# D3.2 FIRST USER TRACKING ALGORITHMS

| | |
|---|---|
| Project title | **Collaborative Recommendations and Adaptive Control for Personalised Energy Saving** |
| Project acronym | **enCOMPASS** |
| Project call | **EE-07-2016-2017 Behavioural change toward energy efficiency through ICT** |
| Work Package | **WP3** |
| Lead Partner | **CERTH – Center for Research and Technology Hellas** |
| Contributing Partner(s) | **WVT** |
| Security classification | **PU** |
| Contractual delivery date | **31/07/2017** |
| Actual delivery date | **27/07/2017** |
| Version | **1.0** |
| Reviewers | **PMI**<br>**SUPSI** |

## History of changes

| Version | Date | Comments | Main Authors |
|---|---|---|---|
| 0.1 | 07/06/2017 | First Draft and ToC | CERTH – Stelios Krinidis, Alexandros Boulogeorgos, Alexandros Kanlis |
| 0.4 | 11/07/2017 | Added sections 2,3,4 and 6 | CERTH - Stelios Krinidis, Alexandros Boulogeorgos, Alexandros Kanlis<br>WVT – Kostas Arvanitis |
| 0.7 | 18/07/2017 | Peer review | SUPSI - Cristina Rottondi |
| 0.8 | 19/07/2017 | Quality check | PMI - Piero Fraternali |
| 1.0 | 31/07/2017 | Ready for submission to the EC | CERTH - Stelios Krinidis |

# Disclaimer

This document contains confidential information in the form of the enCOMPASS project findings, work and products and its use is strictly regulated by the enCOMPASS Consortium Agreement and by Contract no. 723059.

Neither the enCOMPASS Consortium nor any of its officers, employees or agents shall be responsible or liable in negligence or otherwise howsoever in respect of any inaccuracy or omission herein.

The contents of this document are the sole responsibility of the enCOMPASS consortium and can in no way be taken to reflect the views of the European Union.

***This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 723059***.

# Table of Contents

# List of Tables

# List of Figures

# List of Definition and Abbreviations

| Abbreviation | Definition |
|---|---|
| AdaBoost | Adaptive Boosting |
| AdaBoost.SAMME | Adaptive Boosting - Stagewise Additive Modelling using Multi-class Exponential loss function |
| ANN | Artificial Neural Networks |
| BMS | Building Management System |
| BPN | Back Propagation Network |
| $CO_2$ | Carbon dioxide |
| CP | Central Power |
| CRF | Conditional Random Field |
| D | Dryer |
| DT | Decision Trees |
| ECO | Electricity Consumption and Occupancy |
| FN | False Negative |
| FP | False positive |
| HMM | Hidden Markov Model |
| IP | Internet Protocol |
| LB | Lower Bound |
| MAC address | Media Access Control Address |
| MI | Mutual Information |
| NRMSE | Normalized Root Mean Square Error |
| OCCUP | Occupancy |
| PC | Personal Computer |
| PCA | Principal Component Analysis |
| PIR | Passive Infra-Red |
| R | Refrigerator |
| RF | Random Forest |
| RMSE | Root Mean Square Error |

| | |
|---|---|
| SVM | Support Vector Machine |
| SVM-POLY | Support Vector Machine – Polynomial |
| SVM-RBF | Support Vector Machine – Radial Basis Function |
| TN | True Negative |
| TP | True Positive |
| TV | Television |
| UB | Upper Bound |
| WM | Washing Machine |

# Executive Summary

Deliverable D3.2 “First User Tracking Algorithms” is specified in the “amended” GA description as follows:

*“Initial prototype, with documentation, of the algorithms for tracking the presence and movement of users in different indoor conditions”.*

Its major goal is to explain the initial version of the algorithms for tracking the presence of users in indoor environments developed in enCOMPASS.

Building occupancy is an essential task for building analysis. It is highly related to the accurate occupancy information in buildings, which in most cases is defined using predefined functions without use of any kind of measuring and training. These tools [Hutchins07, Liao15, Mahdavi09, Yang03] analyse the building occupancy based on stochastic models (e.g. Markov chain or probabilistic distribution), but they are not accurate since the exploited occupant building usage takes into account predefined models that in most cases do not match to the actual operation of the building. The most accurate building occupancy acquisition can be automatically performed by utilizing surveillance sensors. Furthermore, building occupancy can be estimated utilizing indirect information, such energy consumption. Analysing this information, one can infer the occupancy in building with high accuracy.

Deliverable D3.2 provides a description of the algorithms for the occupancy inference in indoor environments:

- algorithms based on direct and indirect sensors:
  - algorithms based on Passive Infra-Red (PIR) motion detectors;
  - algorithms based on fusion of the information of PIR and $CO_2$ sensors;
- algorithms based on energy consumption of devices/ appliances utilizing machine learning techniques:
  - SVM utilizing RBF kernel
  - SVM utilizing Polynomial function kernel
  - Decision Trees
  - Random Forest
  - Back Propagation Network
  - Adaptive Boosting
  - Hidden Markov Models
  - Conditional Random Fields

The main dependencies with other deliverables are as follows:

- Deliverable D3.1 “Datasets with Context Data and Energy Consumption Data”: This deliverable contains the specification of each one of the energy consumption historical data set, which will be collected by the utility companies, as well as the building owners of the enCOMPASS pilots.
- Deliverable D3.3 “First Energy Disaggregation Algorithms” and D3.5 “Final Energy Disaggregation Algorithms”: These deliverables contains the algorithms which will disaggregate and provide information about the energy consumption of individual devices and appliances in buildings, especially in households, where the acquired information will be mainly based on the central building energy consumption.

The deliverable is structured as follows:

- Section 1 is the introduction of the deliverable;
- Section 2 presents an initial overview of the state-of-the-art techniques on the occupancy detection and inference algorithms;
- Section 3 presents the description of the techniques/ algorithms for the detection of the features that will be utilized for the occupancy inference;
- Section 4 provides a brief description of the classification algorithms utilized in this deliverable;
- Section 5 described the proposed algorithms for the occupancy detection and inference;

- Section 6 presents various experiments that have been performed for the validation of the algorithms;
- The final two Sections contain the Conclusions and References.

# 1. Introduction

Knowing the true occupancy, the presence or the actual number of occupants of a building at any given time is fundamental for the effective management of various building operation functions ranging from security concerns to energy savings targets, especially in complex buildings with different internal kind of use. Occupant's locations within the building varies throughout the day, therefore it is difficult to characterize the number of people that occupy a particular space and for what duration because human behaviour is considered stochastic in nature. In general, occupancy monitoring in buildings is of high interest, since occupancy significantly contributes to the performance of the building. Therefore, there is a need for detailed occupancy knowledge.

This document, which is of type Report, is an accompanying document of the code developed within Task T3.2, and describes the algorithmic approaches that have been developed for occupancy detection and estimation in different indoor environments. State-of-the-art related work is presented in the next section. Preliminary results about the performance of the methods developed are also provided.

Occupancy can be estimated using various sources, such as occupancy sensors, cameras, smart meters etc. Each sensor type has different characteristics regarding detection capabilities, cost, obtrusiveness and privacy. In this work, the occupancy estimation problem is regarded as an occupancy inference problem. After proper processing of the data acquired from the sensors which are used as input, the classifier infers the most probable class that corresponds to the occupancy status of a space. The occupancy estimation methodologies that are presented use various machine learning algorithms and non-intrusive sensors, such as Passive Infra-Red (PIR) motion detectors, $CO_2$ sensors and smart electricity meters. Furthermore, the occupancy sensing methods that are utilized, do not require occupants to carry any special equipment in order to be detected, such as tags or mobile devices.

# 2. State-Of-the-Art

In this section, the latest state-of-the-art studies on non-intrusive occupancy detection and occupancy estimation techniques are briefly presented.

Several approaches have been proposed in the literature to address the challenging problem of occupancy detection either in commercial or residential buildings, such as sensor-based, video-based and energy consumption-based. The use of IR (Infrared) and PIR (Passive Infrared) sensors in predictive models is a common method for estimating the total building occupancy [Hutchins07, Meyn09]. Benezeth et al. [Benezeth11] proposed an algorithm based on a video camera system for indoor occupancy detection and activity analysis. In [Yang15], occupancy estimation is performed based on information received from an Arduino-based wireless sensor box which is placed in each room. The sensor box hosts a number of ambient sensors such as a light sensor, a sound sensor, a motion sensor, a $CO_2$ sensor, a temperature sensor, and others. Occupant detection and tracking system which utilizes depth cameras in a multi-space area is presented in [Krinidis14]. In that work, a camera calibration method for transforming the cameras' local coordinate system to a reference global one is presented. Another method for counting the number of occupants in a conference room is proposed in [Labeodan15]. The authors evaluate the performance of the system which uses information from chair pressure sensors, and results showed that it is capable of providing fine-grained occupancy information. A limitation of the system is that it is unable to detect standing occupants.

Occupancy estimation performance usually increases when using data from multiple sensors, as each sensor type captures different aspects of human presence. In order to combine information from different sources, rule-based methods and machine learning methods have been studied in the literature. According to the rule-based approach, a set of rules is defined in advance by a domain expert who is aware of the sensors' characteristics. A rule-based algorithm for occupancy detection is presented in [Agarwal10], combining information from a reed switch and PIR sensors. The use of machine learning methods for occupancy estimation is also very common in the literature when fusing information available from different types of sensors. In such cases, occupancy estimation is regarded as a classification problem and is performed via the use of a selected classifier. Various machine-learning algorithms have been evaluated in the literature, such as Support Vector Machines (SVM), Artificial Neural Networks (ANN), Decision Trees (DT), agent-based models and other. In the study of Yang et al. [Yang14a], learning-based methods such as a Conditional Random Field model and a Hidden Markov Support Vector Machine are used to estimate the number of occupants in a three-person residence by using the readings of installed PIR motion sensors available via the alarm system. In the recent work of Chaney et al. [Chaney16], the authors introduce an approach for analysing data from multiple sensors, in order to estimate occupancy in a home. The method combines the Dempster-Shafer theory with a Hidden Markov Model (HMM). Occupancy information is then fused with consumption behaviour for evaluating how likely it is that a household can participate in demand response.

Occupancy estimation using implicit sensing sources is an alternative approach for inferring occupancy from monitored environmental or consumption parameters. These parameters can be for example energy data received from power meters, signal strength data received from mobile devices such as smartphones, or computer activity. The implicit sensing approach has gained popularity lately due to the proliferation of IoT smart devices such as smart energy meters. Milenkovic et al. [Milenkovic13] combine the use of power meters with PIR motion sensors in order to detect office worker activities at desk level and count the number of occupants at room level. For computer and desk work, an overall recognition accuracy of 95% was achieved and people count was estimated at 87% in the best case. In the work of Kleiminger et al. [Kleiminger13], digital electricity meters are used as occupancy sensors in order to detect when a household is occupied. The results showed that using common classification algorithms it is possible to detect occupancy with accuracy greater than 80%. Activity detection in a home is studied in [Saha14], where an "activity" is defined as the usage of an electrical appliance, its usage duration and its location. The proposed system fuses electricity consumption data with sensor data from smartphones (WiFi and

microphone) to infer which appliance is being used, where it is being used, and who is using it. The experiments revealed an average precision of about 75% and average recall of about 78%. Christensen et al. [Christensen14] present a method for estimating occupancy in office buildings using existing network infrastructure. The information is then used to control devices in user workspaces. IP and MAC addresses in WiFi access points and in routers are monitored and correlated to the occupancy at room, floor or building level. In Corna et al. [Corna15], an occupancy detection system based on the Bluetooth Low Energy standard is proposed. Multiple low-cost Bluetooth 4.0 antennas are installed at a building in different rooms and emit signals that are gathered by nearby occupants' smartphones which send the information to the Building Management System (BMS). The drawback of the method is that a client mobile application must be installed on the occupants' smartphones. Due to the close relation between electricity consumption and occupancy, there are studies that examine the prediction of energy consumption in office and residential buildings using occupancy and other parameters as input [Dagnely15, Jain14]. Kleiminger et al. [Kleiminger15] used classification techniques, such as SVM and Hidden Markov Models (HMMs), on the publicly available Electricity Consumption and Occupancy (ECO) data set, and achieved accuracy between 83% and 94% on occupancy detection at households. Ryu and Moon [Ryu16] use indoor environmental data ($CO_2$ concentration, temperature, humidity etc.) and energy consumption of the lighting system and other appliances obtained from building sensing network along with decision tree classification technique to detect occupancy in the Building Integrated Control Test-bed at Dankook University.

In [Wahl12] a number of PIR motion detectors are installed strategically in order to estimate people count in offices by identifying the occupants' movement direction. In study [Dong10] various sensor types, such as acoustic, PIR motion and $CO_2$ sensors are combined in order to estimate the occupancy in an open-plan office building. The authors evaluate three different machine learning methods. Occupancy detection for cubicles in an office is studied in [Hailemariam11]. In that study, inputs from $CO_2$, acoustic, PIR motion, light and electrical power sensors are combined to determine whether a cubicle is occupied or not. In [Kuutti14] beam sensors and infrared cameras are utilized to count visitors in a building. The test results showed that counting errors easily accumulate over time.

Some other studies [Christensen14] use implicit sensing methods for occupancy detection and estimation, exploiting the existing infrastructure instead of installing sensors dedicated to occupancy detection. For example, in [Christensen14] the authors present a method for occupancy estimation in a building by counting the number of connected Wi-Fi devices and monitoring the keyboard and mouse activity of desktop PCs. In [Corna15], occupancy detection is achieved via the use of the iBeacon protocol (based on Bluetooth) available on smart devices, such as smartphones. The system was built upon low cost Bluetooth antennas and a client mobile application installed on the occupants' smartphones. The authors in [Kleiminger13] study the use of electricity meters and smart plugs as sensors for occupancy detection in domestic environments. Results showed that detection accuracies over 80% are feasible in most scenarios, without taking into account sensor fusion methods. Although such systems can be easily deployed in existing buildings avoiding the installation cost of occupancy sensors, they usually require occupants to carry equipment, such as Wi-Fi enabled mobile phones. Furthermore, they cannot be applied to certain spaces where the necessary equipment, such as desktop PCs, is not available.

When sensor fusion approach is utilized for occupancy monitoring, a processing method for input received from the multiple sensors has to be determined. Two main occupancy monitoring approaches have been used for sensor fusion modelling: a rule-based approach and a probabilistic model approach. In the latter case, a training phase is required in advance, in order for the model to learn the parameters and allow the occupancy estimation process. On the contrary, in the rule-based approach a training phase is not mandatory.

The use of a rule-based system results in logical inference from sensor data [Nguyen13]. According to this approach, a set of rules is defined and applied. The rules need to be defined by a domain expert, therefore knowledge about sensor characteristics is required. The set of applied rules usually depends on the selected combination of sensors. In [Agarwal10] a rule-based algorithm for occupancy detection is presented, combining information from a reed switch and PIR sensors. An occupancy detection and activity recognition method is also presented in [Nguyen12], where binary output values from simple low-cost sensors are combined for estimating the occupancy state and activity, based on predefined rules.

On the other hand, the probabilistic model approach handles occupancy estimation as a classification problem. A probabilistic model is created by training a selected classifier. The objective is to infer the occupancy class based on input coming from various sensors. Different machine-learning algorithms have been used in the literature, such as support vector machines (SVM), artificial neural networks (ANN), hidden Markov models (HMM), decision trees (DT), agent-based models etc. In [Yang14b], six machine-learning methods were evaluated for occupancy estimation in both single-occupancy and multi-occupancy offices, including SVM, DT, k-nearest neighbours and others, with reported accuracy above 90%. In this study, various ambient sensor types were installed in a box and placed in the area of interest, with sensor data being queried every minute.

The application of machine learning algorithms in the occupancy estimation domain, has been proved effective in the recent years as it has been reported in the literature. Many of the proposed systems yield high accuracy when detecting or estimating occupancy, however most of these utilize intrusive sensing equipment such as cameras, or tags/devices that have to be carried by the occupants. The main objective of our approach is to fully exploit machine learning by discovering effective features derived from the data of implicit or non-intrusive sensing methods.

# 3. Feature selection techniques

In order to rank the influence of each feature to occupancy and extract the more useful information and reduce datasets sparsity (if it is possible), we apply two well-known feature extraction methods: Mutual Information (MI) and Principal Component Analysis (PCA), which are briefly described below.

## 3.1 Mutual Information (MI)

Mutual Information is a measure of dependence between two discrete random variables and quantifies the information obtained on variable *X* by observing variable *Y*. The mutual information between two discrete random variables is given by:

$$I(X,Y) = \sum_{x,y} P_{XY}(x,y) log \frac{P_{XY}(x,y)}{P_X(x)P_Y(y)} = E_{P_{XY}} log \frac{P_{XY}}{P_X P_Y} \quad (1)$$

where $P_X(x)$ and $P_Y(y)$ are the marginal probability density functions of random variables and $P_{XY}(x,y)$ is the joint probability density function. Generally, highly mutual information indicates a large reduction of uncertainty, low mutual information indicates a small reduction and zero mutual information between the two random variables indicates that those variables are uncorrelated. In this work, we calculate the mutual information of all features in terms of occupancy.

## 3.2 Principal Component Analysis (PCA)

Principal Component Analysis was introduced by Pearson (1901) [Pearson01] and Hotelling (1933) [Hotelling33] to describe the variation in a set of multivariate data in terms of a set of uncorrelated variables. It is a method used to emphasize variation and bring out strong patterns in a dataset in such way as to highlight their similarities and differences. PCA uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of uncorrelated variables called principal components. The basic concept when using PCA as a tool for feature selection is to select variables according to the magnitude of their coefficients, referred also as eigenvalues. Below, the background theory of PCA is described. For a vector $\vec{V} = [v_1, v_2, \dots, v_n]^T$ where $v_i, i = 1, \dots, n$ are the *n*-sensors outputs, its covariance matrix is:

$$C_V = E\left(\left(\vec{V} - \vec{\mu}_V\right)\left(\vec{V} - \vec{\mu}_V\right)^T\right) \quad (2)$$

where $\vec{\mu}_V = E\left(\vec{V}\right) = [E(v_1), E(v_2), \dots, E(v_n)]^T$ is the expectation of $\vec{V}$. The eigenvector $\vec{e}_f$ of each feature and its corresponding eigenvalue $\lambda_f$ are the outcomes of the equation $C_V \vec{e}_f = \lambda_f \vec{e}_f$. We can obtain the ranking of all features from the most principal to less principal, by ordering the eigenvalues in descending order.

# 4. Machine learning – Evaluation measures

In this section, the most popular machine learning techniques that have been tested for occupancy detection and estimation along with their evaluation measures are briefly presented.

## 4.1 Machine learning techniques

In this section, we briefly describe the aforementioned machine learning methods along with the Adaptive Boosting algorithm and we point out the evaluation measures to be used to evaluate classifiers performance.

### 4.1.1 Support Vector Machines (SVM)

Support Vector Machines (SVMs) classifier is one of the most convenient and widespread classification and regression algorithms and it was first proposed by Boser, Guyon and Vapnik in 1992 [Boser92]. SVM is a machine learning technique based on risk minimization. The main objective of SVM is to construct a hyperplane as a decision boundary as the maximum margin between classified classes based on Kernel functions. Several kernel functions have been deployed so as to improve the predictive performance of the SVM. In our work, we apply two Kernel functions: Polynomial and Radial Basis Function, denoted hereafter as SVM-POLY and SVM-RBF, respectively.

#### 4.1.1.1Radial Basis Function (RBF) kernel

The SVM-RBF case relies on the Gaussian Radial Basis function kernel with its form given by: $K(x, y) = \exp(-\gamma\|x - y\|^2)$, where $\|x - y\|^2$ is the Euclidean distance between the feature vectors *x* and *y,* and $\gamma = \frac{1}{2\sigma^2}$ is a positive constant with $\sigma$ to be a free parameter.

#### 4.1.1.2 Polynomial Function (POLY) kernel

The SVM-POLY case relies on the Polynomial function kernel given by: $K(x, y) = [x^T y + \theta]^p$, where $p$ is the degree of the polynomial and $\vartheta$ is a free parameter that usually is an integer number, although $\theta = 1$ is preferable as it prevents Hessian matrix from becoming zero. As in RBF, a free parameter C is defined.

### 4.1.2 Decision Trees (DT)

Decision Tree learning is a technique for approximating discrete-valued functions, in which the learned function is represented by a decision tree (or Classification Tree or Learning Tree). Decision trees can also be re-represented as sets of if-then rules so as to improve human readability. This tree-shaped structure is capable of generating classification rules for the tested dataset [Lee01].

### 4.1.3 Random Forest (RF)

Random Forest, also known as random decision forest, is an ensemble of decision trees and each decision tree is constructed by using a random subset of the training data, while the output class is the mode of the classes decided by each decision tree. Random Forest is unexcelled in accuracy among current classifiers, it runs efficiently on large databases and it can handle a vast amount of input variables without variable deletion [Breiman].

### 4.1.4 Back Propagation Network (BPN)

Artificial Neural Networks (ANNs) have seen an increasing interest over the last few years and are being successfully applied on a significant range of problem domains. One of the most popular ANNs algorithms is the back propagation algorithm. BPN is a feed-forward model with supervised learning [Rumelhart86].

### 4.1.5 Adaptive Boosting (AdaBoost)

Adaptive Boosting or AdaBoost is a kind of ensemble learning technique where multiple learners, also called weak learners, are employed to build a stronger learning algorithm. The main goal of AdaBoost is to improve the performance of classification. AdaBoost takes as input a weak learner and iteratively improving it by re-calculating its weights for the incorrectly classified cases in the training set. There are many forms of boosting algorithms [Nath03, Schapire12], but the most popular is AdaBoost, where the weak classifiers are decision trees [Freund95]. In this work, we use the AdaBoost.*SAMME* – Stagewise Additive Modeling using Multi-class Exponential loss function, which is an extension of AdaBoost.*M1* algorithm, so as to perform both two-class and multi-class (multi-label) classification scenarios.

A brief description of the AdaBoost.*SAMME* algorithm is provided next. For a training set $TS_n = [(x_1, y_1), \dots, (x_n, y_n)]$ where $y_i, i = 1, \dots, n$ the classification labels, the observation weights $w_{wl}(i)$ for the weak learners are initialized as $w_{wl}(i) = \frac{1}{n}$. These weights are recomputed according to classifier achievements. Iteratively, for $wl = 1, \dots, k$, a weak classifier $T^{(wl)}(x)$ is utilized in order to minimize the error:

$$err^{(wl)} = \frac{E_{wl}}{\sum_{i=1}^{n} w_{wl}(i)} \tag{3}$$

with

$$E_{wl} = \sum_{i=1}^{n} w_{wl}(i) I(T^{(wl)}(x_i) \neq y_i) \tag{4}$$

where *I* is the indicator function, equal to one when the argument is true and zero otherwise. After *wl* iterations the weights are initial updated by the following function:

$$w_{wl+1}(i) = w_{wl}(i) \exp(a_{wl} I\big(T^{(wl)}(x_i) \neq y_i\big) \tag{5}$$

where $a_{wl} = \frac{1}{2} \ln\left(\frac{1-err^{(wl)}}{err^{(wl)}}\right) + \log(K-1)$, where *K* is the number of classes in dataset. After the initial updates the weights are re-normalized. The final strong classifier is

$$T^{(final)}(x) = \underset{j \in y_i}{\operatorname{argmax}} \left\{ \sum_{wl=1}^{k} a_{wl}\, I\big(T^{(wl)}(x_i) = j\big) \right\} \tag{6}$$

### 4.1.6 Hidden Markov Models (HMMs)

A Hidden Markov Model (HMM) [Rabiner89] is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved states. In a Hidden Markov Model, the state is not directly visible to the observer; only the output, dependent on the state, is visible. Each state has a probability distribution over the possible output tokens. Hidden Markov Models are mainly applied in temporal pattern recognition problems such as gesture recognition and part-of-speech tagging. A HMM implementation that applies maximum-likelihood estimation for parameter learning, and online forward estimation for inference, is utilized.

### 4.1.7 Conditional Random Fields (CRFs)

A Conditional Random Field (CRF) [Lafferty01] is a type of discriminative probabilistic graphical model for labelling sequence data. It is used in many fields such as activity recognition [Vail07], gesture recognition [Wang06], sentence analysis [Sha03], etc. CRFs can be represented as undirected graphs, in which vertices denote random variables and edges denote dependencies between random variables. There are various CRF structures and types, with the linear-chain CRF being the most common one. A linear-chain CRF representation is shown in Figure 1. *Yt* represents the hidden state at time *t*, whereas $X_t$ represents the

visible observation at time *t*. Time is divided in time steps, which are intervals of constant length. A transition to a new state is performed at each time step. The objective is to infer the most probable hidden state given the observation, or the most probable sequence of hidden states given an observation sequence. An observation at each time step is consisted of a set of features (feature vector), where each feature is a real value and is associated with a numerical weight. Thus, multiple features are combined in order to infer the probability of each hidden state.

The CRF structure utilized in this work is similar to the one of a Hidden Markov Model. However, HMMs assume that the observations *X* are independent of each other. Moreover, due to the Markov property, they assume that *a)* the next state only depends on the current state and *b)* the current observation only depends on the current state [Kasteren08]. A CRF can model long term dependencies and does not require the independence assumptions of a HMM. A key advantage of a CRF is the ability to include a variety of arbitrary and non-independent features of the observations. Another difference from a HMM is the way the model parameters are learned. The parameter learning for a HMM is related to maximizing the *p(x,y)* distribution, while for a CRF it is related to maximizing the conditional distribution *p(y|x)*. The probability of a state sequence *y* given an observation sequence *x* is calculated by the following equation:

$$p(y \mid x) = \frac{1}{Z(x)} \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, x_t)\right\} \tag{7}$$

where Z(x) is a normalization function, $f_k$ is a feature function and $\lambda_k$ is a learned weight associated with feature $f_k$.

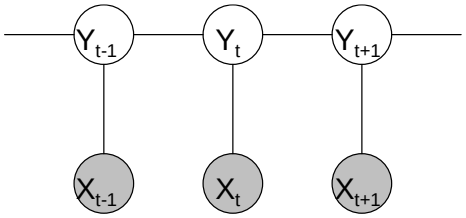


**Figure 1:** Linear-chain CRF: The white and grey nodes represent the hidden states and the observations, respectively

## 4.2 Evaluation measures

For a two-class classification scenario, in order to assess our models, we use the measures of precision, recall, accuracy and F-measure, which are computed from the contents of the confusion matrix of the classification predictions (see Table 1). True positive and false positive cases are denoted as TP and FP, while true negative and false negative are denoted as TN and FN respectively. In order to fit the classification evaluation in occupancy detection problem, we will assign the classes *absence* and *presence*.

*Precision* is the ratio of predicted true positive cases to the sum of true positives and false positives and is given by the equation:

$$Precision = \frac{TP}{TP + FP}$$

*Recall* is the proportion of the true positive cases to the sum of true positives and false negatives and is given by the equation:

$$Recall = \frac{TP}{TP + FN}$$

*Accuracy* is the fraction of the total number of predictions that were correct.

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

Precision or recall alone cannot describe a classifier's efficiency. That's why F-measure is introduced as a combination of these two metrics. It is defined as twice the harmonic mean of precision and recall and is the metric we will be most referring to.

$$F - measure = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

A value closer to one, means better combined precision and recall of the classifier, whereas lower values imply worst accuracy or precision or both.

**Table 1**: Confusion matrix for occupancy detection.

| | | **Predicted class** | |
|---|---|---|---|
| | | **Absence** | **Presence** |
| **Actual Class** | **Absence** | *TP* | *FN* |
| | **Presence** | *FP* | *TN* |

The form of Table 1 is similar in multi-class multi-label classification scenario and the calculation of precision, recall, accuracy and F-measure is based on all the classes and then on averaging them to get a single real number measurement.

When estimating occupancy density or the exact number of occupants, there are more than two possible classes. In these cases, the Root Mean Square Error (RMSE) metric is used in addition to the Accuracy metric. RMSE is defined as below:

$$RMSE = \sqrt{\frac{\sum_{t=1}^{n} (\hat{y}_t - y_t)^2}{n}}$$

Where $y_t$ is the true occupancy class, $\hat{y}_t$ is the estimated occupancy class and $n$ is the total number of instances. RMSE measures the difference between the estimated and the actual occupancy state and provides a more thorough view compared to the accuracy metric. The normalized error (NRMSE) is the RMSE divided by the range of the observed occupancy classes. The use of NRMSE allows direct comparison of results in different scenarios.

# 5. Occupancy Estimation in Indoor Environments

In this section we present an overview of the occupancy detection/estimation methods that have been developed in enCOMPASS. The methods exploited for the occupancy detection and inference utilize machine learning algorithms for classification. As a result, a model has to be created in advance through a training process. Furthermore, both methods support various occupancy granularity outputs, such as two-class occupancy detection, occupancy density or exact number of occupants, depending on the model that is built.

## 5.1 Machine learning based occupancy estimation approach using temporal classifiers

The first method utilizes a temporal classifier such as a HMM or a CRF, in order to estimate the occupancy. The decision engine does not rely on the classifier's output only, but it applies pre-defined rules when necessary, in order to make proper adjustments and corrections.

In the occupancy estimation domain, each feature value, which is a real number normalized to [0, 1], is computed after processing the data of a particular sensor. An observation of the temporal classifier includes the set of features deriving from the installed sensors. Therefore, the observation contains information received from different sensors, allowing the information fusion of different sources. Each occupancy class, represented by an integer value, is modelled as a hidden state. The number of the different hidden states depends on the selected occupancy granularity. For instance, when presence detection is applied, there are two possible states, the occupied state (1) and the unoccupied state (0). The number of different possible states when estimating the exact number of occupants in certain spaces may be high. As possible lack of information for a particular intermediate state, which has not been observed in the training set, can affect the model and the produced estimations, the estimation of occupancy density is more appropriate in these cases. The number of different states when estimating occupancy density is limited. A density class corresponds to a percentage range which shows the occupancy ratio with regard to the maximum occupancy capacity of a space.

Figure 2 depicts the representation of a hidden state and an observation of a temporal classifier in the occupancy estimation domain, when three different sensor types are utilized in the same space. At each time step, the feature *f* of each sensor type is computed. A feature vector which includes all the computed features is formed, representing an observation of the model. The associated state is an occupancy value based on the selected occupancy type.

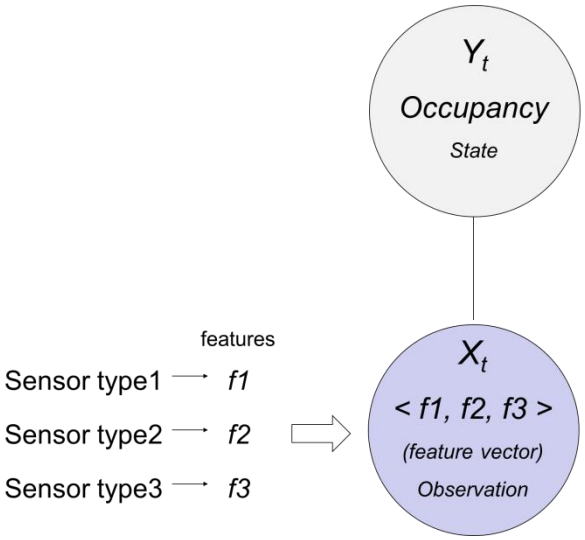


**Figure 2:** Observation and state of a classifier in the case of occupancy estimation

The inference algorithm is able to work in on-line mode, in order to produce estimations in real-time. At each time step, the features that are based on sensorial data are computed, and the observation is passed to the classifier. The classifier updates the probabilities for each occupancy class and returns the estimated state, which is the class with the highest probability.

The parameters of the model are determined in the training phase. The supplied training data are a sequence of observations and provide the actual occupancy label for each observation per time step. During the training process the model learns the dependencies between the states and the observations. The parameter values are chosen such that the logarithm of the likelihood is maximized (maximum-likelihood estimation). Due to the fact that it is not always possible to analytically compute the model's parameter values, iterative approaches such as gradient-based methods are applied. In our implementation, the training algorithm utilized is BFGS[1], which is a quasi-Newton method.

The occupancy estimation engine receives as input a stream of real-time sensor events that are generated by the installed sensors. It includes two main sub-components: the probabilistic classifier and the rule-based component. The engine handles the incoming sensor events and updates the state variables of the sensors accordingly. At the end of each time step, which can be set to be equal to 10 or more seconds, the features of all sensors installed in a monitored space are computed and then forwarded to the corresponding model, which is loaded to the CRF (or HMM) classifier, in order to estimate the current occupancy. The engine logs the classifier's output and in the case there is a change, an occupancy related event is created. The event contains a timestamp, the ID of the space, and the occupancy value. The procedure is repeated for each monitored space. The occupancy estimation engine is also responsible for applying the rules. In general, the rules applied do not directly affect the model's derived estimation but the value of features. The inactivity rule can be applied to all spaces and is triggered when there is no change in the observation vector for the last X minutes and the estimated occupancy is other than zero. The rule initializes each feature value to the default, usually 0, and as a result the estimated occupancy at the next time step will be changed to 0 by the classifier. This can be useful in case a connection with a specific sensor is lost. The inactivity time period is set per each monitored space and its value depends on the space type and the types of installed sensors. Additional rules can be defined based on the types of sensors that are utilized. The occupancy estimation process is illustrated in the figure below.

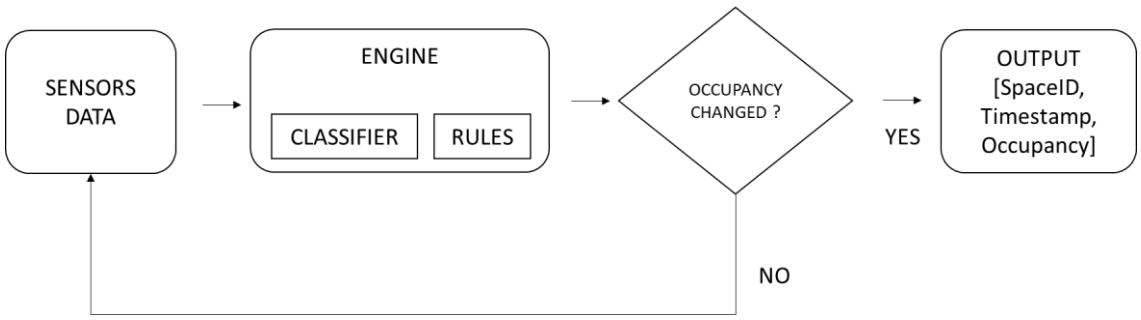


**Figure 3:** Flowchart of the first occupancy estimation approach

The occupancy estimation method has been evaluated at different spaces in a commercial building (CERTH/ITI premises). Two types of non-intrusive occupancy sensors have been used: PIR motion detectors and $CO_2$ sensors. Evaluation results are shown in Section 6.

[1] Broyden–Fletcher–Goldfarb–Shanno

## 5.2 Machine learning based occupancy estimation approach using Adaptive Boosting

According to the second approach for occupancy estimation, the Mutual Information feature selection method is applied in a pre-processing phase. This allows us to evaluate the information gained from each one of the different features in relation with occupancy that are included in the dataset. After this pre-processing step, the most meaningful features that best describe occupancy changes are defined and fed to the selected machine learning algorithm. The different machine learning classification algorithms supported are Support Vector Machines, Decision Trees, Random Forest, and Back-Propagation Neural Network. The occupancy estimation approach also utilizes Adaptive Boosting (*AdaBoost.SAMME*) in an attempt to improve classifiers' predictive performance. We have to note that the feature selection step can be useful for limiting the number of installed sensors required, as only the most important ones can be kept for training the final model which will be then used in the occupancy estimation process. The flowchart describing the approach is shown in Figure 4.

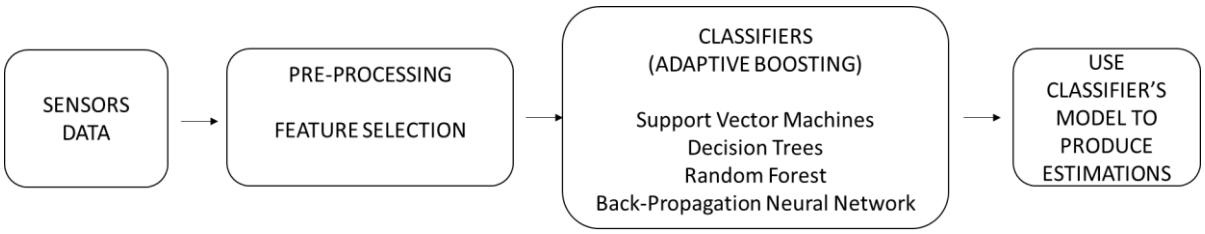


**Figure 4:** Flowchart of the second occupancy estimation approach (Adaptive Boosting)

Preliminary evaluation of the method has been performed in a residential environment which accommodates three occupants. Energy consumption data were used as input in order to detect whether the residence was occupied or not (presence detection). More details about the experimental setup and preliminary results of both approaches are presented in Section 6.

## 5.3 Selecting types of sensors

Besides performance, other aspects have to be also taken into account when selecting a sensor combination for occupancy inference: These are the cost efficiency, privacy preservation and obtrusiveness. Due to the fact that two different setups may yield similar performance, the most appropriate combination can be determined based on the other criteria.

Cost efficiency refers to the total cost, which includes both procurement and setup cost. Total cost is determined by the types of sensors utilized, the total number of sensors and complementary necessary equipment required (gateways, connectors etc.).

Another important criterion is privacy efficiency, since it is very important for occupants to feel that their privacy is not violated by the multi-sensorial installation. Some sensors, such as cameras, may raise concerns since they are deemed to violate privacy (low privacy efficiency), while others, such as environmental sensors, are more easily acceptable by the end-users (high privacy efficiency).

Occupants' sense of obtrusiveness plays also a crucial role in the selection of sensors. Certain sensor installations may be more obtrusive to users than others. For example, a PIR motion sensor is considered very unobtrusive, while a door counter might be more obtrusive since it can be easily spotted by the occupants.

All the aspects mentioned above should be taken into account when evaluating models that utilize different combinations of sensors at a given space. The final selection of the sensors/model that will be applied is based not only on the reported performance for each sensor combination tested, but on the other aspects as well, depending on preferences and priorities.

# 6. Experimental setup and results

In this section, performance evaluation results of the occupancy estimation approaches are presented.

## 6.1 Temporal classification based approach

### 6.1.1 Sensor types and spaces

PIR motion and $CO_2$ sensors were used in order to detect the presence of occupants or to estimate the occupancy density in a commercial building. The PIR motion sensors were already available via the existing alarm system of the building. An activation event is sent to a PIR motion sensor, when movement is detected. After a *N*-second time period (in the experiment *N=3*) of inactivity, a *deactivation* event is sent. At each time step, the feature extracted from a PIR motion sensor is either 0 or 1, based on its state. If during the period of the last time step the sensor had been activated, the value 1 is assigned. Instead, in case the PIR remained inactive during the whole duration of the last time step, the value 0 is assigned, as no activity is considered.

A $CO_2$ sensor measures the carbon dioxide concentration in the air, in ppm (parts-per-million). It can provide information on occupancy density, even though there are some limitations such as the slow response rate and the effect of ventilation systems on readings. $CO_2$ sensors are installed in the breathing zone, at a height of about 1.20 – 1.80 m above floor level. The $CO_2$ sensor outputs an integer value which denotes the ppm of carbon dioxide in the space. In order to extract the feature of the $CO_2$ sensor at each time step as a normalized value in [0, 1], a lower bound (*LB*) and an upper bound (*UB*) are set at 400ppm and 2550 ppm, respectively. An additional feature for the $CO_2$ sensor is also extracted. It is also normalized to [0, 1] and is calculated as:

$$0.5 + \frac{\frac{CO2(t) - CO2(t - 10\,\mathrm{min})}{UB - LB}}{2}$$

It shows the difference between the current value and the value observed 10 minutes ago. Thus, the trend of the carbon dioxide concentration is captured. A value equal to 0.5 means no change, a value greater than 0.5 shows denotes an increase in the $CO_2$ concentration, whereas a value less than 0.5 denotes a decrease in concentration.

The occupancy sensors were installed in different types of spaces (Figure 5). The first office (Office_1) accommodates 10 employees and a PIR motion sensor is used in order to detect the presence of occupants. The second office (Office_2), which has a maximum capacity of 15 occupants, is used for conferences. A $CO_2$ sensor and a PIR motion sensor are installed in order to detect the presence of occupants and estimate the occupancy density (5 discrete occupancy levels). The third office (Office_3), is the administration office and accommodates 4 employees. The space is normally occupied during the working hours. Two PIR motion sensors are available in order to detect the presence of occupants. A PIR motion sensor and a $CO_2$ sensor have also been installed in the building's kitchen. The kitchen usually accommodates around 20-30 occupants during the lunch time and is occasionally occupied throughout the day. The objective was to estimate the occupancy density and detect the presence of occupants as well.

Sensorial data and the actual occupancy to be used as ground truth were collected in order to create the dataset per each space. A number of weekdays were selected per space in order to be used in the training and the testing phase. For the training part, tests showed that 3 – 4 weekdays of data were enough for

effective occupancy detection and occupancy density estimation. Moreover, the use of large datasets for training the models is not always feasible, as such data may not be available in practice. A model per sensor combination was created, in order to evaluate different sensor combinations per space. The ground truth was annotated manually via the use of a webcam which stored images on a PC.

Office_1

Office_2

Kitchen

PIR & $CO_2$ sensors

**Figure 5:** Different types of spaces and sensors in CERTH

### 6.1.2 Experimental results

Table 2 presents the accuracy metric when detecting the presence of occupants in each one of the 4 different spaces. In all cases, Accuracy is greater than 80%. Another observation is that the type of space and its usage pattern affect the detection outcome, as it can be perceived by comparing the results between Office_1 and Office_2. The latter space is not occupied continuously during the working hours, but only when meetings occur. The lower accuracy compared to the one achieved in Office_1 is due to false positives reported (space wrongly estimated as occupied) for a time period after the end of meetings.

**Table 2:** Occupancy detection results (presence/ absence) utilizing CRF classifier

| Space | Sensors utilized | Accuracy |
|---|---|---|
| Office_1 | PIR motion | 94% |
| Office_2 | PIR motion | 82% |
| Office_3 | 2 × PIR motion | 96% |
| Kitchen | PIR motion | 92% |

Table 3 presents the NRMSE and Accuracy metrics when estimating occupancy density using $CO_2$ and PIR motion sensors, in Office_2 and Kitchen.

**Table 3:** Occupancy density estimation results (5 discrete occupancy levels) utilizing CRF classifier

| Space | Sensors utilized | NRMSE | Accuracy |
|---|---|---|---|
| Office_2 | $CO_2$, PIR motion | 0.27 | 80% |
| Kitchen | $CO_2$, PIR motion | 0.177 | 81% |

Comparison of CRF with the HMM classifier

Preliminary tests showed that the CRF classifier outperformed the HMM classifier in most cases. Although in some spaces the difference in performance was negligible, the HMM classifier produced more false negatives at presence detection (Figure 6), which is a behaviour that can negatively affect the comfort of the occupants in the case occupancy-based automated decisions are made. Moreover, in some cases the occupancy density estimation produced by the CRF classifier was more stable with fewer fluctuations among the different occupancy classes.

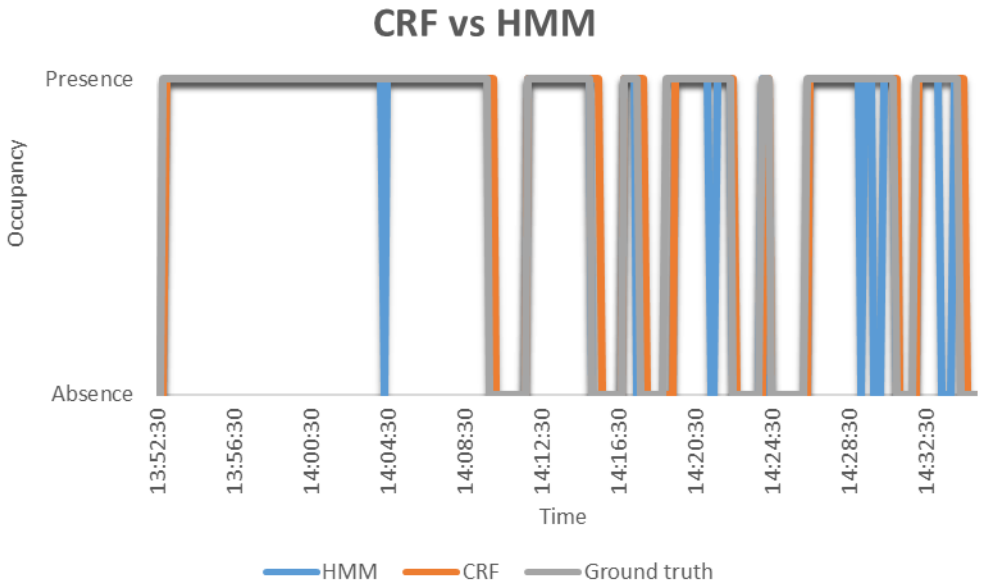


**Figure 6:** CRF versus HMM when detecting presence

## 6.2 Adaptive Boosting based approach

This section describes the Adaptive Boosting approach which has been applied when utilizing data from smart meters (electricity and water consumption) that were installed in a residence. Mutual Information has been applied in order to select the most important input data. Different machine learning classifiers have been evaluated using various parameters through Monte Carlo simulations.

### 6.2.1 Data description and dataset creation

Three different systems were deployed in a domestic environment which accommodates 3 occupants, in order to capture and store the energy and occupancy related data. The data collection was performed for about 1 month.

The power consumption of crucial electrical appliances is monitored via a wireless network of smart plugs that utilize the ZigBee protocol[2]. The installed smart plug modules communicate with each other forming a network of mesh topology. Each one of the following appliances was attached to a smart plug: TV, washing machine, refrigerator, and hair dryer. Furthermore, a special built-in module, which is connected to the

[2] https://www.plugwise.com

wiring, is used in order to monitor the power consumption of the electrical kitchen appliance. An aggregator application requests the current power consumption from each module every minute, receives the corresponding messages which include the measured power consumption of the connected appliance in Watts, and then stores the data directly into the database. It is worth noting that the timestamps of all events that are generated by the three systems are in UTC.
A highly accurate, active infrared door counter sensor which includes four receivers has been installed at the main entrance of the house. The sensor is able to detect entries and exits based on the break-reset timings of the beams. Thus, it keeps track of the number of occupants that are present in the residence and dispatches the occupancy value, on change. Occupancy events are stored along with the timestamp into the database.

### 6.2.2 Data pre-processing

After retrieving the raw data of the systems, a processing step was performed in order to create the final aggregated dataset which includes events per 1-minute intervals of all the measured features. Periods during which the occupants were on vacation were excluded from the final dataset. Moreover, periods in which some of the data were not available due to power failures or network connectivity issues, were excluded as well.

The initial aggregated dataset constructed after processing the raw data contains 5 features [Central Power (lights of the domestic environment), Refrigerator, TV, Washing Machine, Dryer] denoted hereafter as [CP, R, TV, WM, D] and the target Occupancy, denoted hereafter as [OCCUP]. The dataset contains energy consumption data of 1-minute resolution for a time interval of 16 consecutive days during summer time. Thus, the shape of overall dataset is 23040x5 (without taking into account the target feature) and its sparsity is 74.44%. Figure 7 depicts the energy consumption of some electrical devices, with regard to occupancy.

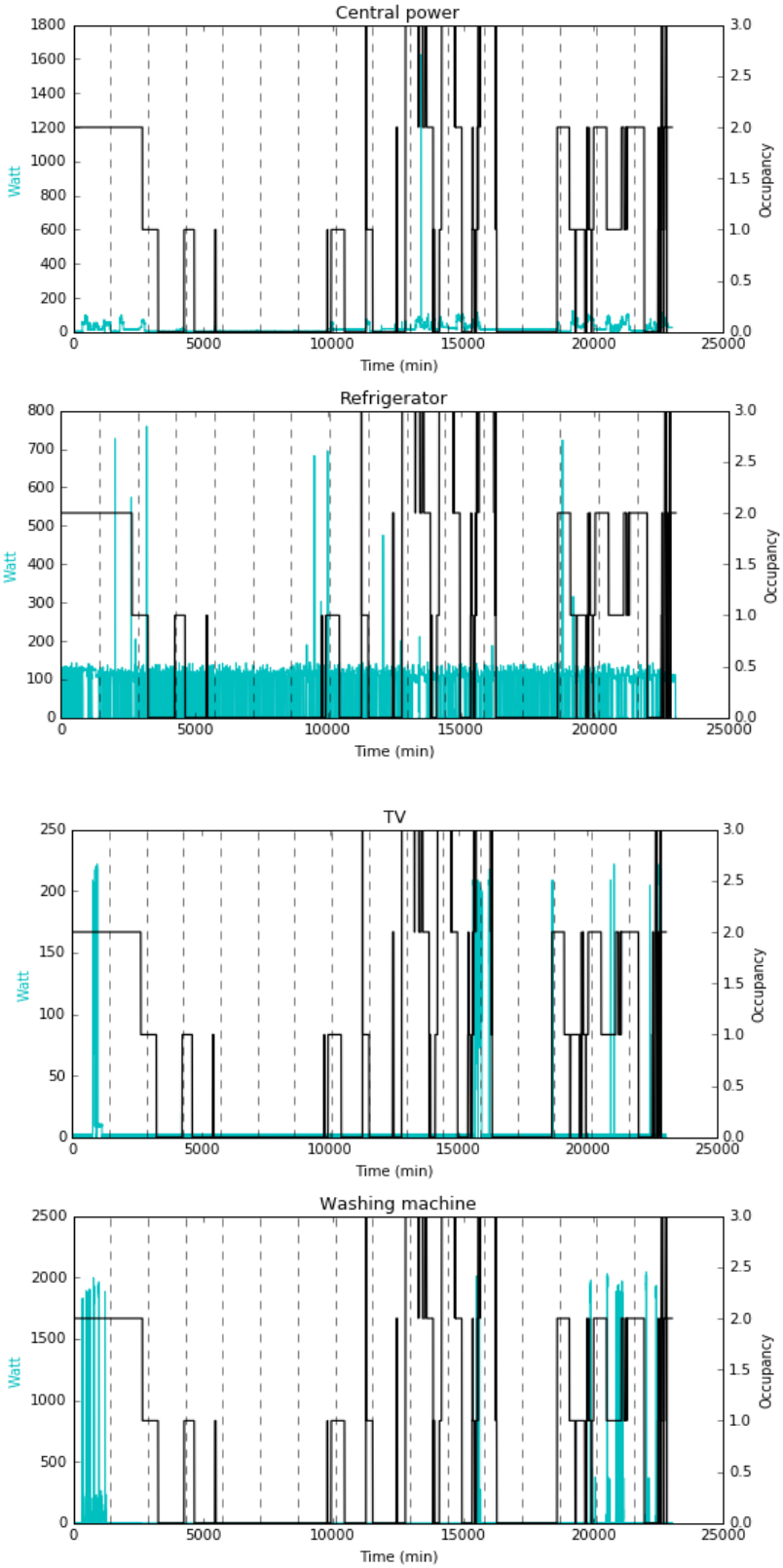


**Figure 7:** Energy consumption of electrical devices central power, refrigerator, TV and washing machine (cyan solid line - left axis) compare to occupancy (black solid line - right axis).

From an optical point of view, one can see from Figure 7 that Central Power, TV and Washing Machine features are somehow correlated with occupancy. On the other hand, the calculation of Pearson's

correlation among dataset reveals a two-dimensional behaviour among features, and cannot provide concrete results about occupancy inference. Table 4 presents the Pearson's correlation among variables.

**Table 4:** Correlation matrix of power consumers with occupancy

| | Central Power | Refrigerator | TV | Washing Machine | Dryer | Occupancy |
|---|---|---|---|---|---|---|
| Central Power | 1 | 0.11 | 0.116 | 0.149 | 0 | 0.497 |
| Refrigerator | 0.11 | 1 | 0.091 | 0.009 | 0 | 0.073 |
| TV | 0.116 | 0.091 | 1 | 0.242 | 0 | 0.219 |
| Washing Machine | 0.149 | 0.009 | 0.242 | 1 | 0 | 0.191 |
| Dryer | 0 | 0 | 0 | 0 | 0 | 0 |
| Occupancy | 0.497 | 0.073 | 0.219 | 0.191 | 0 | 1 |

The results from Table 4 show that Occupancy feature is not strongly correlated with the energy consumption features except from Central Power feature (0.497).

### 6.2.3 Feature selection utilizing Mutual Information

We all acknowledge the fact that interdependency exists among human habits and occupancy inference. In our work, these habits are denoted by the 5 features described previously. Thus, in order to rank the influence of each feature to occupancy inference and extract the most useful information, we have used Mutual Information (MI) as the feature selection technique. MI measures how much one random variable provides information about another. It is a dimensionless quantity, and can be thought of as the reduction in uncertainty about one random variable given knowledge of another. High mutual information indicates a large reduction in uncertainty; low mutual information indicates a small reduction; and zero mutual information between two random variables means the variables are independent.
In Table 5, the features with the highest MI, based on occupancy, are presented in descending order.

**Table 5:** Descending order of features based on feature extraction technique of MI

| **Feature** | **Mutual Information** |
|---|---|
| Central Power | 0.3549 |
| Washing Machine | 0.1646 |
| Refrigerator | 0.1217 |
| TV | 0.0652 |
| Dryer | 0 |

### 6.2.4 Simulation setup

Our main objective is to find the predictive model that is more efficient on occupancy inference based on energy consumption data. To that end, our simulation schema is based on the application of all tested classifiers. For cross-validation of our results, we generate a training set and a testing set, in a percent of 70% and 30%, respectively of the tested dataset. We generate 100 Monte Carlo iterations for different parameter scenarios in each classifier. The tested classifiers were SVM, DT, RF and BPN models. We employed the boosting technique with 5 weak learners to all the classifiers mentioned above in order to improve classification performance. For SVM-POLY, $\theta$ takes the values $\theta = (start = 30, end = 60, step = 6)$ and the polynomial degree takes the values $p = (2,7,1)$. For SVM-RBF $\sigma$ varies same as $\theta$ and the constant C as $C = (1000,10000,2000)$. The classic BPN has a single hidden layer and the number of neurons varies as $n = (100,200,20)$. The RF has an ensemble of $estimators = (20,100,20)$ DTs. The combination of all values of parameters and a size of 100 Monte Carlo iterations for each case, results in an overall of 6100 tested cases.

### 6.2.5 Simulation results

Due to the vast amount of simulation results, we present results from specific simulation cases on the dataset and from the application of AdaBoost.

**Table 6:** Precision, recall, accuracy and F-measure (estimated averages) from 100 Monte-Carlo iterations for RF and BPN classifiers

| **Classifier: RF** | | | | |
|---|---|---|---|---|
| **Estimators** | **Precision (%)** | **Recall (%)** | **Accuracy (%)** | **F-measure (%)** |
| 20 | 72.78 | 85.94 | 78.24 | 78.81 |
| 40 | 73.03 | 86.31 | 78.33 | 79.11 |
| 60 | 73.17 | 86.42 | **79.02** | 79.24 |
| 80 | 73.44 | 86.34 | 78.83 | 79.37 |
| 100 | 73.56 | 86.49 | 78.75 | 79.50 |
| **Classifier: BPN** | | | | |
| **Neurons** | **Precision (%)** | **Recall (%)** | **Accuracy (%)** | **F-measure (%)** |
| 100 | 74.79 | 83.28 | 76.24 | 78.81 |
| 120 | 75.93 | 84.31 | 75.87 | 79.90 |
| 140 | 75.21 | 83.33 | **76.71** | 79.06 |
| 160 | 76.07 | 84.52 | 76.33 | 80.07 |
| 180 | 76.12 | 84.03 | 76.58 | 79.88 |

Table 6 shows the simulation results of RF and BPN, and one can see that the highest accuracy, when RF is applied, is 79.02% (highlighted value on RF case) $(estimators = 60)$ and 76.71% when BPN is applied $(neurons = 120)$ (highlighted value on BPN case). On the other hand, SVM-POLY achieved an accuracy of 76.45% $(p = 2, \theta = 42)$, while SVM-RBF achieved an accuracy of 74.82% $(C = 5000, \theta = 36)$. Preliminary simulations results show a superiority of RF and DT classifiers compare to other for our classification scenario. RF classifier had achieved 80% for accuracy and 83% for F-measure (on average of 100 Monte-Carlo iterations), with DT to follow very closely. The rest of the tested classifiers achieved below 78% on accuracy and around 80% for F-measure. Table 7 present's precision, recall, accuracy and F-measure (on average of 100 Monte-Carlo iterations) with the application of boosting on tested classifiers.

**Table 7:** Precision, recall, accuracy and F-measure (estimated averages) from 100 Monte-Carlo iterations with the application of Adaboost

| **Classifier** | **Parameters of weak learners** | **Precision (%)** | **Recall (%)** | **Accuracy (%)** | **F-measure (%)** |
|---|---|---|---|---|---|
| SVM – POLY | p = 2, 3, 4, 5, 6<br>θ = 30, 36, 42, 48, 56 | 74.79 | 89.34 | 79.83 | 81.42 |
| SVM – RBF | C = 1000, 3000, 5000, 7000, 9000<br>θ = 30, 36, 42, 48, 56 | 74.35 | 89.07 | 80.06 | 81.04 |
| DT | - | 74.89 | 91.37 | **80.94** | 82.31 |
| RF | 20, 40, 60, 80, 100 | 73.91 | 95.17 | 80.23 | **83.20** |
| BPN | n = 100, 120, 140, 160, 180 | 74.01 | 92.83 | 80.21 | 82.36 |

From Table 7, one can see that the DT with AdaBoost achieves the higher performance compared to the other tested classifiers (see highlighted values) with 80.94% accuracy (82.31% F-measure), while the RF follows closely in accuracy (80.23%), but achieves higher F-measure compared to DT (83.20%). The

application of boosting provides an improvement of maximum 2% of the performance of the RF classifier compared to the non-boosted case of RF for the best set of simulation parameters. This outcome, point to the fact that the application of boosting saved us time and effort to search for the appropriate set of simulation parameters, in order to achieve the best predictive performance of the classifier.

# 7. Conclusions

The present document described methods for presence detection and occupancy estimation. The methods utilize well-known machine learning classifiers. They can be used for the estimation of various granularities of occupancy, such as presence detection and occupancy density. They can also be applied to diverse types of spaces, such as different types of offices in commercial buildings, and residences. Results showed that it is possible to achieve high detection performance when using non-intrusive sensor types, such as PIR motion sensors, $CO_2$ sensors and smart meters.

The algorithms have been tested in CERTH's premises, while they will be tested, parameterized and extended with real-data from the pilot buildings. As a next step, we plan to perform experiments, in order to further improve the performance of the methods developed, by discovering and applying more efficient features of the input data. Lastly, further experiments can be performed in order to evaluate the already created models when they are applied to new spaces where the same sensing equipment is installed.

# 8. References


- [Agarwal10] Y. Agarwal, B. Balaji, R. Gupta, J. Lyles, M. Wei, T. Weng, "Occupancy-driven energy management for smart building automation", In Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building (BuildSys'10), pp. 1–6, ACM, New York, NY, USA, 2010
- [Benezeth11] Y. Benezeth, H. Laurent, B. Emile, C. Rosenberger, "Towards a sensor for detecting human presence and characterizing activity", Energy and Buildings, vol. 43, pp. 305–314, 2011
- [Boser92] B.E. Boser, I.M. Guyon, V.N. Vapnik, "A training algorithm for optimal margin classifiers", Proceedings of the Fifth Annual Workshop on Computational Learning Theory (ACM'92), pp. 144–152, 1992
- [Breiman] L. Breiman, "Random forests – random features", Technical Report 567, Statistics Department, University of California, Berkeley, ftp://ftp.stat.berkeley.edu/pub/users/breiman
- [Chaney16] J. Chaney, E.H. Owens, A.D. Peacock, "An evidence based approach to determining residential occupancy and its role in demand response management", Energy and Buildings, vol. 125, pp. 254-266, 2016
- [Christensen14] K. Christensen, R. Melfi, B. Nordman, B. Rosenblum, R. Viera, "Using existing network infrastructure to estimate building occupancy and control plugged-in devices in user workspaces", International Journal of Communication Networks and Distributed Systems, vol. 12(1), pp. 4-29, 2014
- [Corna15] A. Corna, L. Fontana, A. A. Nacci, D. Sciuto, "Occupancy detection via iBeacon on Android devices for smart building management", Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition, March 2015, EDA Consortium, pp. 629-632, 2015
- [Dagnely15] P. Dagnely, T. Ruette, T. Tourwé, E. Tsiporkova, C. Verhelst, "Predicting hourly energy consumption. Can you beat an autoregressive model?", Proceeding of the 24th Annual Machine Learning Conference of Belgium and the Netherlands, pp. 105-122, 2015
- [Dong10] B. Dong, B. Andrews, K.P. Lam, M. Hoynck, R. Zhang, Y.-S. Chiou, D. Benitez, "An Information Technology Enabled Sustainability Test-Bed (ITEST) for Occupancy Detection through an Environmental Sensing Network", Energy Build., vol. 42 (7), pp. 1038-1046, 2010
- [Freund95] Y. Freund, R.E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting", Computational Learning Theory, Springer, pp. 23–37, 1995
- [Hailemariam11] E. Hailemariam, R. Goldstein, R. Attar, A. Khan, "Real-time occupancy detection using decision trees with multiple sensor types", Proceedings of the 2011 Symposium on Simulation for Architecture and Urban Design, Society for Computer Simulation International, San Diego, CA, USA, pp. 141-148, 2011
- [Hotelling33] H. Hotelling, "Analysis of a complex of statistical variables into principal components", Journal of Educational Psychology, vol. 24 pp. 417–441, 498–520, 1933
- [Hutchins07] J. Hutchins, A. Ihler, P. Smyth, "Modeling count data from multiple sensors: a building occupancy model", IEEE International Workshop on Computational Advances in MultiSensor Adaptive Processing (2007) DOI: 10.1109/CAMSAP.2007.4498010
- [Jain14] R. K. Jain, K. M. Smith, P.J. Culligan, J.E. Taylor, "Forecasting energy consumption of multi-family residential buildings using support vector regression: Investigating the impact of temporal and spatial monitoring granularity on performance accuracy", Applied Energy, vol. 123, pp. 168-178, 2014
- [Kasteren08] T.L.M. Kasteren, A.K. Noulas, B.J.A. Krose, "Conditional Random Fields versus Hidden Markov Models for Activity Recognition in Temporal Sensor Data", In Proceedings of the Annual Conference of the Advanced School for Computing and Imaging, Netherlands, June 2008
- [Kleiminger13] W. Kleiminger, C. Beckel, T. Staake, S. Santini, "Occupancy detection from electricity consumption data", In Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings, ACM, November, pp. 1-8, 2013, DOI: 10.1145/2528282.2528295
- [Kleiminger15] W. Kleiminger, C. Beckel, S. Santini, "Household Occupancy Monitoring Using Electricity Meters", Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, 2015, DOI: 10.1145/2750858.2807538
- [Krinidis14] S. Krinidis, G. Stavropoulos, D. Ioannidis, D. Tzovaras, "A Robust and Real-Time Multi-Space Occupancy Extraction System Exploiting Privacy-Preserving Sensors", Proceedings of International Symposium on Communication Control and Signal Processing, pp. 542-545, 2014
- [Kuutti14] J. Kuutti, P. Saarikko, R.E. Sepponen, "Real Time Building Zone Occupancy Detection and Activity Visualization Utilizing a Visitor Counting Sensor Network", 11th International Conference on Remote Engineering and Virtual Instrumentation (REV), IEEE, Polytechnic of Porto (ISEP) in Porto, Portugal, 26-28 February, pp. 219-224, 2014

- [Labeodan15] T.M. Labeodan, W. Zeiler, G. Boxem, Y. Zhao, “Occupancy measurement in commercial office buildings for demand-driven control applications”, Energy and Buildings, vol. 93, pp. 303-314, 2015
- [Lafferty01] J. Lafferty, A. McCallum, and F. C. Pereira, “Conditional random fields: Probabilistic models for segmenting and labelling sequence data”, In Proceedings of ICML, Morgan Kaufmann, San Francisco, CA, June 2001, pp. 282-289, 2001
- [Lee01] S.J. Lee, K. Siau, “A review of data mining techniques”, Industrial Management & Data Systems, vol. 101(1), pp. 41–46, 2001
- [Liao15] C. Liao, P. Barooah, C.L.C. Liao, “An integrated approach to occupancy modeling and estimation in commercial buildings”, In: American Control Conference (ACC’15), pp. 3130-3135, 2015
- [Mahdavi09] A. Mahdavi, C. Proglhof, “Toward empirically-based models of peoples presence and actions in buildings”, In 11th International IBPSA Conference, pp. 537-544, 2009
- [Meyn09] S. Meyn, A. Surana, Y. Lin, S. M. Oggianu, S. Narayanan, T. A. Frewen, “A sensor-utility-network method for estimation of occupancy distribution in buildings”, IEEE 48th Conference on Decision and Control (2009) DOI: 10.1109/CDC.2009.5400442
- [Milenkovic13] M. Milenkovic, O. Amft, “An opportunistic activity-sensing approach to save energy in office buildings”, Proceedings of the fourth international conference on Future energy systems, ACM, pp. 247-258, 2013
- [Nath03] S.V. Nath, R.S. Behara, “Customer churn analysis in the wireless industry: a data mining approach”, Proceedings – Annual Meeting of the Decision Sciences Institute, pp. 505–510, 2003
- [Nguyen12] T. A. Nguyen, M. Aiello, “Beyond Indoor Presence Monitoring with Simple Sensors”, Proceedings of the 2nd International Conference on Pervasive and Embedded Computing and Communication Systems (PECCS), Rome, Italy, pp. 5-14, 2012
- [Nguyen13] T. A. Nguyen, M. Aiello, “Energy intelligent buildings based on user activity: A survey”, Energy and Buildings, vol. 56 (1) pp. 244-257, 2013, http://dx.doi.org/10.1016/j.enbuild.2012.09.005
- [Pearson01] K. Pearson, “On lines and planes of closest fit to systems of points in space”, Philosophical Magazine, vol. 6(2), pp. 559–572, 1901
- [Rabiner89] L. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition”, In Proceedings of the IEEE, vol. 77 (2), pp. 257-286, 1989, http://dx.doi.org/10.1109/5.18626
- [Rumelhart86] D.E. Rumelhart, G.E. Hinton,R.J. Williams, “Learning representations by back-propagating errors”, Nature, vol. 323, pp. 533-536, 1986
- [Ryu16] S.H. Ryu, 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
- [Saha14] M. Saha, S. Thakur, A. Singh, Y. Agarwal, “EnergyLens: combining smartphones with electricity meter for accurate activity detection and user annotation”, In Proceedings of the 5th international conference on Future energy systems, ACM, pp. 289-300, 2014
- [Sha03] F. Sha, F. Pereira, “Shallow parsing with conditional random fields”, In Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL '03), Association for Computational Linguistics, Edmonton, Canada, pp. 134-141, 2003, http://dx.doi.org/10.3115/1073445.1073473
- [Schapire12] R.E. Schapire, Y. Freund, “Boosting: Foundations and Algorithms (Adaptive Computation and Machine Learning Series)”, MIT Press, 2012
- [Vail07] D.L. Vail, M.M. Veloso, J.D. Lafferty, “Conditional random fields for activity recognition”, In Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems (AAMAS '07), ACM, New York, NY, USA, Article 235, 14-18 May, 2007, http://doi.acm.org/10.1145/1329125.1329409
- [Wahl12] F. Wahl, M. Milenkovic, O. Amft, “A Distributed PIR-based Approach for Estimating People Count in Office Environments”, Proceedings of the IEEE 15th International Conference on Computational Science and Engineering (CSE '12), IEEE Computer Society, Washington, DC, USA, pp. 640-647, 2012, http://dx.doi.org/10.1109/ICCSE.2012.92
- [Wang06] S. Wang, A. Quattoni, L.P. Morency, D. Demirdjian and T. Darrell, “Hidden Conditional Random Fields for Gesture Recognition”, Computer Vision and Pattern Recognition, vol. 2, pp. 1521-1527, 2006, http://dx.doi.org/10.1109/CVPR.2006.132
- [Yang03] D. Yang, H. Gonzales-Banos and L. Guibas, “Counting people in crowds with a real-time network of simple image sensors”, Proceedings of the International Conference on Computer Vision, pp. 122–129, 2003
- [Yang14a] L. Yang, K. Ting, M.B. Srivastava, “Inferring Occupancy from opportunistically Available Sensor Data”, IEEE International Conference on Pervasive Computing and Communications, pp. 60–68, 2014

- [Yang14b] Z. Yang, N. Li, B. Becerik-Gerber, M. Orosz, “A systematic approach to occupancy modelling in ambient sensor–rich buildings”, Simulation, vol. 90(8), pp. 960-977, 2014, http://dx.doi.org/10.1177/0037549713489918
- [Yang15] Z. Yang, B. Becerik-Gerber, “Cross-space building occupancy modeling by contextual information based learning”, Proceedings of the 2nd International Conference On Embedded Systems for Energy-Efficient Built Environments, ACM, pp. 177-186, 2015