) with unlabeled data by assuming three conditionally independent views of a sample are available, which limits the practicality of its approach.

Recent studies utilized deep learning to estimate the performance under domain shifts [8, 9, 86]. Deep Embedded Validation (DEV) [86] proposed a model selection method by estimating the ratio between the source and the target density (i.e., density ratio). Specifically, DEV predicts the target risk by training a domain discriminator from the unlabeled target data and estimating the density ratio. A recent work utilizing Domain-Invariant Representations (DIR) [9] estimates performance under domain shift. DIR estimates the performance of a target model by comparing it with a “check model.” Ensemble via Representation Matching (EnsRM) [8] is the state-of-the-art performance estimation algorithm. Similar to DIR, EnsRM pre-trains a check model. EnsRM further improves the estimation accuracy over DIR using an ensemble of check models. All these algorithms assume that their domain discriminator and check models are accurate, which might not hold in mobile sensing due to a variety of factors affecting performance as explained in §2. Note that they require heavy computations after deployment for training the domain discriminator and the check model(s). As DEV focuses on the model selection and does not provide a comparable performance metric, we compared DAPPER with DIR and EnsRM in our experiments.

7 DISCUSSION

7.1 Implications to Mobile Sensing

Previous research on the domain shift problem in mobile sensing [6, 15, 21, 30, 80, 94] has focused on improving the accuracy of DA algorithms. Our work disclosed the pitfall of DA in mobile sensing caused by various factors that often lead to performance degradation or unnecessary computational overhead. This phenomenon motivated our research of performance estimation by leveraging only unlabeled data that are naturally obtained in mobile sensing applications. This performance estimation framework would be beneficial to design quantitatively efficient DA systems in mobile sensing by selecting better models while reducing computational costs.

Furthermore, this study opens up a new direction for human-AI interaction in mobile sensing DA. Under the existing DA frameworks, users have to blindly use mobile sensing applications without knowing how well it operates. Demonstrating the model’s performance can be seen as a way of interaction between AI and users, and it is known to improve user trust in AI systems [35, 85, 87]. This would facilitate involving users in the loop of

personalizing AI, making better models for themselves. We believe that providing performance is key to enabling a human-AI interaction toward overcoming the domain shift problem in mobile sensing.

7.2 Model Calibration

In order to treat uncertain samples, model calibration [22] aims to make a model produce probability estimates representative of the true correctness likelihood for a given sample. Our objective differs from model calibration in that we estimate the accuracy of the adapted model under domain shift given a number of unlabeled data. One might argue that model calibration could be utilized for our scenario, but we instead take a learning-based approach due to the following limitations of model calibration. First, recent studies found that model calibration significantly deteriorates under domain shift [23, 53]. Second, model calibration methods often generate undesirable accuracy degradation [22] or computational overhead [16]. Third, calibrated models still have errors while uncalibrated models show a reasonable correlation with the true likelihood [22]. Lastly, while designing our method, we conducted a preliminary experiment with a widely-used model calibration method, temperature scaling [22], to understand its effectiveness in our scenario. We found that it was not very effective, and the case without calibration (i.e., when the temperature was equal to one) on average showed the best correlation with accuracy.

7.3 Limitations and Extensions

7.3.1 Training and Validation Data Split. In our experiments, we used 500 validation data for performance estimation. While we focused on performance estimation with unlabeled data, the ratio between training and validation data could be an important decision, especially for real deployments. For instance, given 1,000 unlabeled data collected from a target user, the ratio between the number of training and validation samples could affect both the model's accuracy and the estimation accuracy. Furthermore, when only a few labeled data instances are available, utilizing some of them along with unlabeled data could improve performance estimation. Note that there is a trade-off in selecting the ratio between the training data and validation data; a larger number of training data would result in better adaptation, while a larger number of validation data would provide better estimation, as shown in §5.5. This is a similar discussion to the efficient training/validation data split in machine learning [61, 84], and their valuable findings could provide insights into our scenario.

7.3.2 Handling Continuous Data Stream. We focused on predicting the performance within a single adaptation episode given a set of train/validation data. However, for the continuous sensor data streams from the target users, it might involve multiple adaptations as data arrive. We argue that performance validation becomes even more important in such scenarios due to continuous domain shifts that might cause performance drift over time. We believe our insights and findings could be applicable to the continuous data stream case. However, issues that need to be addressed remain, such as how to manage distribution shifts as time passes, how much data can be kept, what data should be discarded, etc. We leave this direction as future work.

8 CONCLUSION

We investigated the uncertainty of domain adaptation performance in mobile sensing and the importance of performance validation. As a solution, we proposed DAPPER, a performance estimation framework that utilizes only unlabeled data from a target domain. As unlabeled data are collected naturally in mobile sensing, it does not require additional manual user effort for labeling. Our evaluation with four real-world sensing datasets showed that DAPPER is the most accurate performance estimator compared to the baselines. In addition, our on-device computation analysis showed that DAPPER reduced the computational overhead up to 396× compared with the existing training-based estimation. We believe our study discloses an important problem of performance estimation in mobile sensing domain adaptation and proposes a promising solution that takes a meaningful stride toward overcoming the performance dynamics in mobile sensing.

ACKNOWLEDGMENTS

This work was supported in part by the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2022-0-00495) and the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-00064).

REFERENCES

- [1] Philip Bachman, R Devon Hjelm, and William Buchwalter. 2019. Learning Representations by Maximizing Mutual Information Across Views. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2019/file/ddf354219aac374f1d40b7e760ee5bb7-Paper.pdf>
- [2] Sourav Bhattacharya, Petteri Nurmi, Nils Hammerla, and Thomas Plötz. 2014. Using unlabeled data in a sparse-coding framework for human activity recognition. *Pervasive and Mobile Computing* 15 (2014), 242–262. <https://doi.org/10.1016/j.pmcj.2014.05.006> Special Issue on Information Management in Mobile Applications Special Issue on Data Mining in Pervasive Environments.
- [3] Xiao Bo, Christian Poellabauer, Megan K. O'Brien, Chaithanya Krishna Mummidisetty, and Arun Jayaraman. 2018. Detecting Label Errors in Crowd-Sourced Smartphone Sensor Data. In *2018 International Workshop on Social Sensing (SocialSens)*. 20–25. <https://doi.org/10.1109/SocialSens.2018.00017>
- [4] John Bridle, Anthony Heading, and David MacKay. 1991. Unsupervised Classifiers, Mutual Information and 'Phantom Targets. In *Advances in Neural Information Processing Systems*, J Moody, S Hanson, and R P Lippmann (Eds.), Vol. 4. Morgan-Kaufmann. <https://proceedings.neurips.cc/paper/1991/file/a8abb4bb284b5b27aa7cb790dc20f80b-Paper.pdf>
- [5] Fabio M. Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. 2019. Domain Generalization by Solving Jigsaw Puzzles. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, New York City, 2224–2233. <https://doi.org/10.1109/CVPR.2019.00233>
- [6] Youngjae Chang, Akhil Mathur, Anton Isopoussu, Junehwa Song, and Fahim Kawsar. 2020. A Systematic Study of Unsupervised Domain Adaptation for Robust Human-Activity Recognition. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 1, Article 39 (March 2020), 30 pages. <https://doi.org/10.1145/3380985>
- [7] Jagmohan Chauhan, Jathushan Rajasegaran, Suranga Seneviratne, Archan Misra, Aruna Seneviratne, and Youngki Lee. 2018. Performance Characterization of Deep Learning Models for Breathing-Based Authentication on Resource-Constrained Devices. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 4, Article 158 (Dec. 2018), 24 pages. <https://doi.org/10.1145/3287036>
- [8] Jiefeng Chen, Frederick Liu, Besim Avci, Xi Wu, Yingyu Liang, and Somesh Jha. 2021. Detecting Errors and Estimating Accuracy on Unlabeled Data with Self-training Ensembles. *Advances in Neural Information Processing Systems* 34 (2021).
- [9] Ching-Yao Chuang, Antonio Torralba, and Stefanie Jegelka. 2020. Estimating Generalization under Distribution Shifts via Domain-Invariant Representations. *International conference on machine learning* (2020).
- [10] Shuhao Cui, Shuhui Wang, Junbao Zhuo, Chi Su, Qingming Huang, and Qi Tian. 2020. Gradually Vanishing Bridge for Adversarial Domain Adaptation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, New York City, 12452–12461. <https://doi.org/10.1109/CVPR42600.2020.01247>
- [11] Bharath Bhushan Damodaran, Benjamin Kellenberger, Rémi Flamary, Devis Tuia, and Nicolas Courty. 2018. DeepJDOT: Deep Joint Distribution Optimal Transport for Unsupervised Domain Adaptation. In *Computer Vision – ECCV 2018*, Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (Eds.). Springer International Publishing, Cham, 467–483.
- [12] Yunbin Deng. 2019. Deep learning on mobile devices: a review. In *Mobile Multimedia/Image Processing, Security, and Applications 2019*, Sos S. Agaian, Vijayan K. Asari, and Stephen P. DelMarco (Eds.), Vol. 10993. International Society for Optics and Photonics, SPIE, 52 – 66. <https://doi.org/10.1117/12.2518469>
- [13] Qi Dou, Daniel Coelho de Castro, Konstantinos Kamnitsas, and Ben Glocker. 2019. Domain Generalization via Model-Agnostic Learning of Semantic Features. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2019/file/2974788b53f73e7950e8aa49f3a306db-Paper.pdf>
- [14] Hady Elsahar and Matthias Gallé. 2019. To Annotate or Not? Predicting Performance Drop under Domain Shift. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 2163–2173. <https://doi.org/10.18653/v1/D19-1222>
- [15] Biying Fu, Naser Damer, Florian Kirchbuchner, and Arjan Kuijper. 2021. Generalization of Fitness Exercise Recognition from Doppler Measurements by Domain-Adaption and Few-Shot Learning. In *Pattern Recognition. ICPR International Workshops and Challenges*, Alberto Del Bimbo, Rita Cucchiara, Stan Sclaroff, Giovanni Maria Farinella, Tao Mei, Marco Bertini, Hugo Jair Escalante, and Roberto Vezzani (Eds.). Springer International Publishing, Cham, 203–218.

- [16] Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of the 33rd International Conference on Machine Learning - Volume 48* (New York, NY, USA) (*ICML '16*). JMLR.org, 1050–1059.
- [17] Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised Domain Adaptation by Backpropagation. In *Proceedings of the 32nd International Conference on Machine Learning - Volume 37* (Lille, France) (*ICML '15*). JMLR.org, 1180–1189.
- [18] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-Adversarial Training of Neural Networks. *J. Mach. Learn. Res.* 17, 1 (Jan. 2016), 2096–2030.
- [19] Muhammad Ghifary, W. Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. 2016. Deep Reconstruction-Classification Networks for Unsupervised Domain Adaptation. In *Computer Vision – ECCV 2016*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (Eds.). Springer International Publishing, Cham, 597–613.
- [20] Taesik Gong, Yeonsu Kim, Ryuhaerang Choi, Jinwoo Shin, and Sung-Ju Lee. 2021. Adapting to Unknown Conditions in Learning-based Mobile Sensing. *IEEE Transactions on Mobile Computing* (2021), 1–1. <https://doi.org/10.1109/TMC.2021.3061130>
- [21] Taesik Gong, Yeonsu Kim, Jinwoo Shin, and Sung-Ju Lee. 2019. MetaSense: Few-Shot Adaptation to Untrained Conditions in Deep Mobile Sensing. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems* (New York, New York) (*SensSys '19*). Association for Computing Machinery, New York, NY, USA, 110–123. <https://doi.org/10.1145/3356250.3360020>
- [22] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70* (Sydney, NSW, Australia) (*ICML '17*). JMLR.org, 1321–1330.
- [23] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. 2020. AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty. *Proceedings of the International Conference on Learning Representations (ICLR)* (2020).
- [24] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
- [25] Alexander Hoelzemann and Kristof Van Laerhoven. 2020. Digging Deeper: Towards a Better Understanding of Transfer Learning for Human Activity Recognition. In *Proceedings of the 2020 International Symposium on Wearable Computers* (Virtual Event, Mexico) (*ISWC '20*). Association for Computing Machinery, New York, NY, USA, 50–54. <https://doi.org/10.1145/3410531.3414311>
- [26] Sergey Ioffe and Christian Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning - Volume 37* (Lille, France) (*ICML '15*). JMLR.org, 448–456.
- [27] Xu Ji, João F Henriques, and Andrea Vedaldi. 2019. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*. 9865–9874.
- [28] Xiaotang Jiang, Huan Wang, Yiliu Chen, Ziqi Wu, Lichuan Wang, Bin Zou, Yafeng Yang, Zongyang Cui, Yu Cai, Tianhang Yu, Chengfei Lyu, and Zhihua Wu. 2020. MNN: A Universal and Efficient Inference Engine. In *Proceedings of Machine Learning and Systems*, I. Dhillon, D. Papailiopoulos, and V. Sze (Eds.), Vol. 2. 1–13. <https://proceedings.mlsys.org/paper/2020/file/8f14e45fcee167a5a36dedd4bea2543-Paper.pdf>
- [29] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G. Hauptmann. 2019. Contrastive Adaptation Network for Unsupervised Domain Adaptation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 4888–4897. <https://doi.org/10.1109/CVPR.2019.00503>
- [30] Md Abdullah Al Hafiz Khan, Nirmalya Roy, and Archan Misra. 2018. Scaling Human Activity Recognition via Deep Learning-based Domain Adaptation. In *2018 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. 1–9. <https://doi.org/10.1109/PERCOM.2018.8444585>
- [31] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR (Poster)*. <http://arxiv.org/abs/1412.6980>
- [32] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.), Vol. 25. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf>
- [33] Jacob Leon Kröger, Philip Raschke, and Towhidur Rahman Bhuiyan. 2019. Privacy Implications of Accelerometer Data: A Review of Possible Inferences. In *Proceedings of the 3rd International Conference on Cryptography, Security and Privacy* (Kuala Lumpur, Malaysia) (*ICCSPP '19*). Association for Computing Machinery, New York, NY, USA, 81–87. <https://doi.org/10.1145/3309074.3309076>
- [34] Abhishek Kumar, Prasanna Sattigeri, Kahini Wadhawan, Leonid Karlinsky, Rogerio Feris, Bill Freeman, and Gregory Wornell. 2018. Co-regularized Alignment for Unsupervised Domain Adaptation. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2018/file/99607461cdb9c26e2bd5f31b12dcf27a-Paper.pdf>
- [35] Vivian Lai and Chenhao Tan. 2019. On Human Predictions with Explanations and Predictions of Machine Learning Models: A Case Study on Deception Detection. In *Proceedings of the Conference on Fairness, Accountability, and Transparency* (Atlanta, GA, USA) (*FAT* '19*). Association for Computing Machinery, New York, NY, USA, 29–38. <https://doi.org/10.1145/3287560.3287590>

- [36] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. 2018. Learning to Generalize: Meta-Learning for Domain Generalization. In *AAAI*. 3490–3497. <https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16067>
- [37] Da Li, Jianshu Zhang, Yongxin Yang, Cong Liu, Yi-Zhe Song, and Timothy M. Hospedales. 2019. Episodic Training for Domain Generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 1446–1455.
- [38] Jian Liang, Dapeng Hu, and Jiashi Feng. 2020. Do We Really Need to Access the Source Data? Source Hypothesis Transfer for Unsupervised Domain Adaptation. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, 6028–6039. <http://proceedings.mlr.press/v119/liang20a.html>
- [39] Christian Limberg, Heiko Wersing, and Helge Ritter. 2020. Beyond Cross-Validation—Accuracy Estimation for Incremental and Active Learning Models. *Machine Learning and Knowledge Extraction* 2, 3 (2020), 327–346. <https://doi.org/10.3390/make2030018>
- [40] R. Linsker. 1988. Self-organization in a perceptual network. *Computer* 21, 3 (1988), 105–117. <https://doi.org/10.1109/2.36>
- [41] Mingsheng Long, ZHANGJIE CAO, Jianmin Wang, and Michael I Jordan. 2018. Conditional Adversarial Domain Adaptation. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2018/file/ab88b15733f543179858600245108dd8-Paper.pdf>
- [42] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. 2018. Conditional adversarial domain adaptation. *Advances in neural information processing systems* 31 (2018).
- [43] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. 2017. Deep Transfer Learning with Joint Adaptation Networks. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 2208–2217. <http://proceedings.mlr.press/v70/long17a.html>
- [44] Chris Xiaoxuan Lu, Bowen Du, Peijun Zhao, Hongkai Wen, Yiran Shen, Andrew Markham, and Niki Trigoni. 2018. Deepauth: In-Situ Authentication for Smartwatches via Deeply Learned Behavioural Biometrics (*ISWC '18*). Association for Computing Machinery, New York, NY, USA, 204–207. <https://doi.org/10.1145/3267242.3267252>
- [45] Jin Lu, Chao Shang, Chaoqun Yue, Reynaldo Morillo, Shweta Ware, Jayesh Kamath, Athanasios Bamis, Alexander Russell, Bing Wang, and Jinbo Bi. 2018. Joint Modeling of Heterogeneous Sensing Data for Depression Assessment via Multi-Task Learning. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 1, Article 21 (March 2018), 21 pages. <https://doi.org/10.1145/3191753>
- [46] Xinran Lu, Lei Xie, Yafeng Yin, Wei Wang, Yanling Bu, Qing Guo, and Sanglu Lu. 2020. SpeedTalker: Automobile Speed Estimation via Mobile Phones. *IEEE Transactions on Mobile Computing* (2020), 1–1. <https://doi.org/10.1109/TMC.2020.3034354>
- [47] Dougal Maclaurin, David Duvenaud, and Ryan P. Adams. 2015. Gradient-Based Hyperparameter Optimization through Reversible Learning. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37 (Lille, France) (ICML '15)*. JMLR.org, 2113–2122.
- [48] Alan Mazankiewicz, Klemens Böhm, and Mario Berges. 2020. Incremental Real-Time Personalization in Human Activity Recognition Using Domain Adaptive Batch Normalization. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 4, Article 144 (dec 2020), 20 pages. <https://doi.org/10.1145/3432230>
- [49] Abhinav Mehrotra and Mirco Musolesi. 2018. Using Autoencoders to Automatically Extract Mobility Features for Predicting Depressive States. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 3, Article 127 (Sept. 2018), 20 pages. <https://doi.org/10.1145/3264937>
- [50] Vinod Nair and Geoffrey E. Hinton. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning (Haifa, Israel) (ICML '10)*. Omnipress, Madison, WI, USA, 807–814.
- [51] Andrew Y. Ng. 2004. Feature Selection, L1 vs. L2 Regularization, and Rotational Invariance. In *Proceedings of the Twenty-First International Conference on Machine Learning (Banff, Alberta, Canada) (ICML '04)*. Association for Computing Machinery, New York, NY, USA, 78. <https://doi.org/10.1145/1015330.1015435>
- [52] Babatunde Kazeem Olorisade, Pearl Brereton, and Peter Andras. 2017. Reproducibility of studies on text mining for citation screening in systematic reviews: Evaluation and checklist. *Journal of Biomedical Informatics* 73 (2017), 1–13. <https://doi.org/10.1016/j.jbi.2017.07.010>
- [53] Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua V. Dillon, Balaji Lakshminarayanan, and Jasper Snoek. 2019. *Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty under Dataset Shift*. Curran Associates Inc., Red Hook, NY, USA.
- [54] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 8024–8035. <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>
- [55] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. 2018. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA) (Brisbane, Australia)*. IEEE Press, 1–8. <https://doi.org/10.1109/ICRA.2018.8460528>

- [56] Emmanouil Antonios Platanios, Avrim Blum, and Tom Mitchell. 2014. Estimating Accuracy from Unlabeled Data. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence* (Quebec City, Quebec, Canada) (UAI'14). AUAI Press, Arlington, Virginia, USA, 682–691.
- [57] Emmanouil A. Platanios, Hoifung Poon, Tom M. Mitchell, and Eric Joel Horvitz. 2017. Estimating Accuracy from Unlabeled Data: A Probabilistic Logic Approach. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4–9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.), 4361–4370. <https://proceedings.neurips.cc/paper/2017/hash/95f8d9901ca8878e291552f001f67692-Abstract.html>
- [58] Xin Qin, Yiqiang Chen, Jindong Wang, and Chaohui Yu. 2020. Cross-Dataset Activity Recognition via Adaptive Spatial-Temporal Transfer Learning. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 4, Article 148 (sep 2020), 25 pages. <https://doi.org/10.1145/3369818>
- [59] Joaquin Quiñero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. 2008. *Dataset shift in machine learning*. MIT Press.
- [60] Valentin Radu, Catherine Tong, Sourav Bhattacharya, Nicholas D. Lane, Cecilia Mascolo, Mahesh K. Marina, and Fahim Kawsar. 2018. Multimodal Deep Learning for Activity and Context Recognition. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 4, Article 157 (Jan. 2018), 27 pages. <https://doi.org/10.1145/3161174>
- [61] Zuzana Reitermanova. 2010. Data splitting. In *WDS*, Vol. 10. 31–36.
- [62] Charissa Ann Ronao and Sung-Bae Cho. 2016. Human activity recognition with smartphone sensors using deep learning neural networks. *Expert Systems with Applications* 59 (2016), 235–244. <https://doi.org/10.1016/j.eswa.2016.04.032>
- [63] Aaqib Saeed, Tanir Ozcelebi, and Johan Lukkien. 2019. Multi-Task Self-Supervised Learning for Human Activity Detection. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 2, Article 61 (June 2019), 30 pages. <https://doi.org/10.1145/3328932>
- [64] Swami Sankaranarayanan, Yogesh Balaji, Carlos D. Castillo, and Rama Chellappa. 2018. Generate to Adapt: Aligning Domains Using Generative Adversarial Networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8503–8512. <https://doi.org/10.1109/CVPR.2018.00887>
- [65] Philip Schmidt, Attila Reiss, Robert Duerichen, Claus Marberger, and Kristof Van Laerhoven. 2018. Introducing wesad, a multimodal dataset for wearable stress and affect detection. In *Proceedings of the 20th ACM International Conference on Multimodal Interaction*. 400–408.
- [66] C. E. Shannon. 1948. A Mathematical Theory of Communication. *Bell System Technical Journal* 27, 3 (1948), 379–423. <https://doi.org/10.1002/j.1538-7305.1948.tb01338.x> arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1002/j.1538-7305.1948.tb01338.x>
- [67] Rui Shu, Hung Bui, Hirokazu Narui, and Stefano Ermon. 2018. A DIRT-T Approach to Unsupervised Domain Adaptation. In *International Conference on Learning Representations*. <https://openreview.net/forum?id=H1q-TM-AW>
- [68] Andrea Soro, Gino Brunner, Simon Tanner, and Roger Wattenhofer. 2019. Recognition and Repetition Counting for Complex Physical Exercises with Deep Learning. *Sensors* 19, 3 (2019). <https://doi.org/10.3390/s19030714>
- [69] Jacob Steinhardt and Percy S Liang. 2016. Unsupervised Risk Estimation Using Only Conditional Independence Structure. In *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (Eds.), Vol. 29. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2016/file/f2d887e01a80e813d9080038decbbabb-Paper.pdf>
- [70] Allan Stisen, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærsgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen. 2015. Smart Devices Are Different: Assessing and Mitigating Mobile Sensing Heterogeneities for Activity Recognition. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems* (Seoul, South Korea) (*SenSys '15*). Association for Computing Machinery, New York, NY, USA, 127–140. <https://doi.org/10.1145/2809695.2809718>
- [71] Baochen Sun and Kate Saenko. 2016. Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In *Computer Vision – ECCV 2016 Workshops*, Gang Hua and Hervé Jégou (Eds.). Springer International Publishing, Cham, 443–450.
- [72] Chi Ian Tang, Ignacio Perez-Pozuelo, Dimitris Spathis, Soren Brage, Nick Wareham, and Cecilia Mascolo. 2021. SelfHAR. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 1 (Mar 2021), 1–30. <https://doi.org/10.1145/3448112>
- [73] Hui Tang, Ke Chen, and Kui Jia. 2020. Unsupervised Domain Adaptation via Structurally Regularized Deep Clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, New York City, 8725–8735.
- [74] Hui Tang and Kui Jia. 2020. Discriminative Adversarial Domain Adaptation. In *Association for the Advancement of Artificial Intelligence (AAAI)*.
- [75] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. 2017. Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Vancouver, BC, Canada). IEEE Press, 23–30. <https://doi.org/10.1109/IROS.2017.8202133>
- [76] Yunus Emre Ustev, Ozlem Durmaz Incel, and Cem Ersoy. 2013. User, Device and Orientation Independent Human Activity Recognition on Mobile Phones: Challenges and a Proposal. In *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication* (Zurich, Switzerland) (*UbiComp '13 Adjunct*). Association for Computing Machinery, New York, NY, USA, 1427–1436. <https://doi.org/10.1145/2494091.2496039>

- [77] He Wang, Ted Tsung-Te Lai, and Romit Roy Choudhury. 2015. MoLe: Motion Leaks through Smartwatch Sensors. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (Paris, France) (MobiCom '15)*. Association for Computing Machinery, New York, NY, USA, 155–166. <https://doi.org/10.1145/2789168.2790121>
- [78] Haoran Wang, Tong Shen, Wei Zhang, Lingyu Duan, and Tao Mei. 2020. Classes Matter: A Fine-grained Adversarial Approach to Cross-domain Semantic Segmentation. In *The European Conference on Computer Vision (ECCV)*.
- [79] Ximei Wang, Ying Jin, Mingsheng Long, Jianmin Wang, and Michael I Jordan. 2019. Transferable Normalization: Towards Improving Transferability of Deep Neural Networks. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2019/file/fd2c5e4680d9a01dba3aada5ece22270-Paper.pdf>
- [80] Garrett Wilson, Janardhan Rao Doppa, and Diane J. Cook. 2020. Multi-Source Deep Domain Adaptation with Weak Supervision for Time-Series Sensor Data. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (Virtual Event, CA, USA) (KDD '20)*. Association for Computing Machinery, New York, NY, USA, 1768–1778. <https://doi.org/10.1145/3394486.3403228>
- [81] Shaoan Xie, Zibin Zheng, Liang Chen, and Chuan Chen. 2018. Learning Semantic Representations for Unsupervised Domain Adaptation. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 5423–5432. <http://proceedings.mlr.press/v80/xie18c.html>
- [82] Minghao Xu, Jian Zhang, Bingbing Ni, Teng Li, Chengjie Wang, Qi Tian, and Wenjun Zhang. 2020. Adversarial Domain Adaptation with Domain Mixup. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*. AAAI Press, 6502–6509.
- [83] Xiangyu Xu, Jiadi Yu, Yingying Chen, Yanmin Zhu, Linghe Kong, and Minglu Li. 2019. BreathListener: Fine-Grained Breathing Monitoring in Driving Environments Utilizing Acoustic Signals (*MobiSys '19*). Association for Computing Machinery, New York, NY, USA, 54–66. <https://doi.org/10.1145/3307334.3326074>
- [84] Yun Xu and Royston Goodacre. 2018. On splitting training and validation set: A comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning. *Journal of Analysis and Testing* 2, 3 (2018), 249–262.
- [85] Ming Yin, Jennifer Wortman Vaughan, and Hanna Wallach. 2019. *Understanding the Effect of Accuracy on Trust in Machine Learning Models*. Association for Computing Machinery, New York, NY, USA, 1–12. <https://doi.org/10.1145/3290605.3300509>
- [86] Kaichao You, Ximei Wang, Mingsheng Long, and Michael Jordan. 2019. Towards Accurate Model Selection in Deep Unsupervised Domain Adaptation. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 7124–7133. <http://proceedings.mlr.press/v97/you19a.html>
- [87] Kun Yu, Shlomo Berkovsky, Ronnie Taib, Jianlong Zhou, and Fang Chen. 2019. Do I Trust My Machine Teammate? An Investigation from Perception to Decision. In *Proceedings of the 24th International Conference on Intelligent User Interfaces (Marina del Ray, California) (IUI '19)*. Association for Computing Machinery, New York, NY, USA, 460–468. <https://doi.org/10.1145/3301275.3302277>
- [88] Mattia Zeni, Wanyi Zhang, Enrico Bignotti, Andrea Passerini, and Fausto Giunchiglia. 2019. Fixing Mislabeling by Human Annotators Leveraging Conflict Resolution and Prior Knowledge. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 1, Article 32 (March 2019), 23 pages. <https://doi.org/10.1145/3314419>
- [89] Hanbin Zhang, Gabriel Guo, Chen Song, Chenhan Xu, Kevin Cheung, Jasleen Alexis, Huining Li, Dongmei Li, Kun Wang, and Wenyao Xu. 2020. PDLens: Smartphone Knows Drug Effectiveness among Parkinson's via Daily-Life Activity Fusion. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking (London, United Kingdom) (MobiCom '20)*. Association for Computing Machinery, New York, NY, USA, Article 12, 14 pages. <https://doi.org/10.1145/3372224.3380889>
- [90] Hanbin Zhang, Chen Song, Aosen Wang, Chenhan Xu, Dongmei Li, and Wenyao Xu. 2019. PDVocal: Towards Privacy-Preserving Parkinson's Disease Detection Using Non-Speech Body Sounds. In *The 25th Annual International Conference on Mobile Computing and Networking (Los Cabos, Mexico) (MobiCom '19)*. Association for Computing Machinery, New York, NY, USA, Article 16, 16 pages. <https://doi.org/10.1145/3300061.3300125>
- [91] Hanbin Zhang, Chenhan Xu, Huining Li, Aditya Singh Rathore, Chen Song, Zhisheng Yan, Dongmei Li, Feng Lin, Kun Wang, and Wenyao Xu. 2019. PDMove: Towards Passive Medication Adherence Monitoring of Parkinson's Disease Using Smartphone-Based Gait Assessment. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 3, Article 123 (Sept. 2019), 23 pages. <https://doi.org/10.1145/3351281>
- [92] Yu Zhang, Tao Gu, and Xi Zhang. 2020. MDLdroid: a ChainSGD-reduce Approach to Mobile Deep Learning for Personal Mobile Sensing. In *2020 19th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 73–84. <https://doi.org/10.1109/IPSN48710.2020.00-45>
- [93] Bing Zhou, Jay Lohokare, Ruipeng Gao, and Fan Ye. 2018. EchoPrint: Two-Factor Authentication Using Acoustics and Vision on Smartphones. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking (New Delhi, India) (MobiCom '18)*. Association for Computing Machinery, New York, NY, USA, 321–336. <https://doi.org/10.1145/3241539.3241575>
- [94] Zhijun Zhou, Yingtian Zhang, Xiaojing Yu, Panlong Yang, Xiang-Yang Li, Jing Zhao, and Hao Zhou. 2020. XHAR: Deep Domain Adaptation for Human Activity Recognition with Smart Devices. In *2020 17th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. 1–9. <https://doi.org/10.1109/SECON48991.2020.9158431>