Unsupervised Domain Adaptation for Estimating Occupancy and Recognizing Activities in Smart Buildings

Jawher Dridi

A Thesis

in

The Department

of

Concordia Institute for Information Systems Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Master of Applied Science (Quality Systems Engineering) at

Concordia University

Montréal, Québec, Canada

August 2023

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Jawher Dridi**

Entitled: **Unsupervised Domain Adaptation for Estimating Occupancy and Recognizing Activities in Smart Buildings**

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Quality Systems Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

_____ Chair and Examiner
Dr. Amr Youssef

_____ Examiner
Dr. Chadi Assi

_____ Supervisor
Dr. Manar Amayri

_____ Supervisor
Dr. Nizar Bouguila

Approved by _____
Chun Wang, Chair
Department of Concordia Institute for Information Systems Engineering

_____ 2023

_____
Mourad Debbabi, Dean
Faculty of Engineering and Computer Science

Abstract

Unsupervised Domain Adaptation for Estimating Occupancy and Recognizing Activities in Smart Buildings

Jawher Dridi

Activities Recognition (AR) and Occupancy Estimation (OE) are topics of current interest. AR and OE can develop many smart building applications such as energy management and can help provide good services for residents. Prior research on AR and OE has typically focused on supervised machine learning methods. For a specific smart building domain, a model is trained using data collected from the current environment (domain). The created model will not generalize well when evaluated in a new related domain due to data distribution differences. Creating a model for each smart building environment is infeasible due to the lack of labeled data. Indeed, data collection is a tedious and time-consuming task. Unsupervised Domain Adaptation (UDA) is a good solution for the considered case. UDA solves the problem of the lack of labeled data in the target domain by allowing knowledge transfer across domains. In this research, we provide several UDA methods that mitigate the data distribution shift between source and target domains using unlabeled target data for OE and AR with and without direct access to labeled source data. Firstly, we consider techniques that use only a trained source model instead of a huge amount of labeled source data to make domain adaptation. We adapted and tested several UDA methods such as Source HypOthesis Transfer (SHOT), Higher-Order Moment Matching (HoMM), and Source data Free Domain Adaptation (SFDA) on smart building data. Secondly, we adapt and develop several UDA methods that use labeled source data to estimate the number of occupants and recognize activities. The developed methods that have direct access to the source data are the Virtual Adversarial Domain Adaptation (VADA), Sliced Wasserstein Discrepancy (SWD), and Adaptive Feature Norm (AFN). Finally, we make a comparative analysis between several newly adapted deep UDA methods, applied to the tasks of AR and OE, with and without access to labeled source data.

Acknowledgments

I would like to thank my supervisor Dr. Nizar Bouguila for his support and encouragement during my entire journey at Concordia University.

I would like to thank also my supervisor Dr. Manar Amayri for her advice and valuable comments that helped me develop further my research.

Many thanks to my friends, my family, and especially my lovely parents who were there when I needed them all the time.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem statement

Activities Recognition (AR) [1, 2, 3, 4, 5, 6, 7, 8, 9] and Occupancy Estimation (OE) [10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21] are interesting smart building [22, 23, 24] tasks that can improve several applications such as energy management. Indeed, OE can help provide optimal energy management for smart homes by reducing energy consumption and by optimally distributing energy across all the building's apartments [25, 19, 20, 26, 27, 28, 29, 30]. Detecting humans and recognizing their activities provide more security for smart homes by identifying authorized actions [31, 32, 33]. Recognizing activities allows managers of buildings to optimize HVAC (heating, ventilation, and air conditioning) systems by understanding occupants' behavior in each room, providing inhabitants with more comfort [34, 35]. Smart building models created in a specific environment [25, 13, 27, 36] (room or apartment) can not generalize well to other domains due to different apartment architectures (e.g., locations and types of the used sensors). Training a model for each room or apartment is an expensive and challenging task because of labeled data scarcity [25, 37, 38]. Labeling smart home data is a tedious, costly, and time-consuming task that can not be feasible in some cases due to privacy issues [25, 13, 24, 39, 40, 41, 42]. Domain adaptation (DA) [43, 44, 45, 46, 47, 48, 49, 50] can mitigate data distribution shifts between source and target domains in smart homes. Thus, it can solve the problem of data scarcity [25, 37, 38], and share knowledge gained in a smart home where labeled data are available to other smart buildings where

labeled data are scarce or not available. Sharing knowledge between domains helps increase the performance of models as well as their robustness. Indeed, creating models that generalize well for related domains decreases sensitivity to domain shift, providing robust methods. Some approaches of DA can solve privacy issues by avoiding access to source data while sharing information across environments. Indeed, they get access only to trained models from source domains without accessing original data.

1.2 Theoretical background and related works

1.2.1 Domain adaptation (DA) fundamentals

Definition of DA

Before giving the definition of domain adaptation, let us start by defining some terms related to DA.

Domain: A domain is an environment usually denoted $\mathcal{D}$. It is composed of a feature space and a marginal distribution. We denote the feature space with $\mathcal{X}$ and the marginal distribution with $P(X)$ [51, 52], where X is an instance of the feature space $\mathcal{X}$, $X = \{x_i | x_i \in \mathcal{X}, i = 1, \ldots, n\}$.

Task: A task is a prediction that is usually denoted $\mathcal{T}$. It is composed of a label space $\mathcal{Y}$ and a decision function $\mho$, $\mathcal{T} = \{\mathcal{Y}, \mho\}$. The decision function is learned from the input data [51, 52].

Domain adaptation: Let us define a source domain as $\mathcal{D}_{\mathcal{S}}$, a target domain as $\mathcal{D}_{\mathcal{T}}$, a source task as $\mathcal{T}_{\mathcal{S}}$, and a target task as $\mathcal{T}_{\mathcal{T}}$. Domain adaptation is a type of transductive transfer learning where the source and target tasks are the same but the domains differ. The goal of DA is to improve and enhance the performance of the decision function $\mho_{\mathbb{T}}$ of the target task by using the knowledge gained from the source domain $\mathcal{D}_{\mathcal{S}}$ and transferred by minimizing domain discrepancy between source and target domains [51, 52, 53].

Domain adaptation methods

Homogeneous DA: Homogeneous domain adaptation is defined as the scenario where we have the same feature representation and task between source domains and the target domain ($\mathcal{X}_S = \mathcal{X}_T, \mathcal{Y}_S = \mathcal{Y}_T$), but the feature distributions differ between domains ($P(X_S) \neq P(X_T)$) [54]. Several methods have been considered for homogeneous domain adaptation as follows. Instance re-weighting is the case where we assume that the conditional distributions of the tasks given the feature representations are the same between source and target domains ($P(Y|X_S) = P(Y|X_T)$). In this case, a model learned from a source domain can be used to estimate the target instances ($P(Y|X_T)$) [54]. However, given that $P(X_S) \neq P(X_T)$, the target model may show poor performance in predicting target instances [54]. Instance re-weighting solves the encountered problem by re-weighting source and target instances giving large weights to target instances that are miss-classified while training and decreasing weights for source samples that are miss-classified to ignore them in the training [54]. Parameter adaptation does not require the assumption that conditional distributions of tasks given the features representations are the same between domains ($P(Y|X_S) = P(Y|X_T)$). Indeed, it works in a semi-supervised way by using a small number of labeled target data to enhance and adapt the performance of a source-trained classifier for a target task. It uses the knowledge collected by the small number of labeled sets to generalize the classification process and reduce error [54]. For instance, the domain transfer SVM [55] uses the maximum mean discrepancy (MMD) to reduce the difference between domains and creates the target decision function to estimate target samples. Feature space alignment seeks an alignment between source domains and target domain by minimizing the discrepancy between the sub-spaces obtained using principal component analysis (PCA) of the source and target domains [54]. The minimized distance is the Bregman divergence distance [54]. The unsupervised feature transformation technique is based on methods that do not use labeled data from the target domain to align data distribution across different domains. Transfer Component Analysis (TCA) [54, 56] looks for common feature distributions between source and target domains without changing the intrinsic structure of the original domains. Domain Invariant Projection [54, 57] improves the discrepancy distance used to compare the distributions from a simple distance in the lower dimensional space to a distance defined in

the RKHS (Reproducing Kernel Hilbert Space). The supervised feature transformation technique is based on methods that use labeled data while adapting source and target domains. Semi-Supervised TCA [54, 56, 58] makes domain adaptation using an objective function that contains, additionally to the distance between the domains, a label discrepancy term to maximize domain adaptation. Other approaches use conditional distributions including target predictions and source labels as described in the adaptive kernel approach [59]. Source domain weighting is a technique in the case of multiple source domain adaptation. It selects the most significant domains that provide the most interesting information that helps align data distribution. Indeed, the process is done by giving large weights to the important domains and by penalizing the other ones [54].

Heterogeneous DA: Heterogeneous domain adaptation is the scenario where the source and target domains have different feature representations. Several methods have been considered for the defined scenario as follows. Symmetric feature transformation is a technique used to create a new common feature space between source and target domains. Indeed, it projects the two domains into a new domain where knowledge transfer is possible and the target decision function can be learned in an efficient way [54, 25]. PCA has been used in several methods considered in [25] to create a common domain between source and target feature spaces, then mapping between features has been made using Jensen-Shannon divergence (JSD) distance. Asymmetric feature transformation is a domain adaptation technique that instead of learning a new common domain between the source and target domains, transforms the source feature representation to fit the target domain. Thus, we minimize the discrepancy between source and target domains and we allow the target decision function to be enhanced using source knowledge [54].

1.2.2 Literature review

Activities recognition

For the activities recognition task, several domains adaptation methods have been introduced to transfer knowledge across domains in an efficient way. Also, different types of domain adaptation have been considered such as supervised, semi-supervised, and unsupervised approaches. Indeed, for smart building tasks, unsupervised approaches are recommended to be used due to the lack of

labeled data. [60] has proposed a hybrid deep domain adaptation neural network framework called "Hydranet" which is a supervised DA approach. It is called "hybrid" because we have a heterogeneous domain adaptation with some common features between source and target domains that uses sensor data (magnetometer, gyroscope, and accelerometer) based on motions to train the target model. The proposed framework maps common features between domains, then it approximates the missing features from both domains to obtain a homogeneous common domain between source and target data. [61] has proposed supervised approaches based on several machine learning models such as naive Bayes, nearest neighbor, neural network, support vector machines, and random forest. The main objective of these models is to recognize hand activities using sensor data gathered from the accelerometer and gyroscope attached to the wrist. [62] has shown several sensor data-based approaches that have been developed previously to recognize activities in a supervised and unsupervised manner. The presented approaches use different machine learning models such as naive Bayes, neural networks, support vector machines, decision trees, hidden Markov models, and conditional random fields. [25] has considered several supervised DA methods that are based on PCA to perform domain alignment between source and target spaces. The approaches used binary reed switch sensor data to evaluate the performance of the developed methods. Using PCA, the original data from the source and target domains are transformed into a new common domain, and the transformed features from both domains are mapped based on the values of their divergence calculated using JSD distance. [63] has proposed a convolutional neural network (CNNs)-based model called Heterogeneous Deep Convolutional Neural Network ("HDCNN") which is a supervised DA method. The considered approach uses accelerometer data to adjust weights in the neural network layers to minimize the divergence in data distributions between source and target domains [63]. [64] has proposed an approach to recognize hand gestures and daily activities for a human being using wearable sensors attached to several parts of the body such as fingers, waist, and ankle. The method is based on neural networks and hidden Markov models, and it uses sensing data such as an accelerometer. [65] has used accelerometers in smartphones to gather required sensing data to recognize human physical activities (sitting, standing, laying, walking, and jogging). [65] has developed a supervised approach that uses the K-nearest neighbors algorithm to predict activities. [66] has proposed a supervised DA approach based on deep generative domain adaptation that uses

wearable sensors (gyroscope and accelerometer). The method approximates the posterior of the feature distributions and uses it as a tool to align feature spaces between source and target domains. [67] has proposed a semi-supervised DA approach called "AdaptNet" that uses accelerometer data with a small number of labeled data in the target domain to align data distributions between domains. The method makes bilateral DA using deep translation networks [67]. [68] has considered an unsupervised DA method called local domain adaptation (LDA). The method groups data into clusters and maps the clusters from each domain to make data distribution alignment at the cluster level. The data used to train the models is collected using wearable sensors such as accelerometers, gyroscopes, and magnetometers. [69] has developed an unsupervised adversarial deep domain adaptation method called "XHAR" that uses motion sensor data such as data collected from accelerometers and gyroscopes. "XHAR" uses CNNs and Bidirectional Gated Recurrent Units (BiGRU) to extract significant features. Then, it removes domain discrepancies using domain discriminators. [70] has used several types of sensor data (wireless motion sensor, passive infrared, switch, and pressure sensors) to evaluate an unsupervised DA approach called "UDAR". The method applies variational autoencoder (VAE) to adapt feature distributions across source and target domains.

Occupancy estimation

For the OE task, a lot of DA approaches have been introduced like the AR task. Also, the methods are of different types such as supervised, semi-supervised, and unsupervised learning. [25] has used PCA transformation to perform supervised domain adaptation to transfer knowledge from a source domain to a target domain. [25] has used ambient sensors to collect the used data such as pressure sensors, CO2 concentration sensors, motion sensors, etc. The considered methods are based on PCA to create a new common domain. Then, the mapping between the transformed features has been done based on JSD distance values. [71] has developed a semi-supervised/unsupervised DA technique based on recurrent neural networks (RNNs) to transfer knowledge of models trained on source domains to a target domain where there is a lack of labeled data. The method has been tested on BMS sensor data such as temperature sensors, humidity sensors, motion sensors, etc. [72, 73] have introduced a semi-supervised DA method called domain adaptation method for carbon dioxide - Human Occupancy Counter (DA-HOC). It uses CO2

concentration sensors to collect data to train models.

1.3 Contributions

This thesis has several contributions that can be listed as follows:

- **Unsupervised Domain Adaptation without Source Data for Estimating Occupancy and Recognizing Activities in Smart Buildings**: In this research, we have adapted several UDA methods, that have no direct access to labeled source data, from a 2-dimensional environment (image data) to a 1-Dimensional environment (sensor data). The adapted methods are Source HypOthesis Transfer (SHOT) [74], Source HypOthesis Transfer with Information Maximization (SHOT-IM) [74], Higher-Order Moment Matching (HoMM) [75], and Source data Free Domain Adaptation (SFDA) [44]. We have provided new feature extractors and classifiers architectures so that they fit sensor data. The adapted methods can fit any 1-Dimensional data and not necessarily smart building data. In a lot of testing scenarios, the obtained scores of the adapted methods are better than supervised machine learning methods scores.

  This research has been submitted to Energy and Buildings - Special Issue: "Efficiency and Energy Integration in Buildings - Smart Cities" [76].

- **Unsupervised Domain Adaptation with Source Data for Estimating Occupancy and Recognizing Activities in Smart Buildings**: In this research, we have adapted several methods for smart buildings data (AR and OE) that have direct access to labeled source data, and they can fit sensor data. The considered unsupervised domain adaptation approaches are Virtual Adversarial Domain Adaptation (VADA) method [77], Sliced Wasserstein Discrepancy (SWD) method [78], Hard Adaptive Feature Norm (HAFN) [79], Stepwise Adaptive Feature Norm (SAFN) [79], and SAFN with entropy minimization (SAFN+ENTM) [79]. We provided new deep neural network architectures for the feature extractors and classifiers in most of the considered methods. We evaluated these methods on AR and OE datasets and showed their strengths and weaknesses by analyzing the findings and by applying a data poisoning technique [80].

This research has been submitted to IEEE Transactions on Artificial Intelligence [81].

- **Unsupervised Domain Adaptation With and Without Access to Source Data for Estimating Occupancy and Recognizing Activities in Smart Buildings**: In this research, we have adapted several UDA approaches for smart buildings data (AR and OE) so that they can work with any 1-Dimensional data such as sensor data, and we have tested them on smart buildings datasets (private and public datasets). We have 6 adapted approaches for UDA with access to labeled source data: domain separation networks (DSN) [82], cluster alignment with a teacher (CAT) [83], CAT+ gradient reversal (RevGrad) [83], CAT + robust RevGrad (rRevGrad) [83], Auxiliary Target Domain-Oriented Classifier (ATDOC) with nearest centroid classifier (NC), and ATDOC with neighborhood aggregation (NA) [84]. Also, we have 6 adapted methods for UDA without labeled source data: confidence score weighting adaptation using joint model data structure (CoWA-JMDS) [85], CoWA-JMDS without weights mixup [85], divide and contrast (DaC) [86], attracting and dispersing (AaD) [87], source hypothesis transfer with information maximization (SHOT-IM) [74, 88], and source hypothesis transfer with self-supervised pseudo-labeling (SHOT-Pseudo-labeling) [74, 88]. We have created novel deep neural network architectures for most of the considered methods. Indeed, all the feature extractors and classifiers have been changed with new ones based on convolutional neural networks that are adaptable to sensor data. We have made a comparative analysis between UDA methods' performance with and without direct access to labeled source data, and we have chosen the best approach based on several factors such as privacy issues.

  This research has been accepted in Building and Environment [89].

1.4 Thesis Overview

- In chapter 1, we introduce some theoretical background of domain adaptation and some related works of activities recognition and occupancy estimation.

- In chapter 2, we considered techniques that use only a trained source model instead of a huge amount of labeled source data to make domain adaption in order to estimate the number of

occupants and recognize activities.

- In chapter 3, we adapt and develop several UDA methods that have direct access to labeled source data to estimate the number of occupants and recognize activities.
- In chapter 4, we make a comparative analysis between several adapted deep UDA methods, applied to the tasks of AR and OE, with and without access to labeled source data.
- In conclusion, we summarize our main findings and contributions.

Chapter 2

Unsupervised Domain Adaptation without Source Data for Estimating Occupancy and Recognizing Activities in Smart Buildings

2.1 Introduction

Nowadays, buildings consume more than 32% of the world's electrical energy [90]. Smart buildings [22, 23, 24] have been introduced in the last two decades to ease people's lives and to solve many problems such as energy management [27, 28, 29, 30]. It helps provide optimal energy distribution and reduce energy consumption. Using sensor data, researchers considered multiple smart building tasks such as AR [1, 2, 3, 4, 5, 6, 7, 8, 9] and OE [10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21]. Prior research on AR and OE typically has focused on supervised machine learning methods [13]. Models are trained and evaluated using data from the same domain. Researchers have assumed that data distribution does not change across environments which is not true in most cases.

In this research, we consider domain adaptation methods that do not require labeled data in

the target domain [74, 75, 44]. Labeled data have been a primary problem for researchers in smart building tasks. The collection of labeled data is tedious, time-consuming, and can affect the behavior of the collector with time. Knowledge transfer using domain adaptation is a good solution to avoid data collection and to obtain models with good performance. It helps gain knowledge from source domains and apply it to a target domain in order to improve performance or solve problems like computational power and lack of data. Domain adaptation (DA) methods typically require access to the source data, but the UDA methods that we are considering in this research do not require explicit access to source data. They only use a trained source model instead of source data. Using a trained model has two direct benefits related to privacy and storage [74, 91]. Indeed, it protects data privacy which is one of the blocks to data collection in most cases. Also, adapting domains using models instead of data will help us avoid the huge amount of storage needed to store source data. The first considered method called Source HypOthesis Transfer (SHOT) [74] is inspired by [92] and it consists of a feature encoding module and a classifier module (hypothesis). SHOT freezes the source hypothesis and learns a target feature extractor that gives a target data representation like the source data representation without having access to source data or target labels. In this case, the source classifier can infer the labels of target instances. The feature encoding module of the target is learned by information maximization (IM) [93] and a novel self-supervised pseudo-labeling method [74]. Even though IM helps align feature representation to the source classifier, it is not an optimal method. That is why a self-supervised pseudo-labeling method has been used. To enhance the performance of domain adaptation, batch normalization [94], weight normalization [95] and label smoothing [96] have been introduced in the source network. The second considered method called Higher-Order Moment Matching (HoMM) [75] freezes the source feature extractor module and learns a target classifier to make data alignment using a source-trained model. It also uses pseudo-labeling for target samples to enhance model performance [75]. The third considered method called Source data Free Domain Adaptation (SFDA) [44] uses a target model that is initialized by the weights of the source pre-trained model. Then, the target model is trained in a progressive way using two losses without using any labeled source data. The first loss aims to prevent the target model from biasing by using source pseudo labels of target samples generated by the source pre-trained model [44]. The second loss trains the target model using target pseudo labels generated by

the trainable target model in each iteration [44]. In this research, we contributed by providing new model architectures for the feature extractors and the classifiers to fit sensor data. We also tested the considered methods in a new field of data (sensor data instead of image data). Moreover, we are the first to apply these methods to the fields of smart buildings (AR and OE). In a lot of cases, the obtained results using the considered methods are better than the results of supervised machine learning methods even though we are training target models with unlabeled data. All these findings prove the efficiency of the considered methods.

The rest of the chapter is organized into 2 sections. In section 2, we introduce the considered methods and, in section 3, we discuss the experimental results.

2.2 The considered methods

In this research, we adapted unsupervised DA methods called SHOT [74], HoMM [75] and SFDA [44] for smart building tasks (AR and OE). The tested methods use only a pre-trained model from the source domain without the need of getting access to source data. From the source environment, we are given a set of source samples $\mathrm{X_s} = \left\{x_s^i\right\}_{i=1}^{n_s}$ and a set of source labels $\mathrm{Y_s} = \left\{y_s^i\right\}_{i=1}^{n_s}$, where n_s is the number of source instances. From the target environment, we are given a set of unlabeled target samples $\mathrm{X_t} = \left\{x_t^i\right\}_{i=1}^{n_t}$, where n_t is the number of target instances. Using the source prediction function $f_s : \mathrm{X_s} \longrightarrow \mathrm{Y_s}$ and the target unlabeled samples $\mathrm{X_t}$, the goal of the algorithm SHOT is to learn the target prediction function $f_t : \mathrm{X_t} \longrightarrow \mathrm{Y_t}$ and predict $\mathrm{Y_t} = \left\{y_t^i\right\}_{i=1}^{n_t}$, where Y_t is a set of target labels [74].

The realization of the considered method is done by generating the source model, transferring the source model to the target domain without accessing the source data, and enhancing the adaptation method by acting on the network architecture.

2.2.1 Creating the source model

The first step in developing the considered methods is to generate the source model. To obtain the source model $f_s : \mathrm{X_s} \longrightarrow \mathrm{Y_s}$, we need to develop a deep learning model and minimize the following objective function with label smoothing [74] for the SHOT method. Label smoothing

helps to enhance the discriminability of the source model [96].

$$\mathcal{L}_s^{ls}(f_s; \mathrm{X_s}, \mathrm{Y_s}) = -E_{(x_s,y_s)\in \mathrm{X_s}\times \mathrm{Y_s}} \sum_{k=1}^{K} q_k^{ls} \log \delta_k(f_s(x_s)), \tag{1}$$

where $\delta_k(u) = \frac{\exp(u_k)}{\sum_i \exp(u_i)}$ represents the k-th element of the softmax output for a given vector u of a dimension K. K is the number of classes, and q is one-of-K encoding of y_s. q_k takes 1 for the correct cases and 0 otherwise. The smoothed label is defined as $q_s^{ls} = (1-\alpha)q_k + \alpha/K$ and the smoothing parameter is set to 0.1.

2.2.2 Source HypOthesis Transfer with Information Maximization (SHOT-IM)

As shown in Figure 2.1 [74], the source model consists of a feature extractor module $g_s : \mathrm{X_s} \longrightarrow \mathbb{R}^d$ and a classifier module $h_s : \mathbb{R}^d \longrightarrow \mathbb{R}^K$, where d is the dimension of the feature space of the input instances and K is the number of classes. Here, the source model is defined as $f_s(x) = h_s(g_s(x))$. SHOT freezes the source classifier $h_s = h_t$ and it learns a target feature encoding module $g_t : \mathrm{X_t} \longrightarrow \mathbb{R}^d$ such that the output target data distribution $p(g_t(x_t))$ matches the data distribution of the source data $p(g_s(x_s))$. Hence, it can be classified accurately using the source classifier. It is important to note that SHOT does not use the source data while creating the target feature extractor and it uses the source feature encoder as initialization [74].

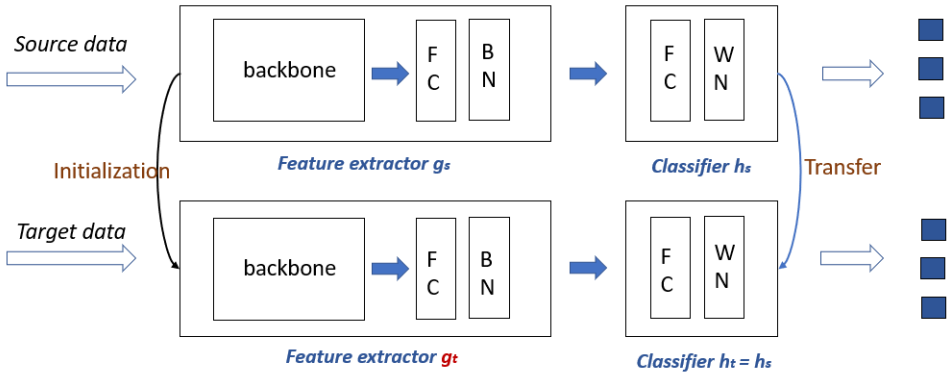

Figure 2.1: SHOT pipeline

Information maximization (IM) [93, 97] loss has been employed to learn the target feature encoder module as follows:

$$\begin{aligned} \mathcal{L}_{ent}(f_t; \mathrm{X_t}) &= -E_{x_t \in \mathrm{X_t}} \sum_{k=1}^{K} \delta_k(f_t(x_t)) \log \delta_k(f_t(x_t)), \\ \mathcal{L}_{div}(f_t; \mathrm{X_t}) &= \sum_{k=1}^{K} \hat{p}_k \log \hat{p}_k = D_{KL}(\hat{p}, \frac{1}{K} 1_K) - \log(K) \end{aligned} \tag{2}$$

where D_{KL} is the Kullback-Leibler divergence, 1_K is a ones vector of dimension K, $\hat{p} = E_{x_t \in \mathrm{X_t}}(\delta_k(f_t^{(k)}(x_t)))$ is the average output embedding of the target environment.

2.2.3 SHOT augmented with self-supervised pseudo-labeling

To further enhance the performance of the target model, pseudo-labeling [98] for target unlabeled data has been introduced and specifically a self-supervised pseudo-labeling strategy inspired from [99]. Pseudo-labeling is defined as giving unlabeled data labels using supervised models. Pseudo-labels of target data can be generated using the source model, but due to the domain shift between source and target domains, a new strategy has been considered to provide unlabeled target data with pseudo-labels [74]. To apply the proposed strategy, we create a centroid for each class of the target environment like in weighted k-means clustering and we infer the pseudo labels as follows [74]:

$$\begin{aligned} c_k^{(0)} &= \frac{\sum_{x_t \in \mathrm{X_t}} \delta_k(\hat{f}_t(x_t)) \hat{g}_t(x_t)}{\sum_{x_t \in \mathrm{X_t}} \delta_k(\hat{f}_t(x_t))}, \\ \hat{y}_t &= \underset{k}{argmin}\, D_f(\hat{g}_t(x_t), c_k^{(0)}) \end{aligned} \tag{3}$$

where $\hat{f}_t$ represents the previously learned target model and it is defined as $\hat{f}_t = \hat{g}_t(h_t)$, and D_f represents the cosine distance.

In the next step, we calculate the target centroids and the new pseudo labels based on the obtained updates.

$$
\begin{aligned}
c_k^{(1)} &= \frac{\sum_{x_t \in \mathrm{X_t}} \mathbf{1}(\hat{y}_t = k)\hat{g}_t(x_t)}{\sum_{x_t \in \mathrm{X_t}} \mathbf{1}(\hat{y}_t = k)}, \\
\hat{y}_t &= \underset{k}{argmin}\, D_f(\hat{g}_t(x_t), c_k^{(1)})
\end{aligned}
\tag{4}
$$

The centroids are generated in an unsupervised way which is why $\hat{y}_t$ are called self-supervised pseudo labels. We keep updating the centroids and the labels until we obtain good pseudo labels.

To sum up, SHOT uses the following general objective function $\mathcal{L}(g_t)$ to determine the feature encoder [74]:

$$
\mathcal{L}(g_t) = \mathcal{L}_{ent}(f_t; \mathrm{X_t}) + \mathcal{L}_{div}(f_t; \mathrm{X_t}) - \beta E_{(x_t, \hat{y}_t) \in \mathrm{X_t} \times \hat{\mathrm{Y}}_\mathrm{t}} \sum_{k=1}^{K} 1_{[K=\hat{y}_t]} \log \delta_k(f_t(x_t)) \tag{5}
$$

where $\beta > 0$ is a balancing hyper-parameter.

2.2.4 Source hypothesis network architecture and general algorithm

For the network architecture, we followed the network **DTN** introduced in [74] but we adapted it to fit sensors data as shown in Figure 2.2.

The general algorithm of the SHOT method [74] is defined in Algorithm 1.

Algorithm 1 SHOT algorithm

Require: Source hypothesis f_s, unlabeled target data, balancing hyperparameter β, number of epochs T.

Initialization: Freezing the source classifier $h_s = h_t$, and using g_s as an initialization.
for $i = 1 \rightarrow T$ **do**
    Generate self-supervised pseudo labels using Eq.(4).
    for $j = 1 \rightarrow n_{batch}$ **do**
        Obtain the pseudo labels for a given batch from target data.
        Using $\mathcal{L}(g_t)$ in Eq.(5), update g_t.
    end for
end for

2.2.5 Higher-Order Moment Matching (HoMM)

HoMM is a deep unsupervised domain adaptation method that transfers weights of the source feature extractor module to the target model, and it performs data alignment on the classifier module

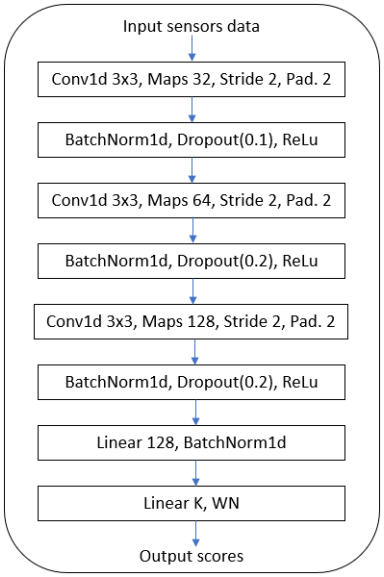

Figure 2.2: Source model network architecture

[75] as shown in Figure 2.3.

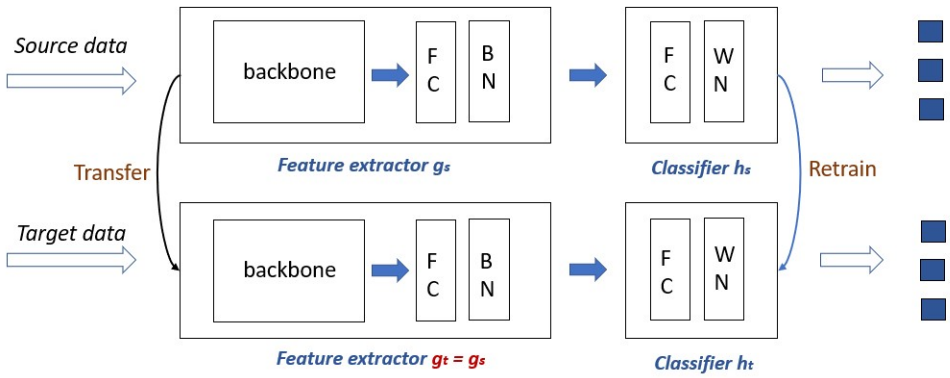

Figure 2.3: HoMM pipeline

Based on [75], a domain adaptation algorithm should include at least the source domain loss $\mathcal{L}_s$ and the domain discrepancy loss $\mathcal{L}_d$ as shown in Eq.(6).

$$\mathcal{L}(\theta|\mathrm{X_s}, \mathrm{Y_s}, \mathrm{X_t}) = \mathcal{L}_s + \lambda_d \mathcal{L}_d \tag{6}$$

$$\mathcal{L}_s = \frac{1}{n_s} \sum_{i=1}^{n_s} J(f_\theta(x_s^i)) \tag{7}$$

where J represents the cross-entropy loss function, θ is the parameter of the model that we want to learn, and λ is a Lagrange multiplier.

Most of the existing discrepancy minimization methods focus on the first and second-order statistics to perform distance minimization between sources and target domains [75]. In this work, we focus on higher-order statistics between domains as presented in the following equation [75].

$$\mathcal{L}_d = \frac{1}{L^p} \left\| \left(\frac{1}{n_s} \sum_{i=1}^{n_s} \Xi_\theta(x_s^i)^{\otimes p} - \frac{1}{n_t} \sum_{i=1}^{n_t} \Xi_\theta(x_t^i)^{\otimes p} \right) \right\|_F^2 \tag{8}$$

where $n_t = n_s =$ batch size, L is the number of hidden cells in the aligned layer, $\otimes^p (p \geq 3)$ is the p-level tensor power, and Ξ_θ represents the output of the activation function of the aligned layer.

Eq.(8) of higher-order moment matching has been generalized into reproducing kernel Hilbert spaces (RKHS) [75].

For a good data distribution alignment between source domain features and target domain features, the unsupervised domain adaptation task turns into a semi-supervised domain adaptation task where we have some target samples with a high level of confidence [75]. We use the predicted samples with a high level of confidence ($probability \geq \eta$) to create pseudo-labels, and apply discriminative clustering using the following loss function [75] by penalizing the distance of the pseudo-label samples and their centers. Discriminative clustering is encouraged for the case of unsupervised domain adaptation where we do not have labeled data on the target domain [75].

$$\mathcal{L}_{dc} = \frac{1}{n_t} \sum_{i=1}^{n_t} \left\| h_t^i - c_{\hat{y}_t^i} \right\|_2^2 \tag{9}$$

where $\hat{y}_t^i$ are the pseudo-labels of the target samples with the high confidence level, $c_{\hat{y}_t^i}$ are the

class centers of the pseudo-labels $\hat{y}_t^i$, and $h_t^i = \Xi_\theta(x_t^i)$.

Finally, we obtain the full loss function defined as the sum of all the loss functions [75].

$$\mathcal{L} = \mathcal{L}_s + \lambda_d \mathcal{L}_d + \lambda_{dc} \mathcal{L}_{dc} \tag{10}$$

where λ_d and λ_{dc} are Lagrange multipliers.

For the network architecture, we created our private architecture defined in Figure 2.4.

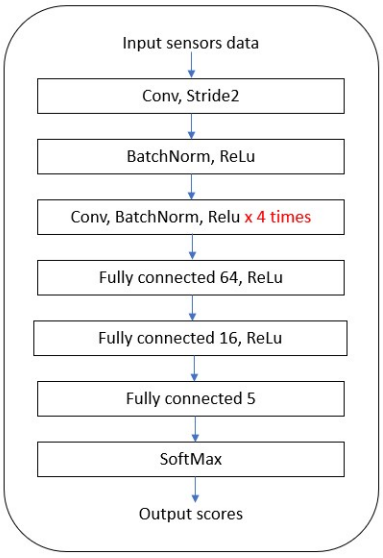

Figure 2.4: HoMM model network architecture

The network is defined as 5 sets of convolutional, batch normalization, and Relu function blocks, followed by 3 fully connected layers using Relu and SoftMax functions.

2.2.6 Source data Free Domain Adaptation (SFDA)

SFDA is a deep unsupervised domain adaptation method that does not use source-labeled data to train the target model, but instead, it uses a pre-trained model from the source domain [44]. It optimizes two losses to train the target model. The first loss uses pseudo labels of target samples using the pre-trained source model, and the second loss uses pseudo labels of target samples generated by the trainable target model periodically.

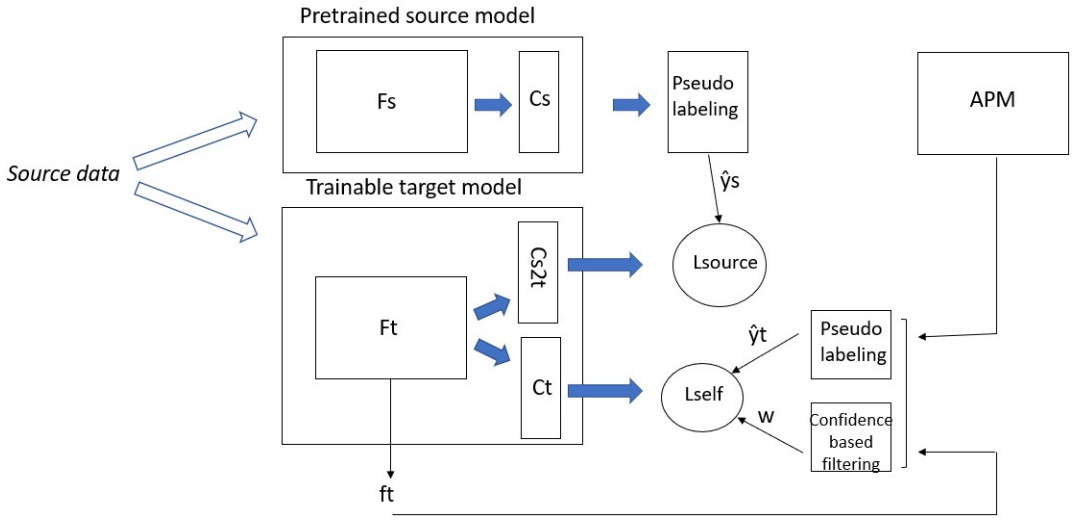

Figure 2.5: SFDA pipeline

Figure 2.5 illustrates the general architecture of the SDFA approach [44]. The considered method contains two types of models. A pre-trained source model with a feature extractor F_s and a classifier C_s. The parameters of the source model remain the same during all the algorithm steps. The target model is initialized with source model parameters at first, then it is trained in a progressive way. It contains a feature extractor F_t and two classifiers: C_{s2t} is trained using source model pseudo labels of target samples $\hat{y}_s$ and C_t is trained using target model pseudo labels $\hat{y}_t$ of target samples generated by adaptive prototype memory (APM). The obtained pseudo labels from APM $\hat{y}_t$ are further filtered using point-to-set distance-based confidence to keep only samples with a high confidence level. SDFA optimizes two loss functions to obtain the final target model. The first loss aims to prevent the target model from biasing by using source pseudo labels of target samples generated by the source pre-trained model. The second loss trains the target model using target pseudo labels generated by the trainable target model in each iteration. The general architecture of the source pre-trained model used in this approach is illustrated in Figure 2.6.

Adaptive prototype memory (APM)

Since the source pre-trained model has fixed parameters during the training of the target model. Then, all the pseudo labels of C_{s2t} of the target samples do not change during the training of the

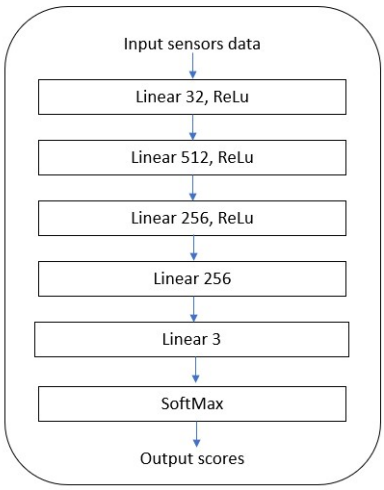

Figure 2.6: SFDA model network architecture

target model [44]. To create the final target model with good performance, we give pseudo labels to all the target samples (multiprototypes) using APM. At first, we compute the normalized entropy for each target sample as follows:

$$\mathcal{H}(x_t) = -\frac{1}{\log N_c} \sum l(x_t) \log(l(x_t)) \tag{11}$$

where $l(x_t)$ is the C_t predicted probability and N_c is the number of classes.

Then, we choose multiprototypes that are going to represent each class based on the values of the entropy. We select the lowest entropy for each class and we set the largest values among the selected entropies as a threshold. The chosen threshold will be used to choose target samples to represent each class. Indeed, only samples with values less than the threshold are considered for each class. The threshold is not static but it decreases as the training goes on, and this improves the performance of the trained target model as small values of the threshold remove the noisy samples for each class [44].

Pseudo labeling and Confidence-Based Filtering

We can calculate pseudo labels for target samples by feeding them into the target feature extractor F_t. Then, we are going to obtain embeddings f_t that we will compare to the multiprototypes of the different classes using a similarity score to make the affectation.

Using the obtained pseudo labels in APM we can train the target model. However, mistakes may happen since we are dealing with unlabeled data. So, a filtering technique has been considered based on point-to-set distance inspired by Hausdorff distance to keep only samples with a high level of confidence [44].

Optimization

The loss function that we are optimizing in this method is a combination of two loss functions from the two classifiers C_{s2t} and C_t. The source model pseudo labels $\hat{y}_s$ are used to train C_{s2t} using the following loss function that maintains knowledge from a source domain to the target model [44].

$$\mathcal{L}_{source}(D_t) = -E_{x_t} \sum_{c=1}^{N_c} 1_{c=\hat{y}_s} \log(\sigma(C_{s2t}(F_t(x_t))) \tag{12}$$

where 1 is an indicator function and σ is a sofmax function.

The pseudo labels $\hat{y}_t$ generated by APM are used to train the C_t classifier using the following loss function:

$$\mathcal{L}_{self}(D_t) = -E_{x_t} \sum_{c=1}^{N_c} w(x_t) 1_{c=\hat{y}_s} \log(\sigma(C_{s2t}(F_t(x_t))) \tag{13}$$

where $w(.)$ is the confidence score of the confident samples.

The total loss function is defined as a weighted sum of the two loss functions.

$$\mathcal{L}_{total}(D_t) = (1-\alpha)\mathcal{L}_{source}(D_t) + \alpha\mathcal{L}_{self}(D_t) \tag{14}$$

where α is a hyperparameter that balances the two loss functions.

2.3 Experimental setup and results

2.3.1 Datasets

For the AR task, we used public datasets of Washington State University (WSU) Center for Advanced Studies in Adaptive Systems (CASAS) [100]. The datasets are single-resident apartment data. They have been collected using several types of sensors (Ambient PIR motion sensors, door/temperature sensors, and light switch sensors). The considered activities are cooking breakfast, cooking lunch, cooking dinner, watching TV, and toileting.

For the OE task, we used our private datasets [13, 14] that are collected from two similar offices (H355 and H358) located at Grenoble Institute of Technology. The data has been collected using many smart building sensors like power consumption sensors, CO2 concentration sensors, humidity sensors, temperature sensors, door and window contact sensors, and acoustic pressure sensors. The considered levels of occupants are 0, 1, and 2.

For both tasks AR and OE, we extracted balanced and unbalanced datasets to test the efficiency of the considered method for different scenarios.

2.3.2 Metrics

For the evaluation, we used 2 metrics depending on the scenario that we are examining. We used accuracy as a metric in the case where we are dealing with balanced datasets.

$$\textbf{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{15}$$

We used the F_1 score as a metric in the scenario where we are dealing with unbalanced datasets.

$$\textbf{Precision (Pr)} = \frac{1}{N} \sum_{i=1}^{N} \frac{TP_i}{TP_i + FP_i} \tag{16}$$

$$\textbf{Recall (Rc)} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FN_i} \tag{17}$$

$$\textbf{F-score } (F_1) = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{18}$$

where TP is true positive, TN is true negative, FP is false positive, and FN is false negative.

2.3.3 Experimental results

AR scores

For unbalanced datasets with 5 activities, SHOT-IM gives 62.71%, and full SHOT improved with self-supervised pseudo-labeling gives 58.74% weighted F_1 scores which are good achievements. However, HoMM gives a lower weighted F_1 score (almost 38%) compared to the previous methods. Indeed, it is less than the supervised method, SHOT-IM, and SHOT pseudo-labeling with 51.63%, 31.34%, and 27.37%, respectively. The obtained results are expected due to the complexity of the task (five activities), the unsupervised manner, and the highly unbalanced datasets. Pseudo-labeling is misleading in the current case of predicting 5 activity levels with a highly unbalanced data set. That is why the obtained results for the full SHOT improved with pseudo-labeling are lower than SHOT-IM. Table 2.1 illustrates all the presented results.

Table 2.1: AR scores for 5 labels in an unbalanced data

Method (Source $\longrightarrow$ Target)	$\mathrm{F}_1 score(\%)$
SHOT-IM	62.71
SHOT-Pseudo-labeling	58.74
HoMM	38.45
Target Supervised Model	90.08

Table 2.2 gives the detailed F1-scores for the different activities in the case of the unbalanced data set. Indeed, the correctness of the prediction of the 5 activities differs between the three methods and inside each method as well. Activities with more numbers of samples (1st activity) are

always well classified. In addition, activities with few instances can be well classified by the proposed methods such as the case of the 3rd and the 5th activities in the SHOT-IM method. However, in some cases, categories with a small number of instances while training cannot be well classified by the model and provide lower results such as the 5th category result (0.0%) in the HoMM method.

Table 2.2: AR scores for 5 labels in an unbalanced data: labels scores

Method—Activity	1st	2nd	3rd	4th	5th
SHOT-IM	88.88	20.66	92.99	15.53	65.49
SHOT-Pseudo-labeling	90.69	31.52	92.99	39.74	32.18
HoMM	72.33	23.301	18.59	40.46	0.0

When we decreased the complexity of the task on only 3 activities, we have seen a huge improvement in the performance of all the methods with almost 30% of f1-score ($+30.60\%$ for SHOT-IM, $+28.96\%$ for full SHOT, and $+31.73\%$ for HoMM). Even though the data sets are unbalanced, SHOT-IM and full SHOT are giving very good weighted F_1 scores (93.31% and 87.70%) that are comparable to supervised approaches results (95.58%) and better than HoMM results (70.18%) for this case. SHOT-IM is better than full SHOT (pseudo-labeling) for this case and this makes us deduce that self-supervised pseudo-labeling created for the unlabeled target data is misleading for the case of AR, as shown in Table 2.3. In other words, the pseudo-labels created using the Full SHOT method are not so accurate for the considered case. SFDA has also given excellent results for the considered task with 92.00% of F1 score which is comparable to supervised learning methods scores.

Table 2.3: AR scores for 3 labels in an unbalanced data

Method (Source $\longrightarrow$ Target)	$\mathrm{F}_1 score(\%)$
SHOT-IM	93.31
SHOT-Pseudo-labeling	87.70
HoMM	70.18
SFDA	92.00
Target Supervised Model	95.58

By decreasing the number of activities, we fixed the issue related to the prediction correctness of each label. Indeed, as shown in Table 2.4, all the activities are classified with a high level of correctness and the models are able to capture information related to all labels better than in

the previous task (5 activities) where some labels have been ignored by the models. The data distribution of the AR task with a huge number of features was one of the reasons that made it difficult for the model to capture all the information of the data. Also, the similarities between some tasks for 5 activities such as preparing lunch and preparing dinner is a reason for the model to get confused while predicting. Indeed, this is an issue that we will work on in the following research trying to obtain DA models that provide good performance for higher levels of tasks. The minimum prediction score that we obtained is with the HoMM method for the 3rd category with a 52.60% f1-score which is acceptable. We can conclude that as the complexity of the task goes up, and that's what we are interested in to obtain powerful models, the accuracies of the models start to decrease, and vice versa. Indeed, these findings are expected since the more complicated the task is the more difficult for the model to predict it correctly.

Table 2.4: AR scores for 3 labels in an unbalanced data: labels scores

Method	1st activity	2nd activity	3rd activity
SHOT-IM	95.26	92.13	90.32
SHOT-Pseudo-labeling	88.84	84.83	88.54
HoMM	83.08	59.21	52.60
SFDA	95.15	93.89	85.67

For balanced data sets, evaluating SHOT on AR task with 5 activities gives significant results for SHOT methods compared to the unbalanced data set results. As shown in Table 2.5, SHOT improved with self-supervised pseudo-labeling outperforms SHOT-IM by almost 3%. Full SHOT provides good results (68.00%) in an unsupervised manner that are comparable to supervised model results. In this case, the created pseudo labels for the unsupervised task have helped to increase the performance of the target model and provide good results compared to SHOT-IM. However, HoMM with the increase of activities number (complexity) for the case of balanced data set did perform poorly with 38.20% weighted F1-score. HoMM is unable to provide good results with the increase of complexity (more than 3 categories) of the target task for both balanced and unbalanced AR data sets.

By reducing the number of classes for AR, we have seen a significant improvement in SHOT performance (88.80%) and HoMM (61.25%) as shown in Table 2.6. The removed activities (Cooking

Table 2.5: AR accuracies for 5 labels in a balanced data

Method (Source $\longrightarrow$ Target)	Accuracy (%)
SHOT-IM	65.40
SHOT-Pseudo-labeling	68.00
HoMM	38.20
Target Supervised Model	80.80

lunch and cooking dinner) have been chosen wisely. Indeed, these 2 activities have a lot of similarities when it comes to the activated sensors while performing these tasks. It is clear that the activated sensors while 'watching TV' or 'toileting' will be most of them different than the tasks related to the kitchen (preparing breakfast, lunch, and dinner). That is why we choose to keep just one task related to the kitchen (preparing breakfast) and remove the comparable tasks to make it easy for the model to better make a difference between the labels. The obtained results are expected since we have reduced the complexity of the task (AR), and the increase in the performance is not influenced by the type of activities that we have removed but influenced only by the task complexity in this case. The scores exceed classical supervised methods performances with 0.8% for the case of SHOT methods. HoMM with reduced target task complexity (less than 3 labels) for both balanced and unbalanced data sets, provides good prediction results that can be considered for further research works. SFDA, by progressively updating the target model in a self-learning manner, has given the best scores and has exceeded all the considered methods with 96.67% of accuracy.

Table 2.6: AR accuracies for 3 labels in a balanced data

Method (Source $\longrightarrow$ Target)	Accuracy (%)
SHOT-IM	88.80
SHOT-Pseudo-labeling	78.80
HoMM	61.25
SFDA	96.67
Target Supervised Model	88.00

Figure 2.7 gives all the obtained scores for balanced (accuracy) and unbalanced (F1-score) datasets compared to their references (target supervised models).

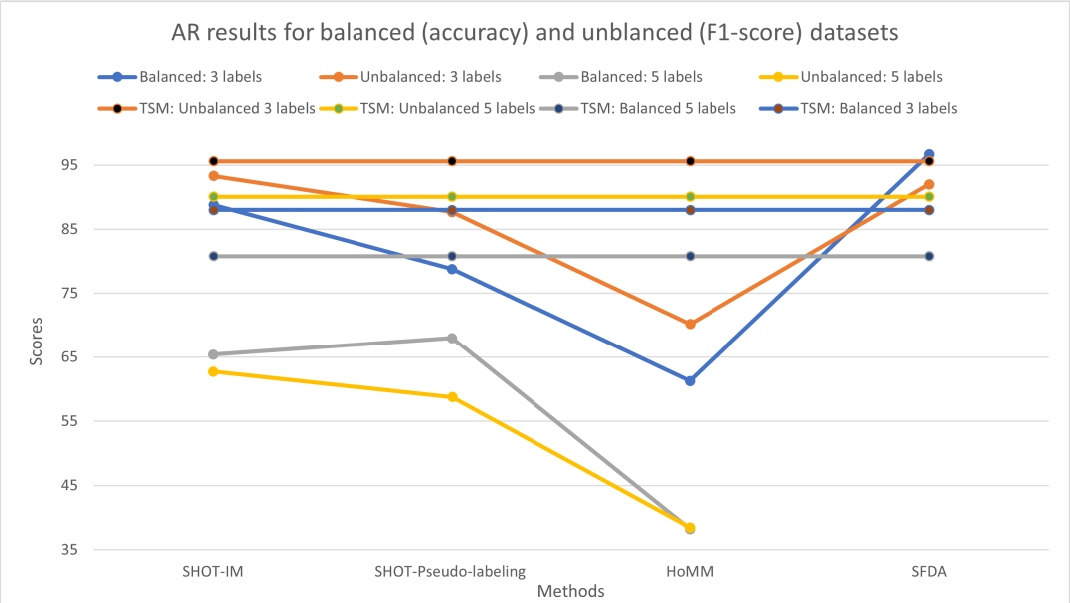

Figure 2.7: AR results for balanced (accuracy) and unbalanced (F1-score) datasets

OE scores

OE data sets that have been tested on the considered methods, have been also used before to test several supervised basic transfer learning methods [25] such as the PCA-like method and PCA-SMOTE method. That is why we included the previously obtained results to make a comparison between basic and deep transfer learning methods.

For unbalanced data sets, full SHOT and SHOT-IM gave great results for the task of OE with 3 levels of occupants. The self-supervised pseudo-labeling method did give the same performance as the SHOT-IM method. Indeed, the created pseudo-labels, in this case, did not improve the results of the basic SHOT method. SFDA gave great results that are less than SHOT methods, but greater than HoMM results with almost 7% of F1 score. Indeed, progressively updating the target model in a self-learning manner has proved its efficiency in the case of the SFDA method. HoMM did give a good result (69.66%), but it is less than the SHOT method results. The results of SHOT methods are better than classical supervised method results with 0.5%, and they are comparable to the supervised PCA-like method [25]. Indeed, these results prove the robustness of the considered methods in the considered unsupervised case. Table 2.7 presents the obtained results for OE with 3 labels.

Table 2.7: OE scores for 3 labels in an unbalanced data

Method (Source $\longrightarrow$ Target)	$F_1 score(\%)$
SHOT-IM	82.57
SHOT-Pseudo-labeling	82.57
HoMM	69.69
SFDA	76.92
Target Supervised Model	82.07
Supervised PCA-like method	85.96

Table 2.8 gives the detailed scores for the different OE levels. Indeed, we can notice some differences in the prediction correctness of the different labels for each method. The difference in scores is due to the unbalanced data sets that we are using. Labels with a huge number of samples get always the best prediction scores. For the considered SHOT methods, the results are good for all the levels of occupancy even though the data sets are unbalanced.

Table 2.8: OE scores for 3 labels in an unbalanced data: labels scores

Method	1st label	2nd label	3rd label
SHOT-IM	92.90	53.85	73.85
SHOT-Pseudo-labeling	92.90	53.85	73.85
HoMM	85.10	32.94	50.76
SFDA	88.49	54.55	12.44
Supervised PCA-like method	92.88	61.01	73.33

By decreasing the complexity of the OE task, we notice an improvement in all the methods, and this is expected as we have explained before. In the case of the OE task, we were able to decrease the occupancy levels without removing instances. Indeed, we have merged levels with high levels of occupancy because it is clear that as the number of occupancy levels increases the values of the activated sensors (such as CO2 concentration, and motion counting) will increase as well. Full SHOT and SHOT-IM keep giving great results (87.62% and 87.63%, respectively) that are close to supervised methods score (88.65%) even though they are unsupervised approaches. Full SHOT enhanced with self-supervised pseudo-labeling is slightly better than SHOT-IM in this case as shown in Table 2.9. HoMM with 2 levels of occupancy provides a better result than SHOT methods (83.23%). Indeed, we can notice that HoMM performs pretty well with the decrease in the target task complexity. All the obtained results for the considered methods are almost the same as the supervised PCA-like method [25] (almost 1.5% of score difference) which is a good achievement

for unsupervised learning methods. Even though, we have noticed an increase in the performance of the SFDA method (+6.41%), it is a bit far from the supervised PCA-like method compared to the rest of the methods. We can conclude that HoMM has benefited from the complexity decrease of the task to improve its performance dramatically and has exceeded all the considered methods.

Table 2.9: OE P/A scores for 2 labels in an unbalanced data

Method (Source $\longrightarrow$ Target)	$\mathrm{F}_1 score(\%)$
SHOT-IM	87.62
SHOT-Pseudo-labeling	87.63
HoMM	88.23
SFDA	83.33
Target Supervised Model	88.65
Supervised PCA-like method	89.88

Table 2.10 illustrates all the scores of the two levels of occupancy for the different considered methods. Indeed, it is clear that the first label has more samples than the second label since all the methods have classified the first label with a high level of correctness. Even though the results between the first and second labels are different, both of labels have very good prediction results.

Table 2.10: OE P/A scores for 2 labels in an unbalanced data: labels scores

Method	1st label	2nd label
SHOT-IM	91.63	73.97
SHOT-Pseudo-labeling	91.63	73.97
HoMM	91.87	69.27
SFDA	91.82	70.23
Supervised PCA-SMOTE method	93.28	79.54

For balanced datasets, evaluating OE task with 3 levels of occupancy gives good performance for SHOT-IM, SFDA, and HoMM (73.80%, 68.00%, and 63.75%, respectively). The obtained results are comparable to classical supervised method results, and they show an important increase compared to SHOT-Pseudo-labeling (6.55% for HoMM, 9.80% for SFDA and 16.60% for SHOT-IM, of accuracies increase). Full SHOT enhanced with self-supervised pseudo-labeling did not perform well compared to SHOT-IM. Indeed, this may be because the considered method requires a big number of samples to learn the target feature encoder. Also, in the used datasets, all the occupancy levels greater than 2 are considered as a 2-level of occupancy and this can affect the created pseudo-labels. Table 2.11 illustrates all the presented scores. In previous works, we tested

a supervised transfer learning method for the balanced data set called PCA-SMOTE method [25], and it has given 90.93% of accuracy. SHOT-IM, as an unsupervised transfer learning method, with 73.80% of accuracy is considered a good achievement compared to the supervised transfer learning method that we have tested before.

Table 2.11: OE accuracies for 3 labels in a balanced data

Method (Source $\longrightarrow$ Target)	Accuracy ($\%$)
SHOT-IM	73.80
SHOT-Pseudo-labeling	57.20
HoMM	63.75
SFDA	68.00
Target Supervised Model	83.00
Supervised PCA-SMOTE method	90.93

When we considered the task of Presence/Absence of Occupancy, we reduced the number of occupancy levels and we obtained better performances as shown in Table 2.12. The improvement in scores is expected due to the complexity decrease. In contrast to the case of the unbalanced dataset for the OE task when HoMM exceeded all the methods by reducing the number of occupancy levels, SFDA, in the current case (balanced dataset), has exceeded all the proposed methods and has given a score 90.00% which is even better than supervised machine learning methods. SHOT enhanced with self-supervised pseudo-labeling (86.40%) gives the same results as SHOT-IM (86.40%) and outperforms HoMM (82.50%). It gives also results so close to classical supervised method scores (88.80%) and supervised PCA-SMOTE method [25] even though it is an unsupervised method. The performances of all the methods are so close and very promising. We can understand that label distribution (balanced or unbalanced datasets) has an impact on the behavior of the methods (HoMM and SFDA).

We can notice that SHOT pseudo-labeling results for the unbalanced data sets are better than its results for the balanced data sets. This can be explained by the usefulness of label weights in the data set to create the pseudo labels with a high level of confidence. Thus, we obtain better results in the model prediction.

Figure 2.8 gives all the obtained scores for balanced (accuracy) and unbalanced (F1-score) datasets compared to their references (target supervised models).

Table 2.12: OE P/A accuracies for 2 labels in a balanced data

Method (Source $\longrightarrow$ Target)	Accuracy (%)
SHOT-IM	86.40
SHOT-Pseudo-labeling	86.40
HoMM	82.50
SFDA	90.00
Target Supervised Model	88.80
Supervised PCA-like method	90.27

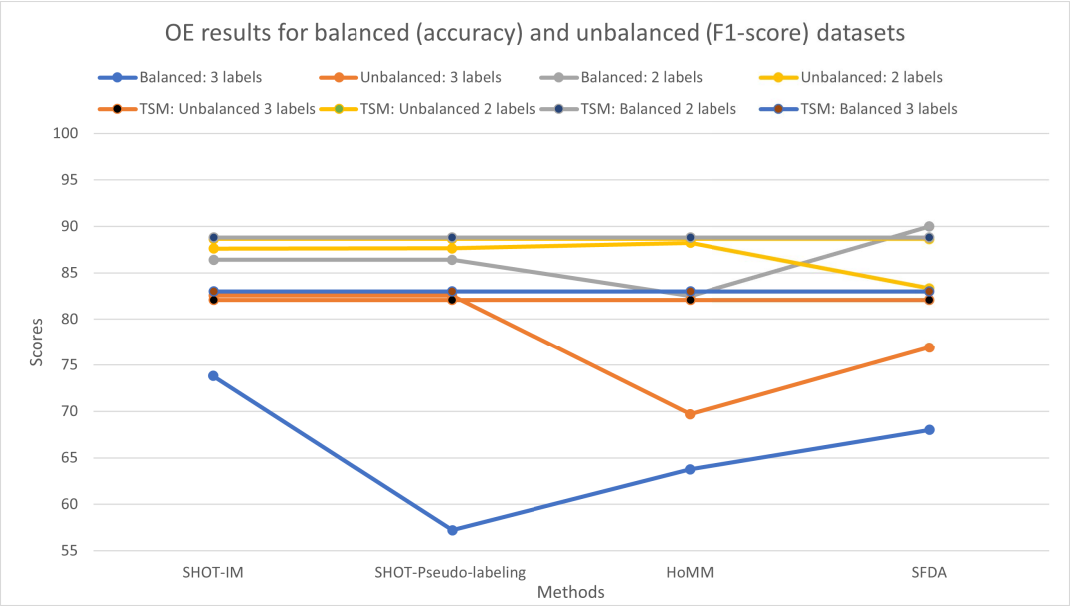

Figure 2.8: OE results for balanced (accuracy) and unbalanced (F1-score) datasets

Chapter 3

Unsupervised Domain Adaptation with Source Data for Estimating Occupancy and Recognizing Activities in Smart Buildings

3.1 Introduction

In the last few years, machine learning [101, 102, 103, 45] gained significant interest in smart buildings [104] due to the increase of data produced by sensors and IoT devices [105, 106, 107, 25, 26]. In particular, researchers have considered two vital smart building tasks which are occupancy estimation (OE) and activities recognition (AR) due to the several advantages generated by them [25, 26, 108, 109, 13, 18, 110, 111, 112, 113, 114]. Indeed, OE can be helpful for energy management by accurately estimating occupancy in the different areas of a building. Thus, managers can optimally distribute energy over all the buildings [25, 26, 115]. Also, detecting and estimating occupancy in a smart building can help identify unauthorized access to the building, providing more security for the inhabitants [31, 32, 33]. Recognizing activities allows managers of buildings to optimize HVAC (heating, ventilation, and air conditioning) systems by understanding occupants' behavior

in each room, providing inhabitants with more comfort [34, 35]. AR and OE machine learning models [13, 116, 117] trained on a particular smart building environment may not generalize well on unknown sensor domains due to several reasons such as feature representation difference (type of sensors) and data distribution shift (sensors locations). Creating a machine learning model for each smart building environment is a costly and time-consuming task due to data scarcity [25, 38]. For that reason, domain adaptation [43, 44, 45, 46, 47, 48, 49, 50] can be used as a solution to solve these problems. Indeed, domain adaptation solves the problem of data requirements by mitigating data distribution shifts between environments, allowing knowledge transfer across related smart building domains [25, 118]. Sharing knowledge between domains helps increase the performance of models as well as their robustness. Indeed, creating models that generalize well for related domains decreases sensitivity to domain shift, providing robust methods.

In this research, we introduce several unsupervised domain adaptation methods where we have access to only labeled data from the source domain and unlabeled data from the target domain. The goal is to mitigate data distribution shifts between source and target domains so that we can transfer knowledge across the environments, and build generalized, accurate, and robust models. We also test the robustness of all the considered methods using a data poisoning technique [80]. Indeed, the data poisoning technique used in this research adds mislabeled data samples to the training and compares the behavior of the model before and after the data poisoning [80]. The considered unsupervised domain adaptation approaches are Virtual Adversarial Domain Adaptation (VADA) method [77], Sliced Wasserstein Discrepancy (SWD) method [78], Hard Adaptive Feature Norm (HAFN) [79], Stepwise Adaptive Feature Norm (SAFN) [79], and SAFN with entropy minimization (SAFN+ENTM) [79]. Firstly, VADA [77] is a deep UDA technique that includes domain adversarial training and a penalty term to punish the violation of the cluster assumption. Domain adversarial training is done using 3 components: a feature extractor, a domain classifier, and a task-specific classifier. The goal of the feature extractor and the task-specific classifier is to predict the target task with high accuracy as well as fooling the domain classifier in distinguishing between data representations coming from different domains. Thus, the feature extractor provides data representations that generalize for different domains. Secondly, the SWD method [78] is an UDA technique that uses task-specific decision boundaries of machine learning models from different domains as

well as the Wasserstein metric to make domain adaptation. The Wasserstein distance is used to measure the discrepancy between the probability distributions defined by the models [78]. Lastly, the 3 Adaptive Feature Norm (AFN): Hard Adaptive Feature Norm (HAFN), Stepwise Adaptive Feature Norm (SAFN), and SAFN with entropy minimization are deep UDA that adapt the feature norms of the models to allow information transferability between domains [79]. HAFN mitigates the domains discrepancy by limiting the expected feature norms of the two domains to a fixed scalar. However, SAFN allows the increase of feature norms in a progressive way for each sample across domains [79], providing an increase in knowledge transfer compared to HAFN. Also, the SAFN+ENTM is improved compared to the SAFN with the entropy minimization term while training [79] which further increases the transfer gains compared to SAFN. This research has several contributions. Indeed, we have adapted all the considered methods for smart buildings data (AR and OE) so that the approaches work without problems with sensor data, providing good performances. Also, we provided new deep neural network architectures for the feature extractors and classifiers in most of the considered methods. Moreover, we evaluated these methods on AR and OE datasets and showed their strengths and weaknesses by analyzing the findings and by applying a data poisoning technique.

The rest of the chapter is organized into 2 sections. In section 2, we introduce and explain the considered approaches and, in section 3, we discuss the research findings.

3.2 Proposed methods

3.2.1 Virtual Adversarial Domain Adaptation (VADA)

Virtual Adversarial Domain Adaptation (VADA) [77] is an unsupervised deep domain adaptation method that uses labeled source data and unlabeled target data to align data distributions between source and target domains in order to enhance target model performance. VADA is a mixture of adversarial domain adaptation that trains a source model to fit a target domain by reducing domain shift, and a conditional entropy loss added to the optimization function that punishes the cluster assumption violation which assumes that samples with close data distributions belong to the same class [77].

Domain adversarial training is defined by solving the following optimization function [77]:

$$\min_{\theta} \mathcal{L}_y(\theta; \mathcal{D}_s) + \lambda_d \mathcal{L}_d(\theta; \mathcal{D}_s, \mathcal{D}_t) \tag{19}$$

where λ_d is a weighting parameter, θ is a hyperparameter, y is the labels vector, $\mathcal{D}_s$ and $\mathcal{D}_t$ are joint distributions for source and target input data x, $\mathcal{L}_y$ and $\mathcal{L}_d$ are loss functions.

$\mathcal{L}_y$ is defined as the cross-entropy loss, and $\mathcal{L}_d$ is a loss function that allows the creation of a feature extractor f that minimizes the distance between the source feature space $f(X_s)$ and the target feature space $f(X_t)$ with X is a marginal data distribution and f is an embedding function [77].

$$\mathcal{L}_y(\theta; \mathcal{D}_s) = E_{x,y\sim\mathcal{D}_s}[y^T \ln h_\theta(x)] \tag{20}$$

$$\mathcal{L}_d(\theta; \mathcal{D}_s, \mathcal{D}_t) = \sup_{D} \{E_{x\sim\mathcal{D}_s}[\ln D(f_\theta(x))] + E_{x\sim\mathcal{D}_t}[\ln(1 - D(f_\theta(x)))]\} \tag{21}$$

where h is a classifier defined with the hyperparameter θ, and D is a discriminator. Domain adversarial training improves the performance of the target task by making the source model invariant to the domain shift. Indeed, the discriminator D tries to predict the domain of samples, and the task classifier tries to confuse the discriminator by creating feature representations that are not informative about the origin of the samples.

The feature extractor, the classifier as well as the discriminator are defined as a series of convolutional and normalization layers as follows.

The feature extractor is created using 2 blocks of layers, each layer is composed of 3 blocks of convolutional, batch normalization, and LeakyReLu activation function, as well as a max-pooling layer and a Gaussian noise as presented in Figure 3.1.

The classifier is composed of 3 blocks of convolutional, batch normalization, and LeakyReLu activation function layers, as well as an adaptive average pooling layer and a convolutional layer as illustrated in Figure 3.2.

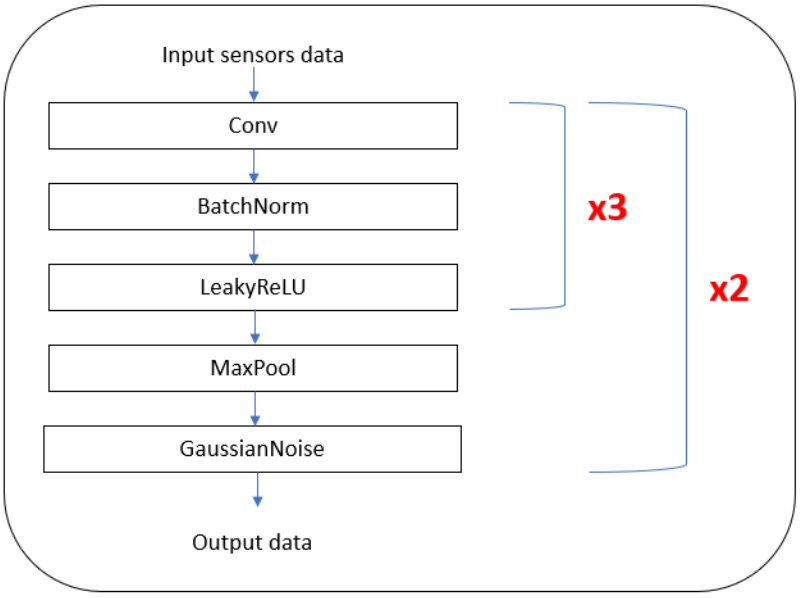

Figure 3.1: Feature extractor architecture.

The discriminator is developed using 2 Linear layers and the Relu activation function as defined in Figure 3.3. Adversarial domain adaptation may fail to generalize for target domains in the case where the feature extractor f has a high capacity or the source-target tasks are different [77]. In our current case, we may encounter the problem of high feature extractor capacity. In other words, the feature extractor is able to capture deep and detailed information of the source data, leading to overfitting to the source domain [77]. To avoid this problem, we add constraints to the loss function to ensure that domain adaptation is being enhanced as defined in Eq. (22).

$$\min_{\theta} \{\mathcal{L}_y(\theta; \mathcal{D}_s) + \lambda_d \mathcal{L}_d(\theta; \mathcal{D}_s, \mathcal{D}_t) + \lambda_s \mathcal{L}_v(\theta; \mathcal{D}_s) + \lambda_t [\mathcal{L}_v(\theta; \mathcal{D}_t) + \mathcal{L}_c(\theta; \mathcal{D}_t)]\} \tag{22}$$

where λ_s and λ_t are hyperparameters, $\mathcal{L}_c$ and $\mathcal{L}_v$ are defined in Eq. (23) and Eq. (24), respectively.

$$\mathcal{L}_c(\theta; \mathcal{D}_t) = -\mathbb{E}_{x \sim \mathcal{D}_t} \left[h_\theta(x)^T \ln h_\theta(x) \right] \tag{23}$$

$\mathcal{L}_c$ is a loss function that applies the cluster assumption. Indeed, it states that every input data is

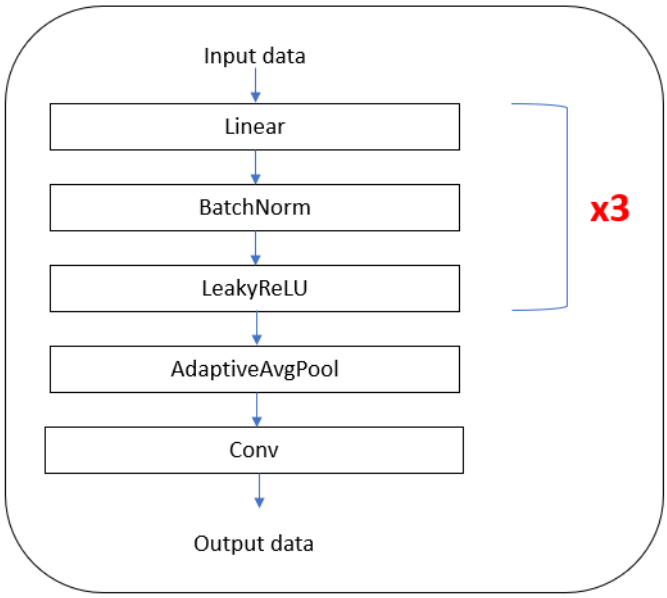

Figure 3.2: Task-classifier architecture

a set of clusters and all samples of each cluster are related to the same class [77]. Thus, the model will push the decision boundaries of each class away from dense areas of unlabeled target data.

$$\mathcal{L}_v(\theta; \mathcal{D}) = \mathbb{E}_{x \sim \mathcal{D}} \left[\max_{\|r\| \leq \epsilon} D_{KL} \left(h_\theta(x) \| h_\theta(x + r) \right) \right] \tag{24}$$

where ϵ is a generalization error function, r is a parameter, and D_{KL} is Kullback–Leibler divergence. $\mathcal{L}_v$ is a loss function that applies the locally-Lipschitz constraint [77] to further enhance the decision boundaries placement and to prevent it to be placed near training samples.

3.2.2 Sliced Wasserstein Discrepancy (SWD)

Sliced Wasserstein Discrepancy (SWD) [78] is a deep learning method that mitigates features distributions shift between source and target domains using labeled source data and unlabeled target data (unsupervised learning). It aims to enhance and improve the performance of a target domain where data is scarce by sharing knowledge from a source domain where labeled data availability is high. SWD is developed using Wasserstein discrepancy distance and task-specific decision boundaries. At first, SWD trains a feature extractor, and two classifiers to fit source data. Then, it freezes the feature extractor and maximizes the discrepancy between the output of the two classifiers using

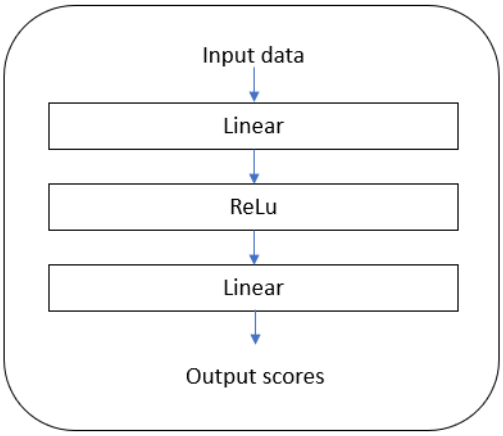

Figure 3.3: Discriminator architecture.

the Wasserstein Discrepancy distance to identify target samples with distributions different from the source data distribution. Finally, it freezes the classifiers' weights, and it updates the feature extractor to minimize the discovered discrepancy, providing domain adaptation between source and target data [78].

To establish unsupervised domain adaptation using SWD, we follow three steps. Firstly, we train a feature extractor G as well as two classifiers C_1 and C_2 on the source data to predict the labels of the samples accurately by solving Eq. (25).

$$\min_{G,C_1,C_2} \mathcal{L}_s(X_s, Y_s) \tag{25}$$

where X_s, Y_s is the source domain, and $\mathcal{L}_s$ is a typical loss function such as cross-entropy [78]. Secondly, we freeze the feature extractor, and we train the two classifiers on the target samples aiming to maximize the difference between the outputs of the two classifiers. Thus, we can detect the target instances that are not supported by the source model. In other words, we identify target samples with data distribution different from the source data distribution [78]. Eq. (26) illustrates the optimization function to solve.

$$\min_{C_1,C_2} \mathcal{L}_s(X_s, Y_s) - \mathcal{L}_{DIS}(X_t) \tag{26}$$

where X_t is the target set, $\mathcal{L}_{DIS}$ is the discrepancy loss, and $\mathcal{L}_s$ is the source loss function that

aims to keep information from source domain [78]. Finally, we mitigate the discrepancy between the source and target domains by training the feature generator on the target data X_t and by freezing the two classifiers C_1 and C_2 as shown in Eq. (27).

$$\min_{G} \mathcal{L}_{DIS}(X_t) \tag{27}$$

The neural network architecture for the feature generator G as well as for the classifiers C_1 and C_2 are given in Figure 3.4 and Figure 3.5.

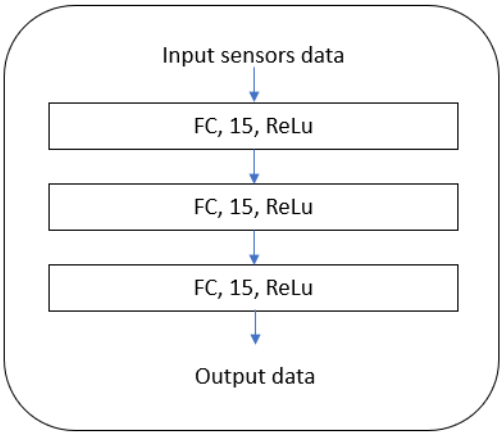

Figure 3.4: Feature extractor architecture.

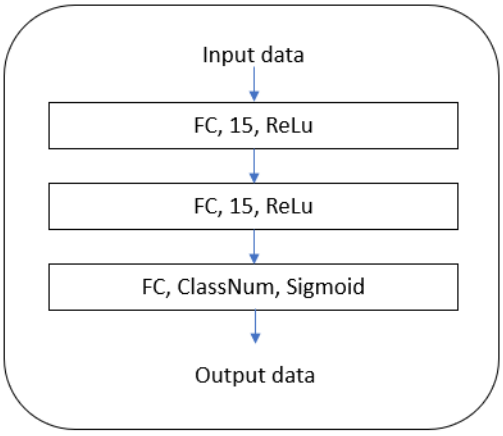

Figure 3.5: Classifiers architecture.

The architectures of the feature extractor G as well as the two classifiers C_1 and C_2 are created using fully connected layers with ReLu activation functions. For the classifiers, they provide

probability values for each class in the last layer using the Sigmoid activation function.

In this research, the Wasserstein distance has been used to enhance the data alignment of source and target domain [78]. It has received huge attention since it is better than several distance metrics such as Jensen-Shannon divergence. Indeed, it takes into consideration the geometry of data distribution space while calculating the distance between sample distributions [78]. For the classification task, the distance is defined in Eq. (28).

$$SWD(\mu, \nu) = \sum_{m=1}^{M} \sum_{i=1}^{N} c(\mathcal{R}_{\theta_m} \mu_{\alpha_i}, \mathcal{R}_{\theta_m} \nu_{\beta_i}) \tag{28}$$

where $c : \Omega \times \Omega \rightarrow \mathbb{R}^+$ is a quadratic geodesic metric that calculates the shortest length between two projections, Ω is a probability space, θ is a uniform measure, μ and ν are probabilities, α and β are permutations, $\mathcal{R}_\theta$ are projections of N samples, and M is the number of uniform measures θ [78].

3.2.3 Adaptive Feature Norm (AFN)

In this research, we consider three deep unsupervised domain adaptation approaches called Hard Adaptive Feature Norm (HAFN), Stepwise Adaptive Feature Norm (SAFN), and SAFN with entropy minimization (SAFN+ENTM). All these methods use labeled source data and unlabelled target data to mitigate domain shifts across both environments by aligning and enlarging feature norms between domains as well as adding new penalty terms to the main objective function [79]. The AFN framework is constructed using a feature generator G and task-specific classifier F. The classifier F is a combination of l fully-connected layers. We calculate label probabilities using the last layer F_y by applying a Softmax activation function [79]. The first $l - 1$ layers called F_f, give as output the bottleneck feature embeddings $\mathbf{f}$ which are specific for each domain and are difficult to transfer across different domains [79]. The architecture of the feature extractor module as well as the classifier are given in Figure 3.6 and Figure 3.7. G is composed of three fully-connected (FC) layers with ReLu activation function, and F is a sequence of a FC-ReLu-Batchnorm-ReLu-Dropout-FC-Softmax layers.

In this research, we aim, using only source labeled data and unlabeled target data, to create a

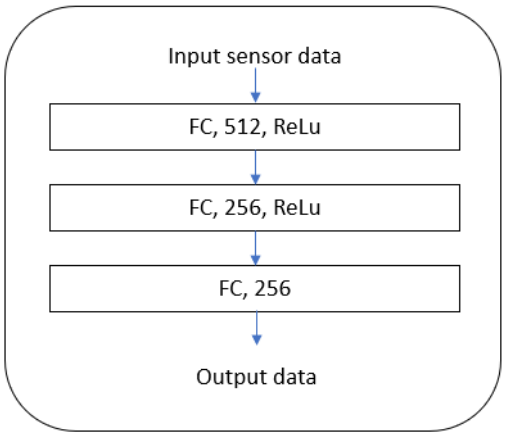

Figure 3.6: Feature extractor architecture for AFN.

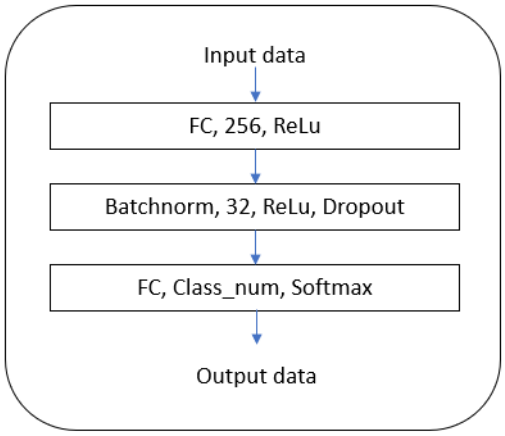

Figure 3.7: Classifier architecture for AFN.

domain adaptation method that will calculate features $\mathbf{f} = F_f(G(\cdot))$ that are transferable across different but related domains. Thus, we considered constraining feature norms of source and target data to a given scalar using HAFN. Then, we tested also SAFN that enlarges feature norms of source and target samples progressively. Finally, we evaluated SAFN with entropy minimization.

Hard Adaptive Feature Norm (HAFN)

To mitigate data distribution shift using HAFN, we solve the objective function defined in Eq. (29) [79]:

$$C1(\theta_g, \theta_f, \theta_y) = \frac{1}{n_s} \sum_{(x_i, y_i) \in D_s} L_y(x_i, y_i) + \lambda(L_d(\frac{1}{n_s} \sum_{x_i \in D_s} h(x_i), R) + L_d(\frac{1}{n_t} \sum_{x_i \in D_t} h(x_i), R)) \tag{29}$$

where $(\theta_g, \theta_f, \theta_y)$ are the parameters of G, F_f, and F_y, respectively. (n_s, n_t) are the numbers of samples in the source domain $D_s = \{(x_i^s, y_i^s)\}_{i=1}^{n_s}$, and the target domain $D_t = \{x_i^t\}_{i=1}^{n_t}$. $\mathcal{H}$ represents all the functions $h(x) = (\| \cdot \|_2 \circ F_f \circ G)(x)$ defined using the L_2-norm operator. λ is a parameter, and R is a scalar that restricts the mean feature norms of the two data distributions to mitigate covariate shifts. L_d, defined as an L_2-distance, is a penalty added to the objective function to mitigate feature-norm differences [79]. L_y is the source loss function to learn source feature distributions defined in Eq. (30).

$$L_y(x_i^s, y_i^s; \theta_g, \theta_f, \theta_y) = -\sum_{k=1}^{|\mathcal{C}_s|} \mathbb{1}_{[k=y_i^s]} \log p_k \tag{30}$$

where $\mathcal{C}_s$ is the source labels space and p is the output of the Softmax activation function in the classifier F. HAFN with small mean feature norms has given acceptable performances. However, with large values of R, we still obtain better results. Thus, we want to explore large values of R aiming to increase model performances using SAFN [79].

Stepwise Adaptive Feature Norm (SAFN)

In this section, we consider an improved version of HAFN where we aim to explore larger values of mean feature norms progressively while learning the task-specific features by solving the objective function in Eq. (31) [79].

$$C_2(\theta_g, \theta_f, \theta_y) = \frac{1}{n_s} \sum_{(x_i, y_i) \in D_s} L_y(x_i, y_i) + \frac{\lambda}{n_s + n_t} \sum_{x_i \in D_s \cup D_t} L_d(h(x_i; \theta_0) + \Delta r, h(x_i; \theta)) \tag{31}$$

where θ is the updating model parameters in the last iteration defined as $\theta = \theta_g \cup \theta_f$, and θ_0 is the updated parameters model in the current iteration [79]. Δr is a positive value that controls

the enlargement of mean feature-norms, defined as the difference between the current and the last mean feature-norm values. The second term in the optimization function allows mean feature-norm increase controlled by Δr. SAFN allows the exploration of more information. Thus, it increases the domain information transferability [79]. It is possible to restrict the mean feature-norms enlargement by changing the second penalty term as indicated in Eq. (32).

$$L_d(\max(h(x_i;\theta_0) + \Delta r, R), h(x_i;\theta)) \tag{32}$$

SAFN with entropy minimization (SAFN + ENTM)

The considered method SAFN can be combined with other domain adaptation techniques such as entropy minimization (ENTM) [79]. ENTM is widely used in domain shift mitigation. Indeed, it encourages target decision boundaries to pass through regions where sample densities are low [79]. In this research, we have explored the impact of entropy minimization by adding it as a penalty term [119] to the main optimization function of the SAFN method, and by comparing its results to the rest of AFN techniques.

3.2.4 Data poisoning technique

In this research, we have evaluated the efficiency and robustness of the considered models with a data poisoning technique. [80] has presented several types of data poisoning techniques that can be used for several tasks. However, for our current research, we have chosen one technique to test the robustness of our domain adaptation approaches. Indeed, the developed method creates mislabeled data from source and target domains and trains the considered domain adaptation methods with poisoned data aiming to fool these approaches. The poison rates that we considered are 0%, 5%, and 10% of the original data (source and target data). The failure of the considered methods demonstrates their limits to generalizing for unknown domains and can allow us to improve the poor methods in future research.

3.3 Experimental setup and results

3.3.1 Datasets

The Washington State University (WSU) Center for Advanced Studies in Adaptive Systems (CASAS) dataset [55] has been used to evaluate the task of activities recognition. It is a public dataset containing data from several apartments with several types of features and tasks. Indeed, it provides several types of activities, but we have focused only on the following activities: cooking breakfast, cooking lunch, cooking dinner, watching TV, and toileting. The choice of these labels was well-studied since they are related to some of our objectives such as energy management. Multiple types of sensors were at the origin of the dataset creation in different single-resident apartments such as Ambient PIR motion sensors, door/temperature sensors, and light switch sensors.

WiFi AR dataset [120, 63] has been used to evaluate the performance of the VADA method with our new setup compared to the previous research. The dataset is public and it uses Wi-Fi Channel State Information (CSI) to recognize human activities such as standing. It has 7 activities: walk, sit, stand, lay, get up, get down, and no activity [120, 63].

Datasets collected from our offices (H355 and H358) located in Grenoble Institute of Technology have been used to evaluate the tasks of occupancy estimation [3]. The considered datasets have multiple levels of occupancy. However, we evaluated three levels of occupancy (0, 1, and 2 occupants) for the current task. Internet of Things (IoT) sensors such as power consumption sensors, CO2 concentration sensors, humidity sensors, temperature sensors, door and window contact sensors, and acoustic pressure sensors, were at the origin of our private datasets that we used for the considered approaches in this research.

For all the considered methods and tasks (AR and OE), we used both balanced and unbalanced datasets while training and evaluating the approaches in order to see the robustness of these methods to label proportion change.

3.3.2 Metrics

To test the considered methods, we choose two metrics to use: accuracy and F1-score. In the case of balanced datasets, accuracy is a great metric to consider in order to evaluate model prediction as defined in Eq. (33) [3].

$$\textbf{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{33}$$

However, for the case where we have unbalanced datasets [67] the F1-score is the best metric to use in order to see the efficiency of the developed methods [3].

$$\textbf{Precision (Pr)} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FP_i} \tag{34}$$

$$\textbf{Recall (Rc)} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FN_i} \tag{35}$$

$$\textbf{F-score } (F_1) = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{36}$$

where TP is true positive, TN is true negative, FP is false positive, and FN is false negative.

3.3.3 Experimental results

Balanced datasets

In this section, we consider AR and OE tasks for the case where we have balanced datasets. For AR with 5 labels classification ('cook breakfast', 'cook lunch', 'cook dinner', 'toileting', 'watching TV'), the accuracies are collected in Table 3.1. For model performances without data poisoning, VADA is the best approach compared to the rest with 69.20% of accuracy which is good for unsupervised domain adaptation methods compared to supervised machine learning methods (80.80%).

HAFN gives acceptable performance (54.66%) when restricting mean feature norms to a given scalar. However, SAFN exceeds the HAFN score (62.00%) when enlarging mean feature norms so that we can discover more feature information. It is expected that SAFN strengthened with entropy minimization will enhance model performance as shown in Table 3.1 (63.33%). For the AR dataset, the SWD gives poor performance with and without data poisoning. The used Wasserstein distance can be the cause of the poor model performance with AR data. We have tested the robustness of the considered approaches by applying data poisoning at 2 levels (5% and 10% of data poison in the training datasets). For the VADA method, we see that the model remains robust for 5% of poison, and it drops a bit (65.10%) at 10% of poison compared to other models presented in other research [121]. Indeed, most of the models presented in [121] have reached 0% of accuracy at 10% of data poisoning. From these results, we can see the robustness of VADA for the considered task. The same observation applies to SAFN where we see a decrease of 3% when adding poison data. However, for HAFN and SAFN+ENTM, we see that the performances are almost the same with or without data poisoning. Thus, these two models are the most robust for this task of AR. We can conclude that entropy minimization helped increase the robustness of SAFN.

Table 3.1: Accuracies for AR with 5 balanced labels

Method	Accuracy (0%P)	Accuracy (5%P)	Accuracy (10%P)
VADA	69.20	71.40	65.10
SWD	20.04	20.04	20.04
HAFN	54.66	56.66	54.00
SAFN	62.00	58.66	58.66
SAFN + ENTM	63.33	62.00	62.66
SMLM	80.80	–	–

For AR with 3 label classifications ('cook breakfast', 'toileting', 'watching TV'), the models scores are presented in Table 3.2. We can notice a huge increase in models performances in general. Indeed, it is expected since we have decreased the number of labels. Thus, it will be easy for the model to predict a task of 3 classification levels than a task of 5 classification levels [3]. SAFN+ENTM has the best performance compared to the rest of the methods with 98.00% of accuracy, exceeding supervised machine learning methods' performances with a huge gap of almost 10%. SAFN+ENTM is totally robust to data poison without any drop in scores for all the data poison levels. SAFN without entropy minimization did also give a good performance that is comparable to

SAFN+ENTM (97.33%). However, without entropy minimization, we can see that the model is less robust compared to SAFN+ENTM. Indeed, it has a decrease of accuracy of around 8%. HAFN has less performance than the rest of AFN methods with 95.33% of accuracy. However, it is robust to data poisoning, and it has performance better than supervised machine learning methods (88.00%). VADA has also shown great performance like the previously stated methods for classifying activities with 95.40% of accuracy. However, data poisoning has a small effect on the model performance with 7% of accuracy reduction at 10% of poisoned training data. The SWD method did not perform well for the task of AR, and the poor results can be due to the used Wasserstein distance while adapting domains.

Table 3.2: Accuracies for AR with 3 balanced labels

Method	Accuracy (0%P)	Accuracy (5%P)	Accuracy (10%P)
VADA	95.40	93.20	88.60
SWD	33.20	33.20	33.20
HAFN	95.33	96.00	94.66
SAFN	97.33	89.33	92.66
SAFN + ENTM	98.00	98.00	98.00
SMLM	88.00	–	–

For OE with 3 levels of occupants, all the scores for the different methods are presented in Table 3.3. VADA has the best performance compared to the rest of the approaches with 76.40% of accuracy, and it is robust to data poisoning even though the small drop in performance while increasing the number of mislabeled samples in the training data (10% of accuracy reduction for 10% of poison data). SAFN is not far away compared to VADA. Indeed, it has a good performance with 74.40% of accuracy, and it is robust to data poisoning (almost the same performance for 10% of data poisoning). SAFN is so much better than SAFN+ENTM in terms of robustness and accuracy. Indeed, in this case, the entropy minimization was not helpful for the model. However, it was a cause to a decrease in its performance with 69.80% of accuracy that decreases till 66.80% when we poison the data. When we restrict mean features norms to a given small scalar (HAFN), we see a decrease in AFN method performance till reaching 63.20%, but it remains robust for data poison. In this case, SWD has shown great robustness for data poisoning with a slight decrease of accuracy from 63.20% till 62.20%, and it has shown acceptable performance in terms of accuracy. VADA and SAFN have performances that are comparable to supervised machine learning methods results,

and this is a great achievement.

Table 3.3: Accuracies for OE with 3 balanced labels

Method	Accuracy (0%P)	Accuracy (5%P)	Accuracy (10%P)
VADA	76.40	69.80	66.40
SWD	63.13	62.80	62.20
HAFN	63.20	58.60	63.20
SAFN	74.40	70.40	75.00
SAFN + ENTM	69.80	65.80	66.80
SMLM	83.00	–	–

For OE with 2 balanced labels, all the obtained scores are presented in Table 3.4. We can see an increase in the models' performances, and it is expected as discussed before [3] since we have decreased the task levels. All the considered methods have almost the same performance as supervised machine learning performance, and this proves their efficiency. VADA and SAFN have the best scores with 88% of accuracy. However, VADA is perfectly robust to data poison, and HAFN is slightly affected by the amount of mislabeled data added to the training process (2% of accuracy reduction). SAFN has shown great robustness while increasing the feature norms, but it has shown a small drop in accuracy (85.60%) as well compared to HAFN (88%). Adding the entropy minimization penalty term to the SAFN loss function has increased the accuracy of the model and kept its robustness. Indeed, it is considered one of the best models for the current task (87.80%).

Table 3.4: Accuracies for OE with 2 balanced labels

Method	Accuracy (0%P)	Accuracy (5%P)	Accuracy (10%P)
VADA	88.00	88.00	88.00
SWD	86.70	87.60	85.60
HAFN	88.00	86.40	86.20
SAFN	85.60	85.80	88.60
SAFN + ENTM	87.80	87.80	88.20
SMLM	88.80	–	–

In Figure 3.8, we present all models scores for the different tasks without data poisoning. Indeed, we see that SWD has better performance with OE tasks than AR tasks. Indeed, for AR it did not perform well, and it has given the lowest scores compared to the rest of the methods. For OE with 2 labels, all the models have comparable performances. Indeed, it can be explained by the simplicity

of the binary task in this case. For all the tasks, VADA has shown all time great performances, and it is one of the best-considered approaches. For AR with 3 labels, except SWD, all the methods have great results. Indeed, as explained before, the simplicity of the task makes the performances of all the methods comparable.

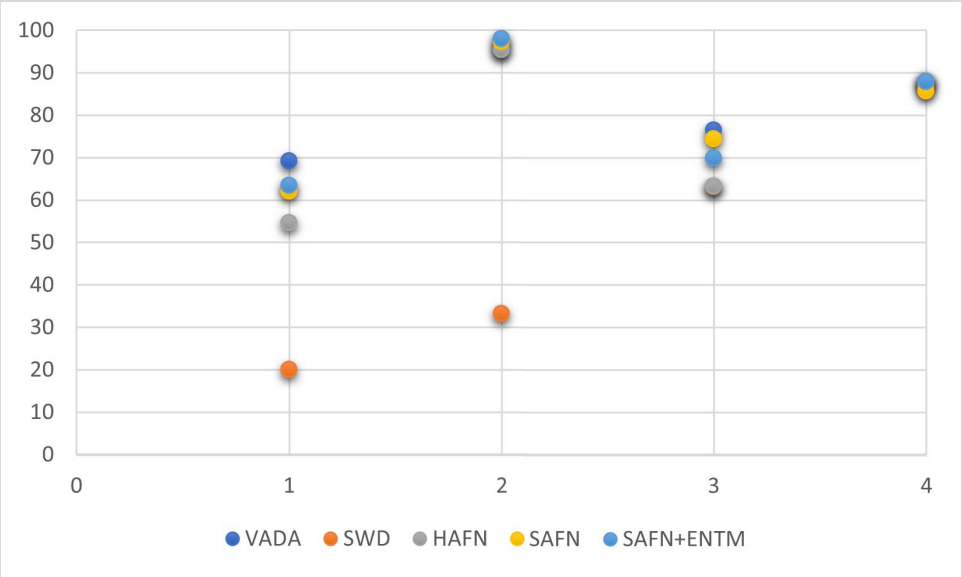

Figure 3.8: UDA methods results for balanced AR 5 labels (1), AR 3 labels (2), OE 3 labels (3), and OE 2 labels(4).

Unbalanced datasets

In this section, we evaluate the models' performances with the unbalanced datasets and test their robustness for the label proportion change for both tasks AR and OE. For AR with 5 unbalanced labels, we can see that VADA is performing well thanks to its domain adversarial training as well as its punishment of cluster violations. It has 71.76% of F1-score, and it is not too far from supervised machine learning methods performance (80.80% of F1-score) even though it is an unsupervised learning method. VADA has also shown good robustness while training it with mislabeled data from source and target domains. Indeed, the models' scores remain stable (around 72%) without any noticeable drop. SWD seems to give acceptable results (51.94%). However, the model scores remain exactly the same even though we have poisoned the data, and this creates doubts about the model's efficiency for the current task. Indeed, the used Wasserstein Discrepancy distance to reduce

domain shift may not perform well with the current features for AR tasks. HAFN has given acceptable results for an unsupervised learning task (48.07%) while restricting mean feature norms to a given scalar. However, it is not perfectly robust to data poisoning since its performance drops while increasing the mislabeled training data (44.59% of F1-score at 10% of poison data). Discovering new mean feature norm scales with SAFN has made the method more robust to data poison. Indeed, with and without data poison, SAFN has scores (47%). Adding entropy minimization to SAFN did not enhance its performance. However, we can see a drop in performance with SAFN+ENTM of 2% compared to SAFN. Table 3.5 presents all the discussed results.

Table 3.5: F1-scores for AR with 5 unbalanced labels

Method	F1-score (0%P)	F1-score (5%P)	F1-score (10%P)
VADA	71.76	71.76	72.14
SWD	51.94	51.94	51.94
HAFN	48.07	43.57	44.59
SAFN	46.80	45.04	47.07
SAFN + ENTM	44.19	43.20	45.25
SMLM	90.08	–	–

For AR with 3 unbalanced labels, Table 3.6 collects all the obtained scores with the different approaches. It is clear that there is a dramatic increase in the scores, and this is expected since we reduced the number of labels from 3 levels of occupancy to 2 levels of occupancy. Except for SWD which is not performing well for the task of AR, all the remaining methods are giving excellent results comparable to supervised machine learning methods and with a high level of robustness. HAFN with a small scale of mean feature norms has shown great performance (95.41% of F1-score) with great robustness as well. Indeed, at 10% of data poisoning, the model is giving 94.72% of F1-score which is almost the same as the performance of the trained model without data poisoning. Enlarging mean feature norms with SAFN has hurt a bit the model performance (4% of F1-score decrease), but it did keep its robustness. However, adding the entropy minimization penalty term for the SAFN optimization function has increased significantly the performance of the SAFN+ENTM model (97.37% of F1-score), allowing it to exceed all the considered methods with a high level of robustness (97.37% of F1-score at 10% of data poisoning). VADA is also giving good results for the current method with 91.01% of the F1-score. However, its robustness (72.48% of F1-score at 10% of poison data) is not that perfect but it is still considered good for an unsupervised domain

adaptation method.

Table 3.6: F1-scores for AR with 3 unbalanced labels

Method	F1-score (0%P)	F1-score (5%P)	F1-score (10%P)
VADA	91.01	71.33	72.48
SWD	36.44	36.44	36.44
HAFN	95.41	92.79	94.72
SAFN	91.42	91.49	89.57
SAFN + ENTM	97.37	96.66	97.37
SMLM	95.58	–	–

For OE with 3 unbalanced labels, all the obtained scores are shown in Table 3.7. SAFN, the method that explores more feature space information by increasing mean features norms, is giving the best scores (81.61% of F1-score), and it is perfectly robust. Indeed, with 10% of data poisoning in the training data, SAFN performance remains the same (82.56% of F1-score) without a noticeable drop. Adding entropy minimization to SAFN has almost no impact on the models' robustness and performance. Indeed, SAFN+ENTM is perfectly robust for data poisoning with 80.99% of the F1-score. Restricting mean feature norms to a given scalar (HAFN) did not hurt the model's robustness. However, we can see a noticeable drop in the model's score (3% of score reduction). VADA has a comparable score to HAFN (79.23%), but it is not perfectly robust to mislabeled samples in training data. Indeed, for 10% of data poisoning, VADA scores decrease to 71.25%. For this task, SWD has shown great robustness, and it has acceptable results (70.42% of F1-score) compared to the rest of the considered techniques.

Table 3.7: F1-scores for OE with 3 unbalanced labels

Method	F1-score (0%P)	F1-score (5%P)	F1-score (10%P)
VADA	79.23	81.18	71.25
SWD	70.42	71.32	71.33
HAFN	78.82	78.53	77.41
SAFN	81.61	79.63	82.56
SAFN + ENTM	80.99	81.56	82.18
SMLM	82.07	–	–

For OE with 2 unbalanced labels, as noticed before, we see a noticeable increase in models performances. Indeed, it is expected due to the task complexity decrease [3]. SAFN and HAFN have almost the same scores without data poisoning (89.70% and 89.78%, respectively). SAFN is

perfectly robust to data poisoning without a noticeable drop in model scores. However, HAFN, with the limited mean features norms scale compared to SAFN, has decreased with almost 4% of the F1-score while poisoning the training datasets. VADA is all the time showing great performances and robustness. In this case, it is perfectly robust to poisoned training data with 85.94% of the F1-score. SWD has also given acceptable results for this task compared to AR tasks. Indeed, it has 75.32% of F1-score, and it is robust to data poisoning with a small drop while adding mislabeled data to training (2% of score decrease).

Table 3.8: F1-scores for OE with 2 unbalanced labels

Method	F1-score (0%P)	F1-score (5%P)	F1-score (10%P)
VADA	85.94	85.95	85.25
SWD	75.32	74.35	73.68
HAFN	89.78	87.97	85.94
SAFN	89.70	88.23	89.62
SAFN + ENTM	88.33	88.23	87.63
SMLM	88.65	–	–

Figure 3.9 compares all the obtained scores for the different models and tasks. We can notice that for unbalanced datasets, SWD is giving better results compared to balanced datasets, especially for AR tasks. Indeed, with balanced datasets, SWD did not perform well for AR tasks. We can conclude that labels proportion change has a good impact on the SWD method. As discussed in balanced datasets, VADA remains one of the best models due to its consistency across all the tasks. Except for AR with 5 labels, most of the methods are giving excellent scores for the rest of the tasks.

VADA with WiFi dataset

VADA has been tested on the WiFi dataset to recognize human activities such as walking. In previous research, VADA has given an accuracy of 53%. However, in our current research, we have exceeded the previous score reaching 53.60% of accuracy as shown in Table 3.9. The new accuracy score is obtained due to new hyperparameters fine-tuning, and data processing.

Table 3.9: VADA accuracies with WiFi dataset

Method	Accuracy (%)
Our VADA	53.60
Previous VADA	53.00

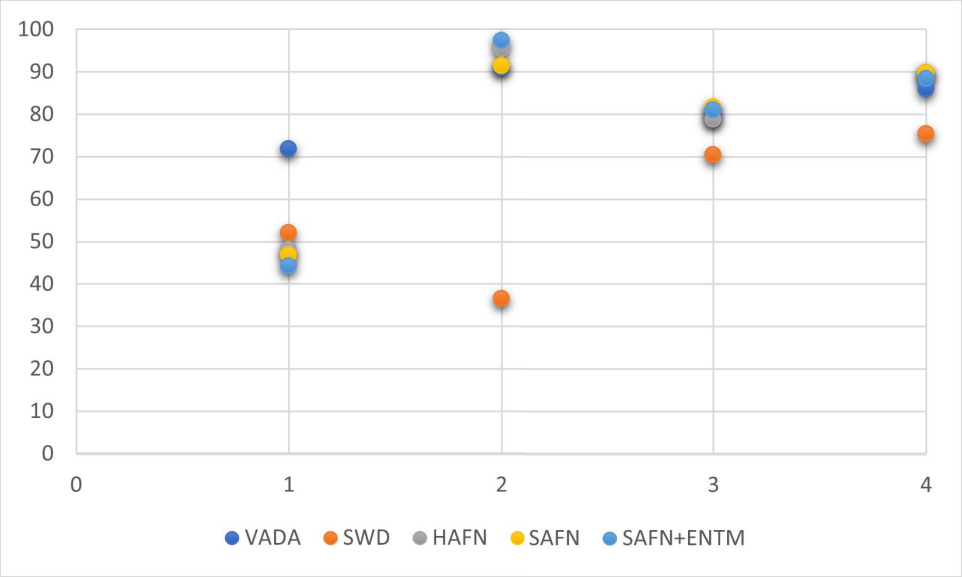

Figure 3.9: UDA methods results for unbalanced AR 5 labels (1), AR 3 labels (2), OE 3 labels (3), and OE 2 labels(4).

Chapter 4

Unsupervised Domain Adaptation With and Without Access to Source Data for Estimating Occupancy and Recognizing Activities in Smart Buildings

4.1 Introduction

In the last few years, it has been estimated that buildings' energy consumption is around 32% of the world's electrical energy [90]. Thanks to the evolution of sensors and IoT technologies [25, 13, 18, 19, 20, 122, 123, 124], smart buildings [25, 13, 19, 20, 30, 125, 126, 127, 128] have emerged, bringing several benefits such as energy management [25, 19, 20, 26, 27, 28, 29, 30, 129, 130, 131]. Researchers have focused on several vital smart building tasks such as activities recognition (AR) [25, 22, 132, 2, 1, 3, 133, 134, 135] and occupancy estimation (OE) [25, 13, 18, 27, 42, 136, 36, 137, 21, 138, 139, 140, 11]. OE can help provide optimal energy management for smart homes by reducing energy consumption and by optimally distributing energy across all the building's apartments [25, 19, 20, 26, 27, 28, 29, 30]. Detecting humans and recognizing their activities provide more security for smart homes by identifying authorized actions [31, 32, 33].

AR can enhance HVAC (heating, ventilation, and air conditioning) systems in smart homes using information gathered from each apartment, providing residents with good services [23]. Smart building models created in a specific environment [25, 13, 27, 36] (room or apartment) can not generalize well to other domains due to different apartment architectures (e.g., locations and types of the used sensors). Training a model for each room or apartment is an expensive and challenging task because of labeled data scarcity [25, 37, 38]. Labeling smart home data is a tedious, costly, and time-consuming task that can not be feasible in some cases due to privacy issues [25, 13, 24, 39, 40, 41, 42]. Domain adaptation (DA) [43, 44, 45, 46, 47, 48, 49, 50] can mitigate data distribution shifts between source and target domains in smart homes. Thus, it can solve the problem of data scarcity [25, 37, 38], and share knowledge gained in a smart home where labeled data are available to other smart buildings where labeled data are scarce or not available. Some approaches of DA can solve privacy issues by avoiding access to source data while sharing information across environments. Indeed, they get access only to trained models from source domains without accessing original data.

In this work, we make a comparative analysis between deep unsupervised domain adaptation (UDA) methods, applied to the tasks of AR and OE, with and without access to labeled source data. We aim to choose between these two types of approaches, taking into consideration several factors such as models' performances and privacy issues. We have considered adapting the latest UDA approaches. Indeed, we have 6 adapted approaches for UDA with access to labeled source data: domain separation networks (DSN) [82], cluster alignment with a teacher (CAT) [83], CAT+ gradient reversal (RevGrad) [83], CAT + robust RevGrad (rRevGrad) [83], Auxiliary Target Domain-Oriented Classifier (ATDOC) with nearest centroid classifier (NC), and ATDOC with neighborhood aggregation (NA) [84]. Also, we have 6 adapted methods for UDA without labeled source data: confidence score weighting adaptation using joint model data structure (CoWA-JMDS) [85], CoWA-JMDS without weights mixup [85], divide and contrast (DaC) [86], attracting and dispersing (AaD) [87], source hypothesis transfer with information maximization (SHOT-IM) [74, 88], and source hypothesis transfer with self-supervised pseudo-labeling (SHOT-Pseudo-labeling) [74, 88]. DSN extracts representations that are shared between source and target domains and representations that are private to each domain [82]. By finding the private subspaces that are orthogonal to the shared

subspace, DSN is able to differentiate between shared information and unique information for each domain. Then, it uses these representations to reconstruct sample features from both environments [82]. CAT trains a teacher model on source-labeled data and uses it to predict pseudo-labels for the unlabeled target samples [83]. CAT uses the generated pseudo-labels to make discriminative clustering by pushing features with the same class to be concentrated together, and features with different classes to be separated from each other [83]. CAT uses an objective loss to align clusters with the same class coming from different domains [83]. CAT has been enhanced with gradient reversal (CAT+RevGrad), and robust gradient reversal (CAT+rRevGrad) [83]. Gradient reversal and robust gradient reversal deal with target samples with low classification confidence and a high chance of being misclassified by the model while training [83]. ATDOC enhances the quality of the generated pseudo-labels by creating two classifiers (NC and NA) and using a memory mechanism to store all the information related to the unlabeled target data [84]. ATDOC+NC trains the nearest centroid classifier where all the information related to the class centroids are being stored in a given memory bank [84]. The centroids are being updated using an exponential moving average strategy [84]. ATDOC+NA creates a large memory bank storing the features with their predictions to train its classifier [84]. Using prediction aggregation and a confidence-weighted classification loss, the NA classifier is being dynamically updated [84]. CoWA-JMDS takes into consideration data points' importance by using JMDS score with exploited knowledge from both domains (source and target domains) [85]. CoWA-JMDS has been tested with and without weights mixup [85]. The goal of weights mixup is to make more use of target data points with low confidence scores [85]. DaC uses a source pre-trained model to divide target data points into source-like and target-specific data points [86], and reduce data distribution shift between them using a loss function based on maximum mean discrepancy (MMD) [86]. The source-like samples with a high level of prediction confidence have been used to learn the global class clustering. However, the target-specific samples with a low level of prediction confidence have been used to learn the local structures of the instances [86]. AaD takes into consideration the assumption that features from the same cluster should have the same class prediction while optimizing the objective. AaD clusters and assigns close features from both domains to reduce data distributions shift and to make domain adaptation [87]. SHOT

methods use a source pre-trained model which has a good performance while classifying data samples from the source domain. They transfer the weights from the source hypothesis module to the target classifier module and freeze them during the whole training process. However, they learn a new feature encoding module so that the discrepancy between the output of the target feature extractor and the source feature extractor is minimum [74, 88]. The information maximization (IM) and a novel self-supervised pseudo-labeling method have been used to learn the feature extractor module of the target model [74, 88, 97, 93]. In Table 4.1, we have provided a classification of the considered UDA composed of 3 categories [141]. Firstly, domain invariant methods learn data representations that are invariant between source and target domains, then they reduce the discrepancy between them. Secondly, self-training methods create pseudo-labels for the unlabeled target data, then they train the target model with the generated pseudo-labels. Finally, self-supervision methods create a pre-trained model, then they fine-tune its parameters to fit the target domain. The considered UDA methods have been evaluated using our private datasets for OE [25] as well as public datasets for AR [100]. The datasets have been collected using ambient sensors such as CO2 concentration sensors [25, 100]. In this research, we have made several contributions. Indeed, we have adapted all the considered methods to smart building tasks (AR and OE) so that they can work with sensor data, and we have tested them on smart buildings datasets (private and public datasets). Also, we have created novel deep neural network architectures for most of the considered methods. Indeed, all the feature extractors and classifiers have been changed with new ones based on convolutional neural networks [83, 85, 86, 87, 100] that are adaptable to sensor data. Moreover, we have made a comparative analysis between UDA methods' performance with and without direct access to labeled source data, and we have chosen the best approach based on several factors such as privacy issues.

The rest of the chapter is organized into 2 sections. In section 2, we introduce and explain the considered methods. In section 3, we present and discuss the research findings.

Table 4.1: Classification of the considered UDA methods

Method	Domain-invariant	Self-training	Self-supervision
DSN	✓		
CAT		✓	
CAT+RevGrad		✓	
CAT+rRevGrad		✓	
ATDOC+NC		✓	
ATDOC+NA		✓	
CoWA-JMDS		✓	
CoWA-JMDS w/o WM		✓	
DaC	✓	✓	✓
AaD			✓
SHOT-IM			✓
SHOT-Pseudo-labeling		✓	✓

4.2 The proposed approaches

4.2.1 Methods with access to source data

Domain separation networks (DSN)

The domain separation networks (DSN) goal [82], as with most domain adaptation methods, is to train a classifier on labeled source data that generalizes well on target data from another smart building environment. The trained model gives source feature representations that are comparable to the target feature space. Thus, a trained model from the source domain can be applied to predict unlabeled target data. However, the newly created representation between source and target domains might include noise that can affect the model predictions and can reduce performance [82]. DSN partitions the feature representations into a shared feature space and a private feature space that is unique to each domain. Splitting data representations in this way allows a trained classifier to generalize well for both source and target domains without being affected by the representations that are unique to each environment [82]. The model has used two loss functions that encourage the separation of shared and private representations and ensure the usefulness of the private representations to make domain adaptation.

Let $X_S = \{(x_i^s, y_i^s)\}_{i=0}^{N_s}$ be a labeled source dataset with N_s instances from the source domain $x_i^s \sim D_S$, and $X_T = \{(x_i^t)\}_{i=0}^{N_t}$ be an unlabeled target dataset with N_t instances from the target domain $x_i^t \sim D_T$. A given input sample x is transformed and mapped to new common and private

feature representations (h_c and h_p) using mapping functions $\mathrm{E}_c(x;\theta_c)$ and $\mathrm{E}_p(x;\theta_p)$, respectively, where θ_c and θ_p are parameters [82]. A reconstructed sample $\hat{x}$ is created by mapping a hidden feature representation h using a decoding function $D(h;\theta_d)$, where θ_d is a parameter [82]. A task-specific prediction $\hat{y}$ is created using a task-specific function $G(h;\theta_g)$, where θ_g is a parameter [82]. For DSN model [82], $\hat{x} = D(\mathrm{E}_c(x) + \mathrm{E}_p(x))$ and $\hat{y} = G(\mathrm{E}_c(x))$. The goal of the approach is to minimize the following optimization function:

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \alpha\mathcal{L}_{\text{recon}} + \beta\mathcal{L}_{\text{difference}} + \gamma\mathcal{L}_{\text{similarity}} \tag{37}$$

where α, β, and γ are weight parameters to control the influence of each loss function. $\mathcal{L}_{\text{task}}$ is the loss function used to train the model on labeled source data assuming that the target domain contains only unlabeled samples. $\mathcal{L}_{\text{task}}$is defined as follows:

$$\mathcal{L}_{\text{task}} = -\sum_{i=0}^{N_s} y_i^s \cdot \log \hat{y}_i^s \tag{38}$$

where y_i^s and $\hat{y}_i^s$ are the source ground-truth and predicted labels [82]. $\mathcal{L}_{\text{recon}}$ is a reconstruction loss based on scale invariant mean squared error that penalizes samples' predictions that are correct by a scaling factor [142]. The reconstruction loss function is defined using Eq.(39) and Eq.(40).

$$\mathcal{L}_{\text{recon}} = \sum_{i=1}^{N_s} \mathcal{L}_{si-mse}(x_i^s, \hat{x}_i^s) + \sum_{i=1}^{N_t} \mathcal{L}_{si-mse}(x_i^t, \hat{x}_i^t) \tag{39}$$

$$\mathcal{L}_{si-mse}(x, \hat{x}) = \frac{1}{k}\left\|x - \hat{x}\right\|_2^2 - \frac{1}{k^2}\left([x - \hat{x}] \cdot \mathbf{1}_k\right)^2 \tag{40}$$

where $\|.\|_2^2$ is the squared L2-norm, k is the number of features in each sample, $\mathbf{1}_k$ is a ones vector. The loss difference, defined in Eq.(41), creates different representation inputs for source and target domains using soft subspace orthogonality constraints applied between private and shared feature representation spaces for source and target environments [82].

$$\mathcal{L}_{\text{difference}} = \left(\left\|\mathbf{H}_c^{s^T}\mathbf{H}_p^s\right\|_F^2 + \left\|\mathbf{H}_c^{t^T}\mathbf{H}_p^t\right\|_F^2\right) \tag{41}$$

where $\mathbf{H}_c$ and $\mathbf{H}_p$ are matrices containing common and private feature representations in source and target domains, and $\|.\|_F^2$ is the squared Frobenius norm [82]. For the similarity loss, its goal is to create feature representations similar between source and target domains so that the model trained on source data can predict samples from the target domain [82]. In this research, we apply a domain adversarial similarity loss to confuse the classifier while predicting the origin (source or target data) of each data point. The confusion process has been done using a domain classifier to predict the domain with a Gradient Reversal Layer (GRL) [82]. The loss functions are defined in the following equations:

$$\mathcal{L}_{\text{similarity}}^{\text{DANN}} = \sum_{i=0}^{N_s+N_t} d_i \log \hat{d}_i + (1-d_i)\log(1-\hat{d}_i) \tag{42}$$

where $d_i \in (0,1)$ depending on the domain of each data point i. Maximum Mean Discrepancy (MMD) has been used as a metric to reduce the discrepancy between the shared hidden representations of source and target domains (h_{ci}^s, h_{ci}^t).

$$\mathcal{L}_{\text{similarity}}^{\text{MMD}} = \frac{1}{N_s^2}\sum_{i,j=0}^{N_s}\kappa(h_{ci}^s, h_{cj}^s) - \frac{2}{N_s N_t}\sum_{i,j=0}^{N_s,N_t}\kappa(h_{ci}^s, h_{cj}^t) + \frac{1}{N_t^2}\sum_{i,j=0}^{N_t}\kappa(h_{ci}^t, h_{cj}^t)$$

$$\kappa(x_i, x_j) = \sum_n \eta_n \exp\left\{-\frac{1}{2\sigma_n}\|x_i - x_j\|^2\right\} \tag{43}$$

where σ_n and η_n are the standard deviations and the weights.

Cluster Alignment with a Teacher (CAT)

Let us consider a set of labeled source data Let $X_s = \{(x_s^i, y_s^i)\}_{i=1}^N$ with $y_s^i \in (1, 2, ..., K)$, and a set of unlabeled target data $X_t = \{x_t^i\}_{i=1}^M$. Typically, the source and the target data have different feature distributions (domains shift), and a trained source model will not perform well on unlabeled target data [83]. Thus, researchers have tried to mitigate domains shift so that classifiers $(h = g \circ f)$ from the source domain can accurately predict inputs from unlabeled target data points [83], where f is a function that creates domain-invariant feature representations, and g predicts the

novel inputs created by f. Normally, mitigating discrepancy between domains is done by optimizing the following supervised and discrepancy loss functions [83]:

$$\min_{\theta} \mathcal{L}_y = \frac{1}{N} \sum_{i=1}^{N} l(h(x_s^i; \theta), y_s^i) \tag{44}$$

$$\min_{\theta} \mathcal{L}_d(X_s, X_t) = \mathcal{D}(f(X_s, \theta), f(X_t, \theta)) \tag{45}$$

where l is a cross-entropy loss and $\mathcal{D}$ is a distance. Most UDA methods minimize the expected error on the source data points and the distance between source and target data distributions. However, they ignore minimizing the expected error between source and target labeling functions [83], which can lead to unsatisfying model accuracy. In this research, we present cluster alignment with a teacher (CAT) that takes into consideration the class-conditional distributions of two domains to mitigate the discrepancy between source and target labeling functions [83]. CAT solves the optimization function in Eq.(46).

$$\min_{\theta} \mathcal{L}_y + \alpha(\mathcal{L}_{cl} + \mathcal{L}_a), \tag{46}$$

where $\mathcal{L}_{cl}$ creates discriminative clusters for features from both domains, $\mathcal{L}_a$ aligns clusters with the same labels from both environments and α is a hyperparameter. A teacher has been developed to provide pseudo-labels for unlabeled target data points. The obtained pseudo-labels will be used by the two losses $\mathcal{L}_{cl}$ and $\mathcal{L}_a$. The discriminative clustering with a teacher is done using $\mathcal{L}_{cl}$, and it can be adjusted based on obtained pseudo-labels from the teacher [83]. $\mathcal{L}_{cl}$ is defined in the following equation:

$$\mathcal{L}_{cl}(X_s, X_t) = \mathcal{L}_{cl}(X_s) + \mathcal{L}_{cl}(X_t) \tag{47}$$

$$\mathcal{L}_{cl}(X) = \frac{1}{|X|^2} \sum_{i=1}^{|X|} \sum_{j=1}^{|X|} [\delta_{ij} d(f(x^i), f(x^j)) + (1 - \delta_{ij}) \max(0, m - d(f(x^i), f(x^j)))]$$

where m is a given margin, d is the squared Euclidean distance, and δ_{ij} is an indicator function that gives 1 in case the prediction of the source model or the teacher model for samples x^i and x^j are the same [83]. $\mathcal{L}_{cl}$ clusters features from the same labels together, and pushes away features with high discrepancy from each other using margin m. After obtaining features with good discriminative cluster structure, the source-trained classifier g may continue to fail to classify the obtained clusters due to the mismatch between source and target clusters with the same labels [83]. That is why, a cluster alignment process is required at this step to enhance the framework prediction accuracy. The cluster alignment is done using $\mathcal{L}_a$ defined in Eq.(48):

$$\mathcal{L}_a(X_s, X_t) = \frac{1}{K}\sum_{k=1}^{K} \|(\lambda_{s,k} - \lambda_{t,k})\|_2^2 \tag{48}$$

where $\lambda_{s,k}$ and $\lambda_{t,k}$ are defined in the following equation with subsets $X_{s,k}$ and $X_{t,k}$ from X_s and X_t, respectively, whose labels are k:

$$\lambda_{s,k} = \frac{1}{|X_{s,k}|}\sum_{x_s^i \in X_{s,k}} f(x_s^i), \lambda_{t,k} = \frac{1}{|X_{t,k}|}\sum_{x_t^i \in X_{t,k}} f(x_t^i) \tag{49}$$

$\mathcal{L}_{cl}$ and $\mathcal{L}_a$ combined together provide us with aligned class-conditional structures for source and target environments [83]. CAT creates a student model trained on the labeled source data, and it generates a teacher model as an implicit ensemble of the source model. The teacher model is used to provide pseudo-labels for the unlabeled target data. Using $\mathcal{L}_{cl}$, CAT forces features from the same label to be clustered together. Then, CAT uses $\mathcal{L}_a$ to mitigate the discrepancy between clusters from source and target domains which updates the teacher model. We combined CAT with other domain alignment methods to enhance the cluster alignment between source and target domains [83]. Indeed, CAT has been extended with gradient reversal (CAT+RevGrad) [83, 143], and robust gradient reversal (CAT+rRevGrad) [83]. Gradient reversal and robust gradient reversal are confidence-thresholding techniques used to prevent certain instances from being aligned with a confidence score less than a given value p. rRevGrad is an upgraded version of RevGrad with the following loss function:

$$\min_{\theta}\max_{\phi}\mathcal{L}_d(X_s,X_t) \quad = \quad \frac{1}{N}\sum_{i=1}^{N}\log c(f(x_s^i;\theta);\phi) \; + \; \frac{1}{\tilde{M}}\sum_{i=1}^{\tilde{M}}[\log(1 \; - \; c(f(x_t^i;\theta);\phi))\gamma_i] \tag{50}$$

where γ_i is an indicator function with value 1 if the model's confidence for the sample x_t^i exceeds the value p, ϕ is a parameter, and c is the critic model.

Auxiliary target domain-oriented classifier (ATDOC)

ATDOC reduces data distribution shifts between source and target domains in this research by learning a target classifier F_t with unlabeled target data. ATDOC exploits the source-like instances in the target data, which have close data distribution to source samples, to build the target classifier and label the remaining target samples [84]. To obtain pseudo-labels with high confidence levels, ATDOC exploits a memory module to store all the information related to the target samples while training [84]. ATDOC has generated 2 types of target classifiers: nearest centroid classifier (NC) and neighborhood aggregation (NA).

Nearest centroid classifier (NC): NC uses label centroids to reduce domain shift between source and target data distribution [84]. It uses a memory bank to store all the gathered information while training (class centroids), and it iteratively labels the target unlabeled samples (pseudo-labels) [84]. The centroids in the memory banks are being dynamically updated using the target pseudo-labels obtained using Eq.(51) and the exponential moving averaging (EMA) technique [84].

$$\hat{y}_i = \arg\max_k p_{i,k}, \quad i = 1,2,\ldots,N_t \tag{51}$$

$$c_j = \frac{\sum_{i\in B_t}\mathbf{1}_{[j=\hat{y}_i]}G(x_t^i)}{\sum_{i\in B_t}\mathbf{1}_{[j=\hat{y}_i]}}, \tag{52}$$

$$c_j^m = \gamma c_j + (1-\gamma)c_j^m, \quad m = 1,2,\ldots,K \tag{53}$$

where N_t is the number of target samples, $p_i = F(G(x_i^t))$ is a prediction vector of dimension K the number of classes, F is a classifier module, G is a feature extractor module, x_i^t is a sample from the target domain $\mathcal{D}_t = \{x_i^t\}_{i=1}^K$, γ is a hyperparameter for smoothing, and B_t is a target batch set. The nearest centroid classifier is built using the obtained centroids. Then, it generates updated pseudo-labels for the target samples using the following equation:

$$\hat{y}_i = \arg\min_{j=1}^{K} d(G(x_i^t), c_j^m), \quad i = 1, 2, \ldots, N_t \tag{54}$$

where $d(.,.)$ is the cosine distance. Then, we develop a cross-entropy loss function as follows:

$$\mathcal{L}_{\text{nc}} = -\frac{\lambda}{N_t} \sum_{i=1}^{N_t} \log p_{i,\hat{y}_i} \tag{55}$$

Neighborhood aggregation (NA): NA classifier generation requires a large memory bank to store all local information related to each target sample structure which is different from NC strategy where we consider only the global domain structure [84]. While updating the memory bank, we do not use the moving average technique. However, to avoid obtaining ambiguous predictions for the unlabeled target data $x_i \in \mathcal{D}_t$, we scale the predictions p_i using a hyperparameter T as defined in the following equation [84]:

$$\tilde{p}_{i,k}^m = \frac{p_{i,k}^{\frac{1}{T}}}{\sum_k p_{i,k}^{\frac{1}{T}}} \tag{56}$$

The neighborhood aggregation strategy uses the cosine similarity distance to measure the discrepancy between the sample target features and the memory bank module features, to get m nearest neighbors [84]. The nearest neighbors' predictions $\tilde{p}_j$ are aggregated using the following equations:

$$\hat{q}_i = \frac{1}{m} \sum_{j \neq i, j \in \mathcal{N}_i} \tilde{p}_j, \tag{57}$$

where $\mathcal{N}_i$ represents the neighbors set for a given target sample x_i^t in the memory bank module. This strategy gives new probability predictions while training on all the target data [84]. Based on these probabilities, we obtain pseudo-labels for all unlabeled target data using Eq.(51). We define a

confidence-weighted cross-entropy loss that uses $\hat{q}_{i,\hat{y}_i}$ as confidence weights for each pseudo-label defined as follows:

$$\mathcal{L}_{\text{na}} = -\frac{\lambda}{N_t} \sum_{i=1}^{N_t} \hat{q}_{i,\hat{y}_i} \log p_{i,\hat{y}_i} \tag{58}$$

For a labeled source domain $\mathcal{D}_{\int}$ and unlabeled target domain $\mathcal{D}_{\sqcup}$, we combine several loss functions: $\mathcal{L}_{\text{na}}(\mathcal{D}_t)$ or $\mathcal{L}_{\text{nc}}(\mathcal{D}_t)$ with $\mathcal{L}_{lsr}(\mathcal{D}_s)$ the stand cross-entropy loss with label-smoothing regularization [144] to obtain the final objective function defined as follows:

$$\mathcal{L} = \mathcal{L}_{lsr}(\mathcal{D}_s) + \mathcal{L}_{\text{nc/na}}(\mathcal{D}_t) \tag{59}$$

For binary tasks (2 levels of occupancy), we can add additional domain shift mitigation techniques such as CDAN [145] to enhance performance.

4.2.2 Methods without access to source data

Confidence score weighting adaptation using the joint model data structure (CoWA-JMDS)

For CoWA-JMDS, we have access to unlabeled target data $X_t = \{x_i^t\}_{i=1}^{n_t}$ with n_t samples, and a model M trained using labeled source data $X_s = \{x_i^s, y_i^s\}_{i=1}^{n_s}$ with n_s data points. Using the pre-trained model M, we generate the pseudo-labels $\hat{Y}_t = \{y_i^t\}_{i=1}^{n_t}$ that we will be using them in the UDA process [85]. A pre-trained source model M is a combination of a feature encoding f with d features and a classifier g with K labels [85]. The model gives its prediction for a given instance based on the values of probabilities obtained using Eq.(60):

$$p_M(X_t) = \text{softmax}(f(g(X_t))) \tag{60}$$

where softmax is an activation function applied to the output layer of g.

In this research, the proposed JMDS confidence score is a combination of the Log-Probability Gap (LPG) score based on data-structure-wise probability and Model Probability of Pseudo-label (MPPL) score based on the model-wise probability [85]. Gaussian Mixture Modelling (GMM) has been used to create clusters and pseudo-labels for the unlabeled data from the target environment.

GMM gives data-structure-wise probability $p_{data}(X_t)$ which improves its confidence compared to other clustering methods [85]. The LPG score is a data-structure-wise confidence score based on GMM probabilities related to the unlabeled target domain [85]. LPG is defined as follows:

$$LPG(x_i^t) = \frac{\text{MINGAP}(x_i^t)}{\max_j \text{MINGAP}(x_j^t)} \tag{61}$$

$$\text{MINGAP}(x_i^t) = \min_a \left(\log p_{data}(x_i^t)_{\hat{y}_i^t} - \log p_{data}(x_i^t)_a\right) \tag{62}$$

where $\hat{y}_i^t = \arg\max_c p_{data}(x_i^t)_c$, $a \in 1, 2, ..., K$, and $a \neq \hat{y}_i^t$. LPG has high confidence levels for data points far from decision boundaries. The MPPL score is a model-wise probability score based on the generated pseudo-label $\hat{Y}_t$ [85]. Indeed, instances with high confidence levels are the ones that have the same pseudo-labels evaluated using $p_{data}(X_t)$ and $p_M(X_t)$ as described in Eq.(63) [85]:

$$MPPL(x_i^t) = p_M(x_i^t)_{\hat{y}_i^t} \tag{63}$$

JMDS is the only probability score defined between 0 and 1 as the product of LPG with MPPL that provides knowledge from both domains. Indeed, MPPL provides information on the model and LPG provides information on the data structure [85]:

$$JMDS(x_i^t) = LPG(x_i^t) \cdot MPPL(x_i^t) \tag{64}$$

The generated pseudo-labels $\hat{Y}_t$ from GMM and JMDS score are used to enhance and train the model M in order to increase its prediction performance on target domain [85]. To deal with noisy labels (pseudo-labels) that are not perfectly accurate, we use a confidence score based on sample weighting as defined in the following equation:

$$\mathcal{L}_{CoWA\text{-}JMDS}(x_i^t) = \text{JMDS}(x_i^t) \cdot \mathcal{L}_{CE}(p_M(x_i^t), \hat{y}_t^i) \tag{65}$$

where $\mathcal{L}_{CE}$ is a cross-entropy loss defined as follows:

$$\text{JMDS}(x_i^t) \cdot \mathcal{L}_{CE}(p_M(x_i^t), \hat{y}_t^i) = -\log p_M(x_i^t)_{\hat{y}_t^i} \tag{66}$$

By using confidence scores, we ignore samples with low scores while training. Indeed, it means that the target domain information is not totally used which can lead to model failure in some particular situations [85]. To solve this issue, we propose to apply CoWA-JMDS with and without weight mixup [85, 146] to allow training with samples with low and high confidence scores so that the model can be more robust to incorrect pseudo-labels while training [85]. The weight mixup loss function is defined in Eq.(67). Indeed, it gives low weights for samples with low confidence scores and vice-versa:

$$\mathcal{L}_{\text{Mixup}}(\tilde{x}^t, \tilde{y}^t) = w(\tilde{x}^t) \cdot \text{E}_{\tilde{y}^t}[-\log p_M(\tilde{x}^t)] \tag{67}$$

where $w(\tilde{x}^t)$ is a defined weight function for each sample based on its confidence score [85].

Divide and Contrast (DaC)

DaC method extracts two types of data from the unlabeled target domain $\mathcal{D}_T$ based on the pre-trained source model predictions: source-like instances $\mathcal{D}_S$ and target-specific outliers $\mathcal{D}_O$ [86]. The method loss function is weighted with α and β parameters, containing three main losses as shown in Eq.(68) [86]:

$$\mathcal{L} = \mathcal{L}_{\text{con}} + \alpha\mathcal{L}_{\text{self}} + \beta\mathcal{L}_{\text{EMMD}} \tag{68}$$

where $\mathcal{L}_{\text{self}}$ is used to apply class-wise adaptation at first using the generated pseudo-labels for unlabeled target data, $\mathcal{L}_{\text{con}}$ is an adaptive contrastive loss that uses target-specific instances for local consistency regularization and uses source-like instances for robust class-wise adaptation, and $\mathcal{L}_{\text{EMMD}}$ is used to mitigate the difference between $\mathcal{D}_S$ and $\mathcal{D}_O$ (distribution alignment) [86].

Firstly, we perform class-wise adaptation by predicting pseudo-labels for the unlabeled target dataset. The pseudo-labels are updated with the centroids multiple times until they converge using the approach proposed in [86, 74]. Indeed, each class has a centroid c_k that groups all the nearest

samples as defined in the following equations:

$$\tilde{\mathcal{Y}} = \{\tilde{y}_i | \tilde{y}_i = \underset{k}{\operatorname{argmax}} \cos(\phi(x_i), c_k), x_i \in \mathcal{D}_T\} \tag{69}$$

$$c_k = \frac{\sum_{i=1}^{n_t} \mathbf{1}(\tilde{y}_i = k) f_i}{\sum_{i=1}^{n_t} \mathbf{1}(\tilde{y}_i = k)} \tag{70}$$

where f_i is the model feature extractor, $\cos(.,.)$ is the cosine similarity, and $\mathbf{1}(.)$ is the indicator function. While training, the training instances from the target data are transformed into two views using two created transformations [86]. Instances are divided into source-like and target-specific samples based on the newly created views [86]. For adaptive contrastive learning, we create an updated memory bank containing all the target domain features $\mathcal{F} = \{z_i\}_{i=1}^{n_t}$, and it creates in each iteration source-like centroids and target specific features [86]. Indeed, with source-like samples, contrastive learning employs class-wise adaptation, considering source-like samples as class centroid [86]. However, with the noisy target-specific samples, contrastive learning takes into consideration the local consistency regularization [86]. Memory bank helps conserve information from the source domain. It is updated at each iteration, and it generates, to the contrastive learning approach, the needed source-like centroids and target-specific samples [86]. The last step performed by this UDA method is distribution alignment. Indeed, we align source-like and target-specific samples using maximum mean discrepancy (MMD) defined as follows:

$$d_{\text{MMD}}(\mathcal{S}, \mathcal{T}) = \frac{1}{m}\sum_{i=1}^{m} s_i \left(\sum_{i'=1}^{m} s_{i'} - \frac{1}{n}\sum_{j'=1}^{n} t_{j'}\right) + \frac{1}{n}\sum_{i=1}^{n} t_i \left(\frac{1}{n}\sum_{j'=1}^{n} t_{j'} - \frac{1}{m}\sum_{i'=1}^{m} s_{i'}\right)$$

where $(\mathcal{S}, \mathcal{T})$ are the source and target domains, (s, t) are the source-like and target-specific features, and (m, n) are the number of features for both domains batch.

Attracting and Dispersing (AaD)

For the AaD method, we have in hand a pre-trained model from the source domain with N_t unlabeled samples $\mathcal{D}_t = \{x_i^t\}_{i=1}^{N_t}$ from the target domain with C labels. In this approach, UDA is

performed without access to source data [87]. The source pre-trained model is a combination of a feature encoding f and a classifier g. The considered method works by grouping features that have close or the same predictions and dispersing features that have different predictions [87]. To make this evaluation, we defined a probability p_{ij} function between two features to measure the similarity between them:

$$p_{ij} = \frac{e^{p_i^T p_j}}{\sum_{N_t} e^{p_i^T p_k}} \tag{71}$$

For each feature z_i, we define a space grouping its close features called $\mathcal{C}$ and space grouping the rest of features called $\mathcal{B}$ [87]. We store all the obtained information about target features in two memory banks while training. To perform feature clustering for AaD, we minimize the following loss function $\tilde{L}$:

$$\tilde{L}_i(\mathcal{C}_i, \mathcal{B}_i) = -\log \frac{P(\mathcal{C}_i)}{P(\mathcal{B}_i)} \tag{72}$$

where P is defined in the next equations for a model with θ parameters [87]:

$$P(\mathcal{C}_i|\theta) = \prod_{j \in \mathcal{C}_i} p_{ij} = \prod_{j \in \mathcal{C}_i} \frac{e^{p_i^T p_j}}{\sum_{N_t} e^{p_i^T p_k}} \tag{73}$$

$$P(\mathcal{B}_i|\theta) = \prod_{j \in \mathcal{B}_i} p_{ij} = \prod_{j \in \mathcal{B}_i} \frac{e^{p_i^T p_j}}{\sum_{N_t} e^{p_i^T p_k}} \tag{74}$$

Optimizing Eq.(72) will simultaneously perform feature clustering and cluster assignment [87]. The considered optimization function can be simplified to a new loss function that does the same function as Eq.(72) which is defined as follows:

$$L = \mathrm{E}(\tilde{L}_i(\mathcal{C}_i, \mathcal{B}_i)), \mathrm{E}(\tilde{L}_i(\mathcal{C}_i, \mathcal{B}_i)) = -\sum_{j \in \mathcal{C}_i} p_i^T p_j + \lambda \sum_{m \in \mathcal{B}_i} p_i^T p_m \tag{75}$$

where λ is a hyperparameter. The first term of the new loss function will encourage prediction consistency features from $\mathcal{C}$, and the second term will disperse prediction of features from $\mathcal{B}$ [87].

Source HypOthesis Transfer (SHOT)

SHOT uses a source pre-trained model which has a good performance while classifying data samples from the source domain. The used source model, defined as $f_s(x) = h_s(g_s(x))$, is a combination of a feature extractor $g_s : \mathrm{X_s} \longrightarrow \mathbb{R}^d$ with d number of features, and a classifier $h_s : \mathbb{R}^d \longrightarrow \mathbb{R}^K$ with K number of labels. SHOT transfers the weights from the source hypothesis module (classifier module) to the target classifier module, and freezes them during the whole training process $h_s = h_t$. However, it learns a new feature encoding module $g_t : \mathrm{X_t} \longrightarrow \mathbb{R}^d$ so that the discrepancy between the output of the target feature extractor and the source feature extractor is minimum. In other words, the data distribution $(p(g_t(x_t)), p(g_s(x_s)))$ difference between source and target domains will be minimized [74, 88].

SHOT with Information Maximization (SHOT-IM): SHOT-IM uses information maximization [74, 88, 97, 93] while updating the weights of the target feature extractor as defined in the following loss function:

$$\begin{aligned} \mathcal{L}_{ent}(f_t; \mathrm{X_t}) &= -E_{x_t \in \mathrm{X_t}} \sum_{k=1}^{K} \delta_k(f_t(x_t)) \log \delta_k(f_t(x_t)), \\ \mathcal{L}_{div}(f_t; \mathrm{X_t}) &= \sum_{k=1}^{K} \hat{p}_k \log \hat{p}_k = D_{KL}(\hat{p}, \frac{1}{K} 1_K) - \log(K) \end{aligned} \tag{76}$$

where 1_K is a vector containing K values of one, $\hat{p} = E_{x_t \in \mathrm{X_t}}(\delta_k(f_t^{(k)}(x_t)))$ is defined as the mean output embedding of the target domain, and D_{KL} is defined as the Kullback-Leibler divergence.

SHOT with self-supervised pseudo-labeling: Pseudo-labeling has been employed in this research to provide unlabeled data from the target domain with labels so that we can see a performance increase compared to SHOT-IM performance [74, 88]. Pseudo-labeling is a technique that uses supervised machine learning models to label unlabeled data. In our case, it can be done by the

source pre-trained model, however, due to the domain's discrepancy between source and target domains a self-supervised pseudo-labeling technique has been considered [74, 88]. the self-supervised pseudo-labeling strategy generates a centroid for each target domain label in an unsupervised way and infers the pseudo labels as defined in the following equation:

$$\begin{aligned} c_k^{(0)} &= \frac{\sum_{x_t \in \mathrm{X_t}} \delta_k(\hat{f}_t(x_t))\hat{g}_t(x_t)}{\sum_{x_t \in \mathrm{X_t}} \delta_k(\hat{f}_t(x_t))}, \\ \hat{y}_t &= \underset{k}{argmin}\, D_f(\hat{g}_t(x_t), c_k^{(0)}) \end{aligned} \tag{77}$$

where D_f is defined as the cosine distance, and $\hat{f}_t = \hat{g}_t(h_t)$ is defined as the previously learned target model. Then, using the previously obtained updates, we calculate the new centroids and pseudo-labels defined as follows:

$$\begin{aligned} c_k^{(1)} &= \frac{\sum_{x_t \in \mathrm{X_t}} \mathbf{1}(\hat{y}_t = k)\hat{g}_t(x_t)}{\sum_{x_t \in \mathrm{X_t}} \mathbf{1}(\hat{y}_t = k)}, \\ \hat{y}_t &= \underset{k}{argmin}\, D_f(\hat{g}_t(x_t), c_k^{(1)}) \end{aligned} \tag{78}$$

We keep doing this step several times until we obtained pseudo-labels confident pseudo-labels. The obtained pseudo-labels are called self-supervised pseudo-labels because the centroids are created in an unsupervised way [74, 88]. SHOT solves the following objective function $\mathcal{L}(g_t)$ to create the target feature extractor [74, 88], and its general steps are defined in Algorithm 2:

$$\mathcal{L}(g_t) = \mathcal{L}_{ent}(f_t; \mathrm{X_t}) + \mathcal{L}_{div}(f_t; \mathrm{X_t}) - \beta E_{(x_t, \hat{y}_t) \in \mathrm{X_t} \times \hat{\mathrm{Y}}_\mathrm{t}} \sum_{k=1}^{K} 1_{[K=\hat{y}_t]} \log \delta_k(f_t(x_t)) \tag{79}$$

where $\beta > 0$ is a balancing hyper-parameter.

Algorithm 2 SHOT algorithm

Require: Source hypothesis f_s, unlabeled target data, balancing hyperparameter β, number of epochs T.

Initialization: Freezing the source classifier $h_s = h_t$, and using g_s as an initialization.
for $i = 1 \rightarrow T$ **do**
    Generate self-supervised pseudo labels using Eq.(78).
    for $j = 1 \rightarrow n_{batch}$ **do**
        Obtain the pseudo labels for a given batch from target data.
        Using $\mathcal{L}(g_t)$ in Eq.(79), update g_t.
    end for
end for

4.3 Experimental setup and results

4.3.1 Datasets

For AR, we have evaluated the considered methods using the public dataset of The Washington State University (WSU) Center for Advanced Studies in Adaptive Systems (CASAS) [100]. It has been collected using several types of sensors such as motion sensors, door contact sensors, temperature sensors, and light switch sensors. It contains multiple tasks but we have only focused on predicting the following labels that are highly related to energy management: cooking breakfast, cooking lunch, cooking dinner, watching TV, and toileting [100]. For the evaluation task, we have considered predicting using five labels at first as mentioned in Table 1. Then, we reduced the number of labels to three labels and compared the obtained results. For OE, we have used datasets collected from two offices (H355 and H358) located in Grenoble Institute of Technology using several types of sensors such as door/window contact sensors, temperature sensors, CO2 concentration sensors, and power consumption sensors [25, 13, 14]. We have evaluated two tasks: three levels of occupancy and two levels of occupancy as mentioned in Table 2. For both tasks (AR and OE), we considered testing all the approaches with both balanced and unbalanced datasets to see the robustness of these methods to class proportion change. The used datasets for both tasks AR and OE contain several features and labels. In Figure 4.1, we visualize some of the features of the 3-label AR (Toileting, watching TV, and Cooking breakfast) dataset as well as some of the features of the 2-label OE. Based on the boxplots of the chosen features, we can see that there is no correlation between the AR features as well as the OE features.

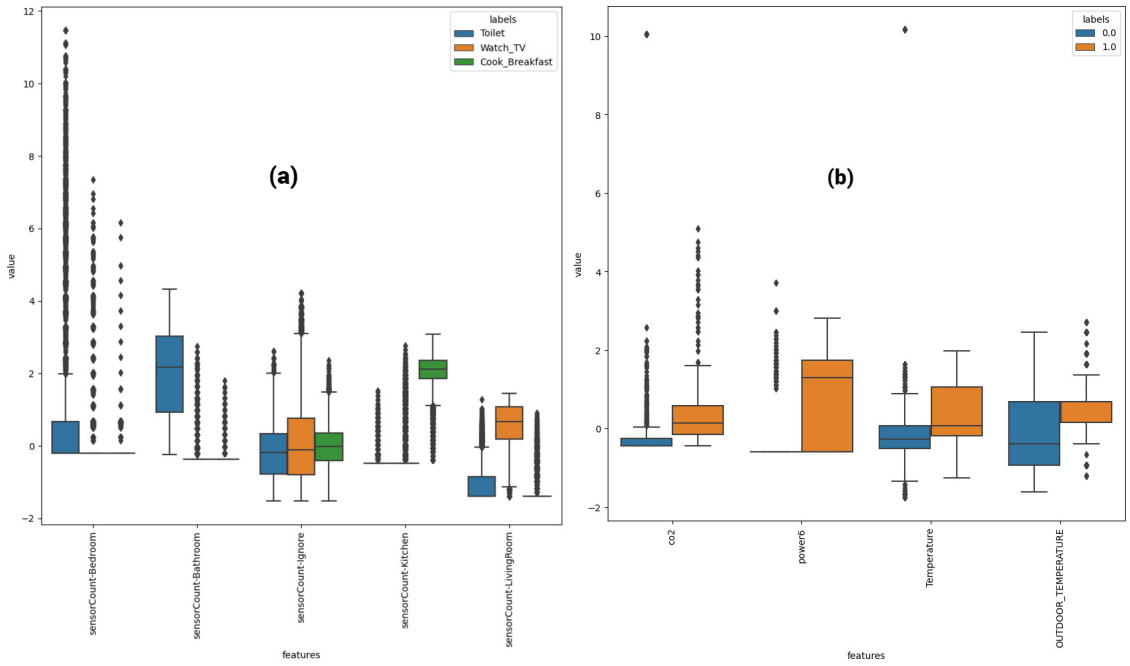

Figure 4.1: Features visualization for AR (a) and OE (b) datasets

4.3.2 Metrics

In this research, we have evaluated the considered approaches with multiple types of metrics depending on the scenario (balanced or unbalanced dataset). For balanced datasets, we have used a typical score which is the accuracy [25] defined as follows:

$$\textbf{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{80}$$

Table 4.2: Number of samples per class for AR and OE dataset

Labels	Source	Target
Watch TV	61312	3577
Cook Breakfast	16279	10167
Toilet	13727	10421
Cook Dinner	5557	7550
Cook Lunch	3197	5165
0 individual	1045	772
1 individual	177	191
2 individuals	148	78
3 individuals	44	27
4 individuals	26	-

For unbalanced datasets, the accuracy is not recommended as a score since it does not show the scores of the model prediction for each label. Thus, it does not show the failure of the model at some specific labels. For this reason, we have considered different scores to evaluate the obtained prediction such as the F1-score [25] defined as follows:

$$\textbf{Precision (Pr)} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FP_i} \tag{81}$$

$$\textbf{Recall (Rc)} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FN_i} \tag{82}$$

$$\textbf{F-score } (F_1) = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{83}$$

where TP is the number of true positive samples, TN is the number of true negative samples, FP is the number of false positive samples, and FN is the number of false negative samples.

4.3.3 Experimental results

We have developed new neural network architectures for most of the considered methods for the feature extractors and classifiers as presented in the shared code. For all the smart buildings tasks, we present and discuss the results for unbalanced and balanced datasets.

5-label AR

For the balanced dataset, the prediction task was poor for most of the methods due to the high complexity of the predicted labels as shown in Table 3. Indeed, we are predicting 5 labels which is a bit challenging to make a difference between them using the available dataset. CAT+RevGrad, using a teacher to cluster the unlabeled target data enhanced with gradient reversal, was able to give almost acceptable results for the considered task with 49% of accuracy. DaC, splitting the target data points into source-like points and target-specific points to make UDA, was able to provide good performance while predicting five labels of activities (56.08%). SHOT methods have given the best

accuracies with (65.40%) for SHOT enhanced with information maximization, and (68.00%) for SHOT with self-supervised pseudo-labels. In the current case, we can see the efficiency of labeling the unlabeled target data while training compared to the use of information maximization.

Table 4.3: Accuracies for AR with 5 balanced labels and F1-scores for AR with 5 unbalanced labels

Method	Accuracy ($\%$)	F1-score ($\%$)
DSN	20.66	23.12
CAT	38.00	55.64
CAT+RevGrad	49.00	51.15
CAT+rRevGrad	41.00	60.82
CoWA-JMDS	30.68	18.20
CoWA-JMDS w/o WM	24.40	10.85
DaC	56.08	36.65
AaD	33.19	14.67
SHOT-IM	65.40	**62.71**
SHOT-Pseudo-labeling	**68.00**	58.74

Extending our work to the unbalanced dataset with different label proportions aims to provide an idea about the robustness of the considered approach. Training with unequal label proportion has increased CAT methods performance giving 51.15% of accuracy for CAT+RevGrad, 55.64% of accuracy for CAT, and 60.82% of accuracy for CAT+rRevGrad as shown in Table 3. Indeed, CAT methods have gained more information from the unbalanced dataset thanks to the difference in label rates. CAT enhanced with robust gradient reversal has given good performance compared to the rest of the CAT methods. However, gradient reversal only was not beneficial for CAT+RevGrad with a decrease in performance compared to CAT due to the high complexity of the predicted task (5 labels). Indeed, gradient reversal and robust gradient reversal deal with target instances with low classification confidence having a high chance of being misclassified by the model while training. It was difficult to cluster data and align clusters using a teacher for DaC in the current scenario due to the predicted task complexity. Thus, we have seen a decrease in DaC performance compared to the previous scenario (balanced dataset). SHOT methods have given good results in this scenario. Unlike the balanced dataset, SHOT-Pseudo-labeling has given acceptable accuracy (58.74%), however, SHOT enhanced with information maximization was better than SHOT-Pseudo-labeling providing the best score of the current task with (62.71%) of accuracy.

3-label AR

By decreasing the complexity of the task (reducing the number of activities), we have seen an increase in most of the approaches' performance except one UDA with source data (DSN) as shown in Table 4. DSN finds private and shared information for source and target domains, and it uses this information to reconstruct features from both environments. The reason behind this low performance from DSN is the difficulty to distinguish between private and shared representations of features for the current OE task. CAT methods have given very good results with CAT+RevGrad having an excellent performance (87%) followed by CAT (78%) and CAT+rRevGrad (65.50%). Gradient reversal has made a huge impact on CAT method performance with 10% of accuracy increase. ATDOC+NC enhances the quality of the generated pseudo-labels by creating the nearest centroid classifier and using a memory mechanism to store all the information related to the unlabeled target data. ATDOC+NC has given a good score with 84.00% of accuracy. Applying CoWA-JMDS with weight mixup has given 65.07% of accuracy. However, eliminating weight mixup has reduced the accuracy of the method to 58%, and this demonstrates the efficiency of weight mixup for the current scenario. CoWA-JMDS considers samples' importance using a score called JMDS that exploits knowledge from both source and target domains. In the current task, weight mixup is so beneficial for the model's prediction by taking into consideration target samples with low confidence scores. AaD has given very good results (58.09%) compared to the previous task thanks to its clustering and assigning methodology. Moreover, the decrease in task complexity has helped increase AaD performance. AaD assumes that features from the same cluster should have the same class, and it uses this assumption to reduce the data distribution shift between the source and target domain by aligning features from the same clusters. SHOT-Pseudo-labeling kept giving good accuracy (78.80%), however, SHOT-IM has given the best score with 88.80%. Indeed, SHOT-IM takes into consideration the efficiency of information maximization while updating the weights of the target feature extractor in order to reduce the discrepancy between source and target domains.

Changing the proportion of labels did show some difference in performance for multiple methods as shown in Table 4. For UDA without source data, DaC keeps giving good results (79.78% of F1-score). Indeed, DaC divides target samples into source-like and target-specific instances using a

Table 4.4: Accuracies for AR with 3 balanced labels and F1-scores for AR with 3 unbalanced labels

Method	Accuracy (%)	F1-score (%)
DSN	39.00	45.54
CAT	78.00	80.00
CAT+RevGrad	87.00	75.79
CAT+rRevGrad	65.50	80.68
ATDOC+NC	84.00	87.35
CoWA-JMDS	65.07	46.79
CoWA-JMDS w/o WM	58.00	33.28
DaC	81.93	79.78
AaD	58.09	25.18
SHOT-IM	**88.80**	**93.31**
SHOT-Pseudo-labeling	78.80	87.70

source pre-trained model, then it uses a loss function based on maximum mean discrepancy (MMD) to mitigate the distribution shift between source and target domains. SHOT methods have given the best results with 87.70% of F1-score for SHOT-Pseudo-labeling, and 93.31% of F1-score for SHOT-IM. The obtained results prove the efficiency of the data distribution alignment between source and target domains employed by SHOT. However, we notice a decrease in performance for the rest of the methods which can be expected with some techniques due to the labels' proportion difference. For UDA with source data, CAT methods have shown good robustness to the unbalanced dataset with 80.68% of F1-score for CAT+rRevGrad. CAT and CAT+RevGrad have also shown good performance with 80% and 75.79% scores, respectively. In the current task, we can notice the efficiency of adding robust gradient reversal for the CAT method by considering samples with low prediction confidence while training. The difference in performance obtained by CAT+RevGrad compared to CAT can be explained by the use of an unbalanced dataset. ATDOC+NC has also given excellent performance which is not far from the best scores with 87.35% of F1-score.

3-label OE

For balanced datasets, estimating occupancy with 3 labels has very good results with CAT methods for UDA with source data as shown in Table 5. Indeed, CAT results were comparable with a small increase in performance when adding gradient reversal (73% of accuracy). Adding robust gradient reversal to the CAT approach did not enhance performance (68.75% of accuracy) compared

to only adding gradient reversal. The small increase in performance while adding gradient reversal shows, for the current task, that the performance is based on the discriminative clustering for features made by the CAT method. ATDOC+NC has given acceptable performance with 68.00% of accuracy. For source-free approaches, CoWA-JMDS has given poor performance compared to the rest of the methods. CoWA-JMDS uses the JMDS confidence score to evaluate samples' importance based on the predicted pseudo-labels for the unlabeled target instances. The reason behind the obtained performance for CoWA-JMDS can be explained by the accuracy of confidence scores obtained using JMDS for the target pseudo-labels. Moreover, weight mixup was not beneficial for CoWA-JMDS. Indeed, its performance without adding weight mixup is greater than when adding it (40.59% to 49.05%), and this proves that weight mixup was misleading in the current task. DaC, based on confidence scores of the predicted pseudo-labels, has given good results with 66.84% of accuracy. AaD has the best performance for source-free methods with 70.04% of accuracy. Indeed, it mitigates distribution shifts between source and target domains by clustering and assigning close features. SHOT-IM using information maximization while mitigating the discrepancy between source and target domains has given the best accuracy which is slightly better than CAT+RevGrad (73.80%).

Table 4.5: Accuracies for OE with 3 balanced labels and F1-scores for OE with 3 unbalanced labels

Method	Accuracy (%)	F1-score (%)
DSN	34.80	57.01
CAT	72.40	66.67
CAT+RevGrad	73.00	61.54
CAT+rRevGrad	68.75	65.31
ATDOC+NC	68.00	64.24
CoWA-JMDS	40.59	54.72
CoWA-JMDS w/o WM	49.05	55.51
DaC	66.84	80.68
AaD	70.04	76.79
SHOT-IM	**73.80**	**82.57**
SHOT-Pseudo-labeling	57.20	**82.57**

Extending the prediction tasks for three unbalanced labels, we aim to evaluate the efficiency of the considered methods with the current tasks (OE). Changing the label's proportion has brought enhancement and new information to almost all the methods as shown in Table 5. Indeed, we can

see that all the methods are giving acceptable and good results. For UDA with source data methods, DSN got enhanced with 57.01% of F-score. Indeed, DSN extracts representations that are shared between source and target domains and representations that are private to each domain to make UDA. Compared to the balanced dataset, CAT methods have seen a small decrease in performance, but they are still having good performances with 66.67% of F-score for the CAT method without gradient reversal. ATDOC+NC has given good performance with 64.24% of F1-score. For source-free methods, both CoWA-JMDS with and without weight mixup got enhanced with 54.72% and 55.51% of F-score, respectively. AaD got a big increase in F-score with 6%, and DaC got a dramatic increase in F-score with 14% reaching the score compared to all the methods (80.68%). SHOT methods keep dominating all the rest of the approaches with an excellent F1-score equal to 82.57% thanks to the self-supervised pseudo-labeling and the information maximization techniques.

2-label OE

Reducing the complexity of occupancy levels from three states to two states has led to a huge increase in the considered methods scores which is expected as explained in [25]. Indeed, for a machine learning model, it is easier to predict data with fewer labels [25]. For the balanced dataset, Table 6 illustrates the obtained accuracies. Indeed, with direct use of source data, CAT methods have the best scores (around 89.80% of accuracy) compared to the DSN with 56% of accuracy which is still considered a good result. For the rest of the methods that do not use labeled source data, CoWA-JMDS has been proven again to have greater results without weight mixup (76.55%) than with weight mixup (85.88%). AaD has greater results than CoWA-JMDS w/o WM with 2% of accuracy. DaC has given the best performance of all the considered UDA methods without source data for the current task with 89.51% of accuracy. SHOT methods, enhanced with pseudo-labels and information maximization, have also given good results with 86.40% of accuracy. ATDOC+NA creates a large memory bank storing the features with their predictions to train its classifier. Using prediction aggregation and a confidence-weighted classification loss, the NA classifier is being dynamically updated. In the current scenario, ATDOC+NA has given excellent scores with 88.68% of accuracy which is not far from the best accuracy (89.80%).

Evaluating the current task of occupancy estimation with the unbalanced dataset has led to a

Table 4.6: Accuracies for OE with 2 balanced labels and F1-scores for OE with 2 unbalanced labels

Method	Accuracy (%)	F1-score (%)
DSN	56.00	63.87
CAT	**89.80**	65.22
CAT+RevGrad	88.80	68.18
CAT+rRevGrad	89.50	73.68
ATDOC+NC	82.65	86.19
ATDOC+NA	88.68	**90.57**
CoWA-JMDS	76.55	72.31
CoWA-JMDS w/o WM	85.88	72.31
DaC	89.51	87.77
AaD	87.30	80.07
SHOT-IM	86.40	87.62
SHOT-Pseudo-labeling	86.40	87.63

change in performance for most of the methods. Indeed, some source-dependent methods like DSN have seen an increase in performance with 63.87%. However, CAT methods have seen a decrease in performance compared to the previous tasks reaching 65.22% for the CAT method without reversal gradient which can be explained by the change in the label's proportion. For source-free methods, most of the approaches remain with the same performance except CoWA-JMDS w/o WM which has seen a decrease reaching 72.31% of F-score, and AaD reaching 80.07% of F-score due to label's proportion change. DaC has given an excellent performance with 87.77% of the score which is a good achievement. However, SHOT methods are not so far from DaC with excellent scores for both approaches (87.62%). ATDOC+NA has given the best performance compared to all the methods with 90.57% of F1-score which is a great achievement for the considered method.

SHOT-IM: label imbalance

In this section, we evaluate the effect of the label's proportion imbalance for the best method SHOT-IM. The method uses a source pre-trained model which has a good performance while classifying data samples from the source domain. It transfers the weights from the source hypothesis module to the target classifier module and freezes them during the whole training process. However, it learns a new feature encoding module so that the discrepancy between the output of the target feature extractor and the source feature extractor is minimum [74, 88]. The information maximization (IM) has been used to learn the feature extractor module of the target model [74, 88, 97, 93]. For

the label's proportion, we considered 4 levels: level 1: labels are balanced, level 2: the major label is 70% of the total labels, level 3: the major label is 80% of the total labels, and level 4: the major label is 90% of the total labels. Based on the obtained results, we can see that although the label's imbalance, SHOT-IM continues to give excellent F1-scores. For almost all the tasks, as long as we increase the unbalanced level, SHOT-IM performance increases or remains the same. This experiment shows how efficient is the SHOT-IM method for label imbalance.

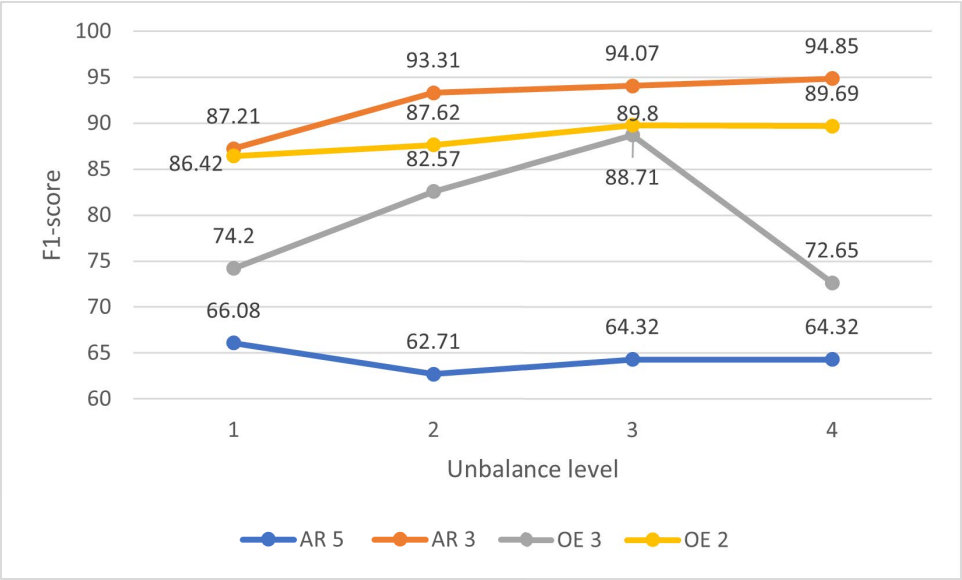

Figure 4.2: Label imbalance for SHOT-IM method

4.3.4 Comparison and discussion

For all the considered UDA methods with and without the use of source data, we notice a close performance for each task (AR and OE) as shown in Figure 4.3 and Figure 4.4. Indeed, if we move from left to right in both figures with the 12 considered methods on the x-axis, we can notice that most of the time the difference in performance (scores) is not huge. Also, we can see that most of the methods have given good scores for all the tasks. Decreasing the complexity of the tasks by decreasing the number of labels (AR and OE) has all the time a positive effect on the models' performances. Indeed, we can see all the time the curve of 2-label OE is above the curve of 3-label OE, and the curve of 3-label AR is all the time above the curve of 5-label AR. The developed methods have been tested on only two smart building tasks (AR and OE). However, they

can generalize to any smart building application that is based on sensor data which is a potential achievement for our research. Also, they can be applied to any AI area as long as the data is 1-dimensional. UDA with source data has direct access to data from the source domains like the behavior of occupants (activities) which can lead to privacy issues [25, 13, 24, 39, 40, 41, 42]. Data privacy is a topic of interest in smart building applications that most researchers try to preserve by developing techniques that get advantage of data while protecting its privacy. In contrast, source-free UDA does not get access to the generated data, but it uses only trained models that do not contain any private information about people which protects residents' privacy. The considered methods are trained with unlabeled target data (unsupervised approaches), and they have given very good results for most of the tasks which is a good achievement for our research.

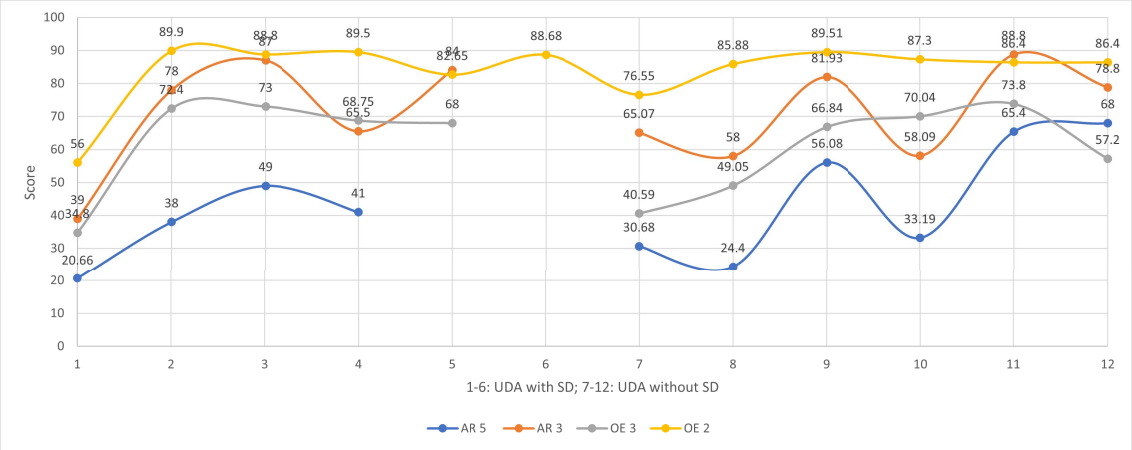

Figure 4.3: Accuracies for all the methods with balanced labels

In our current case, we have good and close performance between both scenarios (UDA with and without source data) and we have source-free methods with the advantages of preserving privacy compared to the rest of the techniques. We suggest choosing the scenario that preserves privacy which is source-free UDA.

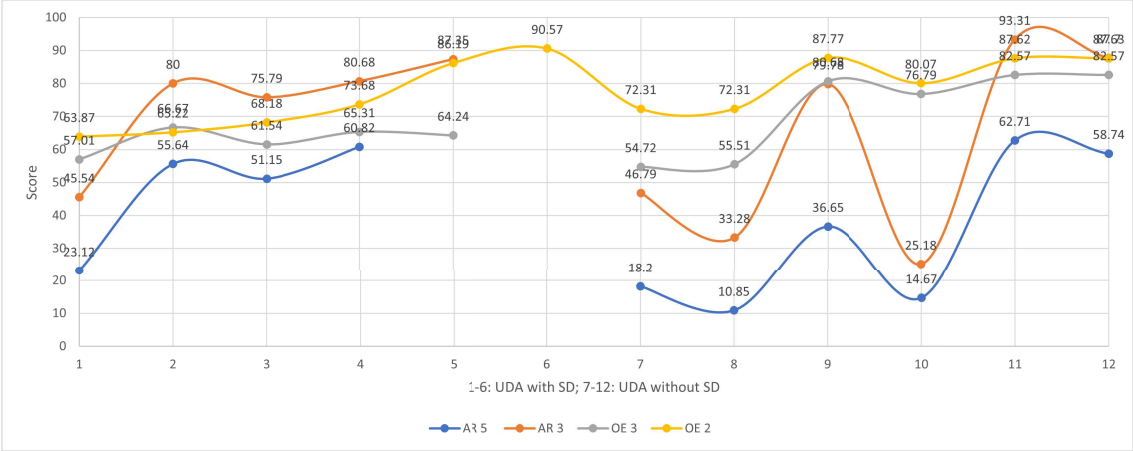

Figure 4.4: F1-scores for all the methods with unbalanced labels

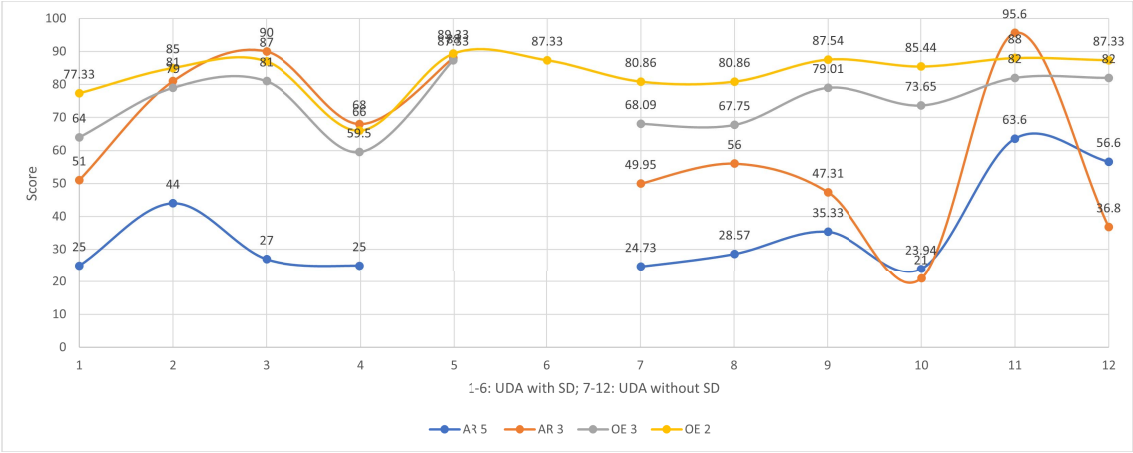

Figure 4.5: Accuracies for all the methods with unbalanced labels

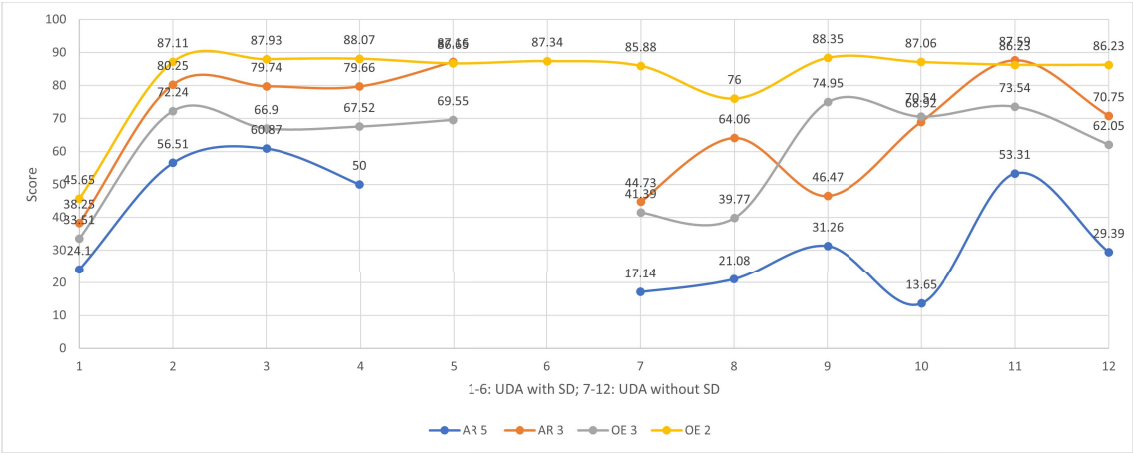

Figure 4.6: F1-scores for all the methods with balanced labels

Chapter 5

Conclusion

In this thesis, firstly, we consider unsupervised domain adaptation methods, that do not have direct access to labeled source data, called SHOT, SFDA, and HoMM. SHOT exploits information maximization and self-supervised pseudo-labeling to create a target feature encoder so that the target data distribution fits the target classifier. It freezes the classifier from the source. HoMM also uses pseudo-labeling for the target samples to enhance data alignment. Indeed, it freezes the feature extractor of the source model and it creates a new classifier for the target model. SFDA progressively updates a target model by selecting reliable target samples using source knowledge. All methods use a pre-trained model from the source domain instead of source data to preserve data privacy. In this research, we have provided new model architectures of the different methods to fit the smart building data. SHOT, SFDA, and HoMM have given good results in all the tasks of AR and OE with balanced and unbalanced datasets. On average, SHOT is better than SFDA and HoMM, but there are some tasks where SFDA and HoMM outperform SHOT. Decreasing the complexity of the task by reducing the number of activities or the number of occupancy levels, has shown a remarkable increase in the performance of the considered approaches. Even though all methods are unsupervised, they give scores comparable to supervised methods results, and in some cases, they exceed supervised machine learning methods' performances.

Secondly, we considered unsupervised domain adaptation methods that use labeled source data and unlabeled target data to mitigate domain shifts between source and target environments. We have considered 5 UDA techniques called VADA, SWD, HAFN, SAFN, and SAFN+ENTM. VADA

combines domain adversarial training and a penalty term to punish the violation of cluster assumption. SWD uses task-specific decision boundaries from source and target domains and the Wasserstein distance to make domain adaptation. HAFN mitigates the domain discrepancy by limiting the expected feature norms of the source and target environments to a given scalar. SAFN enlarges mean features norms progressively to explore more information in features spaces. SAFN+ENTM enhances SAFN with entropy minimization to increase knowledge transfer across domains. All the considered techniques have been evaluated on AR and OE datasets for balanced and unbalanced label proportions. VADA has shown remarkable performance for all the considered tasks, and it has a high level of robustness for data poisoning. AFN methods have also shown great performances for most of the tasks with an acceptable level of robustness for mislabeled samples added to training data. Enlarging the mean features norms scale has enhanced the performance of the model in most cases. Also, entropy minimization has enhanced the performance of SAFN for several AR and OE tasks. SWD with Wasserstein discrepancy minimization did not perform well for the AR task. However, for unbalanced labels proportion, we have seen an enhancement for all AR predictions with SWD. VADA evaluated in the WiFi dataset has shown better performance in recognizing human activities than in previous research.

Finally, we have made a comparative analysis between unsupervised domain adaptation methods with and without labeled source data to estimate occupancy and recognize activities in smart buildings. We have evaluated 12 UDA methods for both scenarios (with and without source data). For UDA with source data, we considered DSN, CAT, CAT+RevGrad, CAT+rRevGrad, ATDOC+NC, and ATDOC+NA. DSN creates more meaningful data representations by extracting information that is shared between source and target domains, and information that is unique to each domain. CAT trains a teacher model on the source data to cluster labeled source samples and unlabeled target samples. Then, it uses the obtained clusters from both domains to align source and target clusters. CAT has been extended with gradient reversal (CAT+RevGrad), and robust gradient reversal (CAT+rRevGrad) to enhance domain adaptation performance. ATDOC methods enhance the quality of the generated pseudo-labels by creating two classifiers (NC and NA) and using a memory mechanism to store all the information related to the unlabeled target data. For source-free UDA,

we considered CoWA-JMDS, CoWA-JMDS w/o weight mixup, AaD, DaC, SHOT-IM, and SHOT-pseudo-labeling. CoWA-JMDS uses the JMDS confidence score to evaluate data points' importance based on the predicted pseudo-labels by exploiting knowledge from both domains. CoWA-JMDS has been extended to weight mixup to make more use of target data points with low confidence scores. AaD optimizes an objective that is based on the assumption that features from the same cluster or from close clusters have closer predictions than other features to adapt domains. DaC uses the confidence scores of the predicted pseudo-labels using the source pre-trained model to split the target data points into source-like points and target-specific points to make UDA. SHOT methods transfer the weights from the source hypothesis module to the target classifier module and freeze them during the whole training process, and they learn a new feature encoding module so that the discrepancy between the output of the target feature extractor and the source feature extractor is minimum. The considered methods can be applied to any sensor-comparable domain. Most of the methods have given excellent and comparable results for all the tasks of OE and AR which is a good achievement for smart building applications. Source-free UDA methods have the advantage of preserving privacy compared to the rest of the methods because they do not have direct access to the source data. Thus, we recommend the use of source-free UDA methods since they preserve privacy, especially with sensitive data.

All the considered methods have been adapted from a 2-dimensional environment (image data) to a 1-Dimensional environment (sensor data), and they can fit any 1-Dimensional data and not necessarily smart building data. The impressive results that we have obtained in this research prove the efficiency of the considered techniques.

Bibliography

[1] Q. Zhou, J. Xing, and Q. Yang, "Device-free occupant activity recognition in smart offices using intrinsic wi-fi components," *Building and Environment*, vol. 172, p. 106737, 2020.

[2] H. Chen, S. H. Cha, and T. W. Kim, "A framework for group activity detection and recognition using smartphone sensors and beacons," *Building and Environment*, vol. 158, pp. 205–216, 2019.

[3] S. H. Cha, J. Seo, S. H. Baek, and C. Koo, "Towards a well-planned, activity-based work environment: Automated recognition of office activities using accelerometers," *Building and Environment*, vol. 144, pp. 86–93, 2018.

[4] Y.-T. Chiang, C.-H. Lu, and J. Y.-j. Hsu, "A feature-based knowledge transfer framework for cross-environment activity recognition toward smart home applications," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 3, pp. 310–322, 2017.

[5] W.-H. Chen, P.-C. Cho, and Y.-L. Jiang, "Activity recognition using transfer learning," *Sens. Mater*, vol. 29, no. 7, pp. 897–904, 2017.

[6] W. Lu, F. Fan, J. Chu, P. Jing, and S. Yuting, "Wearable computing for internet of things: A discriminant approach for human activity recognition," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2749–2759, 2018.

[7] H. Zhang, Z. Xiao, J. Wang, F. Li, and E. Szczerbicki, "A novel iot-perceptive human activity recognition (har) approach using multihead convolutional attention," *IEEE Internet of Things Journal*, vol. 7, no. 2, pp. 1072–1080, 2019.

[8] H. Huang, X. Li, S. Liu, S. Hu, and Y. Sun, "Tribomotion: A self-powered triboelectric motion sensor in wearable internet of things for human activity recognition and energy harvesting," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4441–4453, 2018.

[9] D. Tao, Y. Wen, and R. Hong, "Multicolumn bidirectional long short-term memory for mobile devices-based human activity recognition," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1124–1134, 2016.

[10] Z. Chen, C. Jiang, and L. Xie, "Building occupancy estimation and detection: A review," *Energy and Buildings*, vol. 169, pp. 260–270, 2018.

[11] Y. Zhou, J. Chen, Z. J. Yu, J. Li, G. Huang, F. Haghighat, and G. Zhang, "A novel model based on multi-grained cascade forests with wavelet denoising for indoor occupancy estimation," *Building and Environment*, vol. 167, p. 106461, 2020.

[12] S. H. Ryu and H. J. Moon, "Development of an occupancy prediction model using indoor environmental data based on machine learning techniques," *Building and Environment*, vol. 107, pp. 1–9, 2016.

[13] M. Amayri, A. Arora, S. Ploix, S. Bandhyopadyay, Q.-D. Ngo, and V. R. Badarla, "Estimating occupancy in heterogeneous sensor environment," *Energy and Buildings*, vol. 129, pp. 46–58, 2016.

[14] M. Amayri and S. Ploix, "Decision tree and parametrized classifier for estimating occupancy in energy management," in *2018 5th International Conference on Control, Decision and Information Technologies (CoDIT)*, pp. 397–402, IEEE, 2018.

[15] L. Zimmermann, R. Weigel, and G. Fischer, "Fusion of nonintrusive environmental sensors for occupancy detection in smart homes," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2343–2352, 2017.

[16] S. Hu, P. Wang, C. Hoare, and J. O'Donnell, "Building occupancy detection and localisation using cctv camera and deep learning," *IEEE Internet of Things Journal*, 2022.

[17] B. S. Ciftler, S. Dikmese, İ. Güvenç, K. Akkaya, and A. Kadri, "Occupancy counting with burst and intermittent signals in smart buildings," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 724–735, 2017.

[18] M. Amayri, S. Ploix, N. Bouguila, and F. Wurtz, "Database quality assessment for interactive learning: Application to occupancy estimation," *Energy and Buildings*, vol. 209, p. 109578, 2020.

[19] J. B. Silva, M. Amayri, S. Ploix, P. Reignier, and C. S. Silva, "Cooperative and interactive learning to estimate human behaviours for energy applications," *Energy and Buildings*, vol. 258, p. 111727, 2022.

[20] S. S. Abolhassani, A. Zandifar, N. Ghourchian, M. Amayri, N. Bouguila, and U. Eicker, "Improving residential building energy simulations through occupancy data derived from commercial off-the-shelf wi-fi sensing technology," *Energy and Buildings*, vol. 272, p. 112354, 2022.

[21] O. Bouhamed, M. Amayri, and N. Bouguila, "Weakly supervised occupancy prediction using training data collected via interactive learning," *Sensors*, vol. 22, no. 9, p. 3186, 2022.

[22] Y. Benezeth, H. Laurent, B. Emile, and C. Rosenberger, "Towards a sensor for detecting human presence and characterizing activity," *Energy and Buildings*, vol. 43, no. 2-3, pp. 305–314, 2011.

[23] M. Aftab, C. Chen, C.-K. Chau, and T. Rahwan, "Automatic hvac control with real-time occupancy recognition and simulation-guided model predictive control in low-cost embedded system," *Energy and Buildings*, vol. 154, pp. 141–156, 2017.

[24] M. Aksoezen, M. Daniel, U. Hassler, and N. Kohler, "Building age as an indicator for energy consumption," *Energy and Buildings*, vol. 87, pp. 74–86, 2015.

[25] J. Dridi, M. Amayri, and N. Bouguila, "Transfer learning for estimating occupancy and recognizing activities in smart buildings," *Building and Environment*, vol. 217, p. 109057, 2022.

[26] K. Prabhakaran, J. Dridi, M. Amayri, and N. Bouguila, "Explainable k-means clustering for occupancy estimation," *Procedia Computer Science*, vol. 203, pp. 326–333, 2022.

[27] P. Kumar, C. Martani, L. Morawska, L. Norford, R. Choudhary, M. Bell, and M. Leach, "Indoor air quality and energy management through real-time sensing in commercial buildings," *Energy and Buildings*, vol. 111, pp. 145–153, 2016.

[28] H. Zhang, A. Davigny, F. Colas, Y. Poste, and B. Robyns, "Fuzzy logic based energy management strategy for commercial buildings integrating photovoltaic and storage systems," *Energy and Buildings*, vol. 54, pp. 196–206, 2012.

[29] J. Clarke, J. Cockroft, S. Conner, J. Hand, N. Kelly, R. Moore, T. O'brien, and P. Strachan, "Simulation-assisted control in building energy management systems," *Energy and buildings*, vol. 34, no. 9, pp. 933–940, 2002.

[30] R. Missaoui, H. Joumaa, S. Ploix, and S. Bacha, "Managing energy smart homes according to energy prices: analysis of a building energy management system," *Energy and Buildings*, vol. 71, pp. 155–167, 2014.

[31] V. Chidurala and X. Li, "Occupancy estimation using thermal imaging sensors and machine learning algorithms," *IEEE Sensors Journal*, vol. 21, no. 6, pp. 8627–8638, 2021.

[32] S. Munir, R. S. Arora, C. Hesling, J. Li, J. Francis, C. Shelton, C. Martin, A. Rowe, and M. Berges, "Real-time fine grained occupancy estimation using depth sensors on arm embedded platforms," in *2017 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pp. 295–306, IEEE, 2017.

[33] A. Tyndall, R. Cardell-Oliver, and A. Keating, "Occupancy estimation using a low-pixel count thermal imager," *IEEE Sensors Journal*, vol. 16, no. 10, pp. 3784–3791, 2016.

[34] P. W. Tien, J. K. Calautit, J. Darkwa, C. Wood, S. Wei, C. A. J. Pantua, and W. Xu, "A deep learning framework for energy management and optimisation of hvac systems," in *IOP Conference Series: Earth and Environmental Science*, vol. 463, p. 012026, IOP Publishing, 2020.

[35] R. Rana, B. Kusy, J. Wall, and W. Hu, "Novel activity classification and occupancy estimation methods for intelligent hvac (heating, ventilation and air conditioning) systems," *Energy*, vol. 93, pp. 245–255, 2015.

[36] N. Zamzami, M. Amayri, N. Bouguila, and S. Ploix, "Online clustering for estimating occupancy in an office setting," in *2019 IEEE 28th International Symposium on Industrial Electronics (ISIE)*, pp. 2195–2200, IEEE, 2019.

[37] J. P. Real, C. Rasmussen, R. Li, K. Leerbeck, O. M. Jensen, K. B. Wittchen, and H. Madsen, "Characterisation of thermal energy dynamics of residential buildings with scarce data," *Energy and Buildings*, vol. 230, p. 110530, 2021.

[38] B. Chidlovskii, S. Clinchant, and G. Csurka, "Domain adaptation in the absence of source domain data," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 451–460, 2016.

[39] M. W. Ahmad, M. Mourshed, D. Mundow, M. Sisinni, and Y. Rezgui, "Building energy metering and environmental monitoring–a state-of-the-art review and directions for future research," *Energy and Buildings*, vol. 120, pp. 85–102, 2016.

[40] H. Kazmi, F. Mehmood, and M. Amayri, "Smart home futures: Algorithmic challenges and opportunities," in *2017 14th International Symposium on Pervasive Systems, Algorithms and Networks & 2017 11th International Conference on Frontier of Computer Science and Technology & 2017 Third International Symposium of Creative Computing (ISPAN-FCST-ISCC)*, pp. 441–448, IEEE, 2017.

[41] Z. Luo, M. Amayri, W. Fan, and N. Bouguila, "Cross-collection latent beta-liouville allocation model training with privacy protection and applications," *Applied Intelligence*, pp. 1–25, 2023.

[42] M. Amayri, S. Ploix, N. Bouguila, and F. Wurtz, "Estimating occupancy using interactive learning with a sensor environment: Real-time experiments," *IEEE Access*, vol. 7, pp. 53932–53944, 2019.

[43] A. K. Sharma and N. K. Verma, "Quick learning mechanism with cross-domain adaptation for intelligent fault diagnosis," *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 3, pp. 381–390, 2021.

[44] Y. Kim, D. Cho, K. Han, P. Panda, and S. Hong, "Domain adaptation without source data," *arXiv preprint arXiv:2007.01524*, 2020.

[45] I. Kalita and M. Roy, "Deep neural network-based heterogeneous domain adaptation using ensemble decision making in land cover classification," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 2, pp. 167–180, 2020.

[46] Y. Li, Y. Sun, K. Horoshenkov, and S. M. Naqvi, "Domain adaptation and autoencoder-based unsupervised speech enhancement," *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 1, pp. 43–52, 2021.

[47] T. Kyono and M. Van der Schaar, "Exploiting causal structure for robust model selection in unsupervised domain adaptation," *IEEE Transactions on Artificial Intelligence*, vol. 2, no. 6, pp. 494–507, 2021.

[48] Y. Zhu, X. Wu, Y. Li, J. Qiang, and Y. Yuan, "Self-adaptive imbalanced domain adaptation with deep sparse autoencoder," *IEEE Transactions on Artificial Intelligence*, 2022.

[49] A. Braytee, M. Naji, and P. J. Kennedy, "Unsupervised domain-adaptation-based tensor feature learning with structure preservation," *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 3, pp. 370–380, 2022.

[50] S. Dhar, N. D. Jana, and S. Das, "An adaptive learning based generative adversarial network for one-to-one voice conversion," *IEEE Transactions on Artificial Intelligence*, 2022.

[51] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

[52] D. Cook, K. D. Feuz, and N. C. Krishnan, "Transfer learning for activity recognition: A survey," *Knowledge and information systems*, vol. 36, no. 3, pp. 537–556, 2013.

[53] G. Wilson and D. J. Cook, "A survey of unsupervised deep domain adaptation," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 11, no. 5, pp. 1–46, 2020.

[54] G. Csurka, "Domain adaptation for visual applications: A comprehensive survey," *arXiv preprint arXiv:1702.05374*, 2017.

[55] L. Duan, I. W. Tsang, D. Xu, and S. J. Maybank, "Domain transfer svm for video concept detection," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1375–1381, IEEE, 2009.

[56] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE transactions on neural networks*, vol. 22, no. 2, pp. 199–210, 2010.

[57] M. Baktashmotlagh, M. T. Harandi, B. C. Lovell, and M. Salzmann, "Unsupervised domain adaptation by domain invariant projection," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 769–776, 2013.

[58] G. Matasci, M. Volpi, M. Kanevski, L. Bruzzone, and D. Tuia, "Semisupervised transfer component analysis for domain adaptation in remote sensing image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 7, pp. 3550–3564, 2015.

[59] E. Zhong, W. Fan, J. Peng, K. Zhang, J. Ren, D. Turaga, and O. Verscheure, "Cross domain distribution adaptation via kernel mapping," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1027–1036, 2009.

[60] A. G. Prabono, B. N. Yahya, and S.-L. Lee, "Hybrid domain adaptation with deep network architecture for end-to-end cross-domain human activity recognition," *Computers & Industrial Engineering*, vol. 151, p. 106953, 2021.

[61] C. Shen, Y. Chen, G. Yang, and X. Guan, "Toward hand-dominated activity recognition systems with wristband-interaction behavior analysis," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 7, pp. 2501–2511, 2018.

[62] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu, "Sensor-based activity recognition," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 790–808, 2012.

[63] M. A. A. H. Khan, N. Roy, and A. Misra, "Scaling human activity recognition via deep learning-based domain adaptation," in *2018 IEEE international conference on pervasive computing and communications (PerCom)*, pp. 1–9, IEEE, 2018.

[64] C. Zhu and W. Sheng, "Wearable sensor-based hand gesture and daily activity recognition for robot-assisted living," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 41, no. 3, pp. 569–573, 2011.

[65] J. Wannenburg and R. Malekian, "Physical activity recognition from smartphone accelerometer data for user context awareness sensing," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 12, pp. 3142–3149, 2016.

[66] A. Akbari and R. Jafari, "Transferring activity recognition models for new wearable sensors with deep generative domain adaptation," in *Proceedings of the 18th International Conference on Information Processing in Sensor Networks*, pp. 85–96, 2019.

[67] S. An, A. Medda, M. N. Sawka, C. J. Hutto, M. L. Millard-Stafford, S. Appling, K. L. Richardson, and O. T. Inan, "Adaptnet: Human activity recognition via bilateral domain adaptation using semi-supervised deep translation networks," *IEEE Sensors Journal*, vol. 21, no. 18, pp. 20398–20411, 2021.

[68] J. Zhao, F. Deng, H. He, and J. Chen, "Local domain adaptation for cross-domain activity recognition," *IEEE Transactions on Human-Machine Systems*, vol. 51, no. 1, pp. 12–21, 2020.

[69] Z. Zhou, Y. Zhang, X. Yu, P. Yang, X.-Y. Li, J. Zhao, and H. Zhou, "Xhar: Deep domain adaptation for human activity recognition with smart devices," in *2020 17th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pp. 1–9, IEEE, 2020.

[70] A. R. Sanabria and J. Ye, "Unsupervised domain adaptation for activity recognition across heterogeneous datasets," *Pervasive and Mobile Computing*, vol. 64, p. 101147, 2020.

[71] T. Zhang and O. Ardakanian, "A domain adaptation technique for fine-grained occupancy estimation in commercial buildings," in *Proceedings of the International Conference on Internet of Things Design and Implementation*, pp. 148–159, 2019.

[72] I. B. Arief-Ang, F. D. Salim, and M. Hamilton, "Da-hoc: semi-supervised domain adaptation for room occupancy prediction using co2 sensor data," in *Proceedings of the 4th ACM International Conference on Systems for Energy-Efficient Built Environments*, pp. 1–10, 2017.

[73] I. B. Arief-Ang, M. Hamilton, and F. D. Salim, "A scalable room occupancy prediction with transferable time series decomposition of co2 sensor data," *ACM Transactions on Sensor Networks (TOSN)*, vol. 14, no. 3-4, pp. 1–28, 2018.

[74] J. Liang, D. Hu, and J. Feng, "Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation," in *International Conference on Machine Learning*, pp. 6028–6039, PMLR, 2020.

[75] C. Chen, Z. Fu, Z. Chen, S. Jin, Z. Cheng, X. Jin, and X.-S. Hua, "Homm: Higher-order moment matching for unsupervised domain adaptation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, pp. 3422–3429, 2020.

[76] J. Dridi, M. Amayri, and N. Bouguila, "Unsupervised domain adaptation without source data for estimating occupancy and recognizing activities in smart buildings," *Energy and Buildings*, submitted.

[77] R. Shu, H. H. Bui, H. Narui, and S. Ermon, "A dirt-t approach to unsupervised domain adaptation," *arXiv preprint arXiv:1802.08735*, 2018.

[78] C.-Y. Lee, T. Batra, M. H. Baig, and D. Ulbricht, "Sliced wasserstein discrepancy for unsupervised domain adaptation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10285–10295, 2019.

[79] R. Xu, G. Li, J. Yang, and L. Lin, "Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1426–1435, 2019.

[80] A. Mehra, B. Kailkhura, P.-Y. Chen, and J. Hamm, "Understanding the limits of unsupervised domain adaptation via data poisoning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17347–17359, 2021.

[81] J. Dridi, M. Amayri, and N. Bouguila, "Unsupervised domain adaptation with source data for estimating occupancy and recognizing activities in smart buildings," *IEEE Transactions on Artificial Intelligence*, submitted.

[82] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," *Advances in neural information processing systems*, vol. 29, 2016.

[83] Z. Deng, Y. Luo, and J. Zhu, "Cluster alignment with a teacher for unsupervised domain adaptation," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9944–9953, 2019.

[84] J. Liang, D. Hu, and J. Feng, "Domain adaptation with auxiliary target domain-oriented classifier," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16632–16642, 2021.

[85] J. Lee, D. Jung, J. Yim, and S. Yoon, "Confidence score for source-free unsupervised domain adaptation," in *International Conference on Machine Learning*, pp. 12365–12377, PMLR, 2022.

[86] Z. Zhang, W. Chen, H. Cheng, Z. Li, S. Li, L. Lin, and G. Li, "Divide and contrast: Source-free domain adaptation via adaptive contrastive learning," *arXiv preprint arXiv:2211.06612*, 2022.

[87] S. Yang, Y. Wang, K. Wang, S. Jui, *et al.*, "Attracting and dispersing: A simple approach for source-free domain adaptation," in *Advances in Neural Information Processing Systems*, 2022.

[88] J. Liang, D. Hu, Y. Wang, R. He, and J. Feng, "Source data-absent unsupervised domain adaptation through hypothesis transfer and labeling transfer," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 8602–8617, 2021.

[89] J. Dridi, M. Amayri, and N. Bouguila, "Unsupervised domain adaptation with and without access to source data for estimating occupancy and recognizing activities in smart buildings," *Building and Environment*, accepted.

[90] J. A. Pinzon, P. P. Vergara, L. C. Da Silva, and M. J. Rider, "Optimal management of energy consumption and comfort for smart buildings operating in a microgrid," *IEEE Transactions on Smart Grid*, vol. 10, no. 3, pp. 3236–3247, 2018.

[91] I. Kuzborskij and F. Orabona, "Stability and hypothesis transfer learning," in *International Conference on Machine Learning*, pp. 942–950, PMLR, 2013.

[92] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7167–7176, 2017.

[93] W. Hu, T. Miyato, S. Tokui, E. Matsumoto, and M. Sugiyama, "Learning discrete representations via information maximizing self-augmented training," in *International conference on machine learning*, pp. 1558–1567, PMLR, 2017.

[94] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, pp. 448–456, PMLR, 2015.

[95] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," *Advances in neural information processing systems*, vol. 29, pp. 901–909, 2016.

[96] Y. Gao, W. Wang, C. Herold, Z. Yang, and H. Ney, "Towards a better understanding of label smoothing in neural machine translation," in *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pp. 212–223, 2020.

[97] A. Krause, P. Perona, and R. Gomes, "Discriminative clustering by regularized information maximization," *Advances in neural information processing systems*, vol. 23, 2010.

[98] D.-H. Lee *et al.*, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Workshop on challenges in representation learning, ICML*, vol. 3, p. 896, 2013.

[99] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 132–149, 2018.

[100] D. J. Cook, "Learning setting-generalized activity models for smart spaces," *IEEE intelligent systems*, vol. 2010, no. 99, p. 1, 2010.

[101] S. Liu, P. Reviriego, X. Tang, W. Tang, and F. Lombardi, "Result-based re-computation for error-tolerant classification by a support vector machine," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 1, pp. 62–73, 2020.

[102] U. K. Dutta, M. Harandi, and C. C. Sekhar, "Unsupervised deep metric learning via orthogonality based probabilistic loss," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 1, pp. 74–84, 2020.

[103] S. Niu, Y. Liu, J. Wang, and H. Song, "A decade survey of transfer learning (2010–2020)," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 2, pp. 151–166, 2020.

[104] M. Amiribesheli and H. Bouchachia, "A tailored smart home for dementia care," *Journal of Ambient Intelligence and Humanized Computing*, vol. 9, pp. 1755–1782, 2018.

[105] D. Djenouri, R. Laidi, Y. Djenouri, and I. Balasingham, "Machine learning for smart building applications: Review and taxonomy," *ACM Computing Surveys (CSUR)*, vol. 52, no. 2, pp. 1–36, 2019.

[106] L. Yu, S. Qin, M. Zhang, C. Shen, T. Jiang, and X. Guan, "A review of deep reinforcement learning for smart building energy management," *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 12046–12063, 2021.

[107] K. Alanne and S. Sierla, "An overview of machine learning applications for smart buildings," *Sustainable Cities and Society*, vol. 76, p. 103445, 2022.

[108] J. Guo, M. Amayri, F. Najar, W. Fan, and N. Bouguila, "Occupancy estimation in smart buildings using predictive modeling in imbalanced domains," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–13, 2022.

[109] A. Arora, M. Amayri, V. Badarla, S. Ploix, and S. Bandyopadhyay, "Occupancy estimation using non intrusive sensors in energy efficient buildings," in *14th Conference of International Building Performance Simulation Association, Hyderabad, India, Dec. 7-9, 2015.*, 2015.

[110] M. Amayri, S. Ali, N. Bouguila, and S. Ploix, "Machine learning for activity recognition in smart buildings: A survey," *Towards Energy Smart Homes: Algorithms, Technologies, and Applications*, pp. 199–228, 2021.

[111] N. Manouchehri, O. Dalhoumi, M. Amayri, and N. Bouguila, "Variational learning of a shifted scaled dirichlet model with component splitting approach," in *2020 Third International Conference on Artificial Intelligence for Industries (AI4I)*, pp. 75–78, IEEE, 2020.

[112] O. Dalhoumi, M. Amayri, and N. Bouguila, "A review of neural networks for buildings occupancy measurement," in *2022 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, pp. 29–35, IEEE, 2022.

[113] A. Benmansour, A. Bouchachia, and M. Feham, "Multioccupant activity recognition in pervasive smart home environments," *ACM Computing Surveys (CSUR)*, vol. 48, no. 3, pp. 1–36, 2015.

[114] S. Mohamad, M. Sayed-Mouchaweh, and A. Bouchachia, "Online active learning for human activity recognition from sensory data streams," *Neurocomputing*, vol. 390, pp. 341–358, 2020.

[115] K. Akkaya, I. Guvenc, R. Aygun, N. Pala, and A. Kadri, "Iot-based occupancy monitoring techniques for energy-efficient smart buildings," in *2015 IEEE Wireless communications and networking conference workshops (WCNCW)*, pp. 58–63, IEEE, 2015.

[116] M. Azam, M. Blayo, J.-S. Venne, and M. Allegue-Martinez, "Occupancy estimation using wifi motion detection via supervised machine learning algorithms," in *2019 ieee global conference on signal and information processing (GlobalSIP)*, pp. 1–5, IEEE, 2019.

[117] Q. Zhu, Z. Chen, and Y. C. Soh, "A novel semisupervised deep learning method for human activity recognition," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 3821–3830, 2018.

[118] D. Tuia, C. Persello, and L. Bruzzone, "Domain adaptation for the classification of remote sensing data: An overview of recent advances," *IEEE geoscience and remote sensing magazine*, vol. 4, no. 2, pp. 41–57, 2016.

[119] Y. Grandvalet and Y. Bengio, "Semi-supervised learning by entropy minimization," *Advances in neural information processing systems*, vol. 17, 2004.

[120] I. K. Ihianle, A. O. Nwajana, S. H. Ebenuwa, R. I. Otuka, K. Owa, and M. O. Orisatoki, "A deep learning approach for human activities recognition from multimodal sensing devices," *IEEE Access*, vol. 8, pp. 179028–179038, 2020.

[121] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[122] K. Yan, X. Zhou, and B. Yang, "Ai and iot applications of smart buildings and smart environment design, construction and maintenance," 2022.

[123] M. Shahinmoghadam, W. Natephra, and A. Motamedi, "Bim-and iot-based virtual reality tool for real-time thermal comfort assessment in building enclosures," *Building and environment*, vol. 199, p. 107905, 2021.

[124] Z. D. Tekler, R. Low, C. Yuen, and L. Blessing, "Plug-mate: An iot-based occupancy-driven plug load management system in smart buildings," *Building and Environment*, vol. 223, p. 109472, 2022.

[125] Q. Li, X. Wang, P. Wang, W. Zhang, and J. Yin, "Farda: A fog-based anonymous reward data aggregation security scheme in smart buildings," *Building and Environment*, vol. 225, p. 109578, 2022.

[126] K. Li, J. Zhao, J. Hu, and Y. Chen, "Dynamic energy efficient task offloading and resource allocation for noma-enabled iot in smart buildings and environment," *Building and Environment*, vol. 226, p. 109513, 2022.

[127] J. Xu, D. Li, W. Gu, and Y. Chen, "Uav-assisted task offloading for iot in smart buildings and environment via deep reinforcement learning," *Building and Environment*, vol. 222, p. 109218, 2022.

[128] J. Cui, J. Pan, S. Wang, M. O. Okoye, J. Yang, Y. Li, and H. Wang, "Improved normal-boundary intersection algorithm: A method for energy optimization strategy in smart buildings," *Building and Environment*, vol. 212, p. 108846, 2022.

[129] R. Galvin, "Policy pressure to retrofit germany's residential buildings to higher energy efficiency standards: A cost-effective way to reduce co2 emissions?," *Building and Environment*, p. 110316, 2023.

[130] R. Slabe-Erker, M. Dominko, A. Bayar, B. Majcen, and K. Primc, "Energy efficiency in residential and non-residential buildings: Short-term macroeconomic implications," *Building and Environment*, vol. 222, p. 109364, 2022.

[131] W. Du, M. Li, Y. Wang, X. Ma, C. Hu, Y. Zhang, and Z. Zhang, "Dynamic energy efficiency characteristics analysis of a distributed solar photovoltaic direct-drive solar cold storage," *Building and Environment*, vol. 206, p. 108324, 2021.

[132] W. Duan, Y. Wang, J. Li, Y. Zheng, C. Ning, and P. Duan, "Real-time surveillance-video-based personalized thermal comfort recognition," *Energy and Buildings*, vol. 244, p. 110989, 2021.

[133] J. Kim, K. Min, M. Jung, and S. Chi, "Occupant behavior monitoring and emergency event detection in single-person households using deep learning-based sound recognition," *Building and Environment*, vol. 181, p. 107092, 2020.

[134] K. Rezaee, X. Yang, M. R. Khosravi, R. Zhang, W. Lin, and G. Jeon, "Fusion-based learning for stress recognition in smart home: an iomt framework," *Building and Environment*, vol. 216, p. 108988, 2022.

[135] Q. Zhou, Q. Yang, and J. Xing, "Enabling efficient wifi-based occupant behavior recognition using insufficient samples," *Building and Environment*, vol. 212, p. 108806, 2022.

[136] H. Nguyen, M. Rahmanpour, N. Manouchehri, K. Maanicshah, M. Amayri, and N. Bouguila, "A statistical approach for unsupervised occupancy detection and estimation in smart buildings," in *2019 IEEE International Smart Cities Conference (ISC2)*, pp. 414–419, IEEE, 2019.

[137] S. Ali and N. Bouguila, "Towards scalable deployment of hidden markov models in occupancy estimation: A novel methodology applied to the study case of occupancy detection," *Energy and Buildings*, vol. 254, p. 111594, 2022.

[138] S. M. Islam, A. Droitcour, E. Yavari, V. M. Lubecke, and O. Boric-Lubecke, "Building occupancy estimation using microwave doppler radar and wavelet transform," *Building and Environment*, vol. 236, p. 110233, 2023.

[139] K. Sun, P. Liu, T. Xing, Q. Zhao, and X. Wang, "A fusion framework for vision-based indoor occupancy estimation," *Building and Environment*, vol. 225, p. 109631, 2022.

[140] R. C. Navarro, A. R. Ruiz, F. J. V. Molina, M. J. S. Romero, J. D. Chaparro, D. V. Alises, and J. C. L. Lopez, "Indoor occupancy estimation for smart utilities: A novel approach based on depth sensors," *Building and Environment*, vol. 222, p. 109406, 2022.

[141] S. Sagawa, P. W. Koh, T. Lee, I. Gao, S. M. Xie, K. Shen, A. Kumar, W. Hu, M. Yasunaga, H. Marklund, *et al.*, "Extending the wilds benchmark for unsupervised adaptation," *arXiv preprint arXiv:2112.05090*, 2021.

[142] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in neural information processing systems*, vol. 27, 2014.

[143] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International conference on machine learning*, pp. 1180–1189, PMLR, 2015.

[144] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

[145] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," *Advances in neural information processing systems*, vol. 31, 2018.

[146] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.