Transfer Learning for Estimating Occupancy and Recognizing Activities in Smart Buildings

Jawher Dridi, Manar Amayri, Nizar Bouguila

Highlights

Transfer Learning for Estimating Occupancy and Recognizing Activities in Smart Buildings

Jawher Dridi,Manar Amayri,Nizar Bouguila

- Transfer learning methods are investigated in smart building applications (occupancy estimation and activities recognition).
- Five general apartment activities were recognized, and three levels of occupancy were estimated using real sensors data.
- Simple feature matching method is limited to transferring knowledge between similar contexts and requires a big amount of data.
- Imbalanced datasets problem is tackled using transfer learning with principal component analysis (PCA).
- Transfer learning helped develop general methods that recognize activities and estimate occupancy in case of lacking training data with accuracies between 90% and 91%.

Transfer Learning for Estimating Occupancy and Recognizing Activities in Smart Buildings

Jawher Dridi[a,*], Manar Amayri[b] and Nizar Bouguila[a]

[a]Concordia Institute for Information Systems Engineering (CIISE), Concordia University, Canada
[b]G-SCOP Lab, Grenoble Institute of Technology, France

ARTICLE INFO

Keywords:
Activities recognition
Occupancy estimation
Transfer learning
Smart buildings
PCA
SMOTE
Sparse coding

ABSTRACT

Activities Recognition (AR) and Occupancy Estimation (OE) are vital to many smart systems that work on providing good services in smart buildings. Many applications, such as energy management need information like activities and occupancy to provide good assistance. Most of the previous research about AR and OE focused on applying supervised machine learning methods. Researchers train a model and evaluate it using data collected from the same environment (Domain). A model trained in a specific domain will not generalize well in other domains. Creating a trained model to every environment is not feasible due to the lack of data. Collecting sufficient data can be time consuming and infeasible in some cases. Computational power can be a challenge for researchers by increasing the training time due to the lack of the required computing resources. Using traditional machine learning methods, the obtained performance may be unsatisfactory, and can not lead to optimal solutions. For all these reasons, we need a solution that helps us overcome the stated problem and obtain models with acceptable results. In this work, we present and discuss different transfer learning methods that help us transfer knowledge from a source domain to a target domain. The goal is to reuse as much as possible information from the source domain to enhance the performance of the model at the target domain. This type of approaches will solve the problems mentioned before such as the lack of data and will provide us with good results due to the use of knowledge from multiple source domains. We tested five Transfer learning (TL) approaches: a principal component analysis (PCA)-like method that creates a transformation like the PCA transformation and apply it to the data to create new common domain, a PCA based method that creates common domain using PCA, a PCA-SMOTE method that balances the data and creates common domain, a basic method based on a simple matching between similar features from source and target domain, and a sparse coding-based method that creates a common domain where the data representation will be as sparse as possible. The impressive results that we obtained in both tasks prove that the presented methods can be applied to transfer knowledge across different domains.

1. Introduction

Nowadays, more than 55% of the population are living in urban areas, and it is estimated that 13% more will move to cities in the upcoming 30 years [1]. With the emergence of the sensor technology and with the goal of providing cities with great services, the idea of smart buildings was introduced in the last two decades. Recently, researchers have started focusing on Activities Recognition (AR) [2, 3, 4] and Occupancy Estimation (OE) [5, 6, 7]. Indeed, AR and OE can be very helpful in many applications such as energy management [8]. They can reduce the energy consumption and find the optimal energy distribution. Most of previous works on AR and OE tasks have focused on supervised machine learning approaches [9]. Indeed, a model is trained in a specific environment and evaluated using data from the same domain. The collection of data is required for each environment. However, the task of collecting data is time consuming and can affect the behavior of the collector with time. Also, most of the previous research assumed that the distributions of data from a specific environment is the same in any new smart building environment. But, this assumption is not valid in most of the cases. So, we need to reduce dependency on data by reusing as much as possible knowledge from the source environment in the new target environment. Transfer learning can be helpful to avoid some problems such as computational power. Indeed, the advancement of the research can be blocked due to the lack of the required computing resources. Also, transferring knowledge across domains is a solution to the models where it is difficult to further improve their performances. Most of the recent research on transfer learning have focused on deep learning methods. In this paper, we introduce five basic methods of transfer learning that do not apply deep learning methods. Indeed, deep learning methods require a lot of training data to give good performance. Unfortunately collecting training data is a very complex process in the case of AR and OE tasks. Four of the introduced methods assume that a little number of instances from the target domain added to the training data can significantly improve the performance of the target model [10]. In this paper, we made several contributions. Firstly, we adapted a PCA-like method for OE. Secondly, we implemented a PCA-based method and we tested it on AR and OE. Thirdly, we created a PCA-SMOTE method that balances the data and creates common domain. Fourthly, we implemented a method called simple feature matching to compare it with other methods that do not consider the nature of features. Finally, we adapted a method based on sparse coding to the task of AR.

*Corresponding author.
E-mail address: j_dridi@live.concordia.ca (J. Dridi)

The rest of the paper is organized in 4 sections. In section 2 and 3, we present the literature review and the theoretical background. In section 4, we introduce the proposed approaches. In the section 5, we present and discuss the experimental results.

2. Literature review

2.1. Activities recognition

Previous research for the AR task has developed multiple machine learning methods. Indeed, AR task has involved supervised, semi-supervised, and unsupervised machine learning methods as well as deep learning and transfer learning approaches.

For the supervised machine learning methods, [4] has used data collected from accelerometers to test 5 machine learning classifiers (k-Nearest Neighbor, Discriminant Analysis, Support Vector Machine, Decision Tree, and Ensemble Classifier) on recognizing 7 office activities. The goal is to understand worker's activity patterns, and the considered approaches gave accuracies reaching 96%. [3] has developed a framework focusing on group activities such as working together and taking classes to develop a building management system. The framework is called GADAR (Group Activity Detection And Recognition), and it uses data collected from smartphone sensors (acceleration, gyroscope, compass, Bluetooth, microphone) and Bluetooth beacons data. The decision tree has been tested as a classifier and it has given accuracy greater than 89% for the activities recognition task. [11] has developed a framework to recognize activities such as walking and standing using Radio-frequency signals. The method evaluation was made using supervised classifiers such as Naive Bayes and K-nearest neighbor. [12] has developed a system to recognize activities for elderly people in order to monitor a public health worry which is fall. The system uses non-wearable devices data which is information generated by WIFI devices. The features information is extracted using a Short-Time Fourier Transform (STFT), and the evaluation system is made using an SVM classifier.

For the semi-supervised machine learning methods, [13] has also considered the recognition of falls for the elderly people using data collected from smartphone sensors (accelerometer and orientation sensor) and evaluated using a semi-supervised approach. [14] developed 3 semi-supervised approaches to monitor activities patterns in health and fitness applications. The proposed methods use mobile phone sensing data such as GPS and accelerometer. The first method is self-learning which uses one classifier to classify unlabeled data, and the last two semi-supervised methods are co-learning methods that use multiple classifiers called En-Co-Training and Democratic co-learning.

For the unsupervised machine learning methods, [15] has considered the case where there is a lack of labeled data and developed an unsupervised approach to recognize activities based on Hidden Markov Model (HMM) and evaluated using acceleration data collected from wearable sensors. [16] has developed an unsupervised approach to recognize five activities (walking, running, sitting, standing, and lying down) using smartphone sensors data. For a known activities number, DBSCAN (Density-based spatial clustering of applications with noise) has achieved an accuracy of 90%.

For the deep learning methods, [17] has presented many techniques that use data collected from wearable and non-wearable sensors. The dataset can be generated using smartphone sensors, watch sensors, door sensors, sound, WiFi, etc. A deep neural network (DNN) has been presented as a model containing more hidden layers than an artificial neural network (ANN). Indeed, DNN works as a classifier that recognizes the activities of processed data. Convolutional Neural Network (CNN) has been presented for the task of AR as it performs features extraction from sensing data, then it makes decisions over the obtained data. Recurrent neural network (RNN) has been considered in [17] as it takes into consideration the temporal pattern of the sensing data while recognizing activities. In most cases, LSTM (long-short term memory), which serves as a memory unit, is employed with RNN.

For the transfer learning methods, [18] has given information about several knowledge transfer techniques. The presented methods have used several data sources such as video cameras, wearable sensors (accelerometers, gyroscopes, radio frequency identification sensors, etc), and ambient sensors (temperature and pressure sensors, motion detectors, object and door sensors). Instance transfer techniques transfer knowledge gained from a source domain to a target domain to enhance the prediction function of the AR task. It is usually done by re-weighting instances from the source domain to fit the target domain [18]. Feature representation transfer has been employed by creating a transformation to map source domains to a target domain to allow knowledge transfer across several domains. Thus, the target domain prediction function gets enhanced with knowledge of several source domains [18]. Parameter sharing methods have been presented for the AR task as they learn a target model using pre-trained models of source domains [18].

In summary, recognizing activities using sensors data has been done using several machine learning techniques such as supervised and transfer learning methods that solve different problems such as the limited amount of labeled data.

2.2. Occupancy estimation

Prior research for OE task has also developed multiple machine learning methods as AR task. Supervised, semi-supervised, and unsupervised machine learning approaches as well as deep and transfer learning methods have been tested on the task of occupancy prediction.

For the supervised machine learning methods, [19] has developed a supervised learning approach based on interactive learning to predict the number of occupants in a smart building room. The method has been evaluated using a decision tree classifier on ambient sensors data such as

CO2 concentration and power consumption avoiding the use of cameras for privacy purposes. [20] has presented an occupancy estimation system based on two supervised machine learning methods and uses WiFi and Bluetooth sensors as data sources instead of ambient and cameras sensors. Indeed, they believe that ambient sensors do not provide good accuracies and cameras sensors require high setup costs. [21] has developed several supervised learning methods to estimate occupancy levels that are trained using WiFi data collected in a smart building office. Among the tested methods, decision tree and random forest have given the highest scores with 95% of accuracy. [22] has developed two supervised learning techniques based on random forest and KNN algorithms to estimate the number of occupants in smart building rooms using several types of sensors data such as CO2 data and airflow. The goal of the considered methods is to create energy efficient Heating, Ventilation, and Air-Conditioning (HVAC) systems. [9] has developed a supervised machine learning approach to improve energy management in smart building rooms using ambient sensors data such as CO2 concentration and acoustic pressure. The technique proposed in the research determines at first the most significant features for the occupancy prediction, then it uses a supervised model such as a decision tree and random forest to estimate the occupancy levels in a given office.

For semi-supervised learning approaches, [23] has developed a method called DA-HOC (domain adaptation human occupancy counter) that uses CO2 data to make domain adaptation based on a semi-supervised learning method. DA-HOC is able to predict the number of occupant in a large room using a small number of labeled training data and based on the knowledge gained from a model trained in a small room. [24] has proposed a semi-supervised machine learning method that uses limited ground truth data to estimate the number of occupants in smart buildings. The methodology uses WiFi data for training and evaluation, and its main goal is to provide efficient energy management. The considered method has been evaluated using a combination of artificial neural networks and linear regression, and it has given accuracies around 90%.

For the unsupervised machine learning approaches, [25] has developed a method based on the Hidden Markov Model and Bayesian Network algorithms to estimate the number of occupants in an office. The considered method has achieved accuracies between 89% and 91%, and it works using probabilistic cause effects relations and states. [26] has considered energy consumption planning in a smart building by developing an unsupervised approach to estimate occupants number. The method is based on finite scaled Dirichlet mixture models that use non-wearable sensors data such as door/window position and acoustic pressure. [27] has proposed an unsupervised method that detects occupants in rooms using air quality data. The considered method is based on a k-means clustering algorithm, and its main goal is to manage energy in buildings.

For deep and transfer learning methods, [28] has proposed a deep transfer learning method that uses IoT (Internet of Things) sensors data such as temperature and CO2 data collected from offices in a smart building. In this research, a pre-trained model from the source domain called Convolutional Deep Long Short-Term Memory (CDLSTM) has been used to transfer knowledge to the target domain by sharing model weights. [29] has also a deep transfer learning method based on LSTM using data collected from IoT sensors such as thermometer, PIR, and CO2 sensors. The considered approach, which aims to reduce energy consumption, transfers knowledge by sharing the source model weights to the target model and by retraining the target classifier with a small amount of target data.

In summary, as the AR task, researchers have developed various machine learning methods to estimate the number of occupants in smart buildings for several purposes but mainly for energy management.

For AR and OE tasks, transfer learning has not received much attention as compared to other approaches such as supervised learning methods. Most of the existing approaches of knowledge transfer for AR and OE tasks are focusing on deep learning. Deep learning methods require a huge amount of labeled data which can be difficult to obtain especially for smart buildings tasks due to problems of privacy. In this paper, we present 5 basic transfer learning methods that transfer knowledge from a source domain to a target domain, that are not based on deep learning, to enhance the target prediction function.

3. Theoretical background

3.1. Transfer learning fundamentals

3.1.1. Definition of TL

Before giving the definition of transfer learning, let us start by defining some terms related to TL.

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 $P(X)$, where X is an instance of the feature space $\mathcal{X}$, $X = \{x_i | x_i \in \mathcal{X}, i = 1, \dots, n\}$.

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 f, $\mathcal{T} = \{\mathcal{Y}, f\}$. The decision function is learned from the input data.

Transfer learning: Given a source domain $\mathcal{D}_S$ and a target domain $\mathcal{D}_T$, a source task $\mathcal{T}_S$ and a target task $\mathcal{T}_T$ that correspond to the source and the target domains, respectively. The goal of transfer learning is to improve and enhance the performance of the decision function f_T of the target task by using the knowledge gained from the source domain $\mathcal{D}_S$ [30].

Figure 1 gives an intuitive example about transfer learning. Indeed, a person who knows how to play a violin or ride a bike will easily learn how to play a piano or ride a motorbike, respectively. However, we cannot use the knowledge gained form playing a violin to learn how to ride a motorbike.

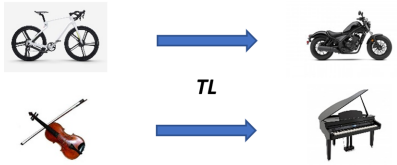

Figure 1: TL example

We call this negative transfer, and this is an interesting topic that researchers take into consideration when applying TL. Transfer learning should be applied between two domains or more that have some similarities.

3.1.2. Categories of transfer learning

Transfer learning can be categorized in different ways. If we take the problem categorization view, we can divide TL into three categories based on the availability of data labels: Transductive TL, Inductive TL, and Unsupervised TL. The scenario can be categorized into Transductive TL if only the source domains have labeled data. In this scenario, the tasks from both source and target domains are similar, but the domains can be different. We speak about Inductive TL when the labeled data exist in the target domain no matter if it exits or not in the source domain. In this case, the tasks from both domains are different. The case where we do not have label information from both domains is called Unsupervised TL [31, 32]. Transfer learning can be divided into two categories based on the feature and the label spaces of the source and the target domains. If we have the same feature space and label space between the source and the target, the scenario is called Homogenous TL. Otherwise, if one of the spaces is different between the source and the target, the scenario is called Heterogenous TL [30]. If we take the solution categorization view, Transfer learning can be divided into four categories: Instance-based approach, Feature-based approach, Parameter-based approach, and Relational-based approach. The Instance-based approach corresponds to the case when we use data from a source domain to train a target model. This is done mainly by re-weighting the instances from the source to fit the target. The Feature-based approach is mainly based on creating a transformation that will take whether the source features to fit the target feature space (Asymmetric transformation) or will take both source and target features to a new common feature space (Symmetric transformation) [30]. The parameter-based approach focuses on the target model by transferring the knowledge at the parameter level. This kind of approaches is mainly used in deep learning when a well-trained model in the source can share a part of its weights to a target model, assuming that both of tasks in the source and the target are related [31]. The Relational-based approach is the newest and the less used method in transfer learning compared to the previous methods. It does not assume that the datum from the source and the target domains are independent and identically distributed ($i.i.d$). It works by assuming that similar relations and dependence exists between data from different domains [31]. Figure 2 illustrates all the transfer learning categories mentioned above.

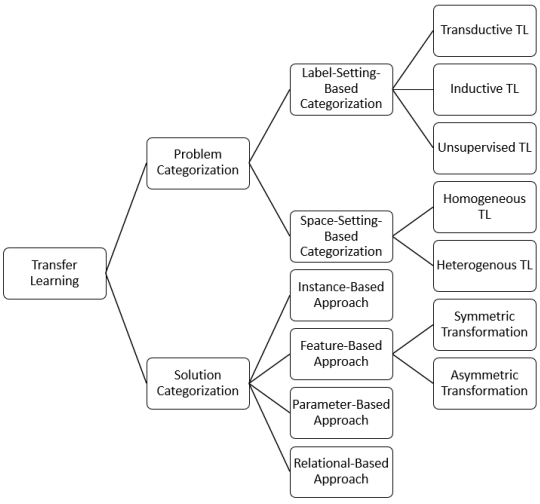

Figure 2: Transfer learning categories

3.1.3. Data-based interpretation

Transfer learning is based on the knowledge transformation and adjustment for the data-based approaches. First, we will speak about the two strategies that adjust and transform the knowledge: Instance weighting and Feature transformation. Then, we will tackle the goals of data-based transfer learning approaches. Figure 3 illustrates our plan.

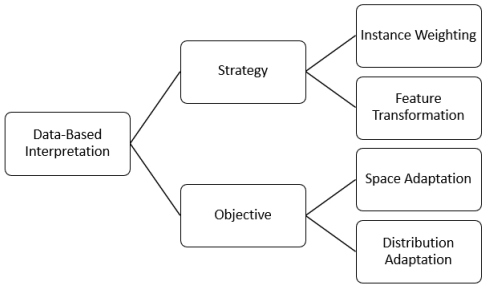

Figure 3: Transfer learning from data perspective

Instance weighting: The first strategy of data-based methods is Instance weighting. Most of Instance-based approaches use Instance weighting for knowledge sharing. This strategy assumes that we have a limited amount of data in a target domain and a lot of data from the source. The source and the target domains differ only on the marginal distribution. In other words, $P_S(X) \neq P_T(X)$ and $P_S(Y|X) = P_T(Y|X)$, where X is an instance set, Y is a label set corresponding to the instance set and P is a probability [30]. In this case, if we want to apply TL, we need to adapt the different marginal distributions between the source and the target. We have to

re-weight the instances from the source by assigning them some weights in the loss function.

Feature transformation: The second strategy called Feature transformation is widely used with feature-based approaches. It creates a transformation to transform the data into a new feature space. The transformation is created using the source and the target data or using only the source data, depending on the case. The goal of this transformation is to reduce the marginal and conditional distributions divergence. In the new created feature space, we can easily find a matching between the source and the target features by calculating the features divergence [30]. There are multiple metrics to measure the divergence or the similarity between source and target domains such as Jensen Shannon divergence, Euclidian distance, etc. However, Maximum Mean Discrepancy (MMD) has been proved to be the most effective metric used in transfer learning:

$$MMD(X_S, X_T) = \left\| \left(\frac{1}{n_S} \sum_{i=1}^{n_S} \Xi(x_i^S) - \frac{1}{n_T} \sum_{i=1}^{n_T} \Xi(x_i^T) \right) \right\|_{\mathcal{H}}^2 \quad (1)$$

where S is the source domain, T is the target domain, X is an instance set, x is a feature vector, n is the number of instances, Ξ is a nonlinear mapping that maps original features into a space $\mathcal{H}$ called a reproducing kernel Hilbert space.

Objectives: The goal of the strategies mentioned above is to achieve distribution adaption, and space adaptation. These strategies will try to reduce the distribution divergence between features. This will lead to a relatedness between the source and target feature spaces, and that is how we obtain space adaptation.

3.2. Principal component analysis (PCA)

PCA is highly used in the machine learning field. It is a dimensionality reduction tool. It transfers a given data to a new basis where the newly created principal components are independent, and the dimensions that capture more variances are ranked first. To calculate the PCA, we follow these steps:

1. Calculating the covariance matrix of the original data instances.
2. Calculating the eigen vectors and eigen values of the covariance matrix.
3. Sorting the eigen vectors by their eigen values in descending order.
4. Selecting the first k eigen vectors that have the biggest eigen values (k is the new feature space dimension).
5. Building the projection matrix.
6. Transforming the original data using the projection matrix to the new basis.

4. The proposed approaches

4.1. PCA-like method

In this section, we will introduce a supervised TL method called PCA-like method inspired from [33, 34, 10]. Indeed, the researchers in these papers applied their methods on AR, but we extended it to the OE task. The method is based on feature transformation, feature similarity calculation, feature mapping and model training. The method maps the source and target feature spaces into a common feature space. Figure 4 gives an illustration of the general architecture.

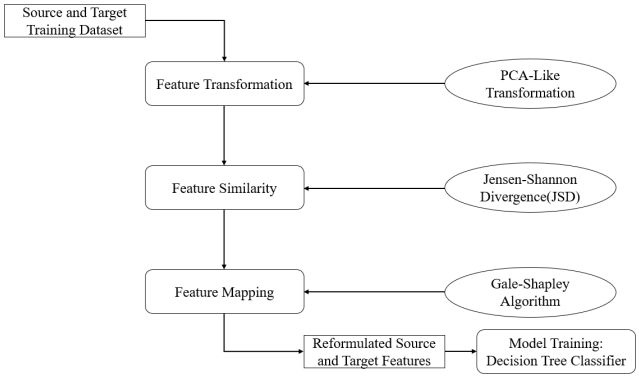

Figure 4: The architecture of the PCA-like method

4.1.1. Feature transformation

Feature transformation in some scenarios is a mandatory step. Otherwise, information will be lost like shown in Figure 5. The figure shows features from different domains but they are dependent. The best that we can do without reformulation is to match the feature g_2 with the feature h_3. It is very clear that the features g_1, h_1 and h_2 share some information that is not visible to the model. After knowledge transfer, the information inside the features g_1, h_1 and h_2 will be lost. That is why a reformulation step before applying transfer learning in this scenario is required. The reformulation will allow the one-to-one mapping between features. Without transformation, the matching may not be one-to-one because features from a space can be a combination of multiple features from other spaces.

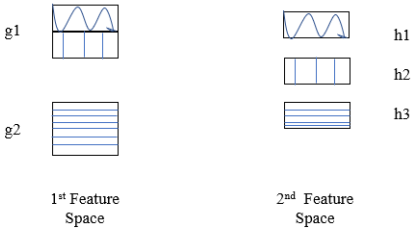

Figure 5: Example of not visible information to models

In our scenario, labeled data from the source and target domains are available. Our method can use the data from both domains to create transformation matrices for both domains. Let us consider $\mathcal{L}$ a transformation with a transformation matrix T, Z a variable of the labels, $\{X_1, X_2, \ldots, X_n\}$ random variables of a dataset. In this scenario, X_i corresponds to the random variable of the feature f_i. The

vector of random variables of the features is called $X = (X_1, X_2, \dots, X_n)^T$. This method works by creating a linear transformation. Our transformation $\mathcal{L}$ should satisfy the two following criteria. Firstly, $H(X) = H(TX)$ where H is the Shannon entropy, and T is the linear transformation that we are looking for. This first criterion will preserve the amount of information before and after the transformation without encountering information loss. Secondly, after applying the transformation, the mutual information between any two features from a specific domain given the labels is null: $I(\mathcal{L}(X_i); \mathcal{L}(X_j)|Z) = 0$, I is the mutual information. This second criterion assumes that reformulated features are independent, and each reformulated feature provides new information. It is difficult to create a transformation that agrees with the second criterion. Indeed, given the different labels, the joint distribution of the features may not remain unchanged. For that reason, we relax this property and introduce a new property as follows: $E[Cov(\mathcal{L}(Xi); \mathcal{L}(Xj)|Z)] = 0$, where Cov is the covariance function and E is the expected value. This criterion provides us with features expectedly uncorrelated and that may still be dependent [33]. To satisfy all these properties, we introduced a PCA-like transformation. First, we have calculated the expected covariance matrix of the random variables given the labels. The expected value is introduced because the covariance matrix is not constant, but it is a function of the labels Z. Indeed, using the labeled features, the system tries to find a new base that is constructed by eigenvectors of the calculated covariance matrix.

$$E(Cov(TX|Z)) = E(TCov(X|Z)T^T) = T\Gamma T^T = \Sigma \quad (2)$$

where $\Gamma = E(Cov(X|Z))$ and Σ is a diagonal matrix.

In case the Σ matrix is not diagonal, we need to normalize the obtained eigenvectors. This procedure gives eigenvectors with Euclidian norm equals to one. We take a percentage (10% for the AR task and 40% for the OE task) of the eigenvectors, and the rest are ignored. The transformation matrix T is computed by letting the selected eigenvectors being its rows vectors [33].

4.1.2. Feature similarity

After feature reformulation, it is easy to make a one-to-one mapping, if the features are the same. However, it is not the case in most of the time. That is why, we need to use a tool to match the features based on their relatedness. There are multiple tools to calculate the relatedness between features distributions that have been widely used in research such as KL (Kullback-Leibler) divergence, correlation coefficient, Euclidian distance, mutual information [10]. The method used in this paper is Jenson-Shannon divergence (JSD). The KL divergence, also known as the relative entropy, is asymmetric and unbounded method, its formula is as follows:

$$D(R \parallel S) = \sum_x R(x) log(\frac{R(x)}{S(x)}) \quad (3)$$

where R and S are two probability distributions. The JSD is a version that satisfies the symmetry and the bounding properties (within [0, 1]), and its formula is as follows:

$$JSD(R \parallel S) = \frac{D(R \parallel U) + D(S \parallel U)}{2}) \quad (4)$$

where $U = \frac{(R+S)}{2}$

Given two reformulated features spaces $\tilde{\mathcal{F}}$ and $\tilde{\mathcal{G}}$ and a label vector Z, the system can compute the JSD between two distributions R_i and S_j of the features $f_i \in \tilde{\mathcal{F}}$ and $g_j \in \tilde{\mathcal{G}}$, respectively given the labels. The general formula is as follows:

$$E(JSD(R_i(f_i|Z) \parallel S_j(g_j|Z))) = \sum_m f_t(z_m) JSD(R_i(f_i|Z = z_m) \parallel S_j(g_j|Z = z_m)) \quad (5)$$

where f_t is a probability distribution function of the labels Z.

4.1.3. Feature mapping

Algorithm: After getting the relatedness between all pairs of features from the source domain and the target domain, it is easy to make a one-to-one matching between features based on the values of their divergence by different matching algorithms. To make the mapping between features, we implemented the Gale-Shapley algorithm [35].

Negative transfer: When transferring knowledge from a domain to another one, the two domains should be dependent. This is required to find useful knowledge, otherwise all the knowledge gained from the first domain will not be helpful. Gained knowledge which decreases the performance of a target model instead of improving it, is called negative transfer. Negative transfer has been taken into consideration in the transfer learning research because of its bad effects. To avoid this problem, we have set a threshold for the mapping step: any two pairs that have a divergence greater than the threshold will not be considered. The threshold is not static and depends on the task. To get the best threshold, we make a grid search for each task.

4.1.4. Model training

After transforming the original features from both domains into a new common domain and matching the different features from the source and the target, we can easily use data from the source domain to help improving the training in the target domain. It has been proven in previous works that small amount of labeled data from the target added to the training improves significantly the performance of the target model [10]. To evaluate the performance of the knowledge transfer, we used a decision tree classifier that has been proven to be effective in the case of AR and OE by previous research.

4.2. PCA-based method

In this section, we introduce the PCA-based method that we tested on both tasks AR and OE. It is an unsupervised domain adaptation method. In this method, we do not have a transformation like the previous one. The method consists of reformulating the features by PCA, calculating the relatedness between new reformulated features, mapping the features, and training the model using the new reformulated features from the source and the target domains. Figure 6 illustrates the different steps that the method will undertake.

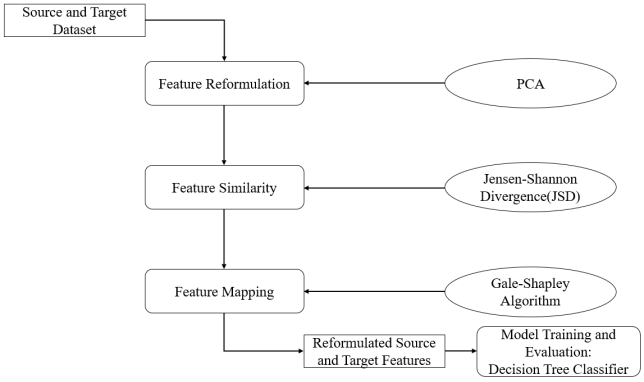

Figure 6: The architecture of the PCA-based method without transformation

4.2.1. Feature reformulation

As we mentioned before, the reformulation step is very important because hidden information can not be exploited without revealing it. Considering the propriety of the principal component analysis: The PCA outputs uncorrelated principal components, we reformulated the random variables of the features from the source and the target by applying a simple PCA to each dataset. Then, we took a percentage (10% for AR task and 40% for the OE task) of the principal components and consider them as our new reformulated features. In this reformulation method, we did not create a transformation matrix to transform the data into a new domain like we did in the previous method. But, by simply applying the PCA we transformed the original data into a new common feature space.

4.2.2. The following steps

After getting the reformulated features, we followed the same path as the previous method by calculating the divergence between the different variables using the Jensen-Shannon Divergence method. Then, we made a one-to-one mapping using the Gale-Shapley algorithm. We take into consideration the negative transfer by defining a threshold for the mapped pairs of features. Finally, we trained the target model using data from both domains. It has been proven in previous works that small amount of labeled data from the target added to the training improves significantly the performance of the target model [10]. We tested this method for both tasks : AR and OE.

4.3. PCA-SMOTE method

The PCA-SMOTE method is an improved version of the the PCA-based method. It is an unsupervised domain adaptation method. It introduces the SMOTE (Synthetic Minority Over-sampling Technique) resampling [36] to balance data. We followed the same steps of the PCA-based method and we introduced the data resampling in the general architecture as shown in Figure 7.

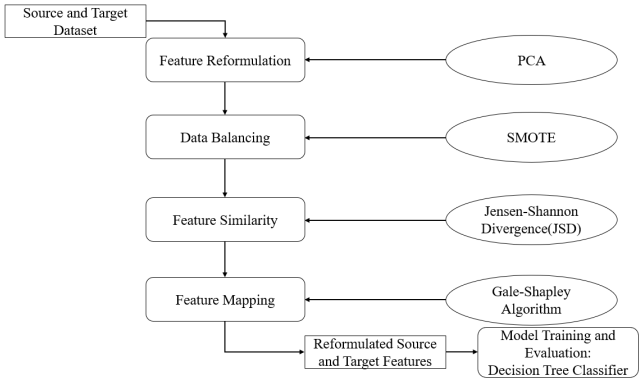

Figure 7: The architecture of the PCA-SMOTE method

4.4. Simple feature matching method

This method is so basic compared to the previous ones. It is an unsupervised TL method that consists of a manual matching between similar features from different domains. In other words, we made a one-to-one mapping between identical features from different feature spaces. After this step, we obtained a huge amount of data from the source domain that can help the training of the target model. Figure 8 explains how the manual matching has been done.

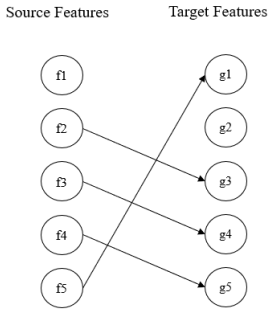

Figure 8: Manual matching

We looked for similar features from the source and the target domains and we matched them together such as (f_2, g_3). The features that do not exist in both domain such as f_1 and g_1, were ignored. This method has been tested for only the OE task because the datasets that we worked on contain similar features as well as a few numbers of features. When the number of features grows, the manual mapping becomes difficult to do. The goal of this method is to make a comparison between the performance of the methods that

do not take into consideration the nature of features such as the 1^{st}, 2^{nd} and 3^{rd} methods, and the methods that take into consideration the type of features such as the current method.

4.5. Supervised transfer sparse coding method (STSC)

In this section, we introduce the algorithm called Supervised Transfer Sparse Coding (STSC). This method takes into consideration the difference of probability between labeled data from the source domain and target domain. It tries to reduce the distributions divergence by creating new feature representations. To discuss the proposed method, we split it into two parts: sparse coding and domain transfer. The algorithm was inspired from [37].

4.5.1. Problem Definition

Let us consider $X = (x_1, \dots, x_N) \in \mathbb{R}^{F \times N}$ a dataset where x_i is the $i-th$ feature vector of the $i-th$ instance. The dataset contains N instances with F number of features. Let us consider the labels space $\mathcal{Y} = (y_1, \dots, y_N) \in \mathbb{R}^N$ as well, where y_i is the $i-th$ label of the $i-th$ instance. For this problem, the training dataset is a combination of a huge number of instances from the source domain and few instances from the target domain. Indeed, it has been proven by [37] that a small number of samples from the target domain added to the training can improve significantly the generalization of the created model. For the evaluation, the testing dataset is a combination of source and target instances as shown in Figure 9.

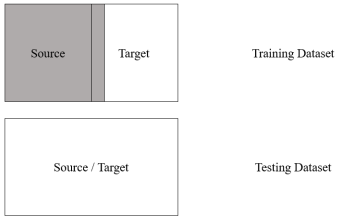

Figure 9: Training and testing datasets

The main goal of this method is to create a robust model that is able to transfer knowledge from a source domain that contains a lot of data to a target domain where we have a lack of labeled data. To do this, we need to create a new representation for the data.

4.5.2. Sparse coding

Sparse coding is a technique that aims of finding a new representation of an input dataset X using a transformation. The new representation is sparse, and it satisfies the assumption that each input vector x is a linear combination of the basis vectors of the created transformation. Figure 10 illustrates the principal objective of sparse coding.

Let us consider an instance $x \in \mathbb{R}^F$ and a dictionary matrix $D = (d_1, \dots, d_G) \in \mathbb{R}^{F \times G}$, where F is the number of features in the data, and G is the number of codewords in the matrix D. Sparse coding works by creating a dictionary

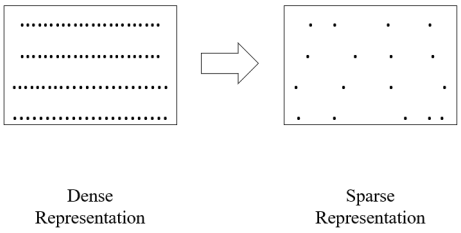

Figure 10: Sparse vs dense

matrix that satisfies the following equation:

$$x = \sum_{j=1}^{G} e_j d_j \tag{6}$$

where $\tilde{e} = (e_1, \dots, e_G)^T$ is the sparse code that represents the sample x.

The optimization problem that sparse coding method uses to calculate the dictionary matrix D as well as the sparse code matrix $E = (\tilde{e}_1, \dots, \tilde{e}_N)$ is the following:

$$\begin{aligned} \min_{D,E} \quad & \| X - DE \|_F^2 \\ \text{s.t.} \quad & \left\| \tilde{e}_i \right\|_0 \leq C, i = 1, \dots, N \end{aligned} \tag{7}$$

where C is constant, $\| \|_F$ is the squared Frobenius norm which is calculated by summing the squares of the absolute elements of a given matrix, $\| \|_0$ is the l_0-norm which is calculated by counting the number of elements of a given vector that are not null.

The development of research has proven that the minimal l_1 solution is the sparsest solution [38]. So, the problem will be updated to the following system:

$$\begin{aligned} \min_{D,E} \quad & \left\{ \| X - DE \|_F^2 + \lambda \sum_{j=1}^{N} \left\| \tilde{e}_j \right\|_1 \right\} \\ \text{s.t.} \quad & \left\| d_i \right\|_2^2 \leq 1, i = 1, \dots, G \end{aligned} \tag{8}$$

In this scenario, the goal of the constraint is to prevent the dictionary matrix from exploding. The λ variable is used as a regularization parameter, and the squared l_2 norm is defined as the sum of the squared elements of a given codeword. We can define the first term of our problem as a reconstruction term that will give us a great representation of the provided input data X. The second term can be interpreted as a sparsity penalty that will give us a sparse representation of the input samples. This type of optimization problem can be efficiently solved by an iterative algorithm like the one proposed by [39].

4.5.3. Domain transfer

In this section, we will introduce the additional terms that should be added to the optimization problem to create unified sparse codes that can be considered from the same domain.

Maximum Mean Discrepancy (MMD) regularization: The MMD has been used in previous research as a regularization term [40]. It calculates the distance between the instances mean of the sparse code representations from both domains ($\mathcal{D}^S, \mathcal{D}^T$). Sparse codes with close distributions will have MMD close to zero. The MMD has the following formula:

$$MMD = \left\| \left(\frac{1}{N^S} \sum_{i:x_i^S} \tilde{e}_i - \frac{1}{N^T} \sum_{j:x_j^T} \tilde{e}_j \right) \right\|_2^2 = Tr(ESE^T) \tag{9}$$

The S matrix is defined as follows:

$$S_{ij} = \begin{cases} 1/(N^S)^2 & if \tilde{e}_i, \tilde{e}_j \in \mathcal{D}^S \\ 1/(N^T)^2 & if \tilde{e}_i, \tilde{e}_j \in \mathcal{D}^T \\ -1/(N^S N^T) & Otherwise \end{cases} \tag{10}$$

Graph-Laplacian regularization: This method has been developed by [41]. It aims to conserve the essential geometrical properties of the data distribution from both domains. In other words, if two samples have close intrinsic geometry of data distribution, then after calculating their codewords, they will remain close. This method has been added as a regularization term to the main optimization system (8). To define the regularization term, let us first define a matrix K and a diagonal matrix L that represent the k-nearest neighbor graph of the data, and the sum of the rows of the matrix K, respectively.

$$\forall i, j \in [1, N] \quad K_{ij} = \begin{cases} 1 & If\ \tilde{e}_i\ is\ one\ of\ the \\ & k-nearest\ neighbors\ of\ \tilde{e}_j \\ 0 & Otherwise \end{cases} \tag{11}$$

$$L = diag(l_1, \dots, l_N) \tag{12}$$

where $l_i = \sum_{j=1}^{N} K_{ij}$

Now, we can define the Graph-Laplacian matrix as well as the Graph-Laplacian regularization term as follows :

$$U = K - L \tag{13}$$

$$GLM = Tr(EUE^T) \tag{14}$$

Using the two regularization terms MMD and GLM, we can create a general regularization term $\tilde{S}$ that consists of their combination using two tuning parameters α and β.

$$Tr(E\tilde{S}E^T) = Tr(E(\alpha S + \beta U)E^T) \tag{15}$$

After defining this general regularization term, we add it to the main optimization system (8) and obtain the following optimization problem that corresponds to a work called transfer sparse coding [40].

$$\begin{aligned} \min_{D,E} \quad & \left\{ \| X - DE \|_F^2 + \lambda \sum_{j=1}^{N} \left\| \tilde{e}_j \right\|_1 + Tr(E\tilde{S}E^T) \right\} \\ \text{s.t.} \quad & \left\| d_i \right\|_2^2 \leq 1, i = 1, \dots, G \end{aligned} \tag{16}$$

SVM-Based transfer correction: The STSC has added a new term to the optimization problem (16) compared to the transfer sparse coding (TSC) algorithm. This term is a supervised transfer correction term that we will explain in this section. The goal of this regularization term is to better merge the classes from the source and the target domains. So, the resulting merged domain is easier to be classified than a domain obtained by TSC method [40]. Figure 11 explains our objective in this section. Indeed, without SVM correction the merging of 4 classes may not be optimal and can be misleading. But, using the SVM transfer correction, the separability of classes after merging is better than before.

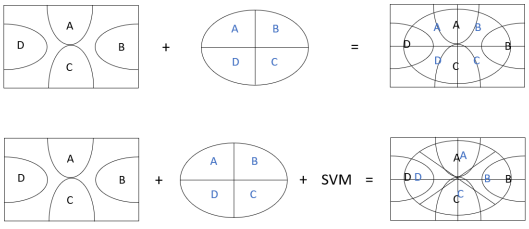

Figure 11: TSC vs STSC

The SVM term improves the performance of the model by using little amount of labeled data from the target space while training. To define this term, we introduce a matrix $W \in \mathbb{R}^{F \times m}$ that contains the hyperplane normal vectors of the different classes. We introduce a bias vector $b \in \mathbb{R}^m$ containing the intercepts of the hyperplane, and we define a matrix $B = (b, \dots, b) \in \mathbb{R}^{m \times N}$ that contains the biases of the different objects based on their original classes. We introduce another matrix $\Theta \in \mathbb{R}^{m \times N}$ that contains the margins of the instances with respect to all the available classes. At this step, we can define the new optimization problem:

$$\begin{aligned} \min_{D,E,W} \quad & \left\{ \| X - DE \|_F^2 + \lambda \sum_{j=1}^{N} \left\| \tilde{e}_j \right\|_1 + Tr(E\tilde{S}E^T) + \right. \\ & \left. \mu(\frac{1}{2} \| W \|_F^2 + c\mathbf{1}^T \Theta \mathbf{1}) \right\} \\ \text{s.t.} \quad & \left\| d_i \right\|_2^2 \leq 1, i = 1, \dots, G \\ & \mathbb{I} - \Theta \preceq Y \circ (W^T E + B), \Theta \succeq 0 \end{aligned} \tag{17}$$

where $\circ$ is Hadamard product,$\preceq\succeq$ are element wise inequalities, μ and c are hyperparameters, $\mathbf{1}$ is a vector of ones and $\mathbb{I}$ is a matrix of ones. This method can create sparse representations that contains classes worse or better separable than before. Indeed, it can sacrifice the separability of some classes to improve the performance of the model in general. We can monitor the way how the model treats the classes by the hyperparameter μ, and for better monitoring we can define a hyperparameter μ_i for each class i.

4.5.4. Solving the optimization problem

To solve the Supervised Transfer Sparse Coding problem, we propose a method based on three iterative steps. Before defining the different steps, let us prepare the optimization problem. The LaGrange function of the optimization problem is the following :

$$\begin{aligned}\mathcal{L}(W,B,\Theta,D,E,\Gamma,\Lambda,\nu) = \|X-DE\|_F^2 + \frac{\mu}{2}\|W\|_F^2 \\ +\lambda\sum_{j=1}^{N}\left\|\tilde{e}_j\right\|_1 + Tr(E\tilde{S}E^T) \\ +\sum_{g=1}^{G}\nu_g(\left\|d_i\right\|_2^2-1) + \mathbf{1}^T[(\mu c\mathbb{I}+\Lambda)\Theta \\ +\Gamma\circ(\mathbb{I}-\Theta-Y\circ(W^TE+B))]\mathbf{1}\end{aligned} \tag{18}$$

where $\Gamma \in \mathbb{R}^{m\times N}$, $\Lambda \in \mathbb{R}^{m\times N}$, and $\nu \in \mathbb{R}^G$ are the LaGrange multipliers.

Based on the duality theory we can solve this new optimization problem:

$$\begin{aligned}\max_{\Gamma,\Lambda,\nu} \quad \min_{W,B,\Theta,D,E} \quad & \mathcal{L}(W,B,\Theta,D,E,\Gamma,\Lambda,\nu) \\ \text{s.t.} \quad & \Gamma\succeq 0,\Lambda\succeq 0,\nu\succeq 0\end{aligned} \tag{19}$$

Applying the First-Order optimality condition over W, B and Θ gives us the following three conditions:

$$\begin{aligned}&W = (\Gamma\circ Y)E^T \\ &(\Gamma\circ Y)\mathbf{1} = 0 \\ &\Gamma\preceq\mu c\end{aligned} \tag{20}$$

Using these conditions, we obtain the final optimization problem:

$$\begin{aligned}\max_{\Gamma,\nu} \quad \min_{D,E} \quad & \mathcal{L}_D(D,E,\Gamma,\nu) \\ \text{s.t.} \quad & (\Gamma\circ Y)\mathbf{1} = 0 \\ & \mu c\succeq\Gamma\succeq 0,\nu\succeq 0\end{aligned} \tag{21}$$

where,

$$\begin{aligned}\mathcal{L}_D(D,E,\Gamma,\nu) = \|X-DE\|_F^2 + \lambda\sum_{j=1}^{N}\left\|\tilde{e}_j\right\|_1 \\ +\sum_{g=1}^{G}\nu_g(\left\|d_i\right\|_2^2-1)+ \\ Tr(E(\tilde{S}-\frac{1}{2}\Psi)E^T) + \mathbf{1}^T\Gamma\mathbf{1}\end{aligned} \tag{22}$$

and,

$$\Psi = (\Gamma\circ Y)^T(\Gamma\circ Y) \tag{23}$$

The previous problem can be solved in three steps. Firstly, using [39] algorithm, it is easy to solve the first optimization subproblem:

$$\min_E \quad \left\{\|X-DE\|_F^2 + \lambda\sum_{j=1}^{N}\left\|\tilde{e}_j\right\|_1 + Tr(E(\tilde{S}-\frac{1}{2}\Psi)E^T)\right\} \tag{24}$$

Secondly, by applying [39] algorithm, we can solve the second optimization subproblem:

$$\begin{aligned}\max_{\nu} \quad \min_{D} \quad & \left\{\|X-DE\|_F^2 + \sum_{g=1}^{G}\nu_g(\left\|d_i\right\|_2^2-1)\right\} \\ \text{s.t.} \quad & \nu\succeq 0\end{aligned} \tag{25}$$

Finally, using the interior-point method we can easily solve the following convex quadratic programming subproblem:

$$\begin{aligned}\min_{\Gamma} \quad & \left\{Tr(E(\frac{1}{2}\Psi)E^T) - \mathbf{1}^T\Gamma\mathbf{1}\right\} \\ \text{s.t.} \quad & (\Gamma\circ Y)\mathbf{1} = 0 \\ & \mu c\succeq\Gamma\succeq 0\end{aligned} \tag{26}$$

The general algorithm of this problem is the following:

Algorithm 1: Supervised Transfer Sparse Coding

Input: Training dataset X, Labels Y, Hyper-parameters $(\alpha,\beta,\mu,\lambda,c)$, Iterations number $Iter-N$.
Creating S the MMD matrix and U the GL matrix
Setting D to a random matrix with zero mean columns
$\Gamma\leftarrow 0$,$\Psi\leftarrow 0$
For loop($Iter-N$)
    Solving (24) to get E.
    Solving (25) to get D.
    Solving (26) to get Γ and computing Ψ.
Output: Dictionary matrix D, Sparse codes matrix E.

The transfer correction that we discussed before can be clearly visible in this algorithm. Indeed, the Ψ learned from the SVM subproblem will be directly subtracted from $\tilde{S}$ which is the transfer and geometry matrix. So, this term will directly influence this matrix and will indirectly influence the dictionary matrix D.

4.5.5. Modeling

After calculating the sparse representation of the testing data using the dictionary matrix, we can easily train a model using the reformulated data, and evaluate it using the testing data. In this method, we used the decision tree classifier as a model because it has been proven to be effective in AR and OE tasks [9][8]. Also, it will make it easy for us to compare the different methods used during this project.

5. Experimental setup and results

5.1. Datasets

5.1.1. Activities recognition dataset

For the AR task, few public datasets are available to use on the Internet. Most researchers use their private datasets. For our case, we used the Massachusetts Institute of Technology (MIT) dataset [42], which is a public dataset that contains data of two single-person apartments. Datasets are available at this website [43]. Each apartment dataset contains 2 weeks of labeled data. The data has been collected using reed switch sensors. These sensors are proximity binary sensors that have been attached to different objects such as doors, windows, refrigerator, containers, and so on. Figure 12 shows some locations where the sensors have been installed.

Figure 12: Sensors locations in the apartments [42]

The first home contains 77 sensors and the second one contains 84 sensors. The apartments owners had been labeling the datasets for 14 days. They had been labeling different types of activities as shown in Table 1, but we have just focused on the five common activities between the two apartments. The common labels are washing dishes, toileting, preparing breakfast, lunch, and dinner.

Table 1
MIT dataset description [42]

Activities	S1	S2
Preparing dinner	8	14
Preparing lunch	17	20
Listening to music	-	18
Taking medication	-	14
Toileting	85	40
Preparing breakfast	14	18
Washing dishes	7	21
Preparing a snack	14	16
Watching TV	-	15
Bathing	18	-
Going out to work	12	-
Dressing	24	-
Grooming	37	-
Preparing a beverage	15	-
Doing laundry	19	-
Cleaning	8	-

5.1.2. Occupancy estimation dataset

For the task of OE, we have the same problem as for the task of AR which is the lack of data. In this project, we used our private datasets [9, 8]. The data has been collected from two similar offices (H355 and H358) that are located in Grenoble Institute of Technology. The H358 dataset contains 20 days of labeled data, and the H355 dataset contains 6 days of labeled data. Figure 13 and Figure 14 show the locations of the used sensors in the two offices.

The sensor network contains video cameras to record the number of occupants, a database to store the data coming from the sensors and the sensors including power consumption sensors, CO2 concentration sensors, humidity sensors, temperature sensors, door and window contact sensors and acoustic pressure sensors.

5.2. Data preparation and implementation

For the AR datasets, we divided the data into intervals of 30 seconds. For each interval, if a sensor is triggered then its value will be set to 1, otherwise its values will be set to 0. After obtaining the prepared data, we selected the labels that are in common between the two apartments (preparing breakfast, preparing lunch, preparing dinner, toileting, and washing dishes). For the OE datasets, we divided the data into intervals of 10 minutes. In each interval, we take the average of values of a given sensor. If the sensor is a motion detector sensor: passive infrared (PIR) sensor, then we take the counting of motions during each interval. Indeed, PIR is a binary sensor that gives the value 1 each time it detects a motion. The labels that exist in these datasets are: No occupant, One occupant, and Two occupants. We prepared two datasets for each office: one dataset for the 4th method, and one dataset for the other introduced methods. We prepared a dataset that contains all the variables from both domains and that will be fed to the models without

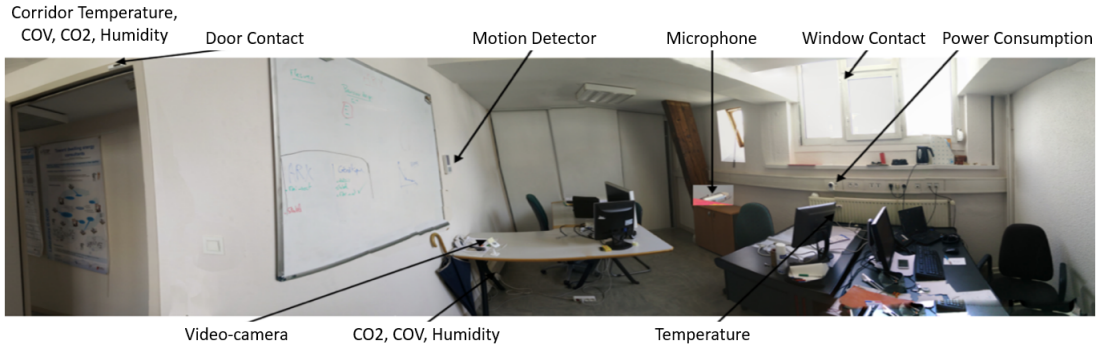

Figure 13: Sensors locations in H355

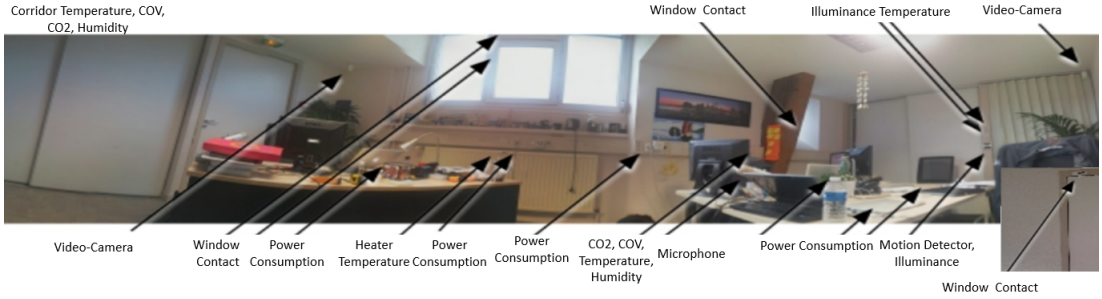

Figure 14: Sensors locations in H358

taking into consideration the nature of the variables. For the 4th method (Simple feature-matching method), it works by the simple matching between similar features from both domains. That is why, we prepared a specific dataset for this method that contains similar features from both domains as well as some created features using the original variables. For the 4th method, we did not take all the similar features, but we selected only the most significant features based on the results of previous research [9]. The method [9] has been introduced to estimate the number of occupants in an office. It starts by determining the most interesting features of the dataset based on information gains. Then, it uses decision trees in a supervised way to estimate occupancy.

Previous research [9] created new features that has been proven to be helpful such as:

- Average of all Power Consumptions: $\sum \frac{Power\ consumption}{Number\ of\ sensors}$
- Derivative CO2 Concentration: $\Delta CO2_k = CO2_k - CO2_{k-1}$
- Motion Counting: sum of the values of the motions
- Sound RMS: The root mean square of the amplitude of a recorded sound

5.3. Metrics

One of the challenges in this research work is the unbalanced data in both tasks. We took this problem into consideration by using different tools such as models hyperparameters, metrics and cross-validation. One of the ways stated before is the use of the appropriate metrics for the evaluation. We evaluated the performance of the methods using the F1-measure as well as the Precision and the Recall scores. For the last method where we introduced SMOTE to balance the data, we used both metrics F1-score and Accuracy.

To calculate these metrics, we need to extract four parameters (TP, TN, FP, and FN) from the confusion matrix:

- **True Positive (TP)** : This is the case where the algorithm correctly predicts the positive label
- **True Negative (TN)** : This is the case where the algorithm correctly predicts the negative label
- **False Positive (FP)** : This is the case where the algorithm incorrectly predicts the positive label
- **False Negative (FN)** : This is the case where the algorithm incorrectly predicts the negative label

Now, we can calculate the desired metrics:

$$\textbf{Precision (Pr)} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FP_i} \tag{27}$$

$$\textbf{Recall (Rc)} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FN_i} \tag{28}$$

$$\textbf{F-Score (F1)} = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{29}$$

$$\textbf{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{30}$$

5.4. Experimental results

5.4.1. STSC method results

The STSC method transfers the knowledge from source domains to a target domain using an approach based on sparse coding. The approach takes as input labeled training dataset of AR (huge amount of source data and small amount of target data), and it applies the presented steps to determine the dictionary matrix and the sparse codes of the input data. The obtained dictionary matrix will be used to determine the sparse codes of the testing dataset (a mixture between source and target data). After calculating the sparse representation of the testing data using the dictionary matrix, we train a decision tree model using the reformulated data and evaluate it using the testing data to obtain the desired prediction. Figure 15 and Table 2 show the performance of the STSC method on the AR task. Transferring knowledge from the source gives acceptable idea about the target without the need of instances from the target. This method takes into consideration the assumption of the importance of a few target data in the training [10, 37]. The obtained result without adding target data in the training is 30%. Adding target data in the training increases the performance of the model to reach 47%. Training with target data can harm the performance of the model as in the case when we add 8 days of target data. In this case, adding 6 days gives better performance than adding 8 days of labeled data from the target. Indeed, this is due to negative transfer coming from instances that do not improve the prediction function of the target task.

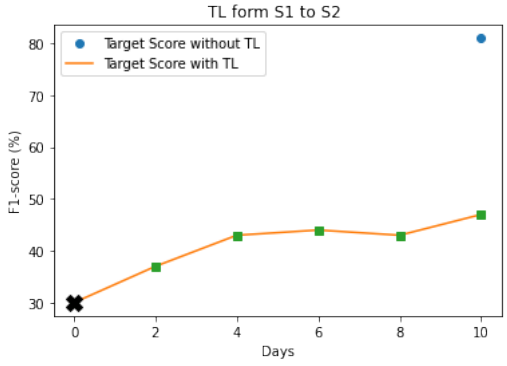

Figure 15: STSC method : AR results

We considered the STSC as unsuitable for our problem. The obtained results are acceptable and give ideas about the target task, but they did not meet our objective which is obtaining better results. The main cause of these results is the insufficient data that we are using. Indeed, the algorithm is of high complexity and requires sufficient labeled data to create the dictionary matrix and the sparse codes. Also, the quality of data such as unbalanced data can impact the results. The datasets that we are using are unbalanced, and this can be the cause of the obtained results.

Table 2
STSC method : training with 10 days from AR target domain

Scores	Without TL(%)	With TL: 10 days(%)
F_1	81.25	46.80
F_1-label1	86.71	53.78
F_1-label2	76.45	40.68
F_1-label3	83.05	31.01
F_1-label4	91.87	40.00
F_1-label5	39.77	74.03

Researchers have used the STSC method in the computer vision field [37]. They have trained it on tens of thousands of labeled data [37], and they got accuracies between 35% and 60%. For the OE task, we did not mention the results because the model did not recognize one label. Indeed, the OE datasets are more unbalanced than the AR datasets, and some labels contain small number of instances. For this reason, the STSC method did not perform well with the OE task. Indeed, it is not suitable for our problem. In future work, we will try to test it with bigger datasets.

5.4.2. Simple feature matching method resutls

We built this method to see the difference in performance between a method that takes into consideration the nature of features like this method, and other methods that do not take into consideration the type of features. The proposed method takes as input source and target labeled data of OE, and it selects and matches only similar features between the two domains. Then, it trains a target model using the source data and evaluates it using the target data to obtain the scores for the OE task. Based on the results, we can notice that the performance with a classical supervised learning method (89.31%) is greater than the results obtained without TL in previous approaches. Indeed, in this method, we applied some feature engineering by creating a lot of new interesting features such as sound RMS, motion counting and derivative CO2 concentration. Also, we can notice that the performance of the model with TL (61.64%) is less than the target results without TL (89.31%). The obtained results are good and can help develop target models without the need for target data which is costly and can be difficult to collect due to problems such as privacy.

We assume that methods that do not consider the nature of features will perform better than this method that considers the nature of variables. Indeed, matching similar features from different domains is not so accurate. For instance, matching door sensors does not mean that both sensors give the same information. An apartment can contain a lot of doors that gives different information such as the main entrance door and the back door. Also, the location of the sensor in doors (Top, Middle, Bottom) can impact the distribution of the feature.

Table 3
PCA-based method : training using 10 days from AR target domain

Scores	Without TL(%)	With TL: 10 days(%)
F_1	81.25	89.09
F_1-label1	86.71	89.25
F_1-label2	76.45	81.54
F_1-label3	83.05	93.85
F_1-label4	91.87	85.71
F_1-label5	39.77	76.19

Table 4
PCA-based method : training with 4 days from presence/absence of occupancy target domain

Scores	Without TL(%)	With TL: 4 days(%)
F_1	83.42	85.39
F_1-label1	88.03	89.43
F_1-label2	73.04	76.36

5.4.3. PCA-based method results

The PCA-based method applies PCA directly to the data without taking into consideration the labels. The approach takes as inputs source and target datasets (AR data or OE data), and it applies PCA directly to the data. The PCA transformation, created without labeled data, transfers data from both domains to a common domain. After selecting the desired principal components, it uses Jenson-Shannon divergence to calculate the divergence between different PCs of the source and target domains and, based on the obtained results it applies the Gale-Shapley algorithm to match features from the two environments. After this step, it trains a target model using decision trees with the source data and evaluates it with testing data for both tasks (AR and OE). Figures 18 and 19 describe the performance of the AR model and the Presence/Absence of Occupancy model, respectively. We started training without target data, and we obtained 28% and 68% as F_1-score for the AR and Presence/Absence of Occupancy tasks, respectively. For a start, it is a good achievement for Presence/Absence of Occupancy, but there is still more work to be done for the AR. The obtained result gives good idea about the target for Presence/Absence of Occupancy and helps to avoid the task of collecting data which is very costly. The obtained results with classical supervised learning methods trained using target data are 81% for the AR and 83% for the Presence/Absence of Occupancy. To exceed these scores, we started adding target data in the training. The F_1-score increases with 8% and 2% for a model trained using 10 days of AR target data and 4 days of Presence/Absence of Occupancy target data, respectively. The obtained models are robust and can create smart buildings applications with high performance. Tables 3 and 4 shows the scores of the different labels in both tasks and Figures 16 and 17 present the confusion matrices.

For the OE task, we increased the complexity for more levels of occupants. Figures 20, Figure 21 and Table 5

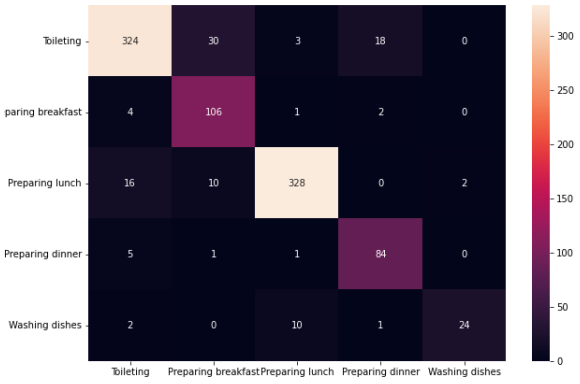

Figure 16: PCA-based method : AR confusion matrix

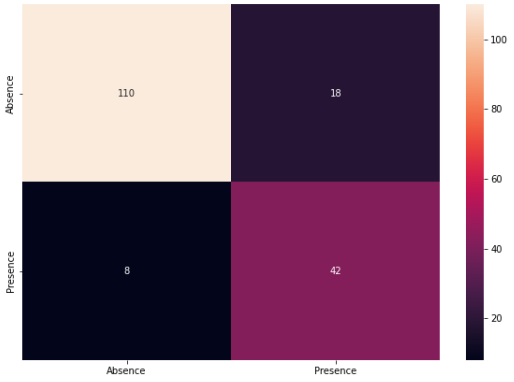

Figure 17: PCA-based method : Presence/Absence of Occupancy confusion matrix

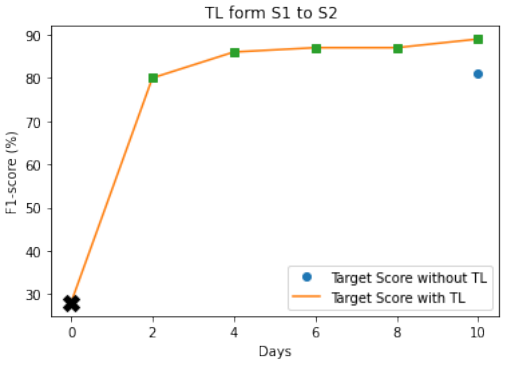

Figure 18: PCA-based method : AR results

illustrate the obtained results. As a start, we obtained 59% as F_1-score when training with only source data. The obtained results are acceptable and can create applications with good

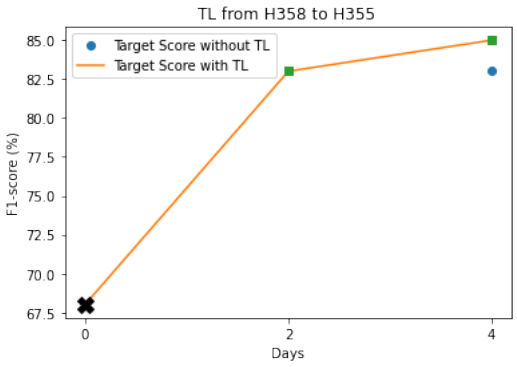

Figure 19: PCA-based method : Presence/Absence of Occupancy results

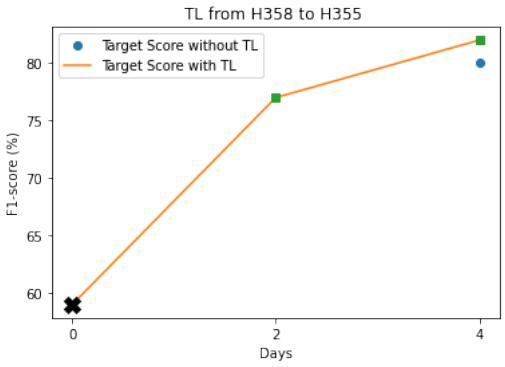

Figure 21: PCA-based method : OE results

Table 5
PCA-based method : training using 4 days from OE target domain

Scores	Without TL(%)	With TL: 4 days(%)
F_1	80.74	82.02
F_1-label1	90.69	92.18
F_1-label2	58.33	51.51
F_1-label3	59.09	64.70

performance. Compared to the Presence/Absence of Occupancy task, the performance of the model trained using only source data decreases slightly (9%). It was expected due to the increase of the occupants levels. Adding target data in the training increases significantly the performance of the target model. Training with 4 days of target data gives 82% as F_1-score which exceeds classical supervised learning methods (81%) with 1% which is a good achievement.

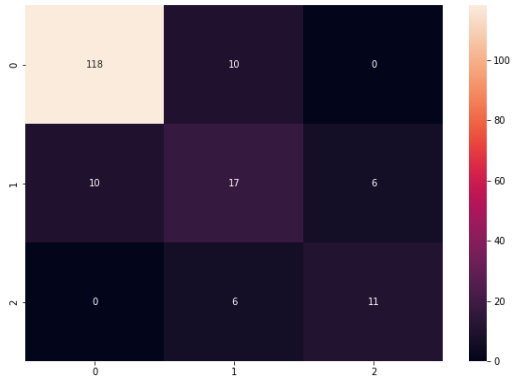

Figure 20: PCA-based method : OE confusion matrix

5.4.4. PCA-like method resutls

PCA-like method transfers knowledge from source domains to a target domain using an approach based on a transformation like PCA. The considered method takes as inputs source and target datasets (AR data or OE data), and it applies a PCA-like transformation using labeled data, for each domain, that takes into consideration the distributions of features given the labels. The transformations transfer data from both domains to a common domain. After selecting the desired reformulated features, it uses Jenson-Shannon divergence to calculate the divergence between the reformulated features of the source and target domains and, based on the obtained results it applies the Gale-Shapley algorithm to match features from the two environments. After this step, it trains a target model using decision trees with the source data and evaluates it with testing data for both tasks (AR and OE) to get the scores. Figure 23 shows the performance of the AR model on testing data. Collecting the required data to create a good classifier using classical supervised learning methods is very costly in smart building applications. Based on [10, 37], training using few data from the target domain improves the performance of the target model. In the beginning, we did not add target data in the training, we obtained 35% as F_1-score which gives an idea about the target task without the need of target data. But, there is still more work to be done to improve the results. To exceed the target score without TL (82%), we started adding target data in the training. Adding 2 days of labeled data in training increase the F_1-score by 47%. It is clear that few labeled data from the target domain increase significantly the scores of the model. Adding 10 days of labeled data from the target space in training improves the F_1-score with 8% compared to a trained model using 10 days of labeled data with a classical supervised learning method, and this proves the robustness of the current method. Table 6 gives the comparison between scores of the different labels and Figure 22 presents the confusion matrix.

For the Presence/Absence of Occupancy task, Figure 25 shows an increase in the target model performance with the increase of target data in the training as assumed in [10, 37]. Training without target data gives 77% as F_1-score which is a good result. The F_1-score increases significantly after knowledge transfer (90%) for a model trained using 4 days of target data and it exceeds the performance of classical supervised learning methods (87%) with 3%.

Table 6
PCA-like method : training using 10 days from AR target domain

Scores	Without TL(%)	With TL: 10 days(%)
F_1	82.52	89.92
F_1-label1	87.80	90.28
F_1-label2	82.59	82.30
F_1-label3	82.41	93.59
F_1-label4	90.82	91.28
F_1-label5	44.13	75.00

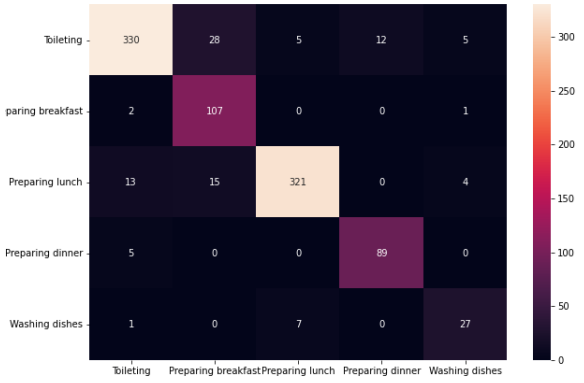

Figure 22: PCA-like Method : AR confusion matrix

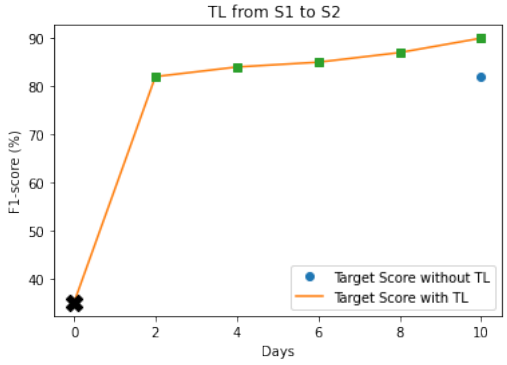

Figure 23: PCA-like Method : AR results

Table 7 shows the comparison between scores of the different labels and Figure 24 presents the confusion matrix.

For OE, we extended the testing to a high levels of occupants. As shown in Figure 27, training with only source data gives 73% as F_1-score which is good and can give an excellent idea about the target task. The obtained results for OE when training using only source data is slightly lower than Presence/Absence of Occupancy task, and this was expected due to the increase of occupants levels. Adding 4 days of target data in the training gives 86% as F_1-score which is greater with 6% than classical supervised learning methods (80%). The obtained results are excellent for the

Table 7
PCA-like method : training with 4 days from Presence/Absence of Occupancy target domain

Scores	Without TL(%)	With TL: 4 days(%)
F_1	87.17	89.88
F_1-label1	91.30	93.28
F_1-label2	75.51	79.54

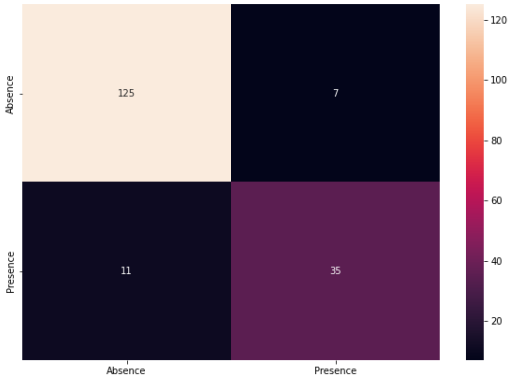

Figure 24: PCA-like method : Presence/Absence of Occupancy confusion matrix

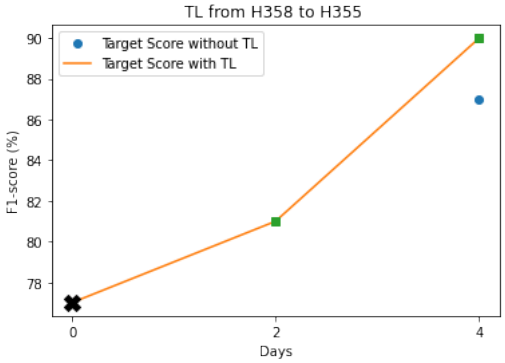

Figure 25: PCA-like method : Presence/Absence of Occupancy results

task of OE, and they are slightly lower than the results of Presence/Absence of Occupancy task which is expected. The obtained results in all tasks with TL are greater than the scores obtained using the PCA-based method. It was expected to obtain these results because the previous method do not take into consideration the labels while reformulating the features. Table 8 shows the comparison between scores of the different labels and Figure 26 presents the confusion matrix.

5.4.5. PCA-SMOTE method results

In the PCA-SMOTE, we introduced data resampling using the SMOTE to solve the problem of imbalanced data. The imbalanced data was a serious issue for us while dealing

Table 8
PCA-like method : training with 4 days from OE target domain

Scores	Without TL(%)	With TL: 4 days(%)
F_1	79.68	85.96
F_1-label1	87.94	92.88
F_1-label2	51.61	61.01
F_1-label3	60.00	73.33

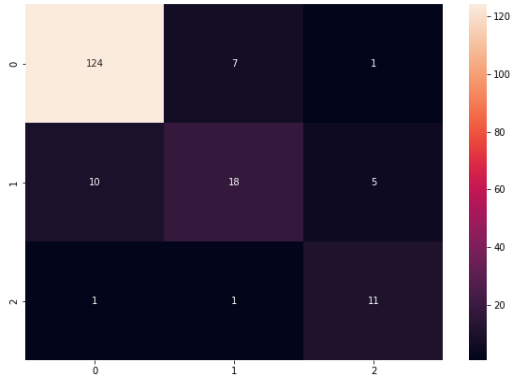

Figure 26: PCA-like method : OE confusion matrix

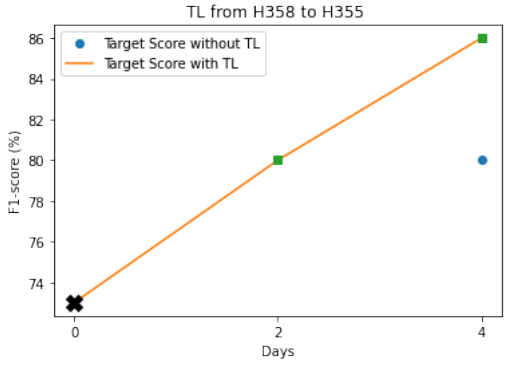

Figure 27: PCA-like method : OE results

Table 9
PCA-SMOTE method : training using 10 days from AR target domain

Scores	Without TL(%)	With TL: 10 days(%)
F_1	89.77	90.67
F_1-label1	86.19	88.25
F_1-label2	92.69	92.88
F_1-label3	85.13	87.54
F_1-label4	96.26	96.96
F_1-label5	88.61	88.09

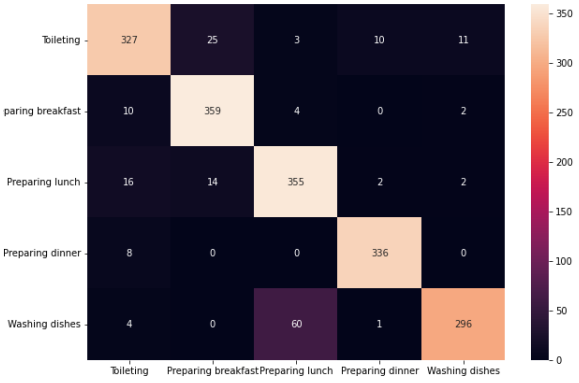

Figure 28: PCA-SMOTE method : AR confusion matrix

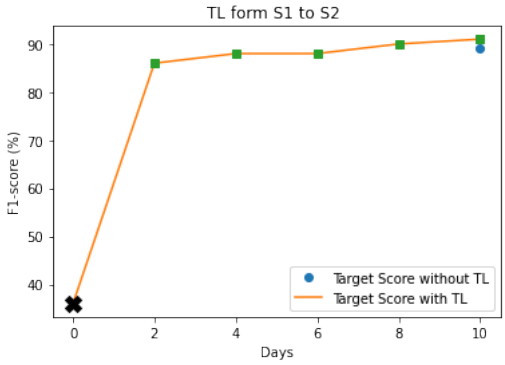

Figure 29: PCA-SMOTE method : AR results

with previous methods, especially for the STSC method. The PCA-SMOTE method is a PCA-based method improved with SMOTE to balance the input datasets. Using this method, we expect that the results of all tasks will exceed the PCA-based method results. For AR, Table 9, Figure 28 and Figure 29 show the obtained results. Training without target data gives 36% as F_1-score which gives an idea about the target without the need of target data. Adding target data in the training increases dramatically the performance of the target model till reaching 91% as F_1-score for 10 days of target data added in the training. The obtained results are excellent and expected.

For the Presence/Absence of Occupancy and OE tasks, Tables 10 and 11 and Figures 30,31, 32 and 33 show the obtained results. Training using only source data gives 69% and 41% as F_1-score for Presence/Absence of Occupancy and OE tasks, respectively. The Presence/Absence of Occupancy model is more accurate than the OE model, and this was expected due to the increase of occupants levels. With 4 days of target data added in the training, we obtained 92% and 91% as F_1-score, respectively. The obtained results are very excellent and show the efficiency of the SMOTE.

As we have balanced the datasets for the PCA-SMOTE method, it is reasonable to test using an accuracy score. Table 12 illustrates the obtained accuracies for all tasks. For OE and P/A of occupancy, we trained the target model with 4 days of target data and for AR, we trained the target model with 10 days of target data. The obtained accuracies are good

Table 10
PCA-SMOTE method : training with 4 days from Presence/Absence of Occupancy target domain

Scores	Without TL(%)	With TL: 4 days(%)
F_1	91.14	91.82
F_1-label1	90.55	91.21
F_1-label2	91.66	92.36

Table 11
PCA-SMOTE method : training using 4 days from OE target domain

Scores	Without TL(%)	With TL: 4 days(%)
F_1	87.43	90.93
F_1-label1	90.63	90.83
F_1-label2	81.50	88.20
F_1-label3	90.03	93.23

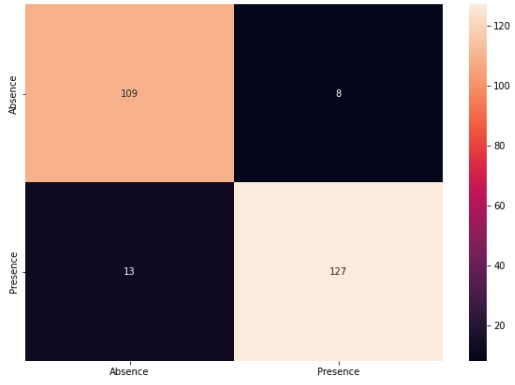

Figure 30: PCA-SMOTE method : Presence/Absence of Occupancy confusion matrix

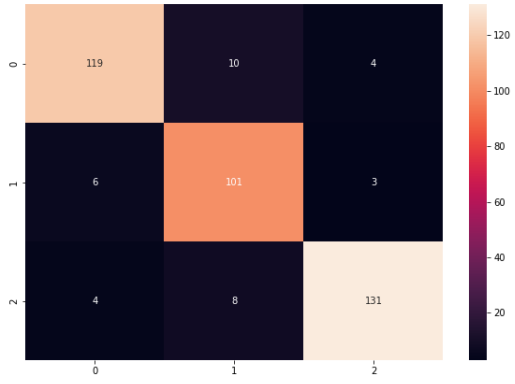

Figure 31: PCA-SMOTE method : OE confusion matrix

and better than target model performance without TL for AR and OE. For P/A of occupancy, we did obtain good results

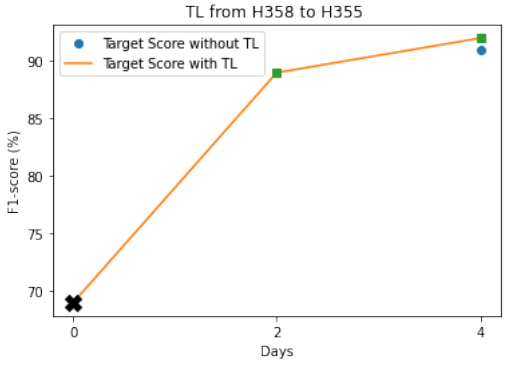

Figure 32: PCA-SMOTE method : Presence/Absence of Occupancy results

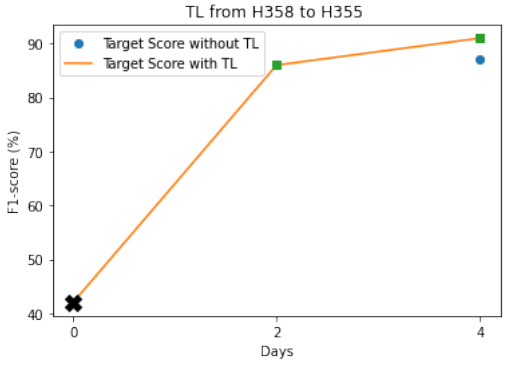

Figure 33: PCA-SMOTE method : OE results

Table 12
Accuracies of AR, P/A of occupancy and OE tasks for PCA-SMOTE method

Task	Without TL(%)	With TL(%)
Accuracy- AR	89.72	90.84
Accuracy- P/A of occupancy	91.51	90.27
Accuracy- OE	87.68	90.93

but, it is slightly lower than the target model score with a supervised machine learning method.

5.4.6. Comparing methods

For the AR, Figure 34 compares the performance of all tested methods. The PCA-SMOTE method has the best performance compared to the rest of the methods. Indeed, this method solves one of the biggest problems that we are facing during this work which is the unbalanced data. The PCA-based method and the PCA-like method have close resutls. The performance of the PCA-like method is in general better than the the PCA-based method. This is expected because the PCA-like method takes into consideration the distribution of the label space while calculating the transformation matrix. The PCA-based method applies PCA without taking into consideration the labels distribution. The STSC method is not good compared to the previous three methods. Indeed,

the quantity and the quality of the used data are the main cause of the obtained results.

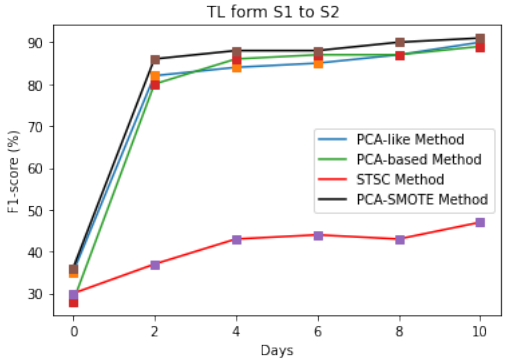

Figure 34: AR results

For Presence/Absence of Occupancy, Figure 35 shows the different tested methods. The PCA-SMOTE remains the best method compared to the rest of the methods as shown previously in other tasks. It exceeds the closest method (PCA-like method) with 2%. PCA-like method is better than PCA-based method because it takes into consideration the distribution of labels while calculating the transformation. It exceeds the PCA-based method with 5%.

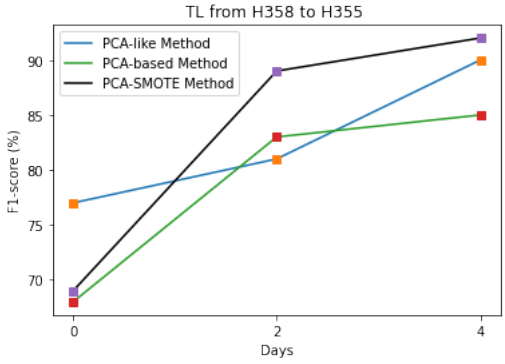

Figure 35: Presence/Absence of Occupancy results

For the OE, Figure 36 compares the obtained results. As we have discussed before, the PCA-SMOTE method is performing better than the rest of the methods. It exceeds the PCA-like method and the PCA-based method with 6% and 9%, respectively, at 4 days of training data from the target domain. The feature matching method trained using only source data and tested on target data performs better than PCA-based method and PCA-SMOTE method. But, it performs poorly compared to the PCA-like method. The feature matching method performs poorly compared to a PCA-SMOTE method, a PCA-like method and a PCA-based method trained using small amount of data from the target. The obtained results for Presence/Absence of Occupancy are greater than OE results for all the tested methods and this was expected due to the increase of occupants levels.

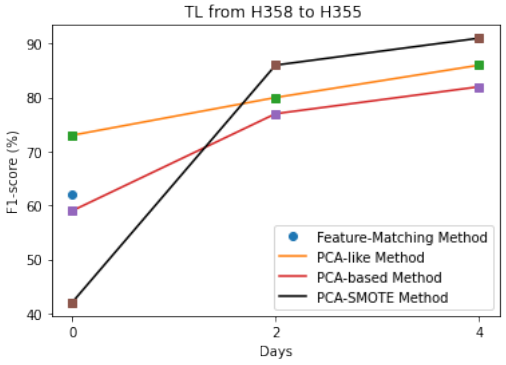

Figure 36: OE results

6. Conclusion

As the development of research in smart buildings is blocked in some directions because of the lack of data, it is necessary to find new tools to overcome this problem. That is why, we considered transfer learning. In this paper, we presented and discussed 5 basic transfer learning methods. The PCA-SMOTE method, the PCA-like method and the PCA-based method performed pretty good compared to the other methods. Indeed, these three methods do not take into consideration the nature of features. The PCA-SMOTE method is the best of all methods because it solves the problem of imbalanced data. The PCA-like method is slightly better than the PCA-based method because it considers the distribution of features given the labels while calculating the transformation. The simple feature matching method performed poorly compared to the first three methods when they are trained using a small amount of data from the target domain. This proves the efficiency of methods that do not consider the nature of features. The STSC method did not perform well in our case because of the unbalanced and small datasets. The tasks of OE (2 levels and 3 levels) performed better than the task of AR thanks to the good results obtained when training without target data. In future work, we will take into consideration the temporal dependency between the instances. Indeed, in this work, we ignored the labels transition and focused only on the labels. Also, we classified the instances using decision trees which do not take into consideration the temporal dependency. In addition, we will try to combine methods using tools such as model ensembling. For the STSC method, we will try to add new supervised transfer correction terms to the main optimization problem of new classifiers such as logistic regression. For the OE task, we will increase the prediction level to more occupants.

To ensure reproducibility of the results by the research community and a potential future improvement of our framework by other researchers the complete source code is provided in the following repository: GitHub repository.

Acknowledgement

The completion of this research was made possible thanks to Natural Sciences and Engineering Research Council of Canada (NSERC) and the ANR-2021 Generic call of the French National Research Agency through the LearningHome project. The authors would like to thank the associate editor and the reviewers for their helpful comments and suggestions.

References

[1] A.-M. Tîrziu, "Urbanization and cities of the future," *International Journal for Innovation Education and Research*, vol. 8, no. 3, 2020.

[2] 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.

[3] 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.

[4] 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.

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

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] S. Sigg, S. Shi, F. Buesching, Y. Ji, and L. Wolf, "Leveraging rf-channel fluctuation for activity recognition: Active and passive systems, continuous and rssi-based signal features," in *Proceedings of International Conference on Advances in Mobile Computing & Multimedia*, pp. 43–52, 2013.

[12] S. Palipana, D. Rojas, P. Agrawal, and D. Pesch, "Falldefi: Ubiquitous fall detection using commodity wi-fi devices," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, no. 4, pp. 1–25, 2018.

[13] P. A. Fahmi, V. Viet, and C. Deok-Jai, "Semi-supervised fall detection algorithm using fall indicators in smartphone," in *Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication*, pp. 1–9, 2012.

[14] B. Longstaff, S. Reddy, and D. Estrin, "Improving activity classification for health applications on mobile devices using active and semi-supervised learning," in *2010 4th International Conference on Pervasive Computing Technologies for Healthcare*, pp. 1–7, IEEE, 2010.

[15] D. Trabelsi, S. Mohammed, F. Chamroukhi, L. Oukhellou, and Y. Amirat, "An unsupervised approach for automatic activity recognition based on hidden markov model regression," *IEEE Transactions on automation science and engineering*, vol. 10, no. 3, pp. 829–835, 2013.

[16] Y. Kwon, K. Kang, and C. Bae, "Unsupervised learning for human activity recognition using smartphone sensors," *Expert Systems with Applications*, vol. 41, no. 14, pp. 6067–6074, 2014.

[17] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognition Letters*, vol. 119, pp. 3–11, 2019.

[18] 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.

[19] M. Amayri, S. Ploix, P. Reignier, and S. Bandyopadhyay, "Towards interactive learning for occupancy estimation," in *ICAI'16-International Conference on Artificial Intelligence (as part of WORLDCOMP'16-World Congress in Computer Science, Computer Engineering and Applied Computing)*, 2016.

[20] E. Longo, A. E. Redondi, and M. Cesana, "Accurate occupancy estimation with wifi and bluetooth/ble packet capture," *Computer Networks*, vol. 163, p. 106876, 2019.

[21] 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.

[22] C. Chiţu, G. Stamatescu, and A. Cerpa, "Building occupancy estimation using supervised learning techniques," in *2019 23rd International Conference on System Theory, Control and Computing (ICSTCC)*, pp. 167–172, IEEE, 2019.

[23] 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.

[24] A. Ashouri, G. R. Newsham, Z. Shi, and H. B. Gunay, "Day-ahead prediction of building occupancy using wifi signals," in *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*, pp. 1237–1242, IEEE, 2019.

[25] M. Amayri, Q.-D. Ngo, S. Ploix, *et al.*, "Bayesian network and hidden markov model for estimating occupancy from measurements and knowledge," in *2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, vol. 2, pp. 690–695, IEEE, 2017.

[26] 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.

[27] U. Habib and G. Zucker, "Automatic occupancy prediction using unsupervised learning in buildings data," in *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*, pp. 1471–1476, IEEE, 2017.

[28] D. Stjelja, J. Jokisalo, and R. Kosonen, "Scalable room occupancy prediction with deep transfer learning using indoor climate sensor," *Energies*, vol. 15, no. 6, p. 2078, 2022.

[29] P. Leeraksakiat and W. Pora, "Occupancy forecasting using lstm neural network and transfer learning," in *2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pp. 470–473, IEEE, 2020.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] C. H. Lu and Y. T. Chiang, "An instantiation of the multiple-transfer framework to reduce efforts in context model learning for new users in smart homes," in *2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pp. 118–121, IEEE, 2014.

[35] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *The American Mathematical Monthly*, vol. 69, no. 1, pp. 9–15, 1962.

[36] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[37] M. Al-Shedivat, J. J.-Y. Wang, M. Alzahrani, J. Z. Huang, and X. Gao, "Supervised transfer sparse coding," in *Twenty-eighth AAAI conference on artificial intelligence*, 2014.

[38] D. L. Donoho, "For most large underdetermined systems of linear equations the minimal ï¿½ï¿½1-norm solution is also the sparsest solution," *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, vol. 59, no. 6, pp. 797–829, 2006.

[39] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," in *Advances in neural information processing systems*, pp. 801–808, 2007.

[40] M. Long, G. Ding, J. Wang, J. Sun, Y. Guo, and P. S. Yu, "Transfer sparse coding for robust image representation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 407–414, 2013.

[41] M. Zheng, J. Bu, C. Chen, C. Wang, L. Zhang, G. Qiu, and D. Cai, "Graph regularized sparse coding for image representation," *IEEE transactions on image processing*, vol. 20, no. 5, pp. 1327–1336, 2010.

[42] E. M. Tapia, S. S. Intille, and K. Larson, "Activity recognition in the home using simple and ubiquitous sensors," in *International conference on pervasive computing*, pp. 158–175, Springer, 2004.

[43] M. I. of Technology (MIT), "Activity recognition in the home setting using simple and ubiquitous sensors." https://courses.media.mit.edu/2004fall/mas622j/04.projects/home/. Accessed: 31/01/2022.