INDOOR OCCUPANCY DETECTION AND TRACKING USING NETWORKED SENSOR NODES

A Dissertation

by

MUHAMMAD EMAD-UD-DIN

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of the Committee,	Dezhen Song
Co-Chair of Committee,	Ya Wang
Committee Members,	Abdullah Muzahid
	Eun Jung Kim
	Theodora Chaspari
Head of Department,	Scott Schaefer

December 2023

Major Subject: Computer Science

ABSTRACT

Indoor occupancy detection and tracking are essential elements of occupant behavior research. One of the recent advancements in this area is the use of networked sensor nodes to create a more comprehensive occupancy picture of an indoor space where multiple sensor nodes can identify human presence while delivering superior accuracy compared to a system that relies on standalone sensor nodes. Standalone sensor nodes often produce false negative and positive detections, due to sensor coverage gaps and shortcomings in the underlying occupancy sensing technologies.

This research aims to improve indoor occupancy detection and tracking performance using network-level sensor fusion and occupancy estimation techniques. These techniques exploit the redundant, correlated, and complementary information in the time-series data that networked occupancy sensor nodes produce. Using a combination of node deployment configurations and indoor environments, several occupancy detection and tracking methods are proposed which can potentially contribute towards applications areas like Occupant comfort, Indoor health, Energy and space utilization, Building Design, and Occupant safety & security.

The presented work focuses on optimizing networked nodes-based occupancy detection and tracking pipeline. Some notable elements of a typical pipeline include data collection and labeling strategies, sensor models, data fusion techniques, node-level and network-level machine learning algorithms, and estimation algorithms. Each sensor node

can use one or multiple node-level occupancy sensors technologies such as microphones, ambient temperature, humidity, carbon dioxide (CO_2), and passive infrared (PIR) sensors.

The main contributions of this dissertation include: (i) A node-level on-device lifelong (ODLL) classifier is proposed that continuously learns evolving occupancy patterns over time; (ii) A network-level algorithm that exploits the inter-node spatial adjacency information and as well as observation correlations between the nodes; (iii) A 1-minute resolution occupancy tracking system that exploits the node adjacency and node correlation constraints to filter-out the unreachable occupancy states.

DEDICATION

To my mother, my sons, and my siblings

ACKNOWLEDGMENTS

I want to express my gratitude to those who helped me accomplish this dissertation.

I especially thank my advisor, Dr. Ya Wang, for being my mentor, for her guidance, support, and encouragement, and for helping me succeed in stressful times. I feel privileged and honored to have her as my advisor. Her support, her mentoring skills, and the opportunities she provided me mean so much to me and everyone who has been praying for my success. She has been my source of encouragement and inspiration. I wish her success so she can keep doing the same for others, for years to come.

I thank my committee chair Dr. Dezhen Song, and committee members, Dr. Abdullah Muzahid, Dr. Eun Kim, and Dr. Theodora Chaspari, for providing feedback and constructive criticism on this dissertation.

Lastly, I want to thank my lab mates for being an irreplaceable help that enabled me to write this dissertation.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a dissertation committee consisting of Dr. Dezhen Song as the committee chair, Dr. Ya Wang as the advisor and the committee co-chair, the committee members Dr. Abdullah Muzahid, Dr. Eun Kim and Dr. Theodora Chaspari of the Department of Computer Science and Engineering.

I was partially helped by Dr. Zhangjie Chen, and Xin Zhao of the Department of Mechanical Engineering, in the data collection process during the Fall of 2021 and Spring of 2022. Qijie Shen of the Department of Electrical and Computer Engineering worked on the SLEEPIR node software while Dr. Libo Wu created the SLEEPIR sensor node.

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

This research was supported and funded by DOE ARPAE DE-AR0000531 (2015-2018) and DE-AR0000945 (2018-2022).

NOMENCLATURE

ADC	Analog-Digital Converter
ADP	Application Desired Parameters
AI	Artificial Intelligence
ANN	Artificial Neural Network
AR	Autoregressive
ARM	Advanced RISC Machine
ASHRAE	American Society of Heating, Refrigerating, and Air-Conditioning Engineers
AWT	Absolute Water Content
BF	Bayes Filter
BPTT	Back Propagation Through Time
BLE	Bluetooth Low Energy
CEC	California Energy Commission
CO_2	Carbon-dioxide
CTRNN	Continuous-Time Recurrent Neural Network
DBI	Davies Bouldin Index
DRAM	Dynamic Random Access Memory
EKF	Extended Kalman Filter
EM	Electromagnetic
FFNN	Feed-Forward NN
FMA	Fundamental Markov Analysis

FoV	Field-of-View
GRU	Gated Recurrent Unit
HCS	Hierarchical Classifier Selection
HMM	Hidden Markov Model
HVAC	Heating Ventilation and Air-Conditioning
IBC	International Building Code
IECC	International Energy Conservation Code
IoT	Internet of things
IR	Infrared
KNN	K-Nearest Neighbor
LDA	Linear Discriminant Analysis
LSTM	Long Short-Term Memory
MCU	Microcontroller Unit
MDP	Markov Decision Process
MGU	Minimal Gated Unit
ML	Machine Learning
MTBF	Mean Time Before Failure
NDD	Node Deployment Density
NEMA	National Electrical and Manufacturers Association
NFPA	National Fire Protection
NN	Neural Networks
ODLL	On-device Lifelong Learning

PDLC	Polymer-Dispersed Liquid Crystals
PF	Particle Filter
PIR	Passive IR
QDA	Quadratic Discriminant Analysis
RF	Radiofrequency
RH	Relative Humidity
RNN	Recurrent NN
SENSOR	Saving Energy Nationwide in Structures with Occupancy Recognition
SoC	System-on-a-Chip
SLEEPIR	Synchronized Low Energy Electronically chopped PIR
SRAM	Static Random Access Memory
SVM	Support Vector Machine
TVOC	Total VOC
VOC	Volatile Organic Compounds
WEKA	Waikato Environment for Knowledge Analysis

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

1. INTRODUCTION

Over the years, progress has been made toward developing methods and algorithms to detect and track human occupancy in enclosed spaces. A varying number of occupancy sensing nodes mounted with a combination of sensor modalities have been used within indoor spaces to estimate occupancy. These nodes act in standalone or networked mode depending upon the application and end-user preferences regarding the occupancy detection or tracking solution. The networked sensor nodes create a more comprehensive occupancy detection and tracking system where multiple sensors can identify human presence with enhanced accuracy compared to a system that relies on standalone sensor nodes. This chapter aims to present the background and motivation for using networked sensor nodes and their associated network-level sensor fusion and estimation techniques. This chapter also mentions the scope and objectives of the research carried out for this dissertation.

1.1. Background and Motivation

1.1.1. Conventional Node-level Occupancy Detection and Tracking Systems

Occupancy detection, in simple terms, means whether there is a human presence in a spatial unit during a period. Thus, occupancy detection is attributed to a spatial unit like a zone, room, floor, or building, as well as to a temporal unit, such as a second, minute, hour, day, week, or month. The resolution for these units depends upon the application for which occupancy information is needed. For example, to evaluate daily energy utilization of a building, the temporal resolution can be as large as a day, while the spatial unit can encompass the whole building. However, the same units are not valid for an application

like HVAC control as temperature may need to be adjusted over a short duration, like a minute, and regulated over a smaller space like a room instead of the whole building. It should be highlighted here that occupancy sensor nodes are limited by underlying sensing technology regarding the range in which they can sense human presence. This range can be referred to as *spatial coverage*. For example, the maximum range of motion-based occupancy sensors is 20 meters [5], which represents the farthest edge of the sensor detection cone that defines the spatial coverage of a typical motion sensor. Moreover, sensor nodes are also limited regarding the frequency and duration in which observations are recorded. The time during which the sensors gather observations can be called *temporal coverage*. The temporal coverage depends on the sensor's sampling frequency, node's duty cycle, and communication polling/push rate.

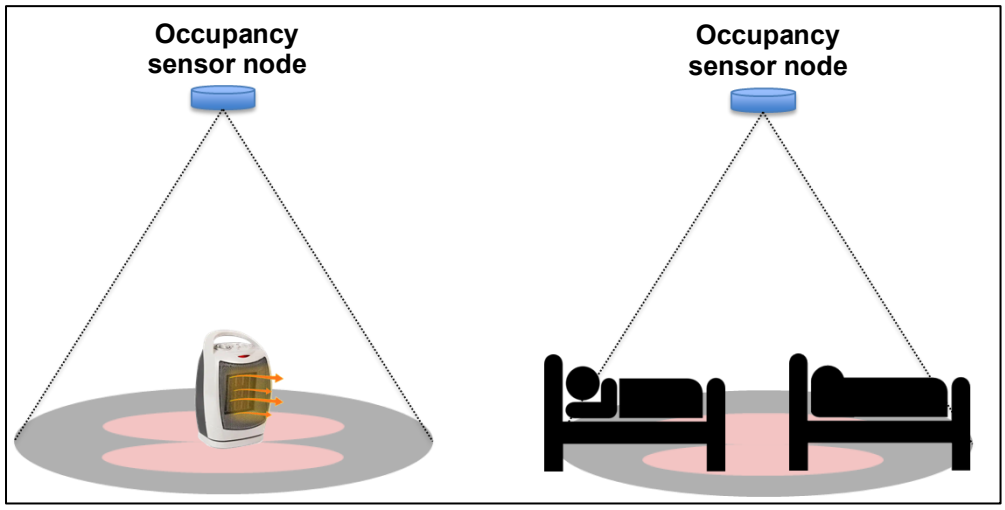

Figure 1. (Left) Parts of the space heater exhibit similar IR radiation to the human body generating false positives. (Right) An occupant with a body covered in a blanket during sleep is not detected by the PIR sensor leading to false negatives.

Besides occupancy detection, occupancy tracking is another crucial element of occupant behavior. Occupancy tracking is performed by detecting occupancy over time

and maintaining the history of movement of the occupants in space. Since research conducted for this dissertation aims to preserve occupant privacy, individual occupants are not tracked. Still, it is the occupancy status of the spatial unit that changes from zone to zone (or room to room), that is tracked. Similar to occupancy detection, the temporal and spatial units need to be defined for occupancy tracking. The noticeable difference here is that multiple sensor nodes are required to track the movement of occupants between rooms or zones (each installed with at least one sensor node).

1.1.2. Gaps in Node-level Occupancy Detection

Conventional occupancy detection systems need to be improved; these depend on the underlying sensing technology used by the sensors. For example, PIR sensors, the most commonly used technology for occupancy sensing, especially for lighting control [6], produce false negatives by failing to detect stationary occupants [7]. PIR sensors also

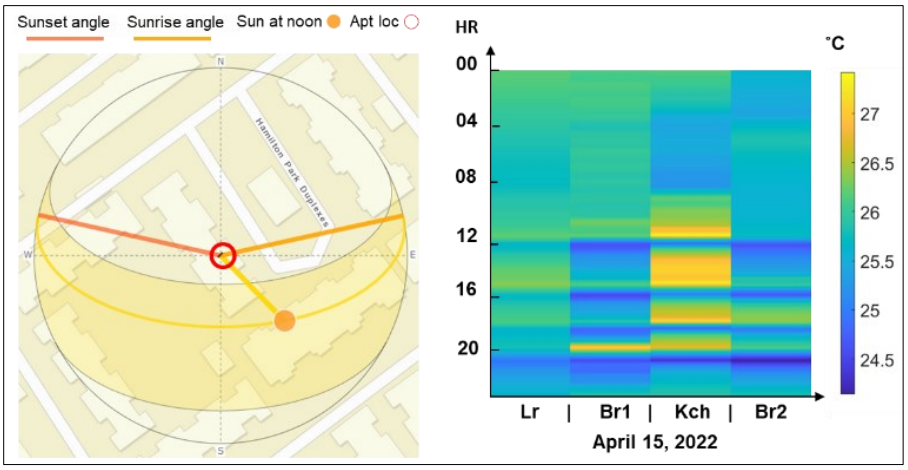

Figure 2. (Left) Apartment location depicted along sunrise and sunset angles on Apr 15, 2022. (Right) Temperature profile recorded by each node for the same day. Livingroom (Lr) is exposed to sunlight IR via large glass sliding door thus warmer. Kitchen (Kch) used for cooking during the day. Bedroom 2(Br2) is also exposed to sunlight during daytime, but HVAC cooling is more effective due to its small size. Alternating periods of blue indicate HVAC setpoint triggers from noon until evening.

suffer from a phenomenon known as IR shielding [8]. This happens when certain clothing,

wearable materials or obstructions inhibit IR radiation from occupants from reaching the PIR sensor. This causes false negatives to occur. It has been observed in literature [9] that occupants with shorter heights and with less area of exposed skin, are less likely to be detected via PIR sensors due to lower amount IR radiation reaching the sensor. This is because pyroelectric detectors with PIR sensors respond to changes in the 7-to-14-micron portion of the EM spectrum, which happens to be the range of the IR energy radiated by human skin[10]. PIR sensors are also prone to errors that lead to false positives. An example can be warm tap water that has the potential to reach human body temperatures and can present a human-like IR radiation profile to the PIR sensor [3]. Examples of IR sources that can potentially cause false positives and negatives are shown in figure 1. Among other sensors, ambient temperature sensors produce observations that are correlated [11] with occupancy in a relatively smaller and confined space, but in practice, these observations are also dependent on factors like HVAC setpoints, which can be triggered due to the outdoor temperature, which is a completely unrelated phenomenon to occupancy. Changes in ambient temperature in a 2 bed 2 bath apartment caused due to HVAC setpoints on summer day are shown figure 2. In short, each indoor environment is unique in terms of the observable environmental variables including ambient temperature, humidity, CO_2 levels, IR radiation noise and the IR radiation profile for the occupants. Since occupancy tracking is a high-order property of the occupancy detection [6], the occupancy detection error produced by standalone sensor nodes also impacts the occupancy tracking accuracy. An obvious solution is to work towards improving the underlying sensing technologies for each sensor modality. Yet another practical approach

is to consider the multivariate time-series nature of the occupancy data received from multiple networked sensor nodes. The following section highlights this approach.

1.1.3. Networked Sensor based Occupancy Detection and Tracking

The use of multiple sensing nodes is a common technique for improving sensing accuracy and reliability. The availability of a greater volume of timestamped raw data can make the inferences and predictions more accurate and robust than if these were achieved via simple aggregation techniques like, average, maximum, minimum or a union of outputs of standalone sensor nodes[12].

Data fusion techniques are most common and have been widely employed in literature to overcome shortcomings of standalone occupancy sensor nodes. Whenever a node-level failure or communication breakdown occurs, robust inferences can be achieved through networked sensor nodes via fusion. This can happen as nodes may redundantly or complementarily observe common observation zones. Occupancy detection methods that implement various fusion approaches can be divided into three broad categories. We closely adopt the categorization done in [13]. Regardless of the category, any fusion-based occupancy detection method would use the following standard properties[12] of a fusion-based approach to their advantage.

1.1.3.1. Cooperative

Multiple networked sensor nodes under a fusion strategy which ensures occupancy detection in an area which is larger than spatial coverage of any one sensor node, is called a *cooperative fusion strategy*. This strategy is ideal for exploiting the spatial relationships

between the neighboring zones. Proposed methods described in Chapters 4 and 5 make extensive use of this strategy.

1.1.3.2. Redundant

Sensor or communication failures are commonplace with networked occupancy sensor nodes[14], especially if the distance between the communicating nodes is near the maximal range of the employed radio communication technique or protocol. A fusion strategy that ensures sensor data delivery despite node or communication failure, is called as a *redundant fusion strategy.* This is generally possible via having redundant nodes or redundant communication modules onboard the sensor nodes.

1.1.3.3. Complimentary

Under a wide variety of scenarios, multiple sensor modalities are required to detect occupancy as a single sensor modality is not able to detect occupancy as the sensor is unable to detect signal that it is sensitive to. For example, CO_2 sensor has been used

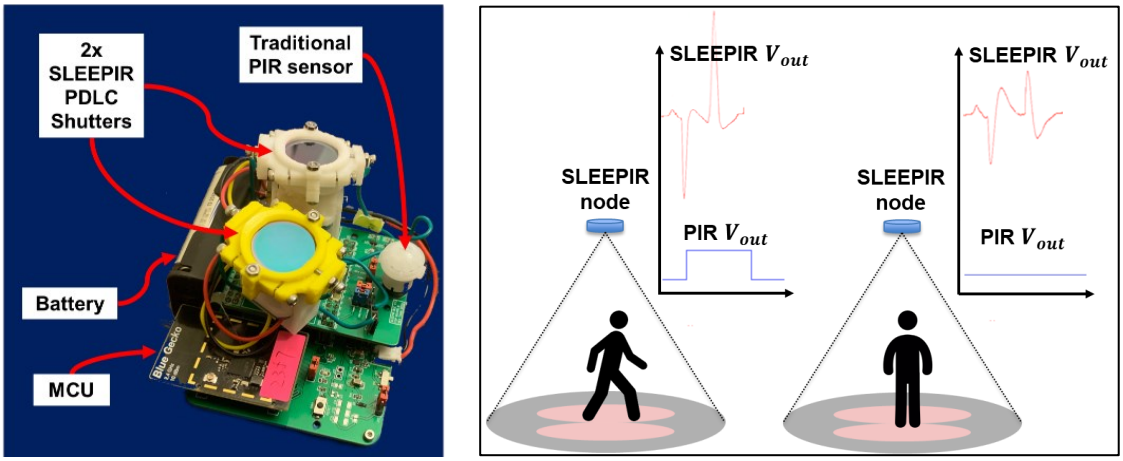

Figure 3. (Left) SLEEPIR sensor node is shown. (Right) Sensor voltage response Vout for both SLEEPIR and traditional PIR sensors are compared. Spatial coverage area for the node is shown as well. Here lighter shade represents the coverage area for 2x SLEEPIR sensors with 4m diameter, while the darker grey shade indicates the spatial coverage area for the PIR sensor with 5m diameter.

historically to detect stationary occupancy as PIR based motion sensor fail to detect such occupancy. Any fusion strategy that uses multiple sensing modalities to reliably detect occupancy is known as a *complimentary fusion strategy*. Proposed methods in Chapter 3, 4 and 5 make limited use of this strategy.

1.1.4. Networked Sensor Node

There exist a wide variety of sensors, IoT based processing platforms and communication standards employed in literature, that are used to evaluate occupancy detection and tracking methods for networked sensors. However, this study does not aim to evaluate underlying sensor node technologies. We choose most widely deployed[6], non-intrusive, privacy-preserving PIR sensor-based nodes to evaluate the proposed occupancy detection and tracking methods. These nodes are termed as SLEEPIR[7] nodes and can detect stationary occupancy.

As shown in figure 3, each SLEEPIR sensor node includes two SLEEPIR sensor modules. Each sensor module consists of an analog PIR sensors (EKMC2691111K, Panasonic Inc) mounted behind an infrared shutter made of PDLC and sandwiched by two Germanium windows that can significantly reduce the power consumption, weight, volume, and noise level, compared to mechanical choppers[15-17]. Optimal design parameters of such a sensor are provided in our previous work[18]. Onboard, Silicon Labs MCU with model no. EFR32BG13, reads the analog signals from both the SLEEPIR and the PIR sensors via ADC at a sampling frequency of 20Hz. Afterwards, the collected values are sent out as observations to a server IoT device (Raspberry Pi) via BLE connection for permanent memory storage. Alongside SLEEPIR sensor modules and

MCU, a traditional PIR sensor (EKMB1391111K, Panasonic Inc), a PDLC driving circuit and two AA batteries connected in serials (3V DC voltage supply) are also present onboard.

Within the analog PIR sensor, a pyroelectric sensing element, which is made up of pyroelectric material, converts the change of heat flux to current. If the radiation power received by the pyroelectric material is $W(t) = W_0 e^{i\omega t}$, which is modulated at frequency ω, then voltage response $V_{out}(t)$ for the preamplifier stage is in the following equation form.

$$V_{out}(t) = \frac{R_{fb}\eta p' A\omega}{G_T(1+\omega^2\tau_T^2)^{\frac{1}{2}}(1+\omega^2\tau_E^2)^{\frac{1}{2}}} W(t) \quad (1)$$

Here, p' is the perpendicular component of the pyroelectric coefficient p. A is the area of the sensing element. η represents the emissivity of sensing element; $\tau_T = H/G_T$ and $\tau_E = R_{fb}C_{fb}$ represent the thermal and electrical constant, respectively. Here H, G_T, R_{fb} and C_{fb} stand for thermal capacity, thermal conductance, feedback resistance, and capacitance, respectively. Commercial-of-the-shelf PIR sensors usually consist of two or four sensing elements placed in series with opposite polarizations. By covering the sensing elements with the same polarization, the transmission change of the PDLC shutter would introduce noticeable voltage signals from the PIR sensor. When the PDLC shutter, which is mounted in front of the PIR sensor, changes its transmission periodically, the

received radiation $W(t)$ changes periodically in synchronization as well. This in turn causes the change of the output voltage $V_{out}(t)$.

1.1.5. Gaps in Network-level Occupancy Detection

All the fusion implementation categories mentioned in methods, have namely (1) Analytical methods, (2) Knowledge based methods, (3) Data-driven methods have their own set of shortcomings that either emanate from node-level detection errors or are artifact of the particular network-level occupancy detection method.

The Analytical methods study physical behavior of occupants and its impact on the indoor environment. These methods exploit the relationship between environmental variables and human presence to derive occupancy decisions. For example, [19] presents an occupancy detection method based on various indoor climate sensor data trajectories. Data from CO_2 and VOC sensors were used to evaluate vacancy, while data from PIR sensors were used to judge occupancy. No prior information about the testbed or dataset prerequisites is required by this method. Despite relying on three different sensor

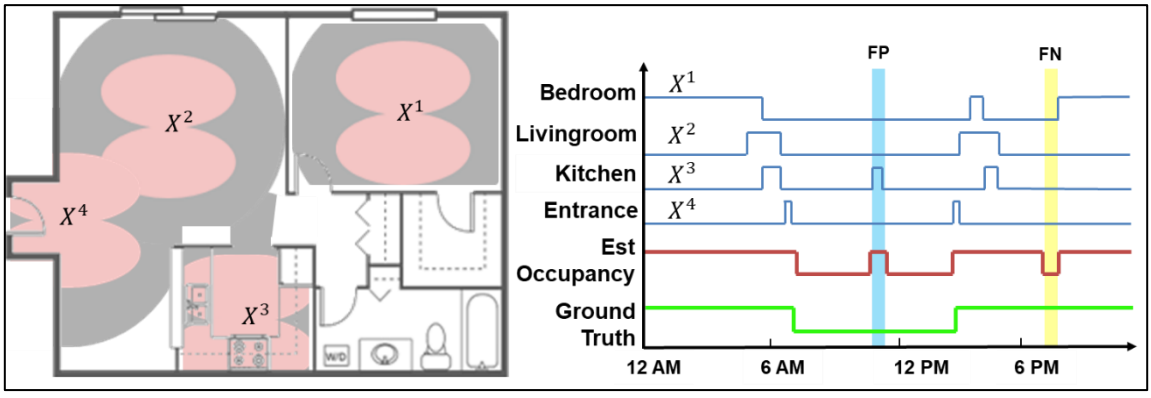

Figure 4. (Left) A deployment diagram showing locations and spatial coverage of ceiling mounted SLEEPIR sensor nodes. (Right) A time-signal chart showing occupancy output from individual nodes in blue and a union of outputs in red. The green signal represents the ground truth. It is shown that only one node-level error can introduce a network-level false positive. In contrast, all node-level outputs must be erroneous to produce a false negative network-level estimate.

modalities, this method reported as much as 43.5% false negatives and 11.8% false positives for a dormitory occupancy scenario.

The Knowledge-based methods, also known as expert systems, use specialized knowledge represented by rules to solve complex problems. A good example is a Finite State Machine based State Switch algorithm[20] that utilizes SLEEPIR nodes. Yet, it cannot robustly handle the node-level detection errors and a transition to a wrong state would be made in case a false occupancy determination was made at the node-level. Similarly, any network-level aggregation algorithm, e.g., union of outputs of standalone sensor nodes, will fail to handle a false positive detection determination made at the node level.

The Data-driven methods include the following sub-categories of methods.

(i) Statistical & deep learning ML methods

(ii) Bayesian inference methods

(iii) HMM-based methods

ML-based network-level occupancy detection methods that statistical process features extracted from raw sensor-node observations can handle uncertainty but have limited application due to the pre-requisite of acquiring labeled training dataset [21]. Not only automated occupancy labeling itself is resource intensive for such datasets but also achieving a class balance between “occupied,” and “un-occupied” label classes is an equally challenging task [21]. It has also been shown that typical ML algorithms only accept training data with fixed sizes; thus a networked node occupancy detection problem which can have a variable number of sensor nodes (due to occasional communication or

hardware failure), would need to reformat the data into a fixed format which often times requires data fusion to happen prior to ML training phase[21, 22]. Furthermore, deep learning ML models need large datasets to train. Thus, it is infeasible to re-collect a large amount of data to retrain to handle any novelty in the occupancy patterns [23]. In the Bayesian inference methods sub-category, node-level occupancy estimation is usually done via ML algorithms or Knowledge-based methods, while the network-level occupancy estimation is performed by fusing the node-level assessment through a Bayesian inference-based framework[22]. Although this approach has produced accuracy up to 93% [21] in uncontrolled experiments, generating ML inference for each node, on-device is resource intensive.

Among all the network-level occupancy detection frameworks found in the literature, two major gaps emerged. Firstly, in general, there was a near linear relationship between the number of sensors and the monitored area in order to maintain the accuracy of the solution[21], secondly, virtually all of the network-level occupancy detection frameworks required historical sensor data (may it be data correlation or training datasets) to be able to filter out the noisy node-level observations. The latter of the gaps is one of the major roadblocks in the way of widespread adoption of network-level occupancy detection methods. Figure 4 shows an example deployment diagram showing locations and spatial coverage of ceiling mounted SLEEPIR sensor nodes. Moreover, it shows a time-signal chart showing occupancy output from individual nodes,

1.2. Methodology

The research effort started with a comprehensive literature review exercise that encompassed the state-of-the-art occupancy estimation methods for networked sensor nodes. The review focused on the employed occupancy detection measures, network-level data filtering & fusion techniques, NDD, and the spatial and temporal resolution of occupancy detection and tracking. The review is novel in assessing the impact, the gap, and the enhanced accuracy networked-node occupancy detection systems offer. The findings of the literature review are listed in Chapter 2.

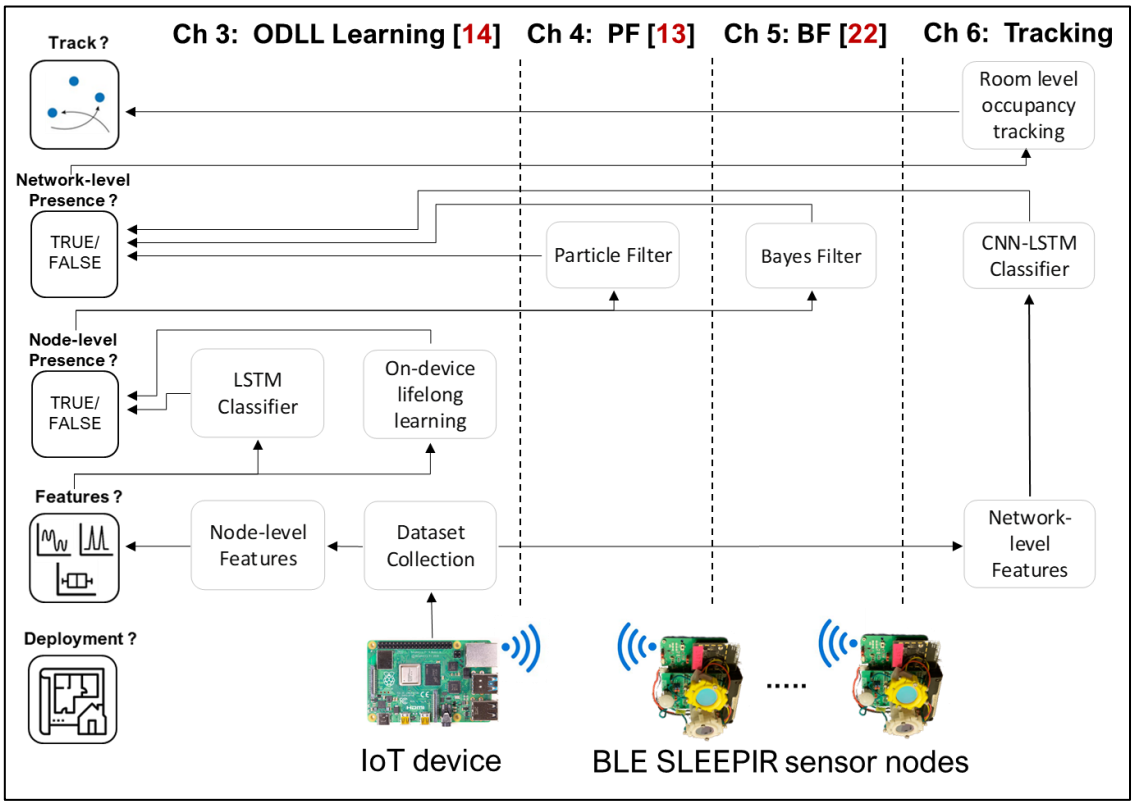

Figure 5. A summary chart of contribution towards various occupancy detection and tracking pipeline elements. Left most elements in the chart represent output for each pipeline level. Lowest level represents the hardware platforms, while the highest level represents the highest order of the functionality i.e., tracking.

The overall research methodology follows a bottom-up approach toward solving the occupancy detection and tracking optimization problem. This approach is illustrated in detail in figure 5. As defined in the last section, there are a number of issues with both the node-level and network-level occupancy estimation methods, that this approach attempts to solve.

The issues of IR shielding, IR noise, and uniqueness of environmental variables, persist in the literature because there is a lack of a comprehensive on-site training dataset that contains patterns encompassing anticipated occupancy scenarios. Such a dataset, while challenging to collect, would consequently require significant computational power to train and may still fail to adapt to novel occupancy patterns detected by the sensor. For the same reason, the ML models are typically trained off-site and model updates require over-the-cloud transfer. Any occupancy pattern which was not part of the off-site training dataset will likely cause degradation in accuracy. To overcome this challenge, in chapter 3, we propose a KNN classifier based ODLL algorithm, that can be trained "near" the SLEEPIR sensor node using an IoT device and a training dataset from the same sensor node where the occupancy inference is needed. Traditionally, ML models that are incapable of being trained locally at the end-user premises where the sensor node is present, need to be updated via over-the-cloud transfer. Although cloud-based transfer has fewer challenges in terms of application design, it is expensive in terms of latency in data transfer, added connectivity, equipment overheads and a host of data privacy and network security issues. The proposed ODLL algorithm in chapter 3 ensures that the ML training

phase happens locally, and no over-the-cloud transfer is required. For comparison purposes, an off-site static LSTM based occupancy model was also trained.

Chapter 4 addresses the gaps in the network-level occupancy estimation methods related to maximizing the spatial coverage for a networked occupancy sensor system. This chapter presents proof of concept that with a limited number of sensors and sparse spatial coverage, a PF can be used to detect the human occupancy of an indoor space regardless of environmental infrared noises. The method exploits the temporal bounds on the change in the occupancy state of the environment. It also factors in the proximity of sensor nodes to each other by evaluating inter-node observation correlation and resultantly structures the PF measurement updates in a way that human occupancy probability is spread spatially in expanded vicinity around the sensor rather than only inside the sensor observation cone.

Chapter 5 attempts to address the gap present in the network-level occupancy estimation methods that pertain to the pre-requisite of the use of historical data for filtering out the noisy node-level sensor observations. This chapter outlines a BF-based algorithm that is more robust to environmental IR disturbances when compared to PF-based occupancy detection and tracking algorithm presented in chapter 4. Moreover, unlike the PF-based algorithm, it also avoids using the historical sensor data to be able to filter out noisy node-level observations. The proposed method also allows the use of a minimal number of adjacent sensor nodes to detect the occupancy of an entire covered space of interest. The presented approach utilizes a MDP formulation[24] to model the indoor occupancy states and occupancy transition probabilities between states. We perform FMA on an underlying MDP, to evaluate transition probability and expected time to travel

between two occupancy states. These two parameters play a crucial role in filtering out environmental IR disturbances.

Chapter 6 introduces the enhanced room-level tracking capability of PF and BF-based occupancy estimation methods. Both methods are flexible enough to increase the size of tracking resolution to zones where a zone can comprise more than one room. The tracking output is also utilized as feedback to fine-tune node adjacency assumptions. As an example, the tracking output can re-adjust the room adjacency relationships because little or no transitions were recorded between two rooms which were initially thought to be adjacent and accessible to each other. Moreover, separate ODLL KNN based sub-classifiers are trained for frequently encountered occupancy scenarios which provide superior accuracy when compared to a base classifier trained for all scenarios. This limitation of the classification space to relevant scenarios improves the classifier accuracy. Overall, this chapter explains the idea of using a resource constrained specialized ML models that are trained with anticipated occupancy scenarios. Classifier selection is performed based upon identified context via feature clustering,

Chapter 7 lists the contributions of this study. It also concludes the dissertation by listing future research avenues for advancing network-level occupancy detection and tracking methods.

1.3. Scope and Expected Outcomes

This dissertation identifies the gaps that impact the performance of networked nodes occupancy detection and tracking pipelines. It proposes data collection strategies, sensor models, data fusion techniques, node-level ML, and network-level occupancy

estimation algorithms. While the evaluation underlying node-level sensor technologies is not the prime focus of this work, PIR sensors are chosen as sensors of reference, so that the performance of networked nodes-based occupancy detection and tracking systems can be compared consistently across the proposed and benchmark methods. Although references to other sensor modalities can be found within the literature review section, it is ensured that any study in the literature that does not include PIR sensors is excluded from comparisons throughout the dissertation.

The expected research outcomes from this study are listed below.

Outcome 1: A node-level ODLL classifier that continuously learns evolving occupancy patterns and reduces the average 15.43% false positive and 17.92% false negative error generated by LSTM-based offline classifier. The primary causes of these errors are environmental IR disturbances and IR shielding.

Outcome 2: A network-level algorithm capable of exploiting the adjacency and correlations between the networked nodes to reduce the average 13.70% FP and 16.31% false negatives error generated by a Finite State Machine network-level estimation algorithm.

Outcome 3: A 1-minute resolution occupancy tracking system that exploits the node adjacency constraints without the use of historical sensor data, to filter the unreachable occupancy states and reduce the average 37.56% tracking error generated by a baseline EKF based occupancy tracking.

2. LITERATURE REVIEW[1]

In the last decade, there has been a considerable shift from high-performance and energy-efficient buildings towards co-optimizing occupant comfort and building energy demand [25, 26]. Multiple studies show that a significant proportion of occupants in US office buildings (up to 75%) are dissatisfied with their thermal environment [27, 28]. Moreover, existing stand-alone occupancy sensors provide limited performance that can cause false negatives (switching off heating and lights during occupancy), resulting in occupant dissatisfaction [26]. Standalone occupancy sensor based methods struggle to achieve the same level of improvement in occupant comfort level [3]when compared to networked occupancy sensor nodes, while deployed under same configuration. Among recent review articles [26, 29-32], an overwhelming majority focuses on the methods that are based on standalone occupancy sensors. As such no review exists dedicated to an algorithmic aspect of multi-node occupancy sensing [13]. In the same set of articles, a basic premise is missing, i.e., the actual occupancy behavior depends upon the building design, sensor node positioning, room connectivity, purpose of each space in a residential or an office unit, and occupant priorities, which tend to be highly time sensitive. Thus, it may be contended that in order to propose a thorough and accurate occupancy detection model for applications like HVAC Control and Occupant Comfort [33], it is necessary to

[1] Part of this chapter is reprinted from "Indoor Occupancy Sensing via Networked Nodes (2012–2022): A Review." By Emad-ud-din, M. and Y. Wang (2023). FUTURE INTERNET JOURNAL 15(3): 116 with the permission of MDPI.

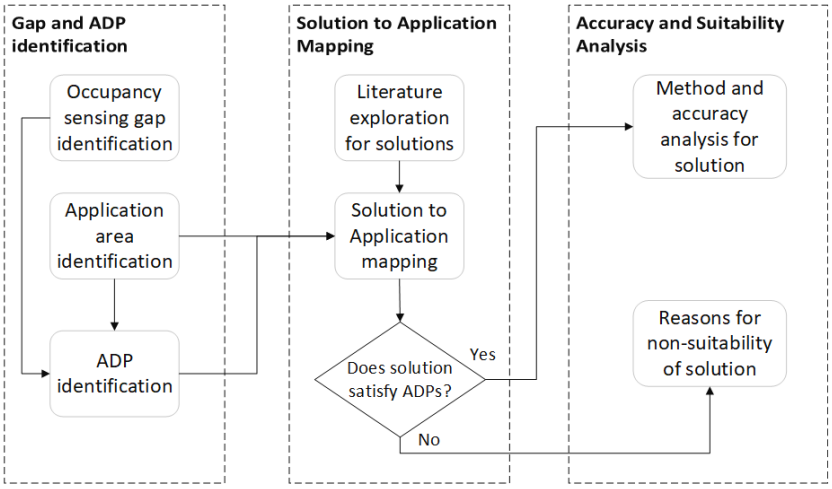

Figure 6. An illustration showing the steps of the review methodology. Reprinted with permission from [4]

have an interconnected network of occupancy sensors that is aware of the room connectivity, time-sensitive occupancy behavior and expected use of each space under observation. The use of multiple sensing nodes to improve detection performance is not novel and has been around for at least a decade. Examples of how this can be achieved include extracting ML network-level features from a multivariate raw-sensor data [34] or determining occupancy via a PF [35] that fuses the node-level ML inference to estimate an occupancy belief. Review studies use evaluation metrics from the cost Field [29] of the proposed solution, to the accuracy and failure rate [5] of the solution. This literature review, however, aims to evaluate the occupancy estimation methods for networked sensor nodes from an application's perspective. Methods are assessed based on the ADPs, i.e., accuracy requirement, information requirement, minimum sensor observation and maximum failure rate and feasible detection area. This review focuses on the employed occupancy detection measure, network-level data filtering & fusion techniques, NDD and the spatial and temporal resolution of occupancy detection. This review is novel in terms

of assessing the impact, the gap, and the enhanced accuracy that networked-node occupancy detection systems offer.

2.1. Review Methodology

This review considers research articles that outline the occupancy estimation methods and algorithms involving two or more networked occupancy sensor nodes. It tabulates the studies that highlight the gaps in the use of networked sensor nodes for occupancy sensing. These occupancy sensing gaps exist in several application areas. Each application area demands a specific set of parameters from the networked sensor nodes and the occupancy estimation methods, to address application area challenges. These parameters are termed ADPs. These ADPs are identified from the reviewed studies and are tabulated and associated with these studies. This phase is referred to as *Occupancy Sensing Gap and ADP Identification* phase. Articles in the literature that address the identified gaps are then evaluated. Comments are added to each article on whether the proposed solutions in the articles contain the ADPs demanded by a particular application area. This phase is termed the *Solution to Application Mapping* phase. The review is concluded by comparing the accuracy delivered by each of the solutions. A discussion is added by the end of this chapter about the occupancy estimation methods used by each solution, and the reasons behind the reported accuracies are highlighted. Explanations are also listed as to why certain proposed solutions are not suitable to some application areas despite the contrary claims by the authors. This phase is termed as *Accuracy and Suitability Analysis phase.* The review thus presents a complete picture to the reader, from identifying sensing gaps to the suitability of each available solution in the literature to the

gaps and application areas. It must be mentioned here that although the review's primary focus lies on networked sensor nodes-based occupancy estimation solutions, the underlying sensor modalities for each solution are listed. The reviewed estimation methods may not be agnostic to the underlying sensing technology. The illustration summarizing the review methodology is shown in figure 6.

Table I. Occupancy Sensing Gaps for Networked Sensor Nodes-Based Estimation Methods. Reprinted with permission from [4]

Studies	Application Area	Gaps
[34],[3],[36],[37],[38],[39],[40],[41]	HVAC Control and Occupant Comfort	- Stationary human detection - Real-time occupancy detection - Privacy concerns - Infrastructure overhead - False Negatives - Historical and expected occupancy data required - Reliability and fault tolerance required
[34],[36],[37, 41], [42], [43], [44], [38],[45],[46],[47]	Health and Safety	- Stationary human detection - Real-time occupancy detection - Privacy concerns - Infrastructure overhead - False positives - Zone level detection - Historical and expected occupancy data required - High accuracy required - Reliability and fault tolerance required
[39],[41],[48],[49],[11],[50],[35],[51]	Energy and Space Utilization	- Stationary human detection - Occupant count required - Zone level detection - Infrastructure overhead
[11],[52],[6],[53],[54],[35]	Security	- Stationary human detection - Infrastructure overhead - Realtime occupant tracking required. - False Negatives - High accuracy required - Zone level detection - Infrastructure overhead - Reliability and fault tolerance required

2.2. Occupancy Sensing Gaps and ADP Identification

First, the occupancy detection gaps found in the literature for networked sensor node-based occupancy estimation methods are identified. Table 1. provides a non-exhaustive but a representative list of most common gaps along with the corresponding occupancy detection application area found within a set of representative studies.

The gaps identified in table 1 point towards a specific set of incapabilities that are present either within the underlying sensing technologies, the occupancy estimation method, or the communication and integration framework that enables the networking between these sensor nodes. As mentioned earlier, this review focuses on the shortcomings of the occupancy estimation methods only and the impact of underlying sensing technologies and wireless sensor networks' communication reliability are separate topics with dedicated studies in the literature dealing with these topics. The shortcomings present within occupancy estimation methods for any given application area can be overcome through a set of parameters that the estimation method must conform to. These parameters are termed ADPs. Table 2 transforms the identified gaps into ADPs with specific values for specific applications. Standard documents from various associations and agencies like the ASHRAE[55], CEC[56], IBC[57], NFPA[58] and IECC[59], are used by the table. The table also lists IoT and Edge AI devices among the potential method execution platforms. The ADPs listed in table 2 serve as the suitability criteria when selecting a particular occupancy estimation solution for a certain application. These ADPs also bring to light certain interesting insights. For example, the accuracy requirement for occupancy sensors used for HVAC controls varies depending on the specific application and building

type. However, generally, the occupancy sensor accuracy should be high enough to correctly detect the presence or absence of occupants in a specific area. The accuracy requirement gets more stringent in the case of both safety and security applications. This is because these applications include critical services such as emergency evacuation, fire detection and suppression, and security depends on the ability of the occupancy sensor to accurately detect the presence or absence of people in a building.

Table II. ADPs for each Application Area. Reprinted with permission from [4].

Application Area	ADPs
HVAC Control and Occupant Comfort	**Accuracy requirement (ASHRAE):** ≥ 90% **Information requirement:** Historical and Expected occupancy info required. **Execution Platforms:** Enterprise core appliances, Datacenters**,** IoT & Edge AI devices **Minimum sensor observation rate (ASHRAE):** ≤ 30 minutes **Maximum sensor failure rate:** No quantification found. Still a research gap [26] **Feasible Detection Area (ASHRAE):** Office ($\leq 250\ ft^2$), Storage ($\geq 50\text{ft}^2$ & $\leq 1000\text{ft}^2$) **Feasible Detection Area (CEC):** Office ($\leq 250\text{ft}^2$), Multipurpose rooms ($\leq 1000\text{ft}^2$), Indoor spaces ($\leq 300\text{ft}^2$) **Feasible Detection Area (IECC):** Indoor spaces ($\leq 300\text{ft}^2$)
Health and Safety	**Accuracy requirement (IBC, NFPA):** ≥ 95% **Information requirement:** Historical and Expected occupancy info required. **Execution Platforms:** IoT, Edge AI devices **Minimum sensor observation rate:** ≤ 1 minute (dictated by sensor limitations) **Maximum sensor failure rate (IBC, NFPA):** 0.01% **Feasible Detection Area (CEC):** Lightening control not permitted for shutoff control in healthcare facilities or Egress lightening where power consumption $\leq 0.1W/\text{ft}^2$
Energy and Space Utilization	**Accuracy requirement (ASHRAE, IECC):** ≥ 90% **Information requirement:** Contiguous Indoor spaces need to be monitored to enable tracking applications. No historical or expected occupancy data needed. **Execution Platforms:** Enterprise core appliances, Datacenters**,** IoT, Edge AI devices **Minimum sensor observation rate:** Hourly **Maximum sensor failure rate:** No quantification found. Still a research gap [26] **Feasible Detection Area (CEC):** Indoor spaces $\leq 300\text{ft}^2$, Storage rooms ($\geq 50\text{ft}^2$ & $\leq 1000\text{ft}^2$), Office space ($\leq 250\text{ft}^2$).
Security	**Accuracy requirement (IBC):** ≥ 95% **Information requirement:** Moderate NDD to enable tracking applications. **Execution Platforms:** IoT, Edge AI devices **Minimum sensor observation rate:**≤ 1 minute (dictated by sensor limitations) **Maximum sensor failure rate (IBC):** 0.01% **Feasible Detection Area (CEC):** Indoor spaces $\leq 300\text{ft}^2$, Storage rooms ($\geq 50\text{ft}^2$ & $\leq 1000\text{ft}^2$), Office space ($\leq 250\text{ft}^2$).

Another important insight is that the response time of the occupancy detection method becomes important for safety and security applications, as it should be fast enough to detect the presence of people and activate the safety system accordingly. Thus, the potential execution platform for such applications excludes time-consuming cloud-based processing options such as Enterprise core appliances and datacenters.

Also worth noting is that CEC standards recommend using occupancy sensors in smaller indoor spaces with high traffic that have areas less than 300ft^2, while at the same time these standards recommend using occupancy sensors in storage rooms or multi-purpose spaces that can go as large as 1000ft^2. It must be mentioned here that these standards do not make recommendations for emergency facilities such as healthcare facilities or fire stations since critical operations may be affected due to automated control. It can also be observed that security applications demand high NDD since it is crucial to tracking occupants indoors. Security applications like intrusion detection are required to detect the path or trajectory (entering or leaving) the occupant is pursuing.

2.3. Solutions to Application Mapping

This review aims to establish the suitability of state-of-the-art networked sensor nodes-based occupancy estimation solutions to occupancy detection application areas. It is critical to mention here that the sensor nodes can be exposed to phenomena that can interfere with sensor measurements. The phenomena can include pronounced variations of temperature, pressure, radiation, IR shielding [8], EM shielding [60], IR noise [3] and EM noise[61]. In short, sensor measurements are error prone. Data fusion techniques have

been widely employed in literature to overcome such errors. Data fusion is "the use of techniques that combine data from multiple sources and gather this information to *achieve inferences"* [12]. The fused inferences or decisions are more informed and are thus more accurate. Moreover, fusion is robust to sensor faults, and communications breakdowns as the fusion technique employs redundant or complementary sensor nodes which can compensate for the lost information. In practice, networked sensor nodes commonly suffer from communication breakdowns [14].

It is a well-known fact that data fusion caters to the spatial and temporal coverage blind spots of sensor nodes [3]. For occupancy sensors the spatial coverage of a sensor usually means the sensor's FoV or its effective volumetric detection range. As for their temporal coverage, it usually depends on the sensor's sampling rate, node's duty cycle and communication delays[12]. Table 3 lists the reviewed methods along with the employed occupancy detection measure, network-level data filtering & fusion techniques, NDD and the spatial and temporal resolution of the occupancy detection. Most reviewed articles make explicit claims about the areas where their research is applicable, while others choose not to comment about the applicable application areas. Table 3 reviews articles that propose the occupancy detection and tracking methods, matured to a point where the authors could claim a particular application area.

Table III. Reviewed Solutions. Reprinted with permission from [4].

Solution	Filtering and Fusion Techniques	Input Data Streams	Detection Measure	NDD	Spatio-temporal Resolution & Avg Accuracy	Author claimed Application Areas
[62]	Bayesian Occupancy Model	PIR sensor nodes	BF based prior probability requires historical data & sensor model	$403\text{ft}^2/node$	Multiple zones, 60 sec, 71%	Energy & Space Utilization
[63]	SVM, LDA, QDA, RF	PIR, Light, Temp, Sound, CO_2	ML Inference	$49\text{ft}^2/node$	Single Zone, 30 sec, 98.4%	HVAC Control & Occupant Comfort
[64]	Decision Tree	PIR, Sound, Power use, CO_2	ML Inference	$28\text{ft}^2/node$	Single Zone, 60 sec, 97.9%	HVAC Control & Occupant Comfort
[40]	RBF based Neural Network	PIR, RH, Light, Sound, Temp, CO_2	ML Inference	$430\text{ft}^2/node$	Multiple Zones, 60 sec, 87.62%	Energy and Space Utilization
[65]	Statistical Feature based FFNN	PIR, Temp, Sound, CO_2	ML Inference	27 sensor nodes, open-plan office space, 8 occupants	Multiple Zones, 5 min, 75%	HVAC Control & Occupant Comfort
[11]	AR HMM	PIR, Temp, Reed switches, Airspeed, CO_2	Expectation Maximization applied to find the local optimal solution	19 sensor nodes, lab, 10 occupants	Multiple Zones, 20 sec, 84%	HVAC Control & Occupant Comfort
[66]	Multinomial Logistic Regression	PIR, Power usage, Temp, RH, Light, Door sensors, CO_2	Predicted probability of the occupants being active, inactive or away	$14\text{ft}^2/node$	Multiple Zones, 60 sec, 94.9%	HVAC Control & Occupant Comfort
[21]	RF, Decision Tree, KNN, SVM	PIR, Temp	ML Inference	$140\text{ft}^2/node$	Multiple Zones, variable time, 99%	HVAC Control & Occupant Comfort
[30]	FFNN	PIR, RH, Light, Pressure, Temp, CO_2,TVOC, Sound, Door & Window sensor	ML Inference	$296\text{ft}^2/node$	Multiple Zones, 60 sec, 94.3%	HVAC Control & Occupant Comfort, Energy & Space Utilization
[19]	Trajectory analysis of climate sensor data	PIR, Temp, CO_2,VOC, RH, AWT, Sound	Occupancy probability via 2 and 5-minute sensor data trends	$54\text{ft}^2/node$	Single Zone, 5 min, 77.8%	HVAC Control & Occupant Comfort

2.4. Accuracy and Suitability Analysis

The following observations can be made about data presented in table 3.

a. High-accuracy solutions are suitable for Energy & Space Utilization only if they're scalable with low NDD. Since this utilization is measured across entire units, solutions with high NDD are inherently non-scalable.

b. Health & Safety and Security apps demand precise occupant tracking. While some solutions offer high accuracy ($\geq 95\%$) at the cost of increased NDD, aren't scalable for HVAC Control or Occupant Comfort & Energy and Space Utilization applications.

c. NN-based classification and regression methods are accurate, but they require fixed training sizes. Missing sensor data must be imputed for inference, and training an NN needs a historical dataset.

Table 4 details the breakdown of how the ADPs for each application area map to each of the reviewed solutions in table 3.

Table IV. ADP Determination for Reviewed Solutions. Reprinted with permission from [4]

Solution	ADPs
[62]	**Accuracy & Detection Area:** 71.0%, $403\text{ft}^2 per\ node$ **Info requirement**: Prior probabilities for Bayesian Model using 4 weeks of historical data. **Execution Platforms:** Samsung SmartThings Hub **Sensor observation rate:** 60 sec **Sensor Failure rate:** High MTBF as per datasheet for ZMOTION® ZEPIR0AA PIR sensor
[63]	**Accuracy & Detection Area:** 98.4%, $49\text{ft}^2 per\ node$ **Info requirement**: Labeled dataset for ML **Execution Platforms:** ARM based Beaglebone SoC **Sensor observation rate:** 30 sec **Sensor Failure rate:** Unspecified PIR sensor
[64]	**Accuracy & Detection Area:** 97.9%, $28\text{ft}^2 per\ node$ **Info requirement**: Labeled dataset for ML **Execution Platforms:** PC/Server **Sensor observation rate:** 60 sec **Sensor Failure rate:** PIR Sensor MTBF unknown (Phidgets 1111 IR Motion Sensor)

Table IV Continued

Solution	ADPs
[40]	**Accuracy & Detection Area:** 87.6%, $430\text{ft}^2 per\ node$ **Info requirement**: Labeled dataset for ML **Execution Platforms:** Arduino Black Widow single-board MCU, MATLAB on PC/Server **Sensor observation rate:** 60 sec **Sensor Failure rate:** Unspecified PIR sensor
[65]	**Accuracy & Detection Area:** 75.0%, $<50\text{ft}^2 per\ node$ **Info requirement**: Labeled dataset for ML **Execution Platforms:** HOBO U series event loggers, MATLAB & WEKA on PC/Server **Sensor observation rate:** 5 min **Failure rate:** Unspecified PIR sensor
[11]	**Accuracy & Detection Area:** 84.0%, $<50\text{ft}^2 per\ node$ **Info requirement**: time-series data correlations need to be evaluated pre-deployment. **Execution Platforms:** wireless measurement nodes, PC/Server **Sensor observation rate:** 20 sec **Failure rate:** Unspecified PIR sensor
[66]	**Accuracy & Detection Area:** 94.9%, $14\text{ft}^2 per\ node$ **Info requirement**: Labeled dataset for ML **Execution Platforms:** BACnet™ for sensor connectivity, R on Workstation **Sensor observation rate:** 60 sec **Failure rate:** Unspecified PIR sensor
[21]	**Accuracy & Detection Area:** 99.0%, $140\text{ft}^2 per\ node$ **Info requirement**: Domain knowledge, Labeled dataset for ML **Execution Platforms:** NI Compact DAQ, scikit-learn on ARM based Beaglebone Black SoC **Sensor observation rate:** Variable **Failure rate:** Unspecified PIR sensor
[30]	**Accuracy:** 94.3%, $296\text{ft}^2 per\ node$ **Info requirement**: Labeled dataset for ML **Execution Platforms:** Arduino Uno, ARM based Kerlink® IoT Wirnet 868 Station **Sensor observation rate:** 60 sec **Failure rate:** >10000 hr (Panasonic® PaPIRs EKMB)
[19]	**Accuracy & Detection Area:** 77.8%, $54\text{ft}^2 per\ node$ **Info requirement**: Method parameters and thresholds are set empirically for each sensor node. **Execution Platforms:** Arduino Mega, PC/Server **Sensor observation rate:** 5 min **Failure rate:** High MTBF as per datasheet (RE 200 B)

2.4.1. Comments on Solution Conformance to the Claimed Application Areas

Method mentioned proposed in [62] is not feasible for the *Energy and Space Utilization* application as claimed by the author. This is because the solution accuracy does not meet ADP accuracy criteria i.e., $71\% < 90\%$. It is a data-driven method thus has a pre-requisite of historical data collection before its deployment.

Authors in [63] claim the method's suitability for *HVAC Control and Occupant Comfort* application. Although the method is suitable on a smaller scale, it cannot scale up well as the spatial NDD is high i.e., $49\text{ft}^2/node$. It is a data-driven method.

The method mentioned in [64] is not feasible for the author claimed *HVAC Control and Occupant Comfort* application even on a smaller scale. This is because the execution platform for the algorithm is a PC/Server and the implementation is not optimized for an IoT or Edge AI [67] device. Also, the solution cannot scale up well as the spatial NDD is high i.e., $28\text{ft}^2/node$. It is a data-driven method and has a dataset pre-requisite. Alternatively, the method is suitable for *Energy and Space Utilization* applications.

The method proposed by [40] is not feasible for the *HVAC Control and Occupant Comfort* application area as claimed by the authors. Although the sensor node data is logged via MCU, the ML algorithm execution platform for the algorithm is a PC/Server. Moreover, solution accuracy does not meet ADP accuracy criteria $87.6\% < 90\%$. It is a data-driven method and has a dataset pre-requisite.

The method presented in [65] is not feasible for the author claimed *HVAC Control and Occupant Comfort* application. Sensor data is logged via third party loggers, the ML algorithm execution platform for the algorithm is MATLAB/WEKA on a PC/Server. Moreover, solution accuracy does not meet ADP accuracy criteria i.e., $75.0\% < 90\%$. It is a data-driven method and has a dataset pre-requisite.

It is not feasible to implement the method proposed in [11] for the author claimed *HVAC Control and Occupant Comfort* application. The algorithm execution platform for

the algorithm is a PC/Server. Moreover, solution accuracy does not meet ADP accuracy criteria i.e., $74.0\% < 90\%$. It is a data-driven method and requires historical sensor data for time-series data correlation evaluation.

Method presented by [66] for the claimed *HVAC Control and Occupant Comfort* application is not feasible. The solution cannot scale up well with the spatial NDD of $14\text{ft}^2/node$. It is a data-driven method and thus requires a labeled dataset. Although the method uses a standard protocol by ASHRAE for sensor communication, yet the regression algorithm execution is not optimized for IoT execution which makes the feasibility of the algorithm to be questionable to be used as solution for occupancy detection.

The method proposed by [21] was found to be suitable for the author claimed *HVAC Control and Occupant Comfort* application. The solution can scale up well as the spatial NDD is sufficient to cover an average sized room i.e., $140\text{ft}^2/node$. It is a data-driven method thus it requires a dataset to be collected pre-deployment. Moreover, a human activities layer is incorporated in the learning model which requires domain knowledge about the occupancy patterns. The solution is matured to the point that it has been implemented over an IoT device.

Method presented in [30] is suitable for the author claimed *HVAC Control and Occupant Comfort* and *Energy and Space Utilization* applications. The solution can scale up as the spatial NDD is low i.e., $296\text{ft}^2/node$. It is a data-driven method and thus requires a labeled dataset to be collected. The solution is optimized in terms of node power consumption and local processing at nodes via an IoT device. The ML pre-processing,

training and inference however is made at a back-end machine. Since a two FFNN is relatively simple to implement over an IoT compatible ML framework such as TensorFlow Lite, a case can be made that the solution is suitable for an IoT based implementation. The solution is also alternatively suitable for *Health and Safety applications* as it meets the desired ADP guidelines for this application area.

The method in [19] was found to be not feasible for *HVAC Control and Occupant Comfort* application as claimed by the author. The solution can also not scale up well with the spatial NDD of $54\text{ft}^2/node$. It is an analytical method thus it may only require domain knowledge yet certain thresholds and parameters for Zero Lag Exponential Moving Average algorithm required empirical tuning. The algorithm can be easily ported to IoT for execution which makes the method suitable for IoT execution, but the accuracy does not meet ADP accuracy criteria i.e., $77.8\% < 90\%$.

2.5. Discussion and Future Trends

One of the important ADP indicators used in the suitability analysis is NDD. It is interesting to note here that this density is a simple indicator that is evaluated by dividing the total area of the monitored indoor space by the number of sensor nodes employed by the method. This indicator has no direct relationship with the sensor FoV or range which is usually significantly smaller compared to the NDD. There is a list of factors that contribute towards determining the NDD. Most impactful ones include the node positioning strategy, estimation method accuracy, network/communication reliability, environment contributed sensor noise and the floorplan of the monitored area. Among these factors, the node positioning strategy, estimation method accuracy and network

reliability are the factors that can be optimized to decrease NDD. In effect, NDD can be thought of as a proposed optimization measure by a method, for the node positioning strategy, estimation method accuracy and network reliability. Although there exist dedicated studies for node positioning strategy[68] and network reliability[14] but all of the reviewed articles device their own positioning strategy.

During the review effort, it was noticed that most studies focused on *HVAC Control and Occupant Comfort* and *Energy and Space Utilization* applications rather than applications like *Health and Safety* and *Security.* This is because the latter of the mentioned applications have ADPs that require high reliability and accuracy which is difficult to achieve given the challenging task of occupancy detection and tracking in dynamic environments. Most of the reviewed works attempted to tackle the challenge of highly noise prone and dynamic environment by adding to the suite of sensor modalities. A small minority of methods[19, 30] presented the sensor data responsible for false positives or negatives and proposed consequent solutions to resolve the errors.

Among the researched literature one of the major gaps for data-driven occupancy detection methods was found to be needed for periodic collection of training set to incorporate novel occupancy scenarios. The dataset also includes ground truth occupancy data. This is a problem because collection and labelling of new training datasets is far from an ideal task for an end-user or in some cases it is infeasible. To address this issue, certain studies[69-71] have suggested unsupervised methods as these algorithms need not label the dataset yet such methods have limited applicability as error-prone prior expert knowledge is used for initialization of classes. This knowledge may be based on

assumptions or data distribution of sensor data that may only be valid once the occupancy patterns evolve.

The future for tacking such a complex issue lies in employing more capable IoT devices like Edge AI devices [72] so that on-device ML training and inference can be made incorporating newer occupancy scenarios. The dedicated field of ODLL[23, 73, 74] offers benefits such as a local learning approach where occupancy patterns are learned on the fly, thus making such methods suitable for practice. Moreover, privacy-preserving automated labeling techniques are the flip side of the coin when the OODL approach is used, as the collected dataset also needs to be labeled. Literature needs reliable privacy-preserving techniques in this regard, but video/image-based automated yet privacy-compromising ground-truth collection techniques[48, 75] can be found.

2.6. Conclusion

This review effort developed a matching strategy for mapping occupancy estimation methods involving networked occupancy sensor nodes to the most suitable application areas. In this effort, multiple application areas were investigated to identify a location of ADPs that can help guide the suitability determination process. The ADPs represent the most demanding requirements from the selected application area. Therefore, they can be used to derive design specifications from developing a solution or can equally be used to assess an already designed occupancy solution. ADPs are determined based on occupancy standard documentation, performance constraints, application area considerations, and the sensing technologies' limitations. As a result of stringent application area requirements placed by standardization agencies, sensor limitations, and

challenging environmental constraints, a limited number of methods were able to conform to the proposed ADPs.

3. NODE-LEVEL ODLL[2]

In this chapter, an ODLL approach [23] is proposed to improve the node level detection accuracy of SLEEPIR occupancy sensors [7, 10]. While the SLEEPIR sensor has an advantage in detecting stationary and near-stationary occupants, its performance is limited when it comes to detection range and FoV when compared to traditional PIR sensors. To resolve this issue, LSTM classifier has been deployed [20] in the past to make the occupancy inference more reliable within the range and FoV of the sensor. Attempts have also been made using Bayesian techniques for improving occupancy estimation, yet due to ever changing environmental and occupancy scenarios, considerable accuracy deterioration is noted in certain studies [3, 76]. The problem with such implementations is that these lack a comprehensive training dataset that contains patterns encompassing anticipated occupancy scenarios. Such a dataset, while challenging to collect, would consequently require significant computational power to train and yet may still fail to adapt to novel occupancy patterns detected by the sensor. For the same reason, the ML models are typically trained off-site and model updates require over-the-cloud transfer. Any occupancy pattern which was not part of the off-site training dataset will likely cause degradation in accuracy.

---

[2] Part of this chapter is reprinted, with permission, from "Promoting Occupancy Detection Accuracy Using On-Device Lifelong Learning." By Emad-ud-din, M. and Y. Wang (2023). IEEE SENSORS JOURNAL 23(9): 9595-9606. Copyright © 2023 IEEE

To overcome this challenge, a KNN classifier based ODLL algorithm is proposed, that can be trained "near" the SLEEPIR sensor node using an IoT device and the training dataset from the same sensor node where the occupancy inference is needed. In any classification problem, an ML model which is tasked with predicting the class of a sample, is expected to correctly classify any input samples that may deviate by a small margin from the target [77]. In the case of occupancy detection, the input samples can deviate from the target class i.e., "occupied" or "unoccupied" by a large margin, depending upon the occupancy or the unoccupancy scenario. In other words, for example, "occupied" class contains several sub-classes which deviate from each other by a large margin as different occupancy scenarios can produce varying IR radiation patterns. Thus, addition of more occupancy scenarios adds more subclasses to the classification problem, which impacts the accuracy of the classifier adversely [78].

Traditionally, any ML model that could not be trained at the end-user premises where the sensor node is present, needs to be updated via over-the-cloud transfer. Although cloud-based training has fewer challenges in terms of application design, it comes at the cost of latency in data transfer, added connectivity and equipment overheads, and a host of security issues such as data privacy and network attacks. The proposed ODLL algorithm ensures that the ML training phase happens locally, and no over-the-cloud transfer is required.

There are multiple factors contributing to rendering any ML model trainable locally. First and foremost is the availability of labeled data. Since most occupancy detection systems deployed at the user premises need the capability of automatically

collecting the ground truth via labeling the data, it becomes essential to collect data and train models off-site. Secondly, even if the ground truth can somehow be collected, expensive computational and memory resources in the form of expensive IoT devices need to be deployed for local training due to the large memory requirements to process the collected dataset. The cost of expensive IoT devices can in turn drive up the cost of the overall solution. The use of high-end IoT devices is feasible for some applications where accuracy is critical, as training observations are gathered from the same sensor node where the inference is made but, in our case, where SLEEPIR sensors are applied for occupancy status detection in residential and commercial buildings, the cost and device power consumption are critical factors. This is because competing traditional PIR sensor-based solutions are extremely low-cost and power-efficient. Thus, a unique observation labeling technique is proposed that makes use of a combination of temporal constraints on human walking velocity, time-elapsed between two consecutive PIR sensor observations, and

Figure 7. We use a 2 bed 2 bath apartment as our testbed. We use 4 SLEEPIR sensor nodes for this floor plan for dataset collection purposes. Floor plan and sensor node locations are shown. Reprinted with permission from [2].

observation distribution. This technique allows to gather ground truth for stationary occupancy on-site without any additional equipment or computing power.

For dataset collection, 4 SLEEPIR sensor nodes were deployed at the testbed as shown in Figure 7. The nodes were deployed at a residential apartment unit with 2 bedrooms and 2 bathrooms. The expanse of the apartment is 10m x 14m. Each node is installed at the height of 2.8 meters. As already mentioned in chapter 1, each node embedded with a traditional PIR sensor can detect human motion in a circular area of radius 2.4 meters. Each node also contains 2x SLEEPIR sensor modules, which can detect stationary and moving occupants in a circular area of radius 1.2 meters. Each node collects one observation every 30 seconds.

In this chapter, the focus lies on the node-level occupancy detection performance improvement. Once completed, a networked multiple SLEEPIR node system is constructed which utilizes the node-level occupancy and reports the accuracy for the network-level occupancy system. A level of network-level accuracy is demonstrated which ensures less than 5% chance of encountering false positives or negatives in any given week. This occupancy sensor performance standard is listed by US Department of Energy in their Saving Energy Nationwide in Structures with Occupancy Recognition (SENSOR) Program overview[79].

The effort in this chapter aims to make the following key contributions. (1) Node-level reliable occupancy detection for SLEEPIR sensors is achieved by ensuring that training observations are gathered from the same sensor node on which the inference is made thus limiting the number of occupancy scenarios in the dataset (2) Efficient

techniques like KNN based ODLL that eliminates the need for costly IoT devices are used. (3) The need for periodic over-the-cloud ML model update feature is eliminated. This feature is traditionally needed to address constantly changing occupancy scenarios.

A brief overview of the state-of-the-art is presented in the next section that outlines the present state of single node-based occupancy estimation methods for human occupancy detection. After this, a description of the proposed method is provided. The next section introduces dataset collection strategy and method performance evaluation. Then a brief discussion on addressing the issues that were encountered during the system design and experimentation phase is presented. In the results section, the impact of various parameters on system accuracy is discussed. The last section gives a brief conclusion of the proposed method.

3.1. Single Node based Occupancy Detection Methods

Although RNNs like LSTM has proven to be superior in accuracy for time-series data when compared to simpler algorithms like Decision Trees and KNN, ML algorithms like KNN are superior in terms of efficiency as these do not require training [80]. The KNN classifier is a conventional nonparametric classifier that provides effective performance for optimal values of the positive integer k. In the KNN rule, a test observation (or sample) is assigned the class most frequently represented among the k nearest training samples. If two or more such classes exist, then the test sample is assigned to the class with a minimum average distance to it. Although the KNN model is not "aware" of the temporal dynamics of the gathered observations like RNN model, yet in case when KNN is provided with a near-identical training and test dataset distribution

which are collected using a specific sensor node, the performance of KNN surpasses that of a more sophisticated RNN which is provided with a non-specific sensor dataset. This is because every sensor has a limited number of local occupancy scenarios for which it is comparatively less challenging to train an ML model. This finding is discussed in the results section of this chapter. RNN models like CTRNN and LSTM use BPTT as the training algorithm. The computational complexity of BPTT is of order $O(n^2)$, where n is the number of non-input neurons [81]. The storage complexity of BPTT is potentially unlimited and is proportional to the number of folds in the network [81]. Thus, the computational and storage resource requirements for an unoptimized BPTT algorithm dictates off-site training for RNN models as most IoT devices are incapable of performing on-site RNN training [23, 82]. Due to expensive training algorithms like BPTT, several attempts in the literature e.g. Parameter Pruning[83], Quantization[84], and gradient compression [85] have been made to reduce the training algorithm complexity to perform RNN learning.

Despite these attempts to optimize on-device RNN learning, only marginal success has been achieved in terms of reducing the RNN training complexity [86]. Most IoT devices are energy constrained. DRAM access consumes two orders of magnitude more energy than on-chip SRAM access[87]. Compared to other Deep Neural Networks, typical RNNs have orders of magnitude larger memory footprint of activations which cannot reside over on-chip SRAM for most IoT devices, thus DRAM access is needed during training. The training memory for an RNN should strictly fit on-chip SRAM to achieve on-device training. This is certainly not the case given the large occupancy datasets that usually span

from a few days to several months [25]. It can be argued that an already proposed version of transfer learning for the on-device learning [73](termed TinyTL) have been employed. Since the PIR signal used by the proposed system has unique noise and occupancy scenarios, an initial foray into the transfer learning approach in this work yielded few encouraging results. Instead of focusing on reducing RNN training complexity, a less expensive KNN algorithm is proposed but with an added ability to automatically label the dataset.

Apart from expensive resource usage, models suggested in works [88, 89], when put to the test, produced a significant number of false positives due to environmental IR noise [18, 88]. This finding is further investigated in the results section. An automated ground truth labeling technique is proposed which exploits the fact that if the traditional PIR sensor triggers intermittently and frequently, due to non-stationary human presence, there must be a stationary human presence even during the periods when the PIR is briefly in an untriggered state. This novel ground truth labeling technique ensures that the human presence IR radiation pattern unique to each sensor gets labeled correctly as true positive. It must be highlighted that in the proposed work, the local ML training dataset is only made possible due to the availability of this labeling scheme that automates the occupancy ground truth collection. This scheme will be explained further in the system design section of this chapter.

One of the attempts made in literature, to avoid collecting training data altogether, was to determine a general occupancy model via a semi-Markov model [90]. This attempt hinged on the notion that there exist unique Markov chains indicating occupancy in a

Markov model, provided that each Markov chain embeds in it, the detection episodes of variance in light, CO_2, humidity, temperature, motion, and acoustic sensor outputs. However, the success in occupancy detection for this work was limited as the work only proposes low-resolution occupancy-centered HVAC control schedules generated by the semi-Markov model. Among the works that label the node-level ground truth, one of the most comprehensive publicly available labeled datasets [91] only has 14 days of labeled data. Here, labeling was done via still images captured at 1 minute resolution. Another work [92] compares performance of KNN, SVM and ANN for occupancy prediction. Interestingly, while several statistical ML algorithms were compared, the dataset spanned no more than 3 days. In [92], the labeling was done manually via monitoring a video feed. The limited availability of labeled datasets for occupancy detection indicates a roadblock in terms of training accurate deep learning-based occupancy classifiers. Alternate established options for ground-truth collection like camera or thermopile arrays[93] based

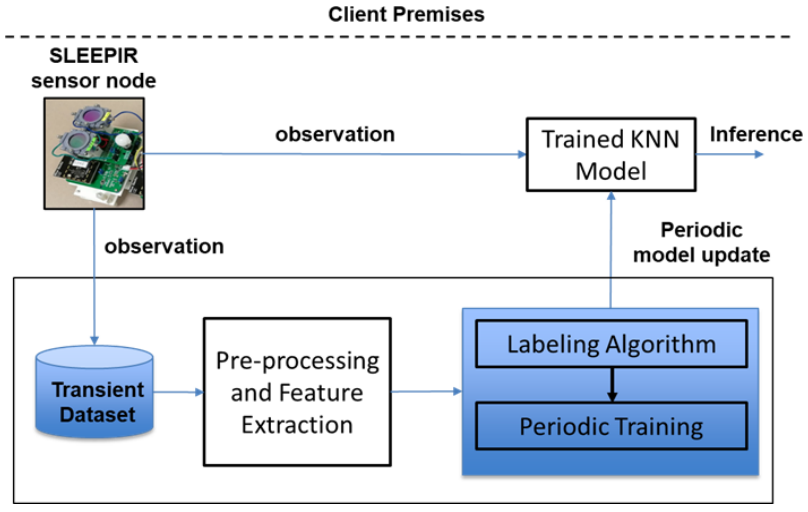

Figure 8. SLEEPIR node generates voltage, ambient temperature, and PIR signal. The voltage data is converted to binary occupancy inference via a KNN binary classifier. Selected observations from a transient dataset are then used to periodically adapt the KNN classifier through an on-site IoT device. Reprinted with permission from [2].

occupancy tracking, are not feasible because of privacy concerns, high cost and computation penalty involved. Apart from cameras, sensors such as Inertial Measurement Unit (IMU), Visible Light Sensors (VLS)[94] and Wi-Fi sensors are either too noisy or require expensive pre-requisites such and radio signal finger-printing [95] in order to be part of a scalable occupancy solution.

Thus, after a careful review of relevant single node-based occupancy detection methods, it can be concluded that a constrained dataset-based classifier such as KNN is presently the only viable alternative to expensive RNN models, for on-device training and inference. Moreover, the proposed automated labeling scheme addresses the gap of collecting ground truth locally via utilizing the onboard PIR sensor.

3.2. System Design

The overall occupancy detection system flowchart is presented in figure 8. A brief algorithm flow is presented below to summarize figure 8.

1. The raw sensor node inputs which include SLEEPIR sensor module voltage, PIR sensor binary output and ambient temperature are collected from each sensor node via BLE communication protocol. The SLEEPIR sensor node and communication platform details have already been presented in chapter 1.
2. An observation consisting of raw voltage values from the SLEEPIR sensors, ambient temperature value and traditional PIR sensor value are normalized and zero centered. A window of l observations is forwarded to the KNN binary classifier. The binary classifier then interprets the window of SLEEPIR sensor observations and outputs in binary whether the sensor has detected human

occupancy or not. This step is explained in sub-section titled "*Data Preprocessing of Sensor Signals*" of this chapter.

3. As and when novel labeled observations are received from Automated labeling algorithm, the KNN classifier training dataset is updated. The automated labeling algorithm is detailed in the section titled "*KNN based Locally Trained Occupancy Classifier*" of this chapter. The primary function of the labeling algorithm is to accept or reject incoming observations from the sensor node into a transient training dataset. This is based on pre-determined criteria.

3.2.1. Data Preprocessing of Sensor Signals

The SLEEPIR sensor node generates time-series observations consisting of SLEEPIR sensor module raw voltage outputs $V_{out}(t)$ (see chapter 1), Ambient temperature, and off-the-shelf digital PIR sensor output. To process these observations and infer whether the observed area is occupied or not, RNNs are an obvious choice, but these are expensive to train and usually require a significantly large dataset as they must be trained over a large number of days, if not weeks, for better accuracy. To realize the ideal scenario of local training and inference using an IoT device like Raspberry Pi that has constrained computational power and memory, a computationally inexpensive algorithm is needed. A potential candidate is KNN that is trained over a bounded dataset. A detailed computational comparison is presented among KNN and RNN algorithms in the next section.

In general, KNN-based classifiers assume that the set of labeled training data is already provided and contains enough training samples to describe the class distributions in the

feature space. In the proposed method, the KNN classification is effective as the extracted features form discernable clusters in feature space. In other words, observations gathered for the cases where there is occupancy, in the feature space, should not be in proximity to the observations that are gathered while there is no occupancy. This places emphasis on the feature determination process, which is discussed in the remaining subsections.

3.2.1.1. Time-series Selection and Formatting

The goal of hand-tuned ML features used widely in the literature is to produce easily distinguishable values for various data classes. A good feature remains invariant to the slight changes in the input pattern for a particular class and tends to produce roughly similar values for patterns belonging to the same class. The elements of the observation $obs(t)$ collected at time *t*, from sensor node include raw voltage signals from two ADC channels of SLEEPIR sensor i.e., $[V_{out1}(t),V_{out2}(t)]$ and a binary traditional PIR signal $PIR(t)$. Notice that Ambient temperature is not considered a part of our dataset. The reasons for not doing so have already been discussed in chapter 1. The training dataset is then initialized by normalizing and zero-centering all obs_T present in the training dataset so that it has a zero mean and a standard deviation of 1.

3.2.1.2. Sliding Window Input Approach

Following the normalization, the observation time-series obs_T which consists of the following elements $[V_{out1}(t),V_{out2}(t), PIR(t)]$ is divided into non-overlapping windows win_T. Here each element win_T is created by sliding a fixed-horizon window of length equaling 8 seconds over the 3-D obs time-series. Here subscript T is the timestamp at which the PDLC shutter permits increased IR radiation to reach the sensor. The shutter

remains in this state for 4 seconds. The PDLC shutter is termed to be in open state for these 4 seconds. The remaining time the shutter is termed to be in a closed state. Thus, the window duration corresponds to the 8 seconds where the sensor completes its response to the 4 seconds PDLC shutter modulation. Figure 9 illustrates the SLEEPIR sensor response to the PDLC shutter where IR radiation that reaches the PIR sensor changes every 4 seconds.

3.2.1.3. Feature Extraction

For each win_T, 6 ML features were computed. These features are hand-crafted and were

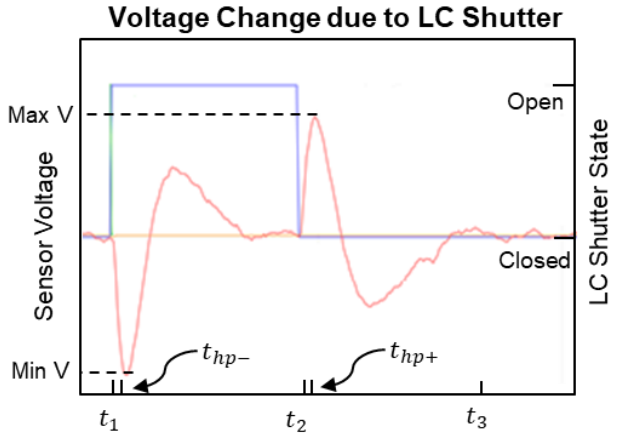

Figure 9. Illustration of the variation of the transmitted IR radiation when PDLC Shutter opens or closes. $t_2 - t_1 = 4\ sec.\ t_3 - t_1 = 8\ sec.\ t_{hp+}$ is timestamp when $V_{out} == \frac{Max\ V}{\sqrt{2}}$ while t_{hp-} is timestamp when $V_{out} == \frac{Min\ V}{\sqrt{2}}$. Reprinted with permission from [2].

tested to perform better in terms of KNN classification compared to a host of other features that were considered during the feature selection effort. Figure 9 describes the measures used in the feature evaluation, figure 10 illustrates the features in the time-series format and table 5 provides a description for each feature.

Table V. Feature Description. Reprinted with permission from [2].

SLEEPIR Features	Description
Max V_T	Maximum V_{out} computed over win_T
Min V_T	Minimum V_{out} computed over win_T
Half-Power Bandwidth for +ve peak (HPB+)	$t_{hp+} - t_2$ where t_{hp+} is timestamp when ($V_{out} == (Max\ V)/\sqrt{2}$))
Half-Power Bandwidth for -ve peak (HPB-)	$t_{hp-} - t_1$ where t_{hp-} is timestamp when ($V_{out} == (Min\ V)/\sqrt{2}$))
Windowed mean (mean V_T)	Mean V_{out} computed over win_T
Windowed Std. Dev (std V_T)	Standard Deviation for V_{out} computed over win_T

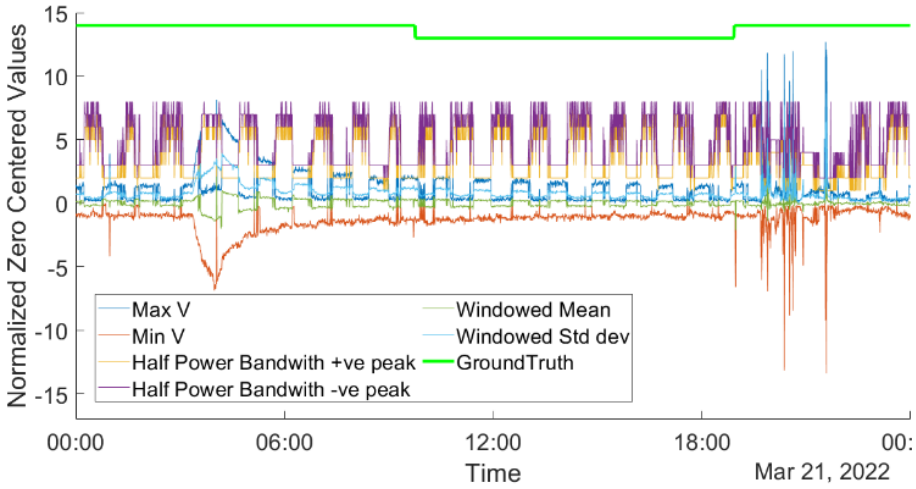

Figure 10. Statistical features are evaluated over a 60 sec observation window and are plotted over 24 hours to illustrate the portions where occupancy was observed i.e., 12am-6am and 6pm-12am. These features are used for training/indexing the proposed KNN algorithm. Reprinted with permission from [2].

3.2.2. KNN based Locally Trained Occupancy Classifier

3.2.2.1. KNN Network Architecture

KNN is a nonlinear, distance-based method, supervised classification technique. It is a direct classification method that does not require a learning process. Instead, it requires the indexed storage of the whole data. Given a training dataset (x_T, y_T), where T =[t_1, $t_1 + 30$, $t_1 + 60$, ...] and a test sample x_{test}, the distance, d_m, between x_{test} and x_T can be calculated as in equation 2:

$$d_m = \|x_{test} - x_T\| \quad (2)$$

Where $\|\cdot\|$ is the distance. One of the most widely applied distance calculations is Euclidean distance. After obtaining the distance d_m, the labels of $\boldsymbol{k}$ training samples with the smallest distance can be used. Then, a majority voting will be performed to determine the label of the testing sample. It must be highlighted here that as a new sample is assigned to a class, the computation time increases as a function of the existing samples in the dataset [96].

The proposed implementation, however, keeps the training dataset bounded via a cap on the total number of observations. This is done by periodically eliminating observations that are farthest (in terms of Euclidean distance) from the respective cluster center. An *Elbow method* search s[97] is used to determine the optimal number of neighbors $\boldsymbol{k}$, which is a crucial parameter for KNN inference. This search is performed periodically rather than at every inference. This method calculates the Within-Cluster-Sum of Squared Errors (WSS) for different values of *k* and choose the *k* for which WSS starts to diminish for the first time. In the plot of WSS-versus-k, this is visible as an elbow.

3.2.2.2. Automated Labeling Algorithm

For the initial training of the KNN classifier, the model is trained with observations which are labeled via calibration data collected by the end user. Labels y_l are then initialized where each element corresponds to each observation $x_l = [Max\ v_T, Min\ v_T, HPB+, HPB-, mean\ v_T, std\ v_T]$. The calibration labels are collected via a smartphone app where the end user labels 20 observation windows. 10 windows are labeled as “occupied” while present in the FoV of the SLEEPIR sensor node while 10 windows are labeled as “unoccupied” while the subject ensures that there is no human

presence within the FoV of the SLEEPIR sensor.

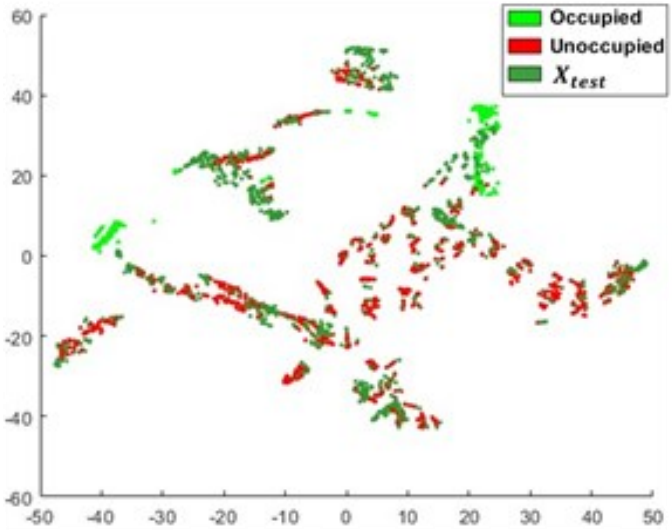

Figure 11. t-SNE feature space plot for training dataset $\{x_l, y_l\}$ for Kitchen sensor X_4. X_{test} represents labels that are yet to be labeled. Reprinted with permission from [2].

For automatic labeling, the range and FoV of traditional PIR sensor embedded within the SLEEPIR node are critical. Experiments conducted in SLEEPIR sensor related works [7, 10], discuss in detail the sensor installation height and orientation choices. As a result of experimentation in [10], it was found that for sensor installed at a height of 2.8 meters, the radius of the SLEEPIR sensor node footprint is 1.2 meters while the radius of concentric PIR sensor footprint is 2.4 meters. Each sensor generates a timestamped log of occupancy status for traditional PIR sensor as follows.

$$D_T^{iPIR} = \{(i, T): i \in N,\ T \in \mathbb{R}^+\} \tag{3}$$

In the equation 3, (i, t) denotes that PIR sensor at location X^i was triggered at time T. Figure 7 shows the set of locations denoted by index i i.e., X^1, X^2, X^3, X^4. The labeling algorithm exploits the time difference between two consecutive PIR activations for a sensor. It is assumed that a human subject is present within the sensor range and FoV if $D_{TT=}D_{T+1}^{iPIR} - D_T^{iPIR}$ is ≤ 60 seconds. It must be mentioned here that no two sensors in the dataset collection scenario overlap in terms of sensor footprint. In other words, it is assumed that the human subject did not leave the sensor footprint area if two consecutive PIR activations for the same node are ≤ 60 seconds apart in time. The assumption that there

Algorithm 1: KNN Training Set Label Generator. Reprinted with permission from [2].

Input: x_{test}, x_l, y_l, $thresh\_occ$, $thresh\_unocc$

Output: Updated Training set x_l, y_l for KNN classifier

```
k_opt = Elbow_search(x_l, y_l);
for all i in x_l where y_l == occupied
    [clust_cents_occ, loc_occ]= KMeans(x_l, k_opt)
for all i in x_l where y_l == unoccupied
    [clust_cents_unocc, loc_unocc]= KMeans(x_l, k_opt)
[dist_occ, idx_occ]=farthest_occ_sample(x_l, y_l);
[dist_unocc, idx_unocc]=farthest_unocc_sample(x_l, y_l);
for all x_test
  for all k in loc
    if dist(x_test,clust_cents_occ) < thresh_occ
      y_test = occupied;
      if dist(x_test,clust_cents_occ) < dist_occ
        if (size(x_l) > 1000)
          [x_l, y_l]= replace_in_dataset(x_test, y_test, idx_occ)
          else
          [x_l, y_l]= add_to_dataset(x_test, y_test)
    if dist(x_test,clust_cents_unocc) < thresh_unocc
      y_i = unoccupied;
      if dist(x_i,clust_cents_unocc) < dist_unocc
       if (size(x_l) > 1000)
        [x_l, y_l]=replace_in_dataset(x_test, y_test, idx_unocc)
        else
        [x_l, y_l]= add_to_dataset(x_test, y_test)
```

certainly would be a stationary occupant in the FoV of sensor if two consecutive PIR

triggers are ≤ 60 seconds, is evaluated to be generally true as the training datasets labeled based on this assumption provide us with high occupancy detection accuracy. The labeling algorithm is better explained via the algorithm steps mentioned in Algorithm 1. The input includes all x_{test} feature observations that were recorded between T and $T + 1$ whenever $D_{TT} \leq 60$ seconds.

Algorithm 1 starts by extracting the cluster centers for multiple clusters formed in feature space via a typical K-means algorithm. Each of these clusters can belong to either an occupied or unoccupied class. The data being clustered belongs to the initial training dataset $\{x_l, y_l\}$ recorded via user calibration. The job of algorithm 1 is to update the initial training dataset to a more comprehensive training dataset that includes occupancy (and unoccupancy) patterns that were not captured during the calibration time period. An unclassified observation is evaluated for proximity to the clusters in the initial training dataset. If the observation is within a threshold distance from the center of a certain class, the observation is assigned the corresponding class to which the cluster belongs. Algorithm 1 bounds the size of the dataset to a limit value of 1000. An example of an updated training dataset divided into "occupied' and "unoccupied" clusters is shown in figure 11. This figure plots distributed stochastic neighbor embedding (t-SNE) projection [98] of the observations. t-SNE gives us an intuition of how the data is arranged in a high-dimensional space.

3.2.2.3. Performance Evaluation Comparison between KNN and other RNNs

To evaluate a performance gain for using dataset bound KNN classifier when compared to an RNN, a comprehensive comparison of the KNN classifier and RNN

classifier performance is performed. The analysis includes testing the collected dataset over LSTM, CTRNN, and proposed KNN architectures. The observation window length *l* is also varied, over a reasonable range to see if certain networks performed better than others. It was found that for *l=60 sec*, the accuracy was highest across all architectures. This indicates that the most effective discriminating features exist over a period of 60 seconds. It is important to mention here that SLEEPIR collects two consecutive observations over a span of 60 seconds.

Table VI. Average Accuracy and Training Duration For 16 Neurons Hidden Layer RNN Models (LSTM, CTRNN) and for 5-Nearest Neighbor Model for a Total of 1000 Observations. Reprinted with permission from [2].

Observation Window length	LSTM		CTRNN		Proposed KNN	
	Acc (%)	training duration (sec)	Acc (%)	training duration (sec)	Acc (%)	indexing duration (sec)
30 sec	96.4	5314	92.3	4109	91.7	19
60 sec	97.1	8722	95.5	8264	94.8	23
90 sec	91.3	9945	88.0	10427	87.3	24
120 sec	82.8	16618	7.1	15730	78.7	28

Table VII. Average Power Consumption for Training 16 Neurons Hidden Layer RNN Models (LSTM, CTRNN) and for 5-Nearest Neighbor Model for a Total of 1000 Observations. Reprinted with permission from [2].

Observation Window length	LSTM	CTRNN	Proposed KNN
	avg consumption (mAh)	avg consumption (mAh)	avg consumption (mAh)
30 sec	1.03	0.96	0.0049
60 sec	2.13	1.71	0.0045
90 sec	2.25	1.98	0.0050
120 sec	3.74	3.06	0.0052

Table 6 outlines the performance evaluation results for the comparison. Although for an unbounded dataset, KNN is not as effective in terms of accuracy as LSTM or CTRNN, it does not require expensive BPTT training as is the case with its RNN counterparts. Training durations for each of the tested algorithms are also provided in table

6. The training durations were measured on a Raspberry Pi 4 using a 64-quad-core Cortex-A72 (ARM v8) processor. Table 7 lists the power consumed by the Raspberry Pi 4 platform during the duration of training for LSTM, CTRNN and KNN algorithms for a training dataset consisting of 1000 observations. Raspberry Pi consumed between 3.8 W to 5.5 W depending on the number of processing cores used during the training process. It may be highlighted here that the accuracy and power consumption values in both table 6 and table 7 are an average for all 4 nodes deployed in the system.

3.3. Results

3.3.1. Dataset

A dataset was collected that employs four SLEEPIR sensor nodes as shown in Figure 7. Certain thresholds were used to remove noisy observations as per the literature presented in [10]. A single surveillance camera was used to label the ground-truth for all rooms as entrances of all rooms and the apartment are visible in the camera FoV. Data for a total of 30 days was collected. 23 days of data were used for training and the remaining 7 days of data for testing. This provided us with a total of 5,184,000 observations for each sensor node within the dataset. A total of 3 subjects (1 adult and 2 children) were employed to gather the dataset. All the accuracy results are reported for the Kitchen Sensor (X^4) which has the highest incidence of IR noise in the collected observations i.e., frequent usage of stove and tap water.

3.3.2. Accuracy Analysis

This work has a unique claim to train the model locally and eliminate the need of periodic over the cloud ML model updates. The work also claims to minimize the

computational resource usage by the ML model while delivering comparable human occupancy accuracy when compared to a traditional RNN method. For this purpose, a previously deployed static LSTM model [88] (trained at the lab for human occupancy) was used and compared to the proposed KNN model which is dynamically updated every 24 hours.

Table VIII. Accuracy Comparison Between Proposed KNN Model and Static LSTM Model. Reprinted with permission from [2].

Date	Static LSTM Classification Accuracy (%)	Static LSTM F1 Score	Proposed KNN Classification Accuracy (%)	Proposed KNN F1 Score
15 April	76.1	0.57	89.9	0.80
16 April	83.4	0.83	94.8	0.95
17 April	69.3	0.69	97.6	0.98
18 April	62.5	0.64	98.0	0.98
19 April	56.9	0.57	97.2	0.97
20 April	74.2	0.80	87.5	0.83
21 April	82.1	0.82	92.6	0.93

It must be highlighted here that lab environment and IR noises were different in many respects from the local environment of the apartment. The results of this comparative study are listed in table 8. As mentioned earlier, this study only involves Kitchen Sensor (X^4). The remining sensors did not involve frequent IR noise like stove and warm tap water. While the data for remaining sensors helped us to reach conclusions about the average power consumption for the algorithm and the impact of data window size and ML architectures over detection accuracy analysis, the data from these sensors contained infrequent IR noise sources and would have skewed the detection accuracy to be high. The accuracy analysis presented in this work is based on a sensor placed in an environment where IR noise is encountered frequently thus representing the model performance in challenging environments.

3.3.3. Results Discussion and Future Work

Despite locally collected training set, more than 10% inaccurate classifications for the days of April 15th and 20th, 2022, were observed. The reasons were investigated and found that both for April 15th and 20th, unusually busy days in terms of cooking and dish washing were observed. The extracted features from observations for both these days rendered observations far away from occupied/unoccupied cluster centers present in the training data. The maximum ambient temperature for the Kitchen sensor for both these days was over 82F when the misclassifications occurred. It was observed that the initial calibration dataset was collected while the ambient temperature was 75F. Although, ambient temperature has a significant impact on the total IR radiation reaching the SLEEPIR sensor [88], it is not strictly correlated to occupancy. Ambient room temperature in our dataset has a constantly changing profile, caused due to weather, HVAC setpoints, difference in daytime and night-time outdoor temperatures as well as seasonal shift in sunrise and sunset angles. Figure 2 (chapter 1) shows this profile for a specific day within our dataset. Moreover, IR noise sources also tend to change ambient temperature. It is thus concluded that ambient temperature must be excluded from the input features for the proposed method as it has no obvious correlation to occupancy.

The training set update via Algorithm 1 slowly modifies the calibration data clusters and may even form new clusters. Since KNN allows more visibility into classification process, it was found that IR noise in combination with high ambient temperature caused the new feature points that were closer to the cluster center of the false class. It was thus deduced that although the local occupancy patterns observed over a

period of time, manage to modify the calibration dataset clusters in the long-term but sudden unusual shifts in the occupancy patterns and IR noise, cause false positives or negatives in the short term. A detailed look at the confusion matrices is presented in figure 12. These matrices, belonging to each test day within the dataset, provide an insight into the false positives caused by IR noise and false negatives caused primarily by IR shielding effect [8]. Certain IR noises such as warm water from a tap in Kitchen sink may produce near identical features to that of human subject. Such features confuse the classifier to produce false positives. As a future work, new features need to be formulated that can distinguish certain IR noises from human subjects while only using a privacy-aware and cost-effective sensor such as a SLEEPIR.

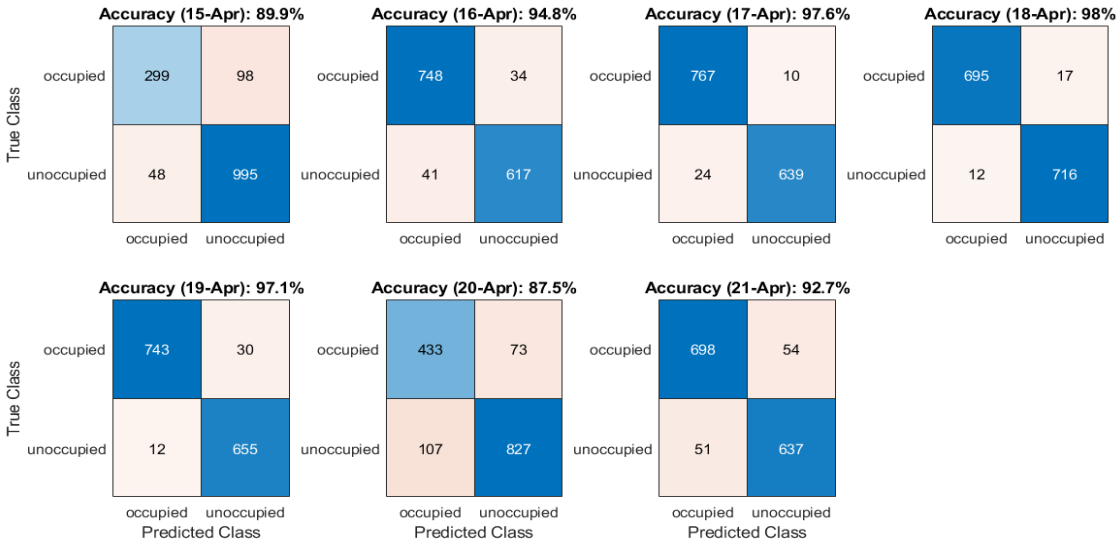

Figure 12. The confusion matrices show the performance for the proposed KNN classification method. The occupancy ground truth was collected via the proposed labeling algorithm (Algorithm 1). Reprinted with permission from [2].

3.4. Conclusion

The proposed ODLL KNN based occupancy classification method claims to provide superior accuracy and training efficiency compared to static RNN models. The

higher accuracy and significantly efficient training hinges on the fact that the KNN model is adapted to the novel observations representing new occupancy scenarios. This is only made possible as the training dataset is labeled locally with the help of the proposed automated labeling algorithm and an initial calibration dataset. The resultant occupancy classification can deal with a host of IR noises and occupancy patterns as the training observations are gathered from the same sensor node where the inference is made. A 21.9% average improvement in accuracy was achieved due to the ability to train the model locally under local occupancy scenarios. The training duration for a limited training set consisting of 1000 records was cut short by the order of magnitude when compared to traditional RNN methods. Apart from accuracy and efficiency gains, the proposed method eliminates the need for over the cloud ML model updates that are usually carried out to update the model to classify newer occupancy patterns.

4. NETWORK-LEVEL: PF + LSTM[3]

The proposed method in this chapter is essentially a proof of concept that with limited number of sensors and sparse spatial coverage, a PF can be used to track the human occupancy of an indoor space regardless of environmental infrared noises. The method exploits the temporal bounds on the change in occupancy state of the environment. It also factors-in the proximity of sensor nodes to each other, and thus PF measurement updates are structured in a way that human occupancy probability is spread spatially in expanded vicinity around the sensor rather than only inside the sensor observation cone. Human occupancy detection is an essential component of many applications like indoor security systems, lightening and HVAC automation systems, activity tracking systems [16, 99, 100], and monitoring systems for elderly people who need around the clock care [101-103]. Alternate options like camera-based occupancy tracking generally fail to deliver because of high infrastructure and computational cost, privacy concerns and failure to track high velocity motions. Apart from cameras, sensors like thermopile arrays [93], IMUs or Wi-Fi sensors are either too noisy or expensive to be part of a scalable solution.

Since PIR are relatively inexpensive and have been traditionally used in human presence monitoring systems, many have jumped on the opportunity and designed efficient and scalable localization systems based on such sensors. The systems employing such sensors

[3] Part of this chapter is reprinted, with permission, from " Indoor Occupancy Estimation using Particle Filter and SLEEPIR Sensor System." By Emad-ud-din, M., Chen, Z., Wu, L., Shen, Q., and Y. Wang (2022). IEEE SENSORS JOURNAL 22(17): 17173-17183. Copyright © 2022 IEEE

can detect heat energy emitted by the human body within a range of roughly 10 meters. However, their incapability of detecting stationary occupants limits their applications in occupancy-centered smart home appliances [104, 105]. To enable PIR sensors to detect stationary occupants, a recently developed SLEEPIR sensor [7, 10] can both detect stationary and moving occupants. However, its detection accuracy is still largely impacted by environmental infrared noises [18, 88]. To address this, a PF based human presence estimation algorithm is presented for the SLEEPIR sensor system. The system includes three sensor nodes sparsely located in an indoor space. The proposed algorithm consumes low-power and is cost-effective and thus provides scalable service which makes the proposed SLEEPIR sensor system eligible for widespread future adoption.

The proposed particle filtering approach involves a PF which relies on three assumptions (1) Any human subjects entering the observed area will trigger at least one SLEEPIR sensor node. (2) The SLEEPIR sensors have a limited Field-of-view (FOV) and thus these only have sparse coverage of the area monitored for human occupancy (3) Collected ground truth data represents the expected traffic conditions within the entire monitored area.

The dataset for this method spans over 15 days. Although the experiment testbed was artificially created in a lab for dataset collection purposes, the experiments were uncontrolled. Three SLEEPIR sensor nodes are installed in two different configurations at the testbed to collect dataset as shown in Figure 13. Primary aim of this study is to not optimally cover the testbed space using SLEEPIR sensors but to determine the minimal number of sensors to achieve a level of accuracy that ensures less than 5% chance of

encountering false positives or negatives in any given week. This occupancy sensor performance standard is listed by US Department of Energy in their SENSOR Program overview[79].

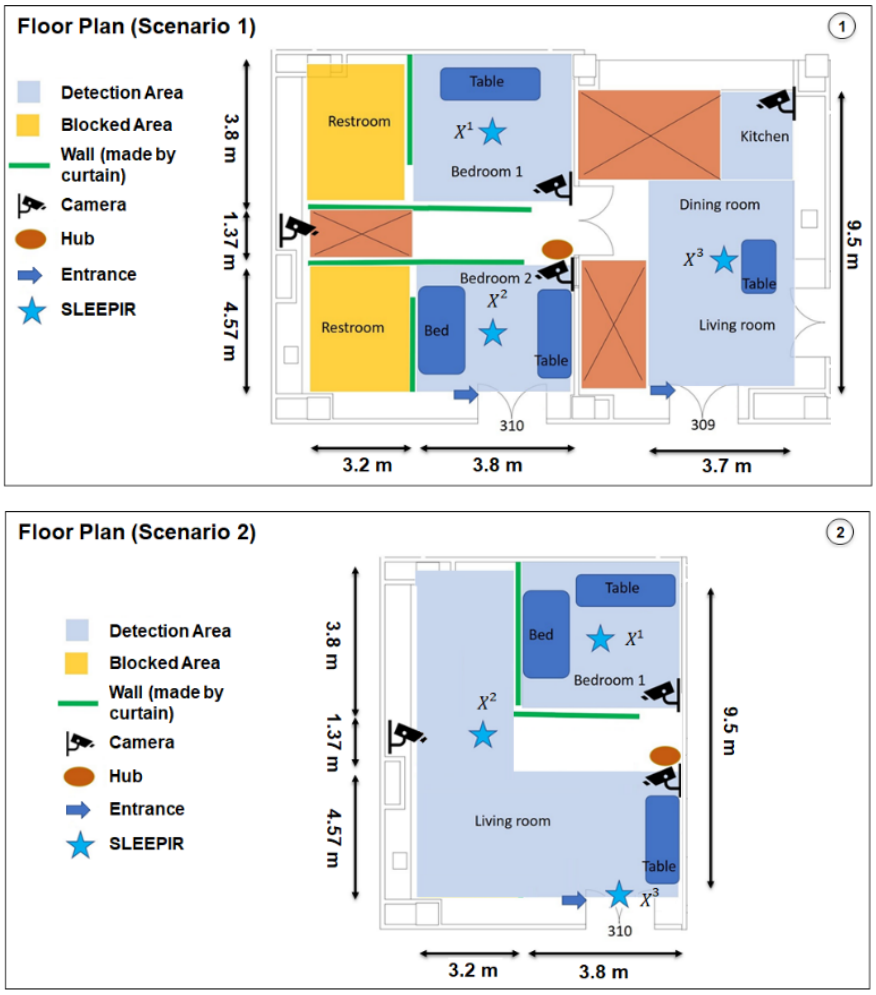

Figure 13. Two different floorplans as testbeds are used. 3 SLEEPIR sensor nodes for each floorplan are employed for dataset collection. Human occupancy is estimated for the locations X1 through X3. Ground truth is collected via surveillance cams. Reprinted with permission from [3].

In the presented work, the use of a PF for the purpose of estimating the human occupancy in an indoor environment is investigated while utilizing a minimal number of low-cost SLEEPIR sensor nodes. This effort aims to make the following key contributions. (1) A particle filter-based occupancy detection method is realized that can achieve superior

or equivalent accuracy when compared to statistical machine learning models. (2) Robust occupancy detection is achieved while maintaining a limited sensor footprint in the monitored area. (3) The solution is scalable to variations in the room size, geometry, and overall monitored area size.

Later in this chapter, in section titled "System Input and Pre-Processing Algorithms", we give a detailed description of the method along with a brief overview of the SLEEPIR sensor system that our method uses for occupancy estimation. The next section titled "PF design" presents a brief discussion on the method and the expected impact of various parameters on system accuracy. Then the comments on the limitations and capabilities of the method are presented in section titled "Discussion". The section titled "Results" introduces dataset collection strategy and method performance evaluation. Lastly a brief conclusion of the work is presented.

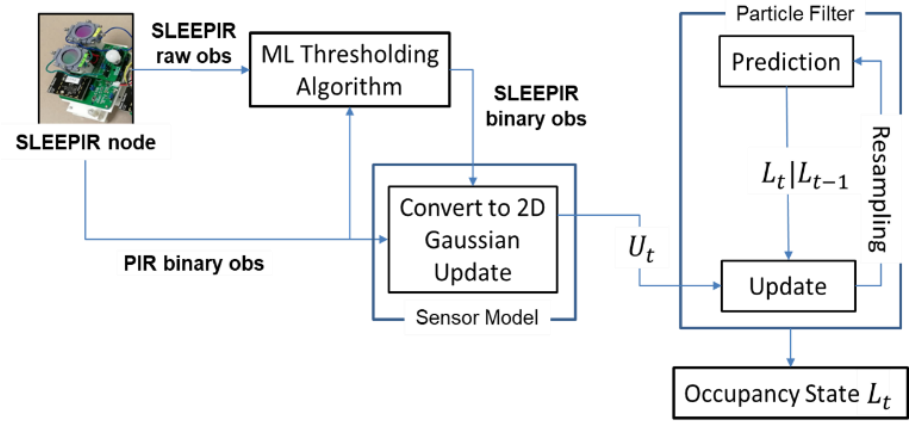

Figure 14. PF based human occupancy detection method flow chart. Networked sensor nodes generate voltage, ambient temperature, and PIR data. The voltage is converted to binary occupancy observations via a thresholding algorithm. The node-level occupancy observations then update a system-level occupancy estimate via a PF. . Reprinted with permission from [3].

4.1. System Input and Pre-Processing Algorithms

The overall system flowchart is presented in figure 2. The raw SLEEPIR sensor observations are extracted from the SLEEPIR sensor using a Bluetooth communication

protocol. The sensor and communication platform details have already been presented in chapter 1. A brief overall algorithm flow is presented below summarizing the flowchart presented in figure 14.

1. The raw sensor inputs which include SLEEPIR sensor voltage, PIR sensor binary output and ambient temperature are collected from each sensor node via a Bluetooth communication protocol.
2. Raw voltage values from SLEEPIR sensor are pre-processed using a machine learning based thresholding algorithm. This algorithm is detailed in sub-section titled "Machine Learning based Thresholding Algorithm". The thresholding algorithm interprets the raw SLEEPIR sensor observations and outputs in binary whether the sensor has detected human occupancy or not. The traditional PIR sensor output is already binary, so it does not require preprocessing.
3. Since the binarized observations need to update a PF, a sensor likelihood model is designed to shape an update for the PF. This sensor likelihood model merges the output from both the SLEEPIR sensor and the traditional PIR sensor into a single PF update.
4. The PF receives these periodic updates from the likelihood function and estimates the probability of human occupancy at each of the locations represented in the PF state. The world constraints are embedded in the likelihood and PF design thus the false positive or false negative observations get filtered and a robust human

occupancy belief is estimated by the PF. Likelihood model and PF are described in detail in section titled “PF Design”.

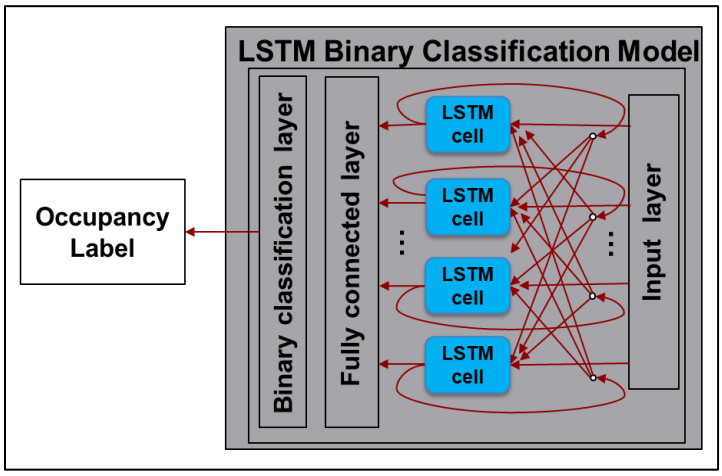

Figure 15. LSTM network architecture for SLEEPIR raw observation binary classifier. Reprinted with permission from [3].

4.1.1. Machine Learning based Thresholding Algorithm

Since sensor node generates time-series observations consisting of SLEEPIR raw voltage output $V_{out}(t)$ (see chapter 1 for details), Ambient temperature $T_{amb}(t)$ and off-the-shelf PIR sensor output $PIR(t)$, we employ RNNs to classify whether the observation indicate human occupancy or not. RNNs, in comparison to the typical FFNNs, have been shown to achieve the highest accuracy with time-series data [106], as they can process and encode the sequential temporal information contained in a time-series data. First, the incoming time-series data from the sensor node is zero-centered and normalized. Then the input quantization step is performed. Reason for choosing to quantize input data is given in upcoming sub-section. Then the input time-series is divided into pre-determined sized observation windows. Each window is then labeled as either occupied or unoccupied based on the available ground-truth gathered via web-camera installed testbed. Lastly, a

Long Short-Term Memory (LSTM) Network is trained with the dataset. The trained network (shown in figure 15) is deployed so that the network can distinguish between the observations indicating occupancy versus those indicating non-occupancy. The thresholding algorithm is detailed in the following sub-sections.

4.1.1.1. Input Formatting and Quantization

The goal of hand-tuned machine learning features used widely in the literature, is to produce easily distinguishable values for various data classes. A good feature remains invariant to the slight changes in the input pattern for a particular class and tends to produce roughly similar values for patterns belonging to the same class. The same effect is achieved by quantizing the input signal so that input signals that bear slight differences with each other, are quantized into similar looking patterns. Input quantization has a proven positive impact on RNN accuracy [107]. So, a quantization strategy is chosen that quantizes the sensor data to three levels (rise/fall/no change). It may be noted that the quantization used here is applied to both the training and test data streams. Equation 4 outlines the quantization function for the incoming observation at time t.

$$\begin{aligned} if\ obs_t > obs_{t+1} + \epsilon \quad & 1(rise) \\ if\ obs_t < obs_{t+1} - \epsilon \quad & -1(fall) \\ if\ abs(obs_{t+1} - obs_t) \leq \epsilon \quad & 0(no\ change) \end{aligned} \tag{4}$$

Literature suggests that a reasonable value for ϵ can be $(\mu_a + \sigma_a/2)$ [108], where μ_a and σ_a are the mean and variance for the input distribution. Thus, the value of ϵ depends upon the distribution of observation elements $[V_{pp1}(t), V_{pp2}(t), T_{amb}(t), PIR(t)]$.

4.1.1.2. Sliding Window Input Approach

The training dataset obs_T is initialized where each element is created by sliding a fixed-horizon window of length $\boldsymbol{l}$ over the 4-D training input time-series consisting of following elements $[V_{pp1}(t),V_{pp2}(t),T_{amb}(t),PIR(t)]$. The labels $label_T$ are then initialized where each element corresponds to each window in obs_T. An element is set to "*occupied*" if a surveillance camera-based $ground\text{-}truth_T$ indicates that human subject was present for more than 50% of observations in the FOV of the sensor. Otherwise, the element is set to "*unoccupied*". A suitable window length (l) is found to be a critical parameter that has a pronounced impact on the over network accuracy. This impact will be later highlighted in the sub-section titled "Performance Evaluation of LSTM and other RNNs" in the chapter.

4.1.1.3. RNN Architecture

A well-cited deep forward RNN model proposed in [109], which contains multiple layers of recurrent units that are connected "forward" in time, is used as a reference model for the proposed RNN architecture. This model architecture is simple yet powerful enough to produce reliable results over publicly available datasets which consist of time-series data. The online LSTM model shown in figure 15 contains a single hidden layer of 16 recurrent neurons. During the evaluation phase, all RNN models use 3, 6, 9 and 16 neurons depending upon the experiment configuration. There are also 4 input neurons to match the number of input time-series from the sensor node i.e., $[V_{pp1}(t),V_{pp2}(t),T_{amb}(t),PIR(t)]$. There are two output neurons to match the output classes corresponding to "occupied" and "unoccupied" status.

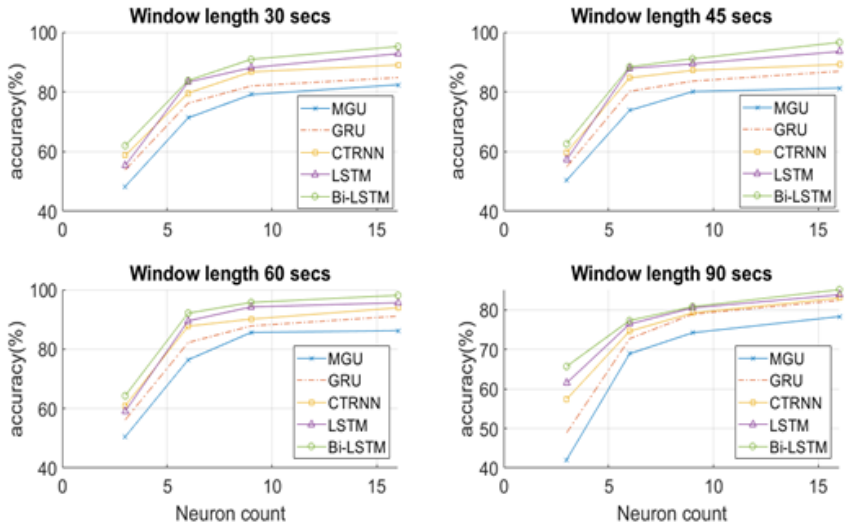

Figure 16. Accuracy comparison between different RNNs with varying network size and observation window length *l*. Reprinted with permission from [3].

4.1.1.4. Performance Evaluation of LSTM and other RNNs

A comprehensive search for suitable RNNs was performed. The analysis included testing the collected dataset over LSTM, Bi-directional LSTM, CTRNN, MGU [110] and GRU networks. The observation window length *l* was also varied over a reasonable range to see if certain networks perform better than others. It was found that for *l=60 sec*, the accuracy was highest across all architectures. This indicates that the most effective discriminating features exist over a period of 60 seconds. It is important to mention here that SLEEPIR collects two consecutive observations over a span of 60 seconds. Moreover, the number of nodes was also varied for each network to see the impact of network size over accuracy. The network size was varied to improve classifier efficiency as the classifier is expected to perform in an online pipeline (refer to figure 14 to see pipeline). Table 9 and figure 16 outlines the performance evaluation results for RNN classifiers. It was observed that LSTM and Bi-LSTM outperformed other RNNs in nearly all configurations. LSTM was selected as our network of choice as it is relatively less expensive in terms of resources. A

relatively high ratio of false negatives was encountered compared to false positives. Reasons for this are discussed in the section titled "PF Design" of this chapter.

Table IX. Impact of Observation Window Size and Network Architecture on Accuracy. Reprinted with permission from [3].

Observation window length	Average accuracy for 16 neurons hidden layer model. (50% labeling threshold) Quantized Observations				
	Bi-LSTM	LSTM	CTRNN	GRU	MGU
30 sec	95.2%	92.8%	89.0%	84.8%	82.4%
45 sec	96.6%	93.5%	89.2%	86.9%	81.3%
60 sec	98.1%	95.6%	93.9%	91.1%	86.2%
90 sec	85.1%	83.8%	83.0%	82.4%	78.3%

4.2. PF Design

After the system observations are binarized via the proposed ML architecture, these observations are used to update a PF. The elements of this filter that include likelihood model, filter state, update and sampling modules, are detailed in the remaining of this section.

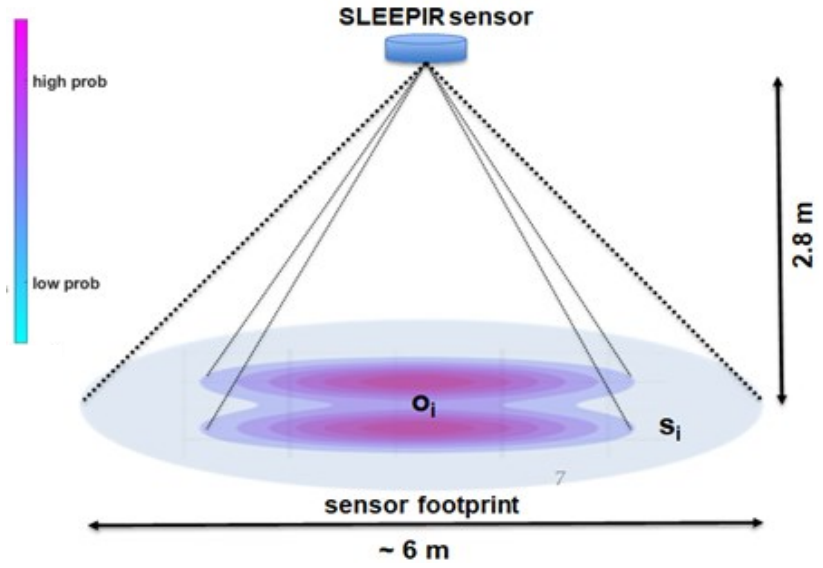

Figure 17. The SLEEPIR and PIR sensor multimodal detection probability distribution is shown. The distribution works as a measurement update for the SLEEPIR and PIR sensors. o_i is the sensor footprint (~$4m^2$) for the SLEEPIR sensor while s_i is the sensor footprint (~$18m^2$) for the PIR sensor. Reprinted with permission from [3].

4.2.1. Sensor Likelihood Model

A likelihood model based on the sensor coverage parameters and inter-sensor correlation measure, is proposed in this section. An observation from a sensor node that includes a SLEEPIR and PIR sensor, has a distribution of detection probabilities associated to the area that the sensor observes. This distribution of detection probability is based on SLEEPIR sensor range and FoV experiments conducted in the earlier works [7, 10]. These works also discuss in detail the sensor installation height and orientation choices. Both FoV and range of the sensor are listed in chapter 1. Moreover, as a result of experimentation in [10], more specifically, it was found that for sensor installed at a height of 2.8 meters, the radius of SLEEPIR footprint is 1.2 meters while the radius of concentric PIR sensor footprint is 2.4 meters. These footprints are visually represented in figure 3 (chapter 1), where a sensor cone is also shown. Each sensor generates a timestamped log of occupancy status observations as follows.

$$\begin{aligned} D_t^{iSLEEPIR} &= \{(i,t): i \in N,\ t \in \mathbb{R}^+\} \\ D_t^{iPIR} &= \{(i,t): i \in N,\ t \in \mathbb{R}^+\} \end{aligned} \tag{5}$$

In the equation 5 (i,t) denotes that sensor i triggers at time *t*. Figure 1 shows the set of locations denoted by index *i* i.e., X^1, X^2, X^3.Whenever an occupancy D_t^i observation indicates a human detection, the variance is adjusted for bivariate Gaussian update in the following way.

$$\begin{aligned} \sigma_t^{vSLEEPIR} &= \frac{1}{\rho_{iv}} \times \sigma_t^{vSLEEPIR} \\ \sigma_t^{vPIR} &= \frac{1}{\gamma_{iv}} \times \sigma_t^{vPIR} \end{aligned} \tag{6}$$

Here matrices γ and ρ represent the Pearson-correlation coefficient between smoothed sensor observations D_t^{iPIR} and $D_t^{iSLEEPIR}$ respectively. For example, ρ_{ii}, would represent the sensor observation correlation with itself which will always be 1. In case sensor *i* is not correlated with sensor *v,* ρ_{iv} will be near the value of 0. In other words, the inter-sensor correlation matrices ensure that if sensor *i* triggers and it happens to have its observations correlated to sensor *v*, the sensor model will indicate the sensor *v* as a triggering sensor as well albeit with a reduced amount of certainty. This amount depends upon the level of correlation present between two sensors. It must be noted that we smooth the observations by a certain time-window τ Thus, the correlation represented in γ and ρ is a correlation over a time-window τ.

The mean μ^i for update distribution for sensor *i*, is set to the 2D sensor coordinates in the map while unadjusted variance σ^i is set according to the following evaluation functions.

$$\sigma_t^{iSLEEPIR} \propto \left(\frac{o_i}{s_i+l_i}\right), \quad \sigma_t^{iPIR} \propto \left(\frac{o_i+s_i}{l_i}\right) \tag{7}$$

$bivariateGaussianGen$ is defined as a function so we can generate a single mode bivariate gaussian distribution for PIR sensor. $mmBivariateGaussianGen$ function is then defined to generate the bimodal Gaussian distribution for SLEEPIR sensor. Both distributions are shown in figure 17 and function definitions are listed in equation 8.

$$\begin{aligned} \pi^{iPIR} &= bivariateGaussianGen\left(\mu^i, \sigma^{iPIR}\right) \\ \pi^{iSLEEPIR1} &= mmBivariateGaussianGen\left(\mu^i, \sigma^{iSLEEPIR}\right) \end{aligned} \tag{8}$$

The variance of these gaussian distributions depends upon the size of areas falling under each of their observation cones. Thus, $\sigma_t^{iSLEEPIR}$ and σ_t^{iPIR} represent the uncertainty

for measurement update distributions π^{iPIR} and $\pi^{iSLEEPIR}$. $\sigma_t^{iSLEEPIR}$ and σ_t^{iPIR} are directly proportional to the ratio of area observed by the sensors to the total area of the room the sensor is installed in.

4.2.2. Particle Filter State

The goal of the filter is to estimate the occupancy of the observed area with a level of certainty. This work represents the occupancy belief over the expanse of observed space via a multimodal Gaussian bivariate distribution represented by L_t. Variable L_t is defined as

$$L_t = \{\pi_t^{uv}\} \tag{9}$$

$$where, \quad u = -lim_x + r, -lim_x + 2r, \dots, lim_x$$

$$and \quad v = -lim_y + r, -lim_y + 2r, \dots, lim_y$$

Here $u\ and\ v$ are indexes that run through the range of weights π which is a bivariate probability density function (pdf) that represents the bivariate probability distribution indicating the occupancy probability across the 2D spatial expanse of area under monitoring. The area dimensions are $2lim_x \times 2lim_y$. Here changing the value of r (or resolution) changes the size of domain of the distribution function. It was found empirically that an optimal value for r is 0.5 meters. The value of r can impact the accuracy and execution efficiency of the PF.

Each particle in the filter contains a varying multimodal bivariate Gaussian distribution for the area under observation. Q particles are initialized under certain initial conditions.

$$P_t^j = initialize\left(L_t^j\right)\ where\ j = 1, \dots, Q \tag{10}$$

Each particle is initialized by adding normal random noise to an initial hypothetical update U_0 where no sensor is triggered. The update U is the based on augmented sensor signals received from all sensors in the system i.e., $X^1, X^2 \; or \; X^3$. We use index j as a particle index invariably for the remaining article.

4.2.3. Prediction Step

It is well known that the PF involves repetitive prediction and update steps. In the prediction step, firstly a uniformly distributed random number $h \in [0,1]$ is generated and then it is used to select sample P_{t-1}^j from all samples at time $t-1$ according to their weights w_t^i. Then the prediction step is performed for each P_t^j as follows

$$L_{t+1}^j = L_t^j + n_{spread} \tag{11}$$

Here $n_s$is a multimodal normally distributed random noise that has modes centered at sensor locations μ^i. Essentially n_s factors in the variation present in the rate at which human subjects move their positions from sensor to sensor within the monitored area.

4.2.4. Update Step

For each particle P_t^j, following steps update the particle variable L_t^j at each time-step via a Gaussian update U_t via the following expression.

$$L_{t+1}^j = L_t^j + \delta_{t+1}(U_{t+1}^i - L_t^j) \tag{12}$$

Here,

$$U_t = \{\pi_t^{uv}\}$$

Variance σ_t^i for U_t is already defined in the earlier section. *u* and *v* have already been defined while defining the PF state. Moreover,

$$\delta_{t+1} = \frac{\sigma_t^2}{\sigma_t^2 + \sigma_{t+1}^2} \tag{13}$$

Here if U_{t+1}^i has high variance relative to U_t^i then δ_{t+1} is small thus it has little impact on value of L_{t+1}^j. This ensures that updates which have more chance of error are factored-in less into our current belief L_{t+1}^j.

4.2.5. Sampling Step

In the sampling step, weight for each particle is set and then perform a weighted sampling to select particles for prediction step. Weights for particles are set higher that have smaller Bhattacharya distance [111] (measures the similarity between two distributions). Since each particle, P_t^j is comprised of bivariate distribution L_t^j, the bivariate Bhattacharya distance between the update U_t and each particle L_t^j, is evaluated. The following step presents the probability of a particle to be sampled via the Bhattacharya distance-based weights.

$$Prob^j\left(w_t^j \middle| U_t\right) = \frac{1}{\sqrt{2\pi}\delta_t} = e^{-\frac{(Bhattayacharya\left(P_t^j - U_t\right))^2}{2(\delta_t)^2}} \tag{13}$$

4.3. Discussion

Sensor model-based updates are critical to the performance of the proposed filter. These updates, if modeled correctly, can optimally select, and help propagate particles that are very close to the real-world occupancy scenario. It is thus useful to visualize and have a critical look at one example of a sensor model-based update. A bivariate Gaussian update in Figure 18 can be seen. Since the sensors at locations X^1 and X^2 have correlated observations, the update factors-in this correlation and hypothesizes a more realistic

update. It may be highlighted here that correlation evaluation can be erroneous in case there is infra-red (IR) noise present in the observed space. This noise can include electronic devices and objects that efficiently absorb the heat radiated by the human body. It may also be observed that particle sensor model heavily relies on certain parameters that are rooted in real world environmental factors and sensor limitations. A list of these tunable parameters is provided in table 10. Any changes in these parameters can have a pronounced impact on PF accuracy.

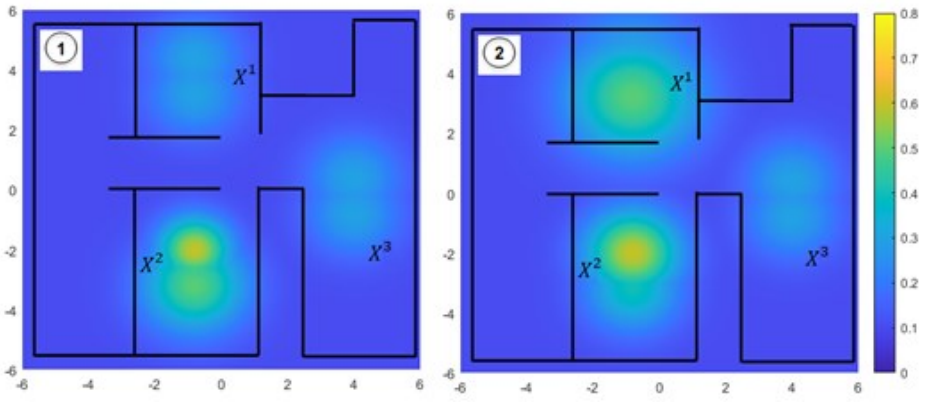

Figure 18. Inset label 1 shows the bivariate multimodal distribution for the update produced by the sensor model when the subject is near sensor X^2. Inset label 2 shows the gaussian profile for the update when the subject leaves the vicinity of sensor X^2 and approaches sensor X^1. Both x and y axis are labeled in meters. Reprinted with permission from [3].

Table X. Brief Description of Method Parameters. Reprinted with permission from [3].

Parameter	Explanation
$Vppr, Vppf$ thresholds	Both thresholds help us deal with the SLEEPIR sensor noise as detailed in [112]
ρ_{uv}, γ_{uv}	Time-window based correlation coefficient for human detection between observation time-series collected at two locations. These helps us configure the sensor model.
β^i	This threshold is applied to the value of bivariate gaussian curve at its mean in L_t^j. If this value is beyond this threshold, human presence is established.
η_t, η_{spread}	These are gaussian noise parameters for the filter. η_t is the noise present in the timestamps of the updates. η_{spread} is the noise present in the velocity of human subjects.

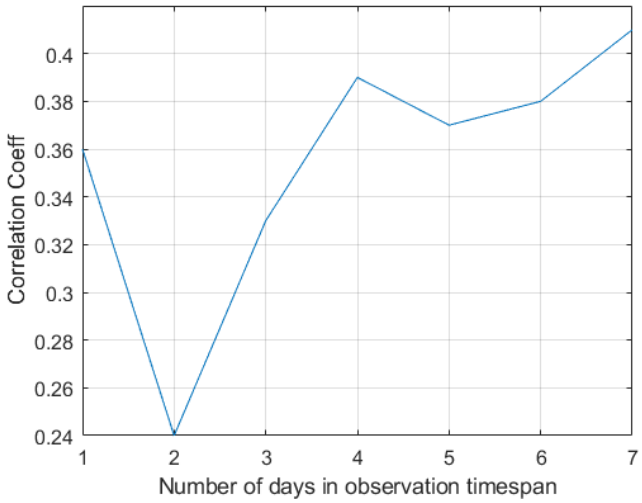

Figure 19. Peterson's correlation coefficient for the sensor pair X^1 and X^2 in scenario 2, is shown to increase as the timespan for observations is increased from 1 day to 7 days. Reprinted with permission from [3].

4.4. Results

4.4.1. Dataset

The used dataset employs three SLEEPIR sensor nodes. The nodes are deployed in two configurations as shown in figure 13. Each node collects the observation every 30 seconds. The SLEEPIR observations $D_t^{iSLEEPIR}$ were evaluated using the raw SLEEPIR sensor voltage values. Certain thresholds were used to remove noisy observations as per the literature presented in [10]. Surveillance cameras were used to label the ground truth Data for a total of 15 days was collected. 7 days were used to extract correlation matrices γ and ρ for sensors. It may be noted here that distinct correlation matrices were extracted for each of the floorplan scenarios presented in Figure 13. The observed correlation between sensor pairs within a scenario plays a crucial role in the performance of particle filter-based occupancy detection. It was observed that Peterson's correlation coefficient evaluated over a longer period was higher and thus produced better results. Figure 19

illustrates the relationship between the observation timespan and the evaluated correlation for a particular sensor pair. After using 7 days of data to extract correlation matrices, we used the remaining 8 days of test data for evaluation. The observations were down sampled to 1 observation per minute. This resulted in a total of 21600 observations within the dataset. A total of 4 subjects (2 males and 2 females) were employed to gather the dataset.

4.4.2. Accuracy Results

This work claims to minimize the sensor footprint and deliver comparable human presence accuracy when compared to statistical ML methods. The impact of the number of sensors on occupancy detection accuracy was also studied. It was found that 3 sensor nodes suffice the 95% true positive accuracy criteria mentioned in the US Department of Energy SENSOR program outline[79]. The results of this comparative experiment are listed in Table 11.

Table XI. Impact of Sensor Nodes Reduction in the Network. Reprinted with permission from [3].

Date	Scenario	1-Sensor Accuracy (X^3) (%)	2-Sensor Accuracy (X^2, X^3) (%)	3-Sensor Accuracy (X^1, X^2, X^3) (%)
25-Feb	1	27.73	76.91	98.83
26-Feb	1	24.04	75.11	94.21
1-Mar	1	19.60	77.87	95.70
2-Mar	1	22.64	67.25	93.37
6-Mar	1	20.01	79.88	95.69
20-Oct	2	46.49	88.53	96.55
3-Nov	2	53.91	81.03	97.34
4-Nov	2	58.27	85.35	98.10

The proposed system-level PF algorithm was used for the experiment results shown in Table 11. As a consequence of this experiment, observations from 3 nodes (3 PIR and 3 SLEEPIR sensors) were employed to compare the accuracy between the sensor-

level algorithm (Statistical ML) [10] and the system-level algorithms (EKF[19], PF and proposed PF with ML algorithms). Sensor-level machine learning-based occupancy detection algorithm results presented in [10] are used as a baseline for this analysis. The EKF-based networked sensor estimation algorithm given in [45] was applied to our dataset and inferior accuracy results were achieved compared to our proposed PF algorithm. Not only, an accuracy comparison is done between EKF and PF approaches, but the performance penalty of choosing to prefer PF over EKF to achieve higher accuracy is also highlighted. Table 12 shows the average processing time for a single observation of a window length of 60 secs. The execution times were measured on a Raspberry Pi 4 using a 64-quad-core Cortex-A72 (ARM v8) processor. Machine learning layers were excluded from the performance comparison as the computation cost for the ML inference is negligible. It is shown in Figure 20 that the proposed Particle Filtering approach can provide superior accuracy when compared to the accuracy achieved via an ML-based statistical thresholding approach [10] and a system-level EKF-based occupancy estimation algorithm [113].

Table XII. Accuracy Comparison between Baseline and Proposed Models. Reprinted with permission from [3].

Date	Scenario	EKF		Statistical ML (%)	PF only (%)		PF with ML (%)
		acc (%)	avg exec time (ms)		acc (%)	avg exec time (ms)	
25-Feb	1	94.61	88	90.82	98.35	461	98.83
26-Feb	1	90.24	86	89.71	91.51	738	94.21
1-Mar	1	86.35	87	81.91	95.47	693	95.70
2-Mar	1	90.02	89	87.96	92.83	727	93.37
6-Mar	1	86.15	88	80.73	94.22	824	95.69
20-Oct	2	84.33	86	76.27	92.36	720	96.55
3-Nov	2	89.70	87	88.25	91.72	602	97.34
4-Nov	2	91.86	87	89.46	94.51	415	98.10

When compared to [10], the proposed work was also able to reduce the number of sensors that are required to determine occupancy in the indoor space. The last two columns of Table 12 show human occupancy accuracy via two input pre-processing approaches (a) The system-level human occupancy is established via a PF that uses a fixed threshold to convert incoming sensor voltage into binary inputs. This is termed the "PF only" approach. (b) The system-level human occupancy is established via the proposed machine learning classifier that is described in the last section. This is termed as "PF with ML thresholding" approach. For both these approaches, the system level-human occupancy is established if the PF output probability density function results in at least a single occupancy peak. Examples of such peaks are shown in Figure 18 in elevated temperature color shades.

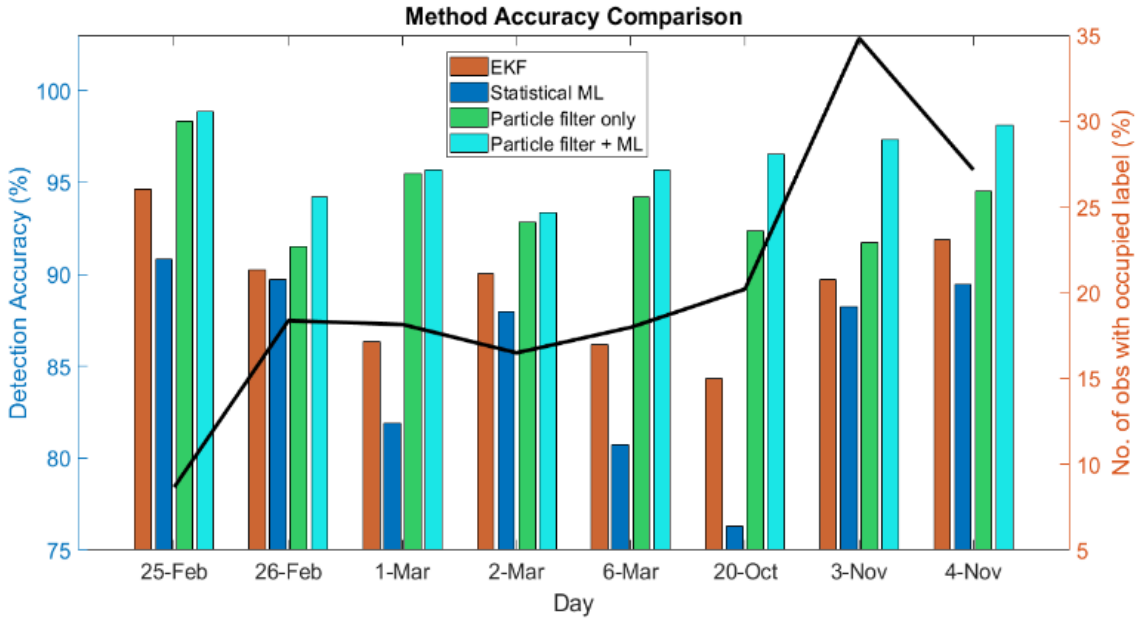

Figure 20. Accuracy comparison with baseline Statistical Machine Learning Model. Accuracy improvement due to PF (green) and added improvement via adding ML in the pipeline (light blue) is shown. The line graph shows the percentage of occupied observations in the test-dataset. Reprinted with permission from [3].

Figure 20, via the line graph, also shows the percentage of observations in the test dataset where at least one occupant was present in the observed area. The presence of the occupant is established via ground truth data. This percentage is important as the system does not encounter significant detection errors whenever human subjects are not around. We also tested the proposed method using two test-bed scenarios (see Figure 13). The first scenario tests the method performance in space that is very restrictive for the sensor's range and the correlation between the sensors is low as each sensor is housed in an independent room. The second scenario has more open space available to the sensor nodes. Moreover, two sensor nodes, although far from each other, are housed within the same enclosure i.e., a large living room.

4.5. Results Discussion

Despite the robust performance at the system level, delivered by the PF, false negatives remain a problem at the sensor level. The IR noise present in the environment is mostly due to the heat transferred to the objects with which the human body gets in contact. These keep emitting IR radiation even after the human subject leaves the observed area. Moreover, the IR radiation emitted by certain bodies that have temperature and emissivity values similar to the human body acts as noise whenever a classifier is trained for occupancy decisions. Figure 21 shows examples of IR noise within the testbed that was discovered during the experimentation. False positives and false negatives quantities for each day of the test dataset are shown in Figure 22.

It must be mentioned here that false positives generated due to objects that acquire human body temperature are transient as these objects cool down within minutes. This

Figure 21. (Left) The mattress still emitting IR after 60 seconds have passed since the subject left the bed. (Right) Top view of a chair, a laptop, and a charger. Chair seat is still radiating IR after the human subject has left. All IR sources mentioned here emit IR noise for a machine learning based classifier. Reprinted with permission from [3].

research while conducted using a system-level occupancy detection algorithm, points towards the need for a node-level ML training algorithm that can discriminate between the human body that maintains a near-constant temperature and objects that are in the process of cooling down. It can be observed that the proposed method consistently produces more false negatives compared to false positives. This is obviously due to the IR noise present in the environment. It can also be seen that certain false positives are also produced. There are two primary contributors to false negatives. Firstly, the IR radiated by certain subjects is simply not enough due to the small body size and clothing. Secondly, certain PF parameters may not be tuned well to suit the test-bed dynamics e.g., Gaussian noise parameters η_t, η_{spread} may need to be hand-tuned to be sensitive to the speed with which human subject approaches and leaves the sensor vicinity. One of the possible solutions is adaptive PF parameter tuning but further analysis and investigation are necessary to propose a parameter tuning strategy. The accuracy results are highly reliant on the correlation measure between the observations from any two sensor nodes used for experimentation.

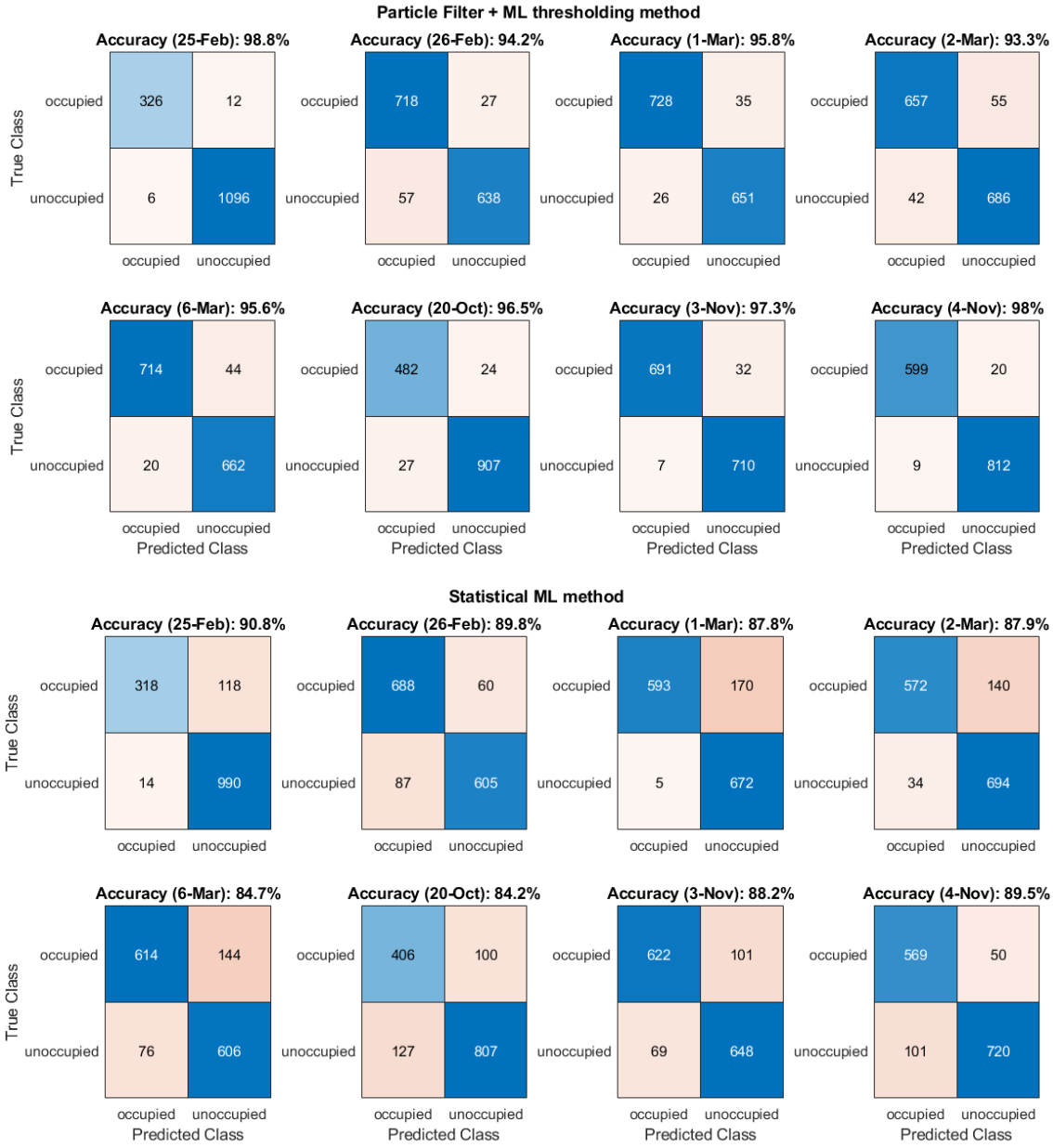

Figure 22. The confusion matrices showing the performance comparison between the state-of-the-art Statistical ML occupancy detection method and the proposed method. The occupancy ground truth was collected via surveillance cameras. Reprinted with permission from [3].

4.6. Conclusion

The proposed method delivers robust results in terms of human occupancy detection while using a small number of low-powered SLEEPIR and PIR sensors. The model exploits the inter-location observation correlation between sensors to generate

close-to real-world measurement updates. Moreover, it exploits the temporal bounds on the position change rate of human subjects within the environment. The novel bivariate update distribution generated by the sensor model ensures that realistic and not random hypotheses (particles) are generated for the PF to sample from. Not only is this method comparable to the contemporary statistical ML method [10] but it also attempts to reduce the number of required sensors to deliver the same accuracy for human presence detection. The method is evaluated over two different testbeds with different sensor configurations. The consistent accuracy reveals the scalability of the proposed method. Moreover, the method's dependence on ground truth to extract observation correlation between different sensors is dependent on the accuracy of ground-truth annotation. As a future effort, alternative methods can be explored to avoid the dependence on historical information like observation correlation between sensors.

5. NETWORK-LEVEL: BF + LSTM[4]

To address the issue that standard PIR sensors can only detect non-stationary occupants, a networked SLEEPIR sensor node [7, 10] is utilized that can detect both stationary and moving occupants by adding an electronic PDLC infrared shutter to a standard PIR sensor [18, 88]. In this chapter, a BF-based algorithm is proposed that uses a network of SLEEPIR sensor nodes to improve the otherwise less-than-perfect occupancy detection capability of individual nodes [88].

The BF-based algorithm is relatively more robust to environmental infra-red disturbances when compared to our previously proposed PF-based algorithm in the last chapter. Moreover, unlike the PF-based algorithm, it also avoids using historical sensor data to be able to filter out node-level noisy observations. The proposed method also allows the use of a minimal number of adjacent sensor nodes to detect the occupancy of an entire covered space of interest. The presented approach utilizes an MDP formulation [24] to model the indoor occupancy states and occupancy transition probabilities between states. FMA is performed on an underlying MDP, to evaluate transition probability and expected time to travel between two occupancy states. These two parameters play a crucial role in filtering out environmental infrared disturbances.

The BF-based algorithm has the following key advantages : (i) superior detection accuracy and computational efficiency compared to the previously proposed PF[3] and

---

[4] Part of this chapter is reprinted, with permission, from “Bayes Filter-based Occupancy Detection Using Networked SLEEPIR Sensors" By Emad-ud-din, M., Chen, Z., Wu, L., Shen, Q., and Y. Wang (2023). IEEE SENSORS JOURNAL Pre-print. Copyright © 2023 IEEE

EKF [114] based method; (ii) superior occupancy estimation accuracy when compared to the union of individual node-level accuracies and (iii) independence from historical sensor data to filter out noisy sensor node observations.

The next section of the chapter provides an overview of the system inputs and pre-processing algorithm. Next, the section titled "Bayes Filter Design" provides a detailed description of the method. Then, in the section titled "Discussion", the proposed method's strengths and weaknesses are listed. The section titled "Results" outlines the dataset collection strategy and highlights the method's comparative performance. The last section of the chapter provides a conclusion to the proposed work.

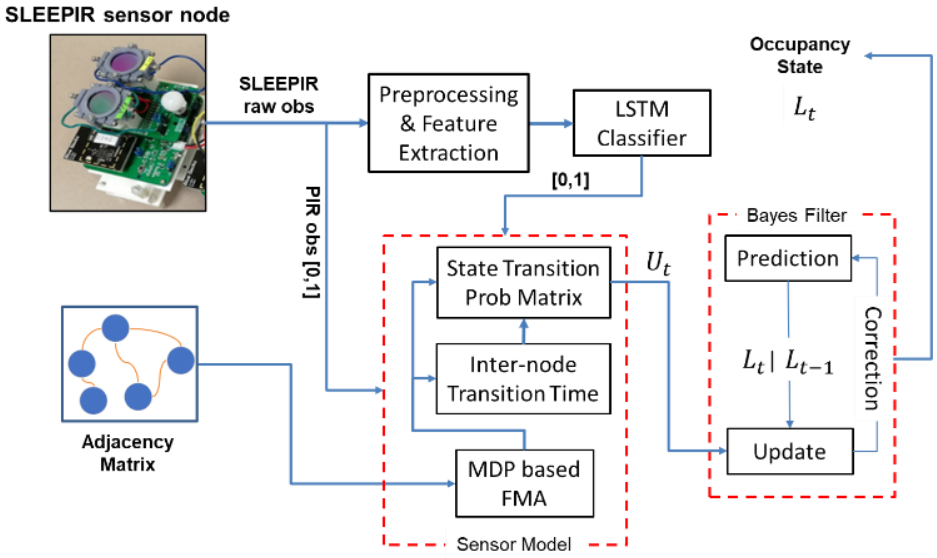

Figure 23. BF based occupancy detection method flow chart. Networked sensor nodes generate voltage, ambient temperature, and PIR data. The voltage is converted to binary occupancy observations via LSTM classifier. The node-level occupancy observations then update a network-level occupancy estimate via BF. Reprinted with permission from [1].

5.1. System Input and Pre-processing Algorithm

The overall system flowchart is presented in Fig. 23. The raw SLEEPIR sensor observations are extracted from the SLEEPIR sensor node using a Bluetooth

communication protocol. The sensor and communication platform details are presented in Chapter 1. A. We present a brief overall algorithm flow below that summarizes the flowchart shown in figure 23.

1. The raw sensor output (which includes SLEEPIR sensor voltage, PIR sensor binary output, and ambient temperature) is collected from each sensor node via a Bluetooth communication protocol.
2. Raw voltage values from the SLEEPIR sensor are pre-processed using an LSTM Network-based thresholding algorithm. This algorithm is detailed in the upcoming sub-section. This thresholding algorithm classifies the raw SLEEPIR sensor observations and outputs in binary whether the sensor has detected human occupancy or not. The traditional PIR sensor output is already binary, so it does not require preprocessing.
3. The binarized observations are converted into a Bayesian update U_t via a sensor model. U_t is used to update our hypothesis L_t that points towards the occupancy state we believe the system to be currently in. This sensor model merges the output from both the SLEEPIR sensor and the traditional PIR sensor into a single BF update. This model is described in detail in the section titled "Bayes Filter Design" in this chapter.
4. The BF receives these periodic updates from the sensor model function and estimates the probability of human occupancy at each of the locations represented in the BF state. The world constraints are embedded in the sensor model and BF design which serves to produce a robust human occupancy belief. Details of the

BF sensor model, state update, and prediction are presented in the section titled "Bayes Filter Design" in this chapter.

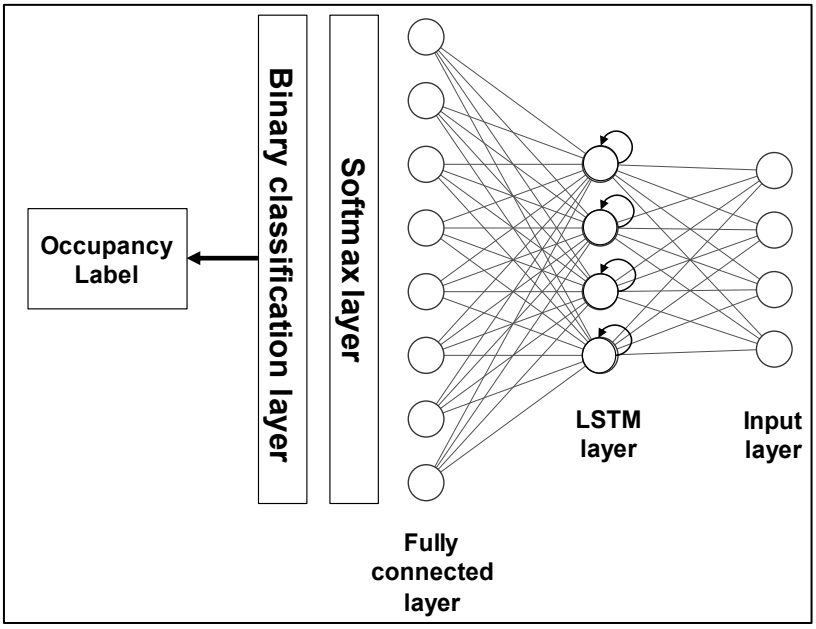

Figure 24. LSTM network architecture for SLEEPIR raw observation binary classifier. Reprinted with permission from [1].

5.1.1. LSTM classifier

As the sensor node generates time-series observations consisting of SLEEPIR raw voltage output $V_{out}(t)$ in equation 1, ambient temperature $T_{amb}(t)$ and digital PIR sensor output $PIR(t)$, we employ RNNs to classify these observations to indicate whether the incoming observation represents human occupancy or not. RNNs when compared to the typical FFNNs, have been shown to achieve higher accuracy with time-series data **[106]**, as these can process and encode the sequential temporal information contained in time-series data. In the implemented pipeline, firstly the incoming time-series data from the sensor node is zero-centered and normalized. The input time series is then divided into pre-determined-sized observation windows. Each window is then labeled as either occupied or unoccupied

based on the available ground truth gathered via the surveillance camera installed at the testbed. Lastly, an LSTM Network is trained with the dataset. The trained network (shown in figure 24) is deployed so that the network can distinguish between the observations indicating occupancy versus those indicating non-occupancy. The Machine Learning (ML) based thresholding algorithm is listed in the following sub-sections.

5.1.1.1. Input Formatting

The hand-tuned ML features are used widely in the literature to produce easily distinguishable values for different data classes [103]. The input could have been chosen to be quantized as input quantization has a proven positive impact on RNN accuracy, provided there is limited information loss [107] but insignificant accuracy improvement was noticed at the cost quantization due to the information loss. Thus, the quantization approach was not opted for.

5.1.1.2. Sliding Window Input Approach

The training dataset obs_T is initialized where each element is created by sliding a fixed-horizon window of length $\boldsymbol{l}$ over the 4-D training input time-series consisting of the following elements $[V_{pp1}(t), V_{pp2}(t), T_{amb}(t), PIR(t)]$. The labels $label_T$ where each element corresponds to each window in obs_T, are then initialized. An element is set to "*occupied*" if a surveillance camera-based *ground-truth*$_T$ indicates that the human subject was present for more than 50% of observations in the Field of view (FoV) of the sensor. Otherwise, the element is set to "*unoccupied*". A suitable window length (l) is known to be a critical parameter that has a pronounced impact on over-network accuracy [115]. This impact will be highlighted in the next sub-section.

5.1.1.3. LSTM Network Architecture

A highly cited deep forward RNN model proposed in [109], which contains multiple layers of recurrent units that are connected "forward" in time, is used as a reference ML model. This model architecture is simple yet powerful enough to produce reliable results over publicly available datasets which consist of time-series data. The online LSTM model shown in figure 24 contains a single hidden layer of 16 recurrent neurons. During the evaluation phase, all RNN models use 3, 6, 9, and 16 neurons depending upon the experiment configuration. There are also 4 input neurons to match the number of input time series from the sensor node i.e., $[V_{pp1}(t),V_{pp2}(t),T_{amb}(t),PIR(t)]$. There are two output neurons to match the output classes corresponding to "occupied" and "unoccupied" status.

In Chapter 4, a comprehensive search for suitable RNNs was performed for the occupancy detection application of SLEEPIR sensors. The analysis included testing the collected dataset over LSTM, Bi-directional LSTM (Bi-LSTM), CTRNN, MGU [110], and GRU networks. Observation window length l was varied over a reasonable range to see if certain networks perform better than others. It was found that for l=*60 sec*, the accuracy was highest across all architectures. This result signified that the most effective discriminating features exist over a window length of 60 seconds. It must be mentioned here that SLEEPIR collects two consecutive observations over a span of 60 seconds. The analysis in Chapter 4 concluded that LSTM and Bi-LSTM outperformed other RNNs in nearly all window length configurations (l=30, 45, 60, 90 seconds). Thus, LSTM was chosen as our network of choice as it is relatively less expensive in terms of resources

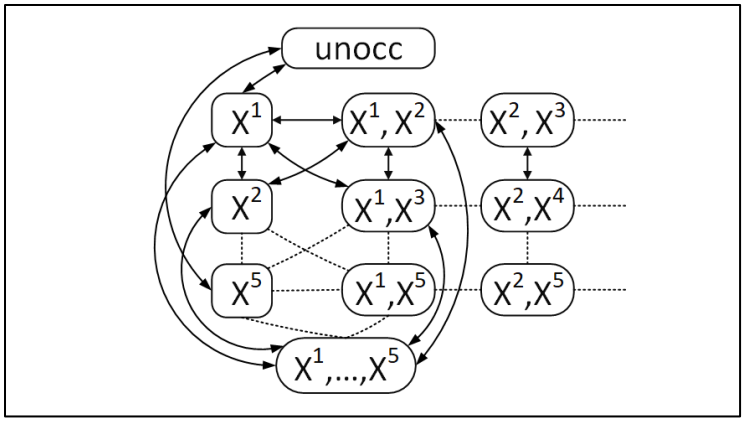

Figure 25. The underlying MDP is used by FMA. Each state *s* consists of a possible combination of sensor nodes where occupancy can be detected. The MDP consists of a total of 2^nstates where *n* is the number of networked sensor nodes. Action *a* connect the states and the probability for each action is defined by the Node Adjacency Matrix and our assumptions about how quickly can the occupancy state change. Reprinted with permission from [1].

when compared to Bi-LSTM.

5.2. Bayes Filter Design

After the node-level observations are binarized via the proposed ML architecture, these observations are used to update a BF which produces a network-level occupancy estimate. BF provides a real-time posterior probability density function (pdf) of the state (occupancy belief) based on available information. The BF is thus 'optimal' as it seeks the posterior distribution which integrates and uses all of the available information expressed by probabilities [116]. The network-level occupancy detection is a two-tiered algorithm. In the first tier, we shape the occupancy state of the monitored space as an MDP as shown in figure 25. The MDP represents the dynamics of the real world. When MDP is presented with a goal state *G* it suggests an optimal policy (a set of actions that results in a set of state transitions) that leads us to *G*. The notion of goal state *G* is useful when we need to evaluate the expected time to transition from any occupancy state within the MDP to the goal state *G*. The second tier consists of a BF that periodically receives the occupancy

status from individual SLEEPIR nodes. When individual SLEEPIR nodes point towards an occupancy state, we consider that state as a goal state *G*. Then an optimal policy π is generated by the MDP model to reach goal G. BF keeps on updating its belief based on how well the incoming occupancy status observations conform to the suggested policy π. In case the incoming occupancy observation conforms well to the MDP suggested policy π, the observation is given a higher likelihood compared to a poorly conforming observation. The sensor model for the BF maps the incoming observations to the likelihood values of the overall observed space as being occupied (or unoccupied).

5.2.1. MDP and State Transition Matrix Evaluation

Before the elements of BF can be described, the steps that help determine the State Transition Matrix must be described in detail i.e., the MDP formulation, policy generation and execution, and Fundamental Markov Analysis for Markov Chains.

5.2.1.1. MDP Formulation

Each state of the MDP represents a possible combination of the occupancy status of each sensor. For example, a state where the Entrance and Kitchen node indicate occupancy and the remaining nodes indicate unoccupancy, is represented by X^4X^5. Similarly, a state where all nodes are indicating occupancy will be represented by $X^1X^2X^3X^4X^5$. A Markov Decision Process with mortality is a tuple as per its standard definition [24].

$$\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$$

where:

- $\mathcal{S}$ is a finite set of states *s*. Each state represents a possible occupancy state for the observed area.

- $\mathcal{A}$ is a finite set of actions where each of its elements a^s is an action intended for a transition to the state s.
- $\mathcal{P}$ is a state transition probability function $\mathcal{P}_{ss'}^{a} = P[\mathcal{S}_{t+1} = s' | \mathcal{S}_t = s, A_t = a]$
- $\mathcal{R}$ is a reward function defined by expected value function E: $\mathcal{R}_s^a = E[R_{t+1} | \mathcal{S}_t = s, A_t = a]$. A[24] large positive reward value can be set for any $s \in \mathcal{S}$, to designate s as G or goal state.
- γ is a discount factor where $\gamma \in [0,1]$. This enables us to model the weight assigned to the future reward at each time step.

The belief Bel about which states are occupied (or unoccupied), changes as state-to-state transitions are made within the MDP via available actions $\mathcal{A}$. $\mathcal{A}$ needs to be defined in detail here. Simply put it is a set of actions for all possible states that add up to 2^5.

$$\mathcal{A} = \{a^{unoccupied}, a^{X^1}, a^{X^2}, a^{X^3}, a^{X^4}, a^{X^5}, a^{X^1X^2}, a^{X^1X^3}, \ldots, a^{X^1X^2X^3X^4X^5}\} \qquad (14)$$

Here the superscript for each action $a^{\{state\}}$ represents the state to which the corresponding action will generate a transition.

5.2.1.2. Node Adjacency Matrix and MDP Policy Generation

For an MDP to be able to reach a solution (to generate a policy), a transition probability function $\mathcal{P}$ must assign outgoing probabilities to each state. This function essentially represents the world dynamics, telling the model how likely a transition is possible between any two states within the MDP model. Each of the states in the MDP model, is assigned a set of outgoing probabilities by $\mathcal{P}$ depending upon the world dynamics. In the case of indoor human occupancy detection, these probabilities are assigned using the *node*

adjacency matrix (ADJ) available to us. The sensor node adjacency matrix must also include the adjacency information for each node to the entrances of the observed space. The following expression is used to assign the probabilities to the state transition matrix.

$$\mathcal{P}_{ss'}^{a} = \begin{cases} 0.8 \times \underset{ss'}{\Pr} \times T^{a} & \text{if } (\mathrm{ADJ}(\mathrm{s}, \mathrm{s}') = 1) \\ 0.2 \times \underset{ss'}{\Pr} \times T^{a} & \text{if } (\mathrm{ADJ}(\mathrm{s}, \mathrm{s}') = 0) \\ 1.0 \times \underset{ss'}{\Pr} \times T^{a} & \text{if } (\mathrm{ADJ}(\mathrm{s}, \mathrm{s}') = 1) \;\&\&\; (\mathrm{s}' = unoccupied) \end{cases} \tag{15}$$

It must be mentioned here that the state transition matrix $\mathcal{P}_{ss'}^{a}$ mentioned above represents daytime transition probabilities. For nighttime transition probabilities, the last rule is modified to $0.5 \times \underset{ss'}{\Pr} \times T^{a}$ if $(\mathrm{ADJ}(\mathrm{s}, \mathrm{s}') = 1)$ && $(\mathrm{s}' = unoccupied)$ as occupancy state is less likely to transition to the *unoccupied* state during the nighttime hours. This will help reduce false negative detections caused by IR shielding. Here adj represents the SLEEPIR sensor node adjacency matrix. T^{a} is the action probability dictated by the transition model. The transition model determines the probability for the outcome for action *a*. It is assumed that $T^{a} = 0.7$ when action *a* is taken and T^{a}=0.3 when an action *a'* is taken where $a \neq a'$. Further explanation about the transition model is provided in [24]. $\underset{ss'}{\Pr}$ is the probability of transitioning from state *s* to state *s'*, evaluated by FMA detailed in section titled "FMA of Markov Chains" in this chapter. It can be observed in equation 15 that the transition probabilities are higher where inter-sensor proximity is higher. Values to $\mathcal{R}$, which represents a specific reward value associated with each state in the MDP, also need to be assigned. A higher reward value makes it highly probable for a policy to generate a Markov Chain that contains the corresponding state. A lower or a negative reward value makes it highly improbable for a policy to generate a Markov Chain

that contains the corresponding state. It is thus ensured to set a high reward value for state that is a goal state G. For remaining states in the MDP, a value near 0 is set to indicate that there exists no preference vis-à-vis states other than the goal state. A standard *Policy Iteration based dynamic programming solution* [24] then uses $\mathcal{P}$ and $\mathcal{R}$, to generate a policy π. 2^5 policies are generated by setting each state as a goal state G, one by one and then generating a corresponding policy for that state. So, the set of policies forwarded to BF will be as follows.

$$\Pi = \{\pi^{\text{unoccupied}}, \pi^{X^1}, \pi^{X^2}, \pi^{X^3}, \pi^{X^4}, \pi^{X^5}, \dots, \pi^{X^1X^2X^3X^4X^5}\} \quad (16)$$

This set of policies Π, is then forwarded to the BF algorithm.

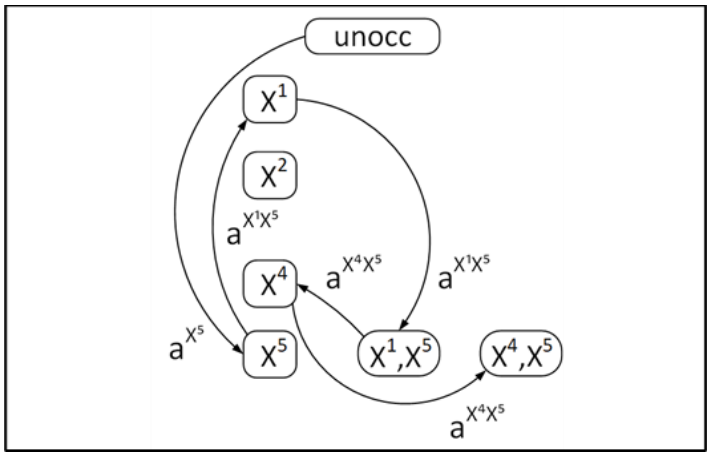

Figure 26. A sample policy $\pi^{X^4X^5}$ evaluated by Policy Iteration Algorithm. A corresponding Markov Chain $MC(\pi^{X^4X^5})$ is generated when the policy is executed. World dynamics essentially dictate that certain state transitions are necessary before state X^4X^5 can be reached by the occupant when starting from the unoccupied state. Reprinted with permission from [1].

5.2.1.3. MDP Policy Execution

In this phase, at the start of execution, a starting state is provided which is the *unoccupied* state, signifying that all observed area is unoccupied. A goal state *G* is also provided which corresponds to the present occupancy scenario as observed by the SLEEPIR sensors. For

example, if the occupancy is detected at the nodes X^4 and X^5 and no occupancy is detected at the remaining nodes, the chosen goal state is $G = \{X^6X^7\}$. A Markov Chain MC^{π^G} is then created as per a standard non-deterministic process [24] and uses pre-computed policy π^G computed during the last step. More specifically, the process executes the action choice by having the transition from the state *s* depending on the actions stipulated in the policy at s, i.e., $\sum_{a\in\mathcal{A}} Pr[\pi^G(s)]\mathcal{P}^a_{ss'}$. Figure 26 shows how a sample policy let's say $\pi^{X^4X^5}$is evaluated. What is important is that how far into the $MC^{\pi G}$, the goal state X^4X^5 occurs. If the G occurs too far into the MC^{π^G} than it will be unrealistic to assign a high credibility to the observation. In case G occurs too early into an MC^{π^G} then again it would not make sense to assign high credibility to the observation. To be able to determine a sensible threshold of where the G should occur in the MC^{π^G}, an MDP based analysis called *FMA of Markov Chains*, must be done. The details of this analysis will be discussed in the next sub-section. It must be mentioned here that the position of *G* in the $MC^{\pi^G}$is evaluated to be able to assign probabilities to each observation via a sensor model, depending upon how much the observation conforms to the real-world temporal and spatial constraints. For the sake of clarity, it must be mentioned that a sequence of non-deterministic actions $\{a^{X^5}, a^{X^1X^5}, a^{X^1X^5}, a^{X^4X^5}, a^{X^4X^5}\}$ was suggested by the policy $\pi^{X^4X^5}$that resulted in the Markov Chain $\{X^5, X^1, X^1X^5, X^4, , X^4X^5\}$.

5.2.1.4. FMA of Markov Chains

For a discrete absorbing Markov chain, with 1 absorbing goal state and $n-1$ transient states, there exists an associated summary of expected temporal behavior that can be

characterized via single $(n-1) \times (n-1)$ matrix called the fundamental matrix [117]. Let us denote the fundamental matrix for MC^{π^G} by N^{π^G}. The entry $n_{ij}^{\pi^G}$ of this matrix gives the expected number of times the occupancy is in transient state s_j given that it started in transient state s_i. Assuming the initial state of the occupancy is at s_i, the following two pieces of information as per [117] need to be computed:

a. **Inter-state transition time:** This gives us the expected number of time steps to reach the goal state G. The time to reach goal is evaluated as: $\tau^G = \sum_{j=1}^{n-1} n_{ij}^{\pi^G}$

b. **Inter-state transition probability:** This gives the probability that the chain is absorbed in state G as $Pr(G) = \sum_{j=1}^{n-1} n_{ij}^{\pi^G} r_{jG}$ where r_{jG} describes the probability of transiting from transient state s_j to absorbing state G, a submatrix of elements from the transition matrix of MC^{π^G}.

The *Inter-state transition time* parameter essentially serves as the gatekeeper for any observation to be able to update the BF belief, as noisy observations often suggest unrealistic transition times for an occupant to travel from one occupancy state to the other. Similarly, *Inter-state transition probability* provides a probability of transitioning between two occupancy states which are crucial to the sensor model of the BF.

5.2.2. Bayes Filter

The occupancy state x needs to be estimated for the overall monitored indoor space based on the observations received from the individual SLEEPIR sensor nodes. The estimation problem is treated via a BF. A combined occupancy snapshot from all

individual SLEEPIR sensors in the network can be referred to as z_t for the observation at time t, then a suitable update equation as per the BF definition [118] will be

$$p(x|z_t, \pi^G) \propto p(z_t|x, \pi^G)p(x|z_{t-1}, \pi^G) \quad (17)$$

In the above expression, the dependence on policy π is explicit. This means that the estimate of the occupancy state is assumed to be dependent upon the latest observation z_t and the policy π^G where state G is the occupancy that the observation z_t is pointing to. For example, if only two sensors X^4 and $X^4$are indicating occupancy at time *t*, state G is set to "X^4X^5". Thus, the above recursive expression gives the complete form of the filter. However, the following essential elements need to be explained in detail i.e., Observation Formulation, Concept of Time and Sensor Model.

5.2.2.1. Observation Formulation

A collective observation z_t generated by a network of SLEEPIR sensor nodes at time *t* is denoted as follows:

$$z_t = \{z_t^{X_1}, z_t^{X_2}, z_t^{X_3}, z_t^{X_4}, z_t^{X_5}\}, where\ z_t^{X_i} \in \{0,1\} \quad (18)$$

For example, in case there is no human detection at the node-level at time *t=1* then $z_1 = \{0,0,0,0,0\}$. Note that {0,0,0,0,0} is special state referred to as "*unoccupied* state" as it represents unoccupancy while all other states represent occupancy. The unoccupied state is shown in figure 26.

5.2.2.2. Concept of Time

Since the individual sensors produce at least one observation about human presence every 30 seconds, the filter can receive z_t every 30 seconds and is able to process the

observations without any lapses the observations. In case there is a lapse in the observations due to a malfunction, the sensor model can address such a scenario. It is essential to state here that we set $\mathrm{G} = z_t$ whenever each observation arrives. This also means that a policy π^G, time to absorption τ^G and $Pr(G)$ will be used for filtering the observation.

5.2.2.3. Sensor Model

The sensor model, $p(x|z_t, \pi^G)$, can now be written in terms of two outcomes of FMA i.e., time to absorption and absorption probabilities.

$$p(x|z_t, \pi^G) = \begin{cases} Pr(G) & \text{if } \Delta t \leq \varepsilon\tau^G \\ 0 & \text{if } \Delta t > \varepsilon\tau^G \\ 0 & \text{if } \Delta t \leq \varepsilon\tau^G \text{ and } z_t = z_t^{\varphi} \end{cases} \quad (19)$$

The sensor model simply makes sure that 0 occupancy probability is assigned to the observation if the observation state cannot be reached in the time$Increment$ is the duration between two consecutive observations. Scalar ε is the safety margin that accounts for any underestimation when it comes to the time to absorption calculation by the FMA. The sensor model also assigns a 0-occupancy probability in case the observation points to the unoccupied state. In all other cases where observation indicates an occupied state, a calculated time to absorption a $Pr(G)$ is suitable estimation of likelihood.

5.3. Discussion

The occupancy output for equation 17 is a pdf. This pdf represents the probability of human occupancy over all possible combinations of locations. Whichever location combination has the maximum probability at time *t,* would be the belief Bel_t about the

state of the occupancy. A time plot that represents the shifting belief Bel_t about the human occupancy over time is shown in figure 27 (top). This plot shows the progression of the network-level output of our proposed method as the BF belief switches between Unoccupied and the remaining occupied states based on the sensor model and the sensor nodes input. The output is for 24 hours of observations collected during the day. It is

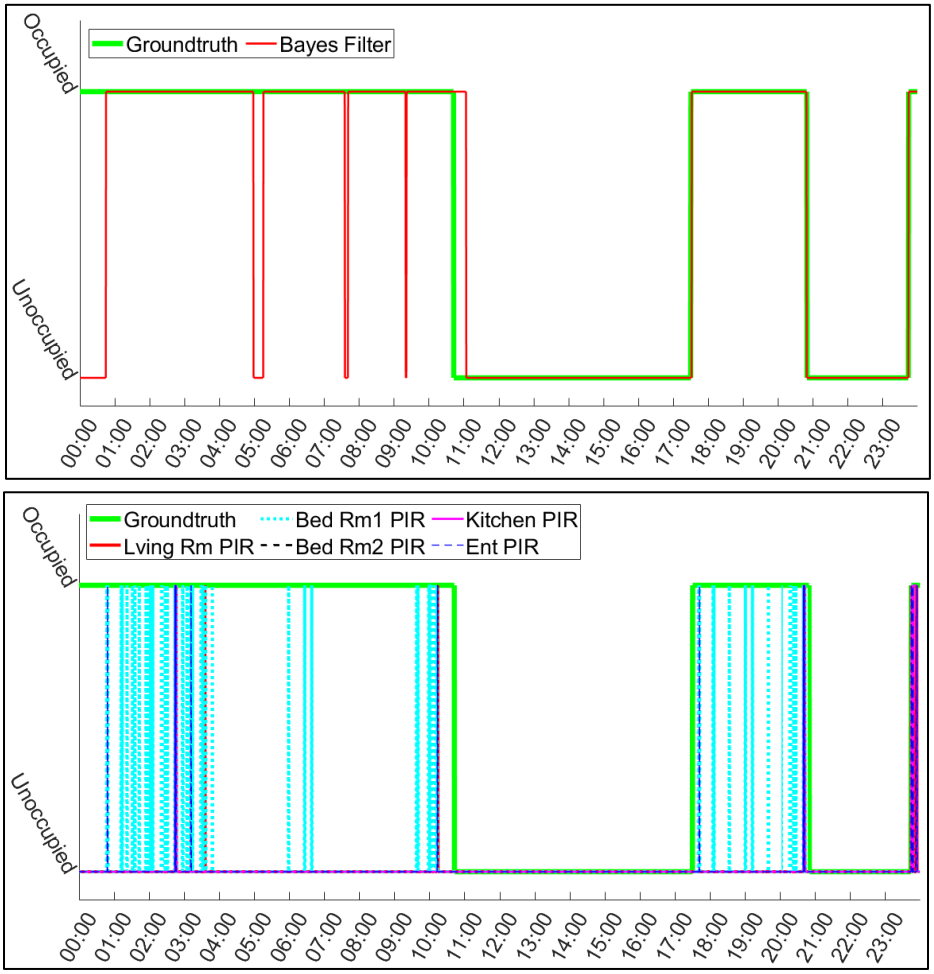

Figure 27. (Top) A time plot that shows the progression of occupancy state through a typical day. The Bayes filter output is compared to the apartment level occupancy ground truth. (Bottom) A time plot that shows the contribution of traditional PIR observations from each of the sensor nodes towards overall occupancy. We notice that PIR activations are few and far between especially between 3 am and 9 am window. Reprinted with permission from [1].

interesting to show here how traditional PIR output corresponds to the SLEEPIR sensor system output.

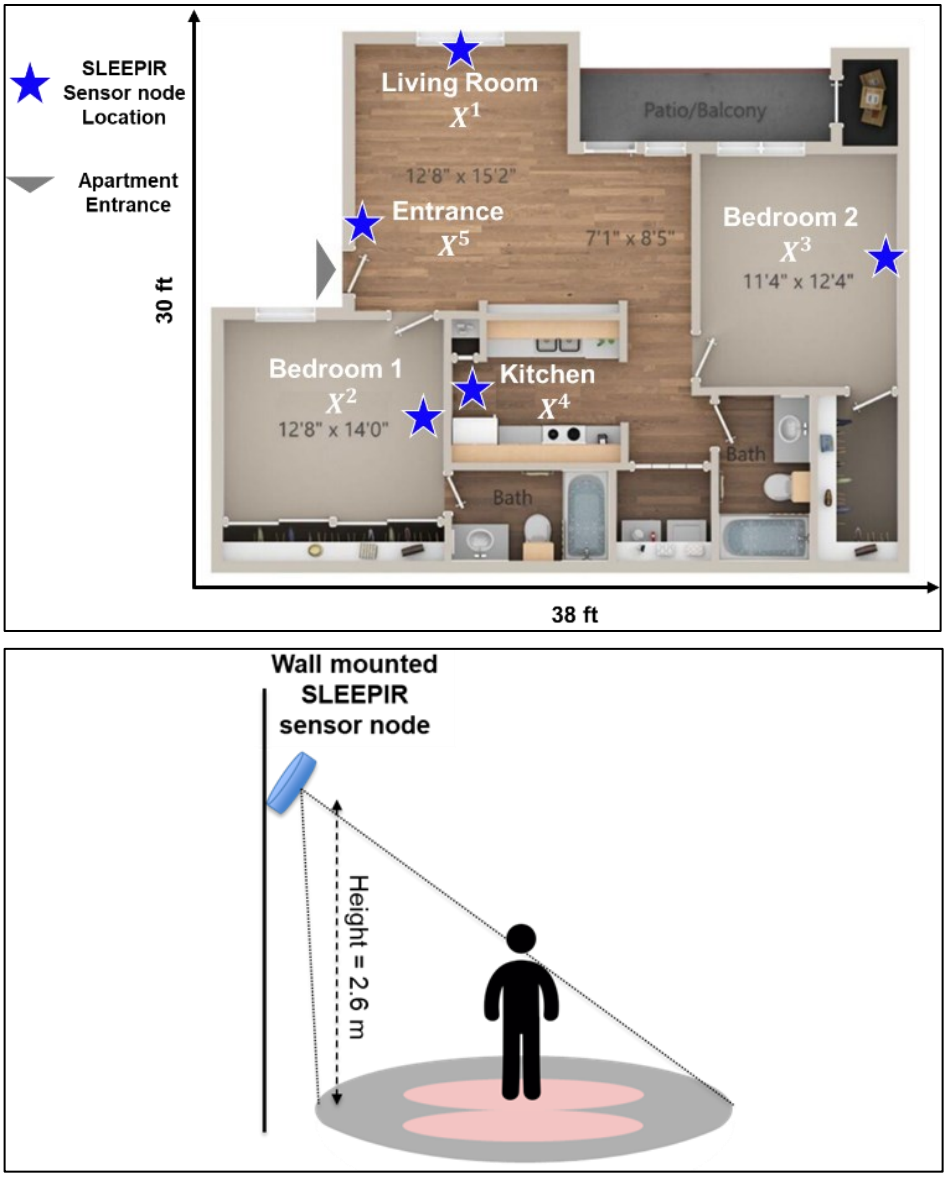

Figure 28. (Top) Human occupancy is estimated for the locations X^1 through X^5. Apartment level occupancy ground truth is collected via manual entries to a log register. (Bottom) Pink footprint is shown for SLEEPIR sensor modules and grey footprint shown for traditional PIR sensor. Reprinted with permission from [1].

Fig. 27 (bottom) superimposes the traditional PIR output over the same 24-hour ground truth period as shown in Fig. 27 (top). Fig. 27 (bottom) shows that traditional PIR only contributes a small fraction towards to overall detected occupancy for the same period. It can also be seen that during the night hours, PIR activity is limited and mostly

restricted to bedrooms. False negatives can also be observed during the night hours (see Fig. 27 – top), as the ML thresholding algorithm can often not detect the sleeping subject covered in a blanket with little skin exposed due to the IR shielding effect.

It can also be observed that Entrance PIR is triggered at the time when the switch between occupied and unoccupied states occurs. This is because there is only a single entrance/exit to the apartment and the entrance node registers a PIR observation whenever an entry or exit event happens. Most PIR-based systems depend on such entrance and exit events, but it is difficult to determine whether the event was an entry or an exit in certain scenarios using only traditional PIR sensors. This is because a single PIR sensor can easily get confused between exit and entry given the complex human motion behavior e.g., there may be multiple subjects involved or simultaneous entry and exit events can happen.

The results section must be preceded by certain assumptions that the proposed method relies on, to provide reliable occupancy, namely (a) Occupants entering the apartment will trigger at least one SLEEPIR sensor node. (b) The SLEEPIR sensors have a limited FoV and consequently have sparse coverage (c) All human subjects within FoV do not use any specialized means to shield the emitted body IR radiation.

5.4. Results

5.4.1. Dataset

The experiment testbed for dataset collection was a 2 bed 2 bath, first-floor residential apartment. There were at least 2 occupants who used this apartment as their primary residence. The sensor node layout configuration is shown in Fig. 28. The experimentation was completely uncontrolled. The optimal sensor node configuration for

best coverage was not explored since the sensor node network is expected to be installed by a non-expert user who may choose to deploy sensors in a sub-optimal configuration. Thus, a single non-optimized deployment configuration was used. Five SLEEPIR sensor nodes were deployed in the months of April, May, and June where the average outdoor temperatures range between 59F to 91F. Each node collects the observation every 30 seconds. The SLEEPIR observations were evaluated using the raw SLEEPIR sensor voltage values. Certain thresholds were used to remove noisy observations as per the literature presented in [10]. Manual logging was used to label the ground truth. Apartment-level occupancy observations were noted down as logbook entries whenever anyone entered or left the apartment. Data for a total of 30 days was collected. The observations were down sampled to 1 observation per minute. This totaled up to 43200 observations for each sensor, within the dataset. For the ML thresholding algorithm, 80% of observations were used for training, 10% for validation, and 10% for testing using 5-Fold Cross-Validation. At least 2 university students (both young males) were the primary subjects for the dataset. The subjects used the apartment as their residence.

5.4.2. Accuracy Results

The proposed method claims to achieve superior accuracy and execution time when compared to already proposed state-of-the-art sensor nodes network-based occupancy detection methods i.e., EKF and PF [3]. An EKF implementation [114] was chosen to be a baseline method for comparison because it is a Gaussian approximation method and a special case of BF with linear, quadratic, and Gaussian assumptions. Secondly, EKF has been compared to BF frequently in literature [119-121] in terms of

algorithmic efficiency. The work in this chapter aims to establish that while the proposed BF method bears similarities to EKF yet is more accurate and efficient compared to existing implementations. The impact of the number of sensor nodes on occupancy detection accuracy is also evaluated. The goal of this evaluation is to prove the efficacy of the proposed method while keeping the cost and infrastructure footprint limited. Firstly, the accuracy results are presented that compare the performance of 1-node, 3-node, and 5-node networks. The results of this comparative experiment are listed in Table 13. The node subsets in Table 13 are selected to maximize coverage in the case of a 3,5-node combination. For the 1-node case, the entrance node was selected as it maximizes the information about the occupancy of the apartment. Table 14 compares the average accuracy of the proposed BF to baseline methods. It also compares the average execution time for processing a single occupancy estimate by the proposed and the baseline methods. The execution time includes the run times for signal preprocessing, feature extraction, LSTM inference, sensor model query, and filter update and prediction steps. The proposed model error is broken down and assigned to known error sources as illustrated by Fig. 29. The error is also broken down into FPs and FNs and Fig. 30 illustrates the average false positives and false negatives reported by the proposed and each of the baseline methods.

Table XIII. Impact of Sensor Nodes Reduction in the Network. Reprinted with permission from [1].

Method	**1-node Accuracy** (X^5)	**3-node Accuracy** (X^5, X^2, X^3)	**5-node Accuracy** ($All\ nodes$)
EKF	33.27±9.77%	64.97±7.63%	76.52±5.01%
PF	37.19±6.81%	77.20±4.81%	84.29±4.38%
BF	40.05±4.06%	88.17±4.59%	92.04±4.40%

Table XIV. Accuracy Comparison between Baseline and Proposed Models. Reprinted with permission from [1].

Method	Accuracy (%)	Avg Execution time (ms)
EKF	76.52±5.01%	87±1.15
PF	84.29±4.38%	698±122.02
$X^1\|X^2\|X^3\|X^4\|X^5$	68.36±9.73%	14±0.0
BF	92.04±4.40%	59±1.35

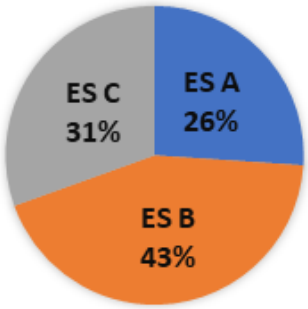

Figure 29. A pie chart illustrating the contribution of each error source towards the false positives and false negatives reported by the proposed method. Reprinted with permission from [1].

5.4.3. Results Discussion

The presented results in the last subsection underscore some key points. Firstly, BF consistently shows superior performance both in terms of accuracy and execution time when compared to EKF[114] and PF[3] implementations. It is important to highlight that each of these implementations has unique strengths and weaknesses and cannot be considered representative of the whole class of EKF or PF implementations. While the PF implementation is similar to the proposed BF implementation, it is important to highlight some important features of the EKF implementation. The EKF implementation is a real-time EKF-based networked sensor-based occupancy estimation algorithm. This system originally estimated the number of occupants in each room of the monitored building. The EKF system was modified to estimate the binary occupancy and use its output to compare

to the proposed BF output. This EKF system handles the non-linearity in the occupancy detection data by placing certain constraints on the model like placing upper bounds and lower bounds on exit/entrance rates, placing upper bounds on occupant flow from one room to another, conservation principle on the number of people in the building, etc.

Table 13 highlights an important aspect of the proposed BF system i.e., scaling down to a smaller number of sensor nodes. In case a sensor node needs to be dropped from the system, the proposed method considers the observations from such sensor nodes as unoccupied. Any node that has an unoccupied status does not determine the present state of the system e.g., for the 3-node scenario, the status of nodes $X^1, fourthX^4$ is set to unoccupied permanently. So, these nodes do not contribute towards the states of the system by design.

The higher accuracy of the proposed BF demonstrated in Table 14 is primarily because of the sensor model deployed in the proposed method. The sensor model was able to integrate the inter-node and entrance proximity information into the Bayesian updates rendering the proposed method simple yet effective enough to surpass an inherently superior PF-based method. The execution time results for BF are not very different from EKF and both methods use effectively similar steps to reach an estimate. Matrix inversion remains the most computationally expensive step for EKF while the transition model matrix model evaluation is the most time-consuming phase for the proposed BF method. It should be mentioned here that the transition model matrix is only evaluated once for each node deployment configuration.

The row for the method titled "$X^1|X^2|X^3|X^4|X^5$" in Table 14 denotes the performance results for the system level output where the algorithm is simply a union of binary occupancy status from each node. This binary status is generated by the ML thresholding algorithm for each sensor node. For this system-level union algorithm, it can be observed that each of the baseline estimation algorithms (EKF, PF, BF) is significantly more accurate than this simple rule-based method.

5.4.4. Error Breakdown

The error sources (ES) illustrated in Fig. 29 were determined by attributing the errors to the location and time of certain IR anomalies like IR noise due to cooking, warm water in a sink, or IR shielding effect [8] due to blanket or extra layers of clothing covering the entire length of the body. The error categories are as follows.

a. **ES A:** FP caused due to the IR noise in the environment. Sources include (1) Stoves, hot utensils (2) Warm water taps (3) Laptops, chargers, and other electronics that can warm up (4) Space heaters (5) Objects that retain body heat after human contact.

b. **ES B:** FN caused due to low human IR radiation reaching the sensor. This happens either due to IR-blocking materials like blankets shielding human subjects or due to sensors failing to detect human IR radiation due to limited FoV.

c. **ES C:** Errors caused by wrong assumptions in network-level estimation algorithms.

We can observe in Fig. 29 that error category B dominates the pie chart as the experiment subjects spend a significant amount of time sleeping covered in blankets during their sleep hours. Errors shown in table 14 are divided into FPs and FNs via confusion charts shown in figure 30.

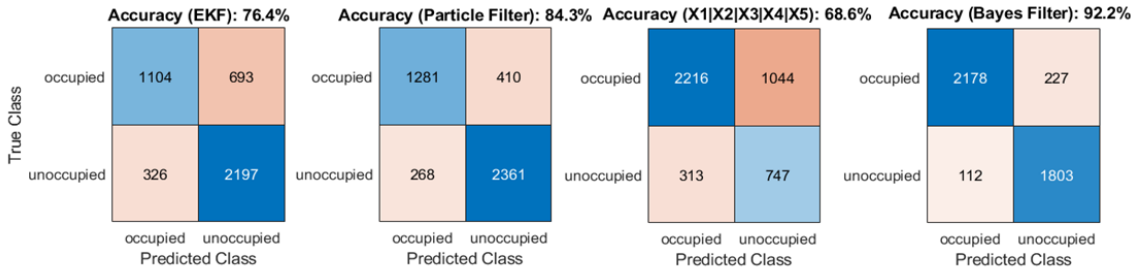

Figure 30. The confusion matrices showing the performance comparison between the EKF, Particle Filter, Simple node output union algorithm, and proposed Bayes Filter method. Reprinted with permission from [1].

5.5. Conclusion

A BF-based occupancy detection method is proposed that detects both stationary and moving subject occupancy using a network of SLEEPIR sensor nodes. The primary claim of this work, i.e., installing a minimal number of SLEEPIR nodes in sub-optimal configuration to achieve high accuracy occupancy detection is proven by the results that are evaluated over a long-term dataset that spans over 30 days. Results indicate an average 23.68% occupancy accuracy improvement when compared to the accuracy state delivered by individual SLEEPIR nodes. Results also indicate an average 7.74% occupancy accuracy improvement when compared to the accuracy state determined by the previously proposed PF-based occupancy estimation algorithm. Furthermore, the proposed BF-based method is shown to be faster by orders of magnitude when compared to a competing PF-based occupancy implementation.

6. CONTEXT-AIDED OCCUPANCY DETECTION AND TRACKING

The problem of occupancy detection is inherently complex as occupancy estimation experiences considerable accuracy deterioration [3, 76] due to constantly evolving environmental and occupancy scenarios. Due to the dynamic nature of occupancy scenarios, it is virtually impossible to collect a comprehensive training dataset that contains patterns encompassing all anticipated occupancy scenarios. Such a dataset, although highly unlikely to exist, would also require significant computational power to train due to its size. For the same reason, the ML models are typically trained off-site, and model updates are pushed to the inference engine which requires over-the-cloud connectivity for the occupancy sensors. For such models, a novel input occupancy pattern that does not belong to the distribution of the training dataset would cause degradation in occupancy detection accuracy. Since occupancy tracking is a higher-order property of occupancy detection [6], it is also expected to suffer a loss in terms of tracking accuracy. The overall occupancy detection challenge is a well-investigated topic [88]. On the other hand, the OODL [2] algorithms are memory constrained and typically have upper bounds on the training dataset size. The proposed method uses the bounded size of the training dataset size to its advantage. The method limits the classification space for a KNN model by determining contextual information, to enhance the occupancy detection accuracy. We term contextual information as simply the context for the occupancy. Determining a context for an occupancy classification model involves clustering the observation features based on Euclidean distance. It was found during the experimentation that each of the clusters approximately corresponded to a specific period during a week e.g., weekend

night, weekday morning, etc. Thus, the context e.g., weekend night outlines a cluster or a subset of the overall training dataset. It was found during the experimentation that an occupancy classifier trained over the subset delineated by the context outperforms a classifier that has been trained for the overall training dataset.

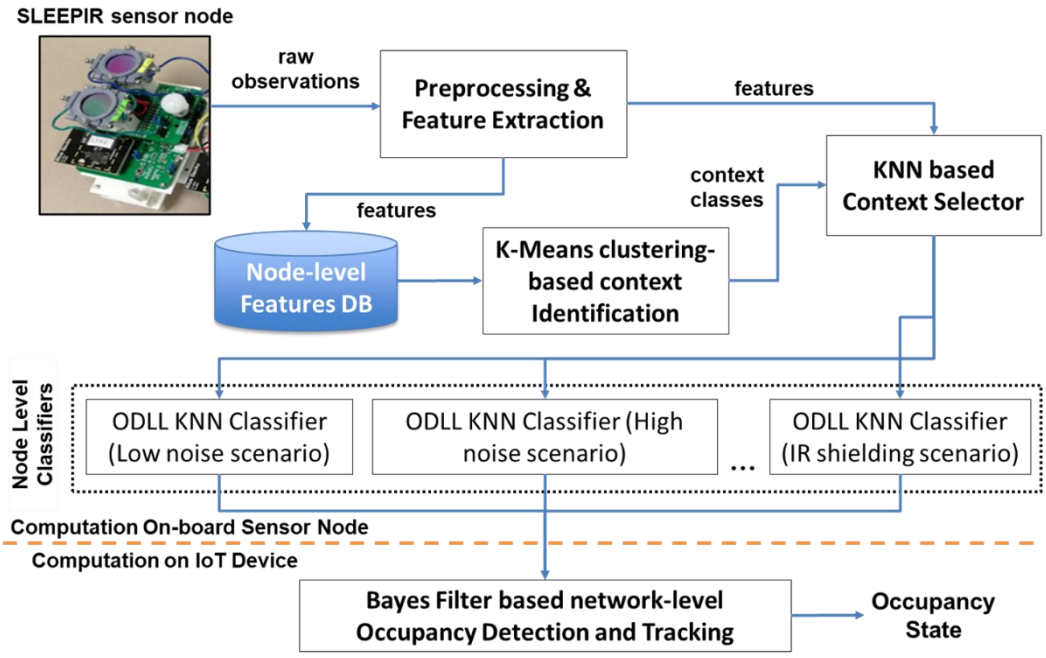

Figure 31. In the Context-aided Occupancy Detection and Tracking System shown above, features are extracted from sensor node observations. These features are then stored in a short-term DB. The DB is assessed for feature clusters using the Davies Bouldin Index (DBI). Clusters link to context classes representing occupancy scenarios. A KNN classifier chooses an ODLL occupancy classifier to determine occupancy state. A Bayes filter system evaluates all node outputs, providing room-level occupancy estimates every 60 seconds.

Figure 31. illustrates the overview of the proposed method. A brief stepwise flow of the method is presented below.

1. Firstly, hand-crafted features are extracted from the temporal segments of zero-centered, normalized occupancy sensor observations. These features are then stored in a bounded feature database.
2. Feature clusters are then identified and subsequently mapped to context classes. Each context class represents an occupancy scenario.

3. For each new incoming observation, a context class is determined via a KNN-based classifier called a KNN-based Context Selector. A corresponding periodically trained ODLL KNN model [2] is selected based on the identified context.
4. The ODLL KNN model then determines the node-level occupancy.
5. The occupancy output and location information of nodes is then provided to a BF-based Occupancy Detection and Tracking algorithm. This algorithm estimates the node-level occupancy state of the system and tracks the node-level occupancy over time.

It is useful to mention here that steps 2, 3, and 4 constitute what is called Hierarchical Classifier Selection (HCS) framework. The key contributions of this work are presented below.

- A context-aided hierarchical classification approach is proposed that is unique within the domain of occupancy detection and tracking.
- The context limits the classification search space for occupancy detection thus improving occupancy classification accuracy and execution time compared to the baseline On-Device Lifelong Learning(ODLL) KNN[2], and static-dataset LSTM [20] algorithms.
- A BF-based tracking algorithm is presented that provides robust occupancy tracking at node-level resolution.

The proposed framework avoids the overhead of offline training using large datasets as well as eliminates the need for over-the-cloud ML model updates.

The details of the underlying networked sensor nodes used in the dataset collection and a description of the steps involved in pre-processing sensor inputs have already been shared in section 3.2. For the remaining chapter, the major features and working of the HCS framework is outlined in section 6.1. Section 6.2 describes the network-level BF algorithm that estimates the system-level occupancy based on the HCS framework and tracks occupancy. Section 6.3 presents a brief discussion of the method. Section 6.4 outlines the dataset collection strategy and lists the method performance results. Section 6.5 presents a conclusion to this chapter.

6.1. Hierarchical Classifier Selection Framework

The purpose of the HCS framework is to train and select the most accurate occupancy classifier among the set of continuously trained classifiers given a context. Here the context is the information that plays the pivotal role in selecting the optimal occupancy classifier. Section 6.1.1 details how the context is evaluated.

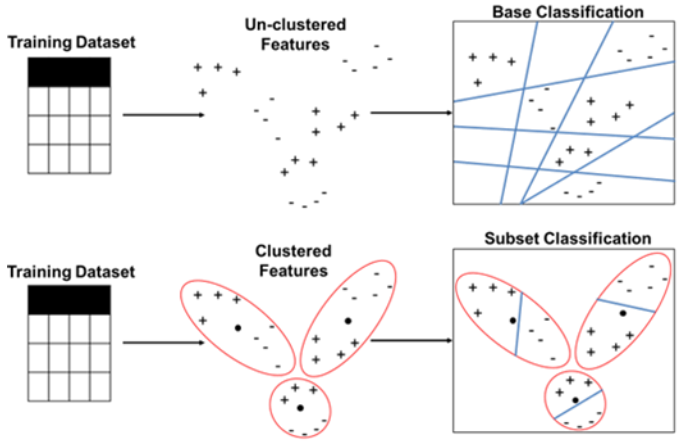

Figure 32. (Top) The un-clustered features being classified for occupancy (+) and non-occupancy (-). (Bottom) The clustered features being classified. The classification is much simpler in case only a single cluster is considered at a time. The caveat here is that the clustering needs to be meaningful and should have minimal outliers.

6.1.1. Context Generation through Data Clustering

We observe in figure 31 that the node-level feature database (DB) contains a labeled base training dataset with labels. The goal of clustering is to identify subsets of the base training dataset that enable more accurate occupancy classification. Figure 32 explains why appropriately clustered feature space is easier to classify compared to un-clustered feature space. The concept of using clustering to improve classification has been investigated thoroughly [122, 123]. This begs the following questions (1) How do feature clusters look like for a real SLEEPIR sensor signal? (2) Can the feature data be clustered in a meaningful way so that it facilitates classification? Figure 32 answers these questions by showing that the evaluated occupancy features, when clustered, can be easily classified into occupied and unoccupied classes. Figure 33 shows that the raw observation clusters overlap and thus have a significantly higher susceptibility to false positives and negatives.

To evaluate meaningful clusters i.e., clusters that correspond to an occupancy scenario, we utilize a clustering technique based on the K-Means algorithm. K-Means clustering is

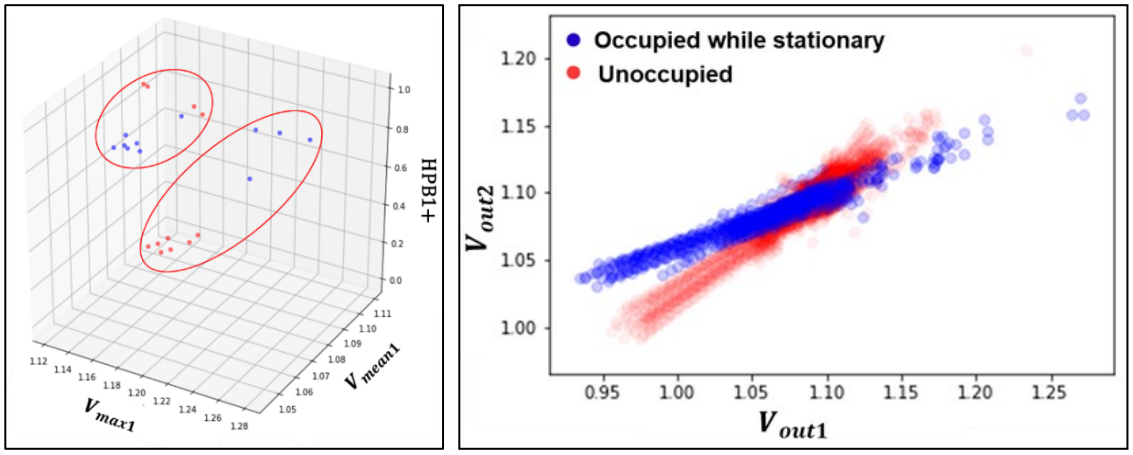

Figure 33 (Left) Clustered features for SLEEPIR sensor module 1, evaluated from the automatically labelled raw observations. (Right) Raw observations from the SLEEPIR sensor node for 24-hour period.

a method that groups data points into k clusters, with each cluster being represented by a centroid. The algorithm begins by randomly selecting k data points as initial cluster centers. It then assigns each data point to the nearest cluster center based on their similarity measured using a distance metric such as the Euclidean distance. The average of the data points within each cluster is calculated to update the centroid, and this process is repeated iteratively until the centroids stabilize.

To determine the appropriate number of clusters k, the DBI *[124]*was employed. The DBI quantifies the ratio between the scatter within clusters and the separation between clusters. By calculating the DBI for varying numbers of clusters, the number of clusters k that yield the lowest value can be identified, indicating a reasonable and meaningful clustering solution. It was proven during the experimentation that DBI enhances the probability of clusters corresponding to an occupancy scenario. It must be highlighted here that one occupancy scenario can cause multiple feature clusters.

6.1.2. Sub-classifier Architecture and Training

The node-level or sub-classifier forms the second layer of the HCS framework as shown in Figure 31. These classifiers are ODLL classifiers having training datasets that evolve as the occupancy scenarios consistently change in almost every real indoor setting. The training dataset for each of these classifiers is an assigned cluster from the context generation phase detailed in section 6.1.1. For example, while the subject sleeps the SLEEPIR observations are frequently clustered together for the collected dataset. Such clusters are shown in figure 34. Thus, these clustered observations can serve as the training dataset for a dedicated ODLL KNN classifier in this sub-classifier layer of the HCS

framework. Similarly, the subject working on a laptop produces feature clusters that are distinct from the "sleep scenario cluster". The "sleep scenario cluster" and "laptop work scenario" clusters may be close in terms of Euclidean or Cosine distance which may cause misclassifications. When both of these scenarios are evaluated for occupancy by two separate classifiers each trained on its respective cluster, the results show marked improvement. The architecture for these sub-classifiers is inspired by the similar KNN classifiers deployed in [2].

In general, KNN is a supervised classification technique that operates on the principles of nonlinear distance-based analysis. Unlike other methods, KNN doesn't involve a learning process but relies on direct classification. It requires the indexed storage of the entire training dataset.

Given we have a training dataset that falls with a cluster C_1 (x_{C_1}, y_{C_1}), and a new observation x_{new},, we can calculate the distance, denoted as d_m, between x_{new} and x_{C_1} using Equation 1:

$$d_m = \left\| x_{new} - x_{C_1} \right\| \tag{20}$$

Distance calculation typically employs the Euclidean distance measure, which is widely used. Once the distance d_m is obtained, the labels of the $\boldsymbol{k}$ training samples with the smallest distances are selected. A majority voting scheme is then applied to determine the label of the new observation. It is worth noting that as the number of existing samples in the dataset increases the computation time for assigning a new sample to a class also increases [96].

A limit is set on the total number of observations in the training dataset (as determined

by the cluster) to keep the size of the training set bounded. This is achieved by periodically removing observations that are farthest from their respective cluster centroid, determined by the Euclidean distance. To determine the optimal number of neighbors (***k***), a critical parameter for KNN inference, the Elbow search method [97] is used. This method involves periodically calculating the Within-Cluster-Sum of Squared Errors (WSS) for different values of ***k*** neighbors and evaluating the WSS. We select the value of ***k*** at which the WSS starts to diminish for the first time. In the plot of WSS versus ***k***, this is visually recognizable as an elbow. This search is performed periodically rather than for every inference.

Depending on the context, a sub-classifier is chosen which establishes the node-level occupancy based on the new observation x_{new} collected at the last timestamp. The occupancy output of the chosen sub-classifier then reaches a network-level BF occupancy detection and tracking algorithm which has already been described in detail in section 5.2.

6.1.3. Context Selector

Among the sub-classifier layer of ODLL KNN models, a suitable model needs to be selected to perform classification whenever a *context* is evaluated. This is achieved via a KNN-based Context Classifier. We must reiterate here the evaluated *context* in section 6.1.1. represents cluster(s) which are based on occupancy observation similarity. We employ another KNN classifier to establish the *context class* for any new incoming observation from the SLEEPIR sensor node. This step constitutes the top layer of the proposed HCS framework as shown in figure 31. The architecture for this classifier is similar to the architecture described in section 3.2.2 with the exception that the data points

in this KNN classifier belong to the base training dataset instead of the limited training dataset determined by the context (clusters). Thus, the context determines a sub-dataset based upon which a trained ODLL KNN sub-classifier performs occupancy classification for the new observation.

6.2. Bayes Filter-Based Occupancy Detection and Tracking

The BF occupancy detection and tracking algorithm determines the network-wide occupancy while tracking the occupancy state of the observed area i.e., the occupancy

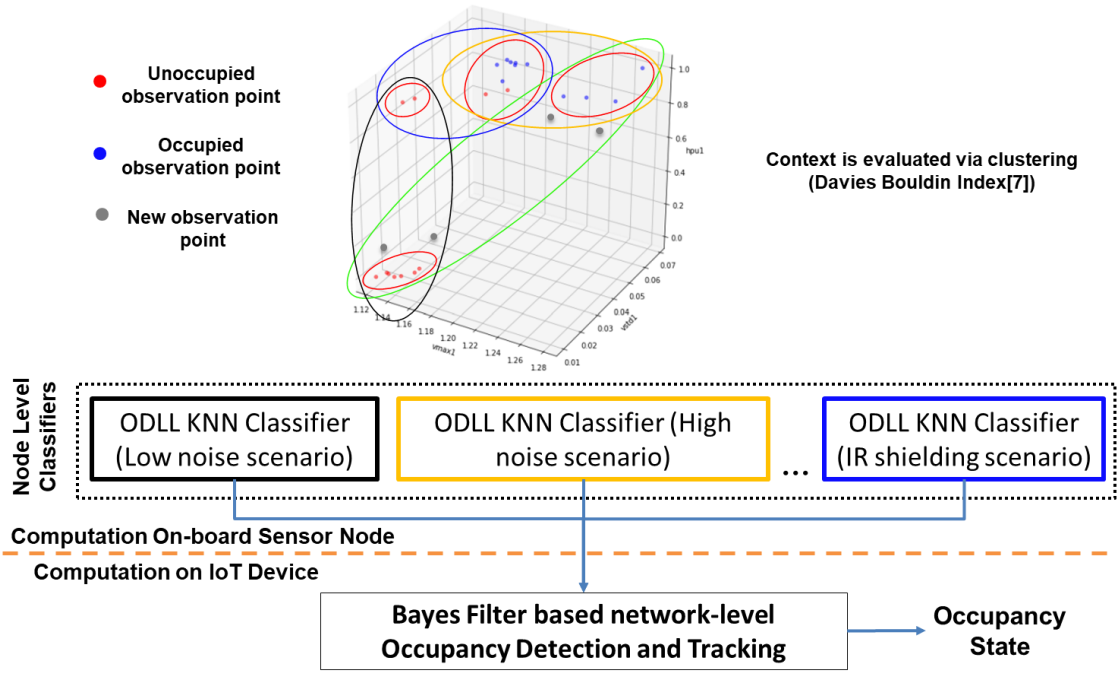

Figure 34. When feature observations (in grey) from a 30-minute window are plotted. Based on the distance between the feature observations (in grey) and the clusters present in vicinity (in red), various groups of clusters are formed (in green, orange, blue, black). The pre-trained sub-classifier for the selected cluster groups is then employed to make occupancy inference.

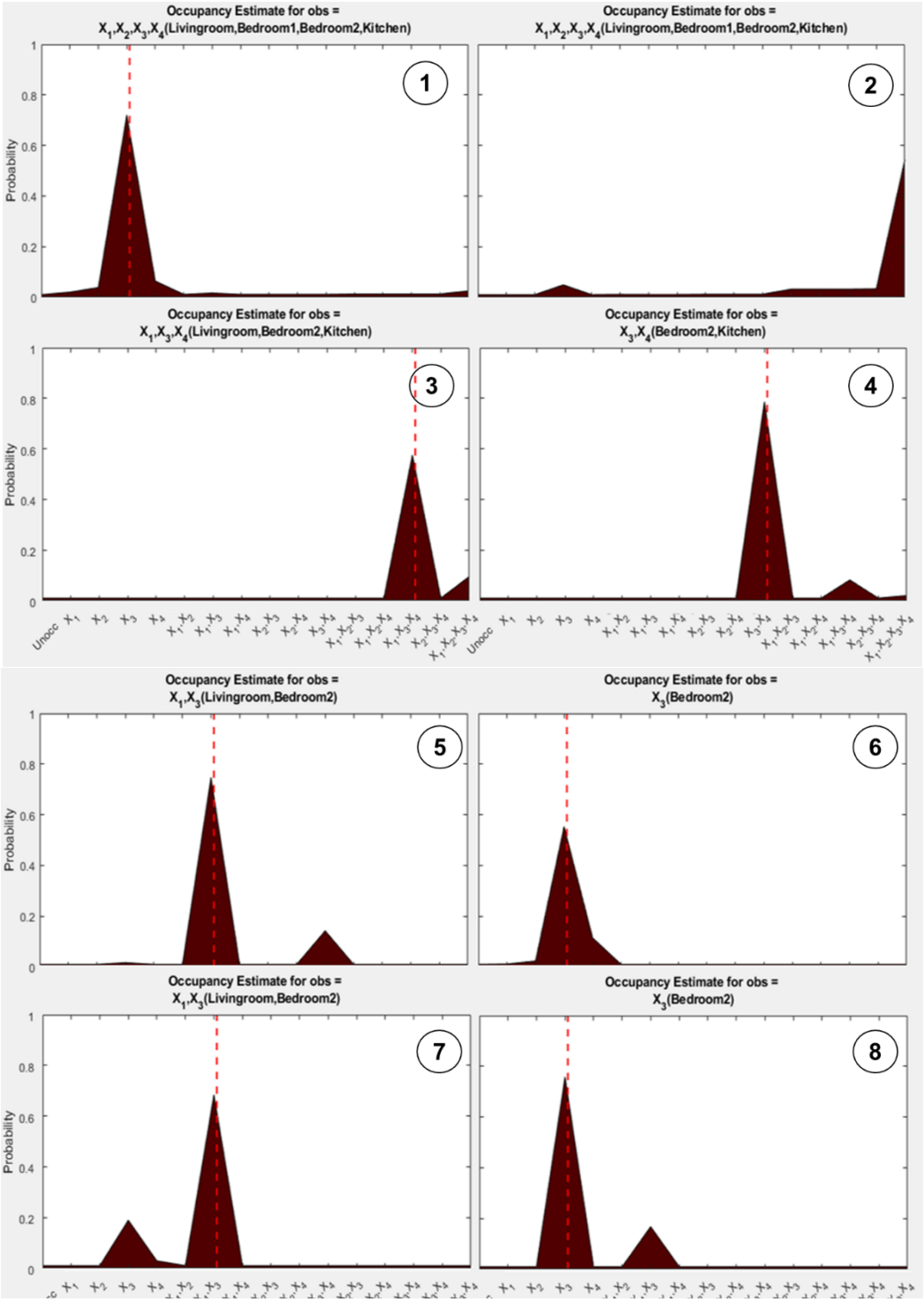

Figure 35. BF output pdf that represents the occupancy belief of the BF based occupancy tracking algorithm that evolves over time. X-axis represents all possible occupancy states while Y-axis represents the probability of occupancy for each state at a time instant.

status for each sensor node in the system. After binarizing the node-level observations using the proposed ML architecture in section 6.1, these observations are utilized to update a BF, which generates an estimate of occupancy at the network level. This BF method proposed in section 5 thus provides a real-time posterior probability density function (pdf)

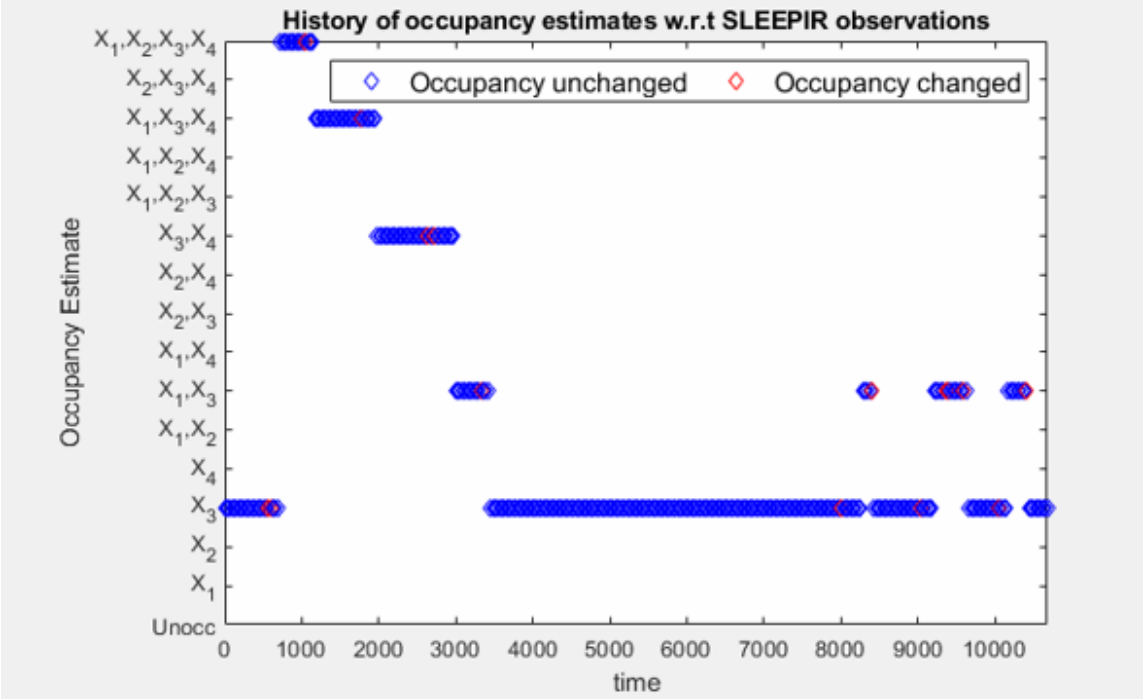

Figure 36. History of Occupancy states corresponding to BF posterior pdfs shown in figure 35.

of the state (occupancy belief) based on the available information. The BF method is considered 'optimal' because it seeks the posterior distribution that integrates and incorporates all the available information expressed as probabilities [116]. The BF-based algorithm for detecting occupancy at the network level employs a dual-stage algorithm. In the initial stage, the occupancy status of the area under surveillance is modeled as an MDP. The MDP is a representation of real-world dynamics. The MDP proposes an optimal policy - a sequence of state transitions that are needed before reaching a goal state. The goal state is the occupancy state detected by the networked SLEEPIR sensor nodes for

example $\{X^2, X^3\}$ which indicates that occupancy was detected at nodes X^2 and X^3. The MDP also needs a starting state which is the previous occupancy state detected by the BF-based method. For example, if the starting state was {Unoccupied} and the goal state was detected to be $\{X^2, X^3\}$, then MDP would propose an optimal sequence of states π that need to be navigated to reach the goal state while beginning at the start state. This suggested sequence π would also require an expected amount of time to be navigated. The work described in section 5 evaluates both the sequence of occupancy states and expected traversal time given a starting and a goal occupancy state.

In the second stage, the BF continually receives updates about occupancy status from individual SLEEPIR nodes, e.g., $\{X^2, X^3\}$. Based upon the suggested sequence of occupancy states π, the BF continually adjusts its belief based on the degree of agreement between the observed occupancy state fed via SLEEPIR node observations and the sequence π. If an incoming occupancy observation aligns well with π suggested by the MDP, it is assigned a higher likelihood compared to an observation that doesn't align well. The sensor model for the BF correlates the incoming observations to the likelihood values of the overall space being observed as occupied (or unoccupied).

The posterior probability density function (pdf) that represents the present probability for all possible occupancy states is provided in Figure 35. Figure 36 shows how the BF-based occupancy tracking algorithm's occupancy belief evolves. Figure 35 also shows the history of tracked beliefs about the occupancy state. The system output shown in the figure represents the real-time output of the proposed context-aided occupancy and tracking system.

6.3. Discussion

It must be highlighted that no prior assumptions are made about the clusters that are essentially the *context classes* as evaluated in section 6.1.1., instead, a data-driven approach i.e., K-Means clustering, is solely used to establish similar occupancy patterns. The dataset gathered for this research indicates that the *context classes* determined by the K-Means clustering algorithm loosely coincided with specified periods of a typical week e.g., weekday mornings/evenings constituted a single *context class* for which the occupant activity pattern was found to be similar. On the contrary, weekend nights and weekday nights constituted two distinct *context classes* as the occupant activities varied drastically between these two time periods.

Given the above discussion, it can be concluded that the clustering technique mentioned in section 6.1 may not necessarily produce clusters that correspond to a specific period within the week. Instead, it was found that although each cluster predominantly contained observations from a typical period in a week such as weekday mornings, similar observations from other time periods also permeated into the cluster under discussion. This can be tolerated by the node-level ODLL KNN classifiers as this does not reduce the classification accuracy in any way. For example, there may be periods of unoccupancy during weekday morning and weekend night or there may exist similar occupancy patterns during weekday morning and weekday evening due to similar activities carried out by the subject(s). Thus, similar-looking occupancy patterns within two different time frames within a week can be part of any cluster depending upon their distance to another cluster within the feature space.

One of the most critical clusters found in the collected dataset is the weekday night cluster. This cluster is unique in the sense that the subject(s) are mostly stationary(sleeping) and during the months with frequent cold nights, most parts of the subject bodies are covered with a blanket or fabric. This inhibits the emitted body IR radiation to reach the sensor and thus unique patterns are formed that are in proximity to clusters representing inoccupancy within the feature space. It would have been a particularly difficult task to perform occupancy classification for the non-clustered data for the cold nights, yet due to the clustering algorithm, both cold night features and inoccupancy features lie within the same cluster. The ODLL KNN classifier for the clustered data encounters significantly lower false negatives as shown in the results section.

6.4. Dataset and Results

The dataset described in Chapter 3 is used for the accurate evaluation of the proposed context-aided detection and tracking method. As a recap, 4 SLEEPIR sensor nodes were deployed at a residential apartment as shown in Figure 7. The apartment unit had 2 bedrooms and 2 bathrooms. The apartment covered area was 10m x 14m. Each node was installed at a height of 2.8 meters. Each node collects one observation every 30 seconds. The duration of data collection was 30 days. Webcams were used to collect ground truth data.

6.4.1. Accuracy Evaluation

As shown in table 15, the current state-of-the-art methods, namely EKF and PF [3], are outperformed by the proposed method for occupancy detection in sensor node

networks. An EKF implementation [114] was selected as a baseline for comparison due to its ability to approximate a Gaussian distribution and its resemblance to BF, which incorporates linear, quadratic, and Gaussian assumptions. In previous literature, EKF has frequently been compared to BF in terms of algorithmic efficiency [119-121].

Table XV. Detection Accuracy Comparison between Proposed Context-Aided KNN Model and Baseline Models

Date	EKF + Static LSTM Accuracy	PF + Static LSTM Accuracy	BF + Static LSTM Accuracy	BF + KNN Accuracy	Proposed Context aided KNN +BF Accuracy
15 April	77.2%	81.9%	83.1%	89.3%	95.9%
16 April	82.1%	89.2%	91.7%	94.0%	95.3%
17 April	70.5%	77.0%	81.4%	90.7%	97.6%
18 April	61.2%	88.5%	82.5%	86.8%	91.0%
19 April	59.0%	88.3%	86.8%	87.1%	95.1%
20 April	71.6%	82.5%	89.5%	91.2%	97.5%
21 April	80.2%	85.2%	91.6%	93.9%	97.8%

Table XVI. Tracking Accuracy Comparison between Proposed Context-Aided KNN Model and Baseline Models (Percentage time spend in correct states)

Date	EKF + Static LSTM Accuracy	PF + Static LSTM Accuracy	BF + Static LSTM Accuracy	BF + KNN Accuracy	Proposed Context aided KNN+ BF Accuracy
15 April	57.9%	70.5%	77.5%	86.9%	93.2%
16 April	63.5%	77.9%	87.1%	91.7%	92.4%
17 April	52.1%	69.4%	75.8%	88.0%	95.4%
18 April	48.5%	77.7%	78.5%	84.8%	85.1%
19 April	44.0%	79.7%	81.7%	86.7%	91.0%
20 April	53.2%	75.2%	84.5%	90.5%	93.9%
21 April	62.9%	78.2%	88.4%	92.2%	94.0%

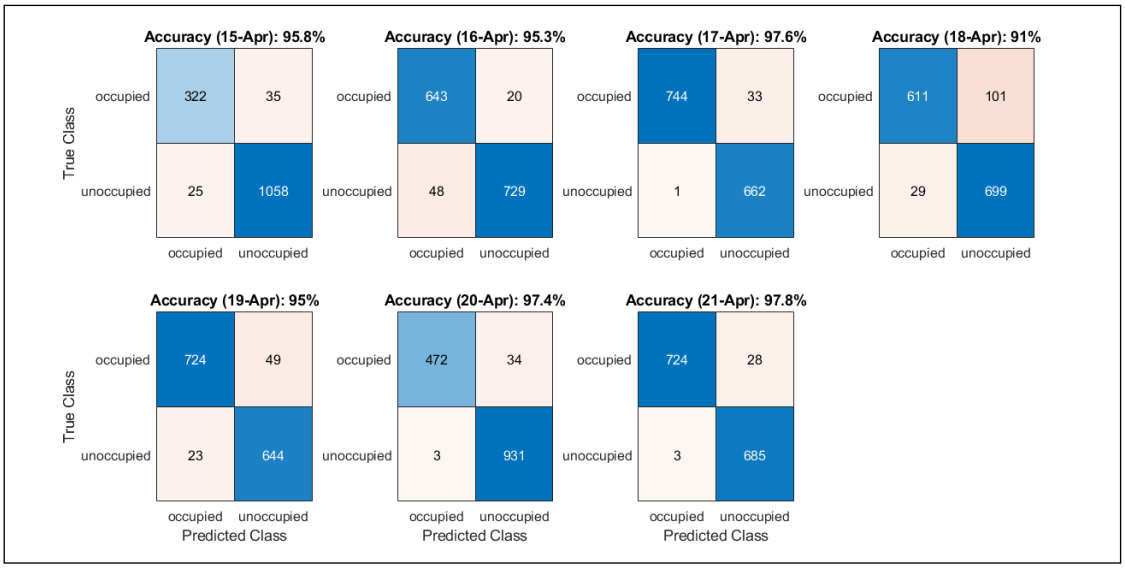

Figure 37. FPs and FNs for 7-day test data showing the performance for the proposed context aided occupancy detection framework.

The objective of this study is to demonstrate that the proposed context-aided BF method, despite its similarities with EKF, surpasses existing implementations in both accuracy and efficiency. The evaluation primarily focuses on establishing the effectiveness of the proposed method in accurately tracking occupancy at the room level. Firstly, the accuracy results that compare the occupancy detection performance using the networked SLEEPIR nodes, are presented. The results of this comparative experiment are listed in Table 15. In Table 16, the tracking performance of the above-mentioned baseline algorithms with the proposed context-aided BF tracking algorithm is compared. The proposed model error is broken down into FPs and FNs and Fig. 37 illustrates the false positives and false negatives reported by the proposed and each of the baseline methods. 77% of the dataset was used for training while the remaining 23% was used for testing. It was ensured that a continuous full week of data is tested.

6.4.2. Error Analysis

- On the 18th of April, it was observed that there was a high false negative (FN) rate across all models, with the lowest accuracy being 61.2% for the EKF + Static LSTM model and the highest being 91.0% for the Proposed Context aided KNN +BF model. This was unusual and could be attributed to the subject spending most of the day in bed. This non-typical behavior might have misled the models, causing them to incorrectly predict the absence of the subject, hence increasing the FN rate. Additionally, the presence of IR shielding was a significant factor that contributed to this anomaly in the models' prediction.
- On the 17th, 20th, and 21st of April, it is noted that there were fewer false positives (FP) than usual across all models. This means that the models performed better in accurately predicting occupancy. A key factor that helped in this improved performance is the capability of the proposed Context aided KNN +BF model to adapt well to the infrared (IR) noise caused by the solar IR in the living room and Bedroom 2. The model was able to minimize false positives, thereby increasing its accuracy, which ranged from 97.6% to 97.8% on these dates.
- The context was evaluated using 30-minute observation windows. A 30-minute window is a tunable parameter. It was found experimentally that evaluating context over less than 30 minutes resulted in non-consistent context while notching up the value to over 30 minutes led to missing certain short-term contexts like having lunch or cooking a small meal etc.

6.5. Conclusion

This research presented a context-aided occupancy detection and tracking framework for sensor node networks. The paper addressed the inherent complexity of occupancy detection due to evolving environmental and occupancy scenarios. It emphasized the challenges of collecting comprehensive training datasets encompassing all anticipated occupancy patterns and the need for model updates in dynamic scenarios. The proposed framework leveraged the bounded size of the training dataset and utilized contextual information to enhance occupancy detection accuracy. The method effectively limited the classification space through data clustering and the hierarchical classifier selection (HCS) framework, improving accuracy and execution time compared to baseline algorithms. The BF-based occupancy detection and tracking algorithm provided a robust estimation of occupancy at both network and node levels, incorporating the real-time posterior probability density function (pdf) of the occupancy state.

The paper demonstrated the effectiveness of the proposed method through a comprehensive evaluation using a dataset collected from a residential apartment. The results showcased superior performance compared to state-of-the-art methods in terms of occupancy detection accuracy and tracking precision. The lowest accuracy was recorded at 61.2% for the EKF + Static LSTM model, while the highest accuracy of 97.8% was achieved by the proposed Context aided KNN + BF model.

The contributions of this research include the novel context-aided hierarchical classification approach, the BF-based occupancy detection and tracking algorithm, and the elimination of offline training and over-the-cloud model updates. The proposed

framework offers potential applications in building automation, energy management, and occupancy-based services.

7. CONTRIBUTIONS & CONCLUSION

7.1. Contribution

The contribution of our research is summarized below.

The ODLL KNN occupancy detection algorithm contributed to our research by

- Present a node-level ODLL classifier that continuously learns evolving occupancy patterns and reduces on average 15.43% FP and 17.92% FN error generated by LSTM-based offline classifier. The primary causes of these errors are environmental IR disturbances and IR shielding.
- Devising a highly efficient algorithm that enables low-power, local periodic computation which eliminates the need for over-the-cloud ML model updates.
- The test distribution and the training distribution are kept the same by training the ML model on the same sensor data which is used for testing.

The Particle and Bayes filter-based network-level occupancy detection algorithms were able to improve the building-level occupancy detection accuracy by

- exploiting the adjacency and observation correlation between the networked nodes to reduce on average 13.70% FP and 16.31% FN error when compared to a benchmark Finite State Machine network-level estimation algorithm.
- Use cooperative, redundant, and complimentary fusion strategies to reduce the impact of sensor noise, small sensor range, FoV, and complicated occupancy scenarios.

The context-aided occupancy detection and tracking framework contributed to gaining superior occupancy detection and occupancy state tracking. More particularly,

- It reduced FP by 48.8% compared to EKF + Static LSTM method.
- It reduced FN by 26.3% compared to EKF + Static LSTM method.
- The FP and FN reduction happened due to the context isolating a subset of training data which is easier to classify.

7.2. Future Challenges

- Although stationary IR noise like space heaters or stoves are not likely to be picked up as occupancy by the auto labeling algorithm mentioned in Chapter 3 but warm objects in motion like window blinds during the day or fan motion within a space heater are still challenging as these tend to trigger voltage response in the PIR sensor. A PIR-based solution that has multiple PIR sensors with PDLC shutters is a recommended future direction. Such a solution should be capable of eliminating small IR-emanating objects by directed FoV.
- Since the ODLL dataset is constantly evolving, low power, low CPU cycle, memory constrained micro-controllers require time to collect observations, label these and then learn the occupancy model. It is a challenge to find a power-optimized hardware platform for this application.
- People counting and differentiating between an adult and a child is not possible using the networked SLEEPIR sensors as the output of these sensors is binary. Yet provided the network-level tracking data and knowledge about the spatial distance between the nodes, an inference can be made about the minimum occupancy within an observed environment.

7.3. Conclusion

This research optimizes indoor occupancy detection and tracking using networked sensor nodes. These nodes yield superior accuracy compared to standalone sensors, which often suffer from coverage gaps and inaccuracies. The presented approach employs network-level sensor fusion and occupancy estimation techniques to leverage the complementary information in time-series data. The work proposes methods for various applications including occupant comfort, health, energy, and space utilization, building design, and safety. Key elements of the proposed system include diverse sensor technologies, data fusion techniques, machine learning algorithms, and estimation strategies. Major contributions of this work include an on-device lifelong classifier that learns occupancy patterns, a network-level algorithm using inter-node spatial information, and a high-resolution system that filters unreachable occupancy states.

REFERENCES

[1] M. E.-u.-d. Z. C. Q. S. L. W. Y. Wang, "Bayes Filter-based Occupancy Detection Using Networked SLEEPIR Sensors," *IEEE Sensors Journal,* 2023, doi: 10.1109/JSEN.2023.3304372.

[2] M. Emad-Ud-Din and Y. Wang, "Promoting Occupancy Detection Accuracy Using On-Device Lifelong Learning," *IEEE Sensors Journal,* vol. 23, no. 9, pp. 9595-9606, 2023, doi: 10.1109/JSEN.2023.3260062.

[3] M. Emad-ud-din, Z. Chen, L. Wu, Q. Shen, and Y. Wang, "Indoor Occupancy Estimation using Particle Filter and SLEEPIR Sensor System," *IEEE Sensors Journal,* pp. 1-1, 2022, doi: 10.1109/JSEN.2022.3192270.

[4] M. Emad-Ud-Din and Y. Wang, "Indoor Occupancy Sensing via Networked Nodes (2012–2022): A Review," *Future Internet,* vol. 15, no. 3, p. 116, 2023, doi: 10.3390/fi15030116.

[5] B. Dong *et al.*, "Sensing and Data Acquisition," in *Exploring Occupant Behavior in Buildings*: Springer International Publishing, 2018, pp. 77-105.

[6] B. Feagin Jr, M. E. Poplawski, and J. T. Day, "A Review of Existing Test Methods for Occupancy Sensors," United States: N., 2020. Web. doi:10.2172/1668746, [Online]. Available: https://www.osti.gov/biblio/1668746

[7] L. Wu, F. Gou, S.-T. Wu, and Y. Wang, "SLEEPIR: Synchronized low-energy electronically chopped PIR sensor for true presence detection," *IEEE Sensors Letters,* vol. 4, no. 3, pp. 1-4, 2020.

[8] S.-M. Jeong *et al.*, "Development of a wearable infrared shield based on a polyurethane–antimony tin oxide composite fiber," *NPG Asia Materials,* vol. 12, p. 32, 2020, doi: 10.1038/s41427-020-0213-z.

[9] *Occupancy Motion Sensors Standard*, N. E. M. Association, Oct, 2021 2021. [Online]. Available: www.nema.org

[10] L. Wu and Y. Wang, "Stationary and moving occupancy detection using the SLEEPIR sensor module and machine learning," *IEEE Sensors Journal,* 2021.

[11] B. Ai, Z. Fan, and R. X. Gao, "Occupancy estimation for smart buildings by an auto-regressive hidden Markov model," in *2014 American Control Conference*, 2014: IEEE, doi: 10.1109/acc.2014.6859372.

[12] A. Abdelgawad and M. Bayoumi, "Data Fusion in WSN," in *Resource-Aware Data Fusion Algorithms for Wireless Sensor Networks*: Springer US, 2012, pp. 17-35.

[13] L. Rueda, K. Agbossou, A. Cardenas, N. Henao, and S. Kelouwani, "A comprehensive review of approaches to building occupancy detection," *Building and Environment,* vol. 180, p. 106966, 2020, doi: 10.1016/j.buildenv.2020.106966.

[14] I. Rodriguez, M. Lauridsen, G. Vasluianu, A. N. Poulsen, and P. Mogensen, "The Gigantium Smart City Living Lab: A Multi-Arena LoRa-based Testbed," in *2018 15th International Symposium on Wireless Communication Systems (ISWCS)*, 2018: IEEE, doi: 10.1109/iswcs.2018.8491077.

[15] H. Liu, Y. Wang, K. Wang, and H. Lin, "Turning a pyroelectric infrared motion sensor into a high-accuracy presence detector by using a narrow semi-transparent chopper," *Applied Physics Letters,* vol. 111, no. 24, p. 243901, 2017.

[16] L. Wu, Y. Wang, and H. Liu, "Occupancy detection and localization by monitoring nonlinear energy flow of a shuttered passive infrared sensor," *IEEE Sensors Journal,* vol. 18, no. 21, pp. 8656-8666, 2018.

[17] L. Wu and Y. Wang, "A low-power electric-mechanical driving approach for true occupancy detection using a shuttered passive infrared sensor," *IEEE Sensors Journal,* vol. 19, no. 1, pp. 47-57, 2018.

[18] L. Wu and Y. Wang, "Performance Optimization of the SLEEPIR Sensor Towards Indoor Stationary Occupancy Detection," *IEEE Sensors Journal,* vol. 21, no. 21, pp. 23776-23786, 2021, doi: 10.1109/JSEN.2021.3111877.

[19] T. Pedersen, K. Nielsen, and S. Petersen, "Method for room occupancy detection based on trajectory of indoor climate sensor data," *Building and Environment,* vol. 115, 2017, doi: 10.1016/j.buildenv.2017.01.023.

[20] Z. Chen, "Data Processing for Device-Free Fine-Grained Occupancy Sensing using Infrared Sensors," Ph.D., Texas A&M University, Ann Arbor, 29241828, 2021.

[21] C. Wang, J. Jiang, T. Roth, C. Nguyen, Y. Liu, and H. Lee, "Integrated sensor data processing for occupancy detection in residential buildings," *Energy and Buildings,* vol. 237, p. 110810, 2021, doi: doi.org/10.1016/j.enbuild.2021.110810.

[22] N. S. Fayed, M. M. Elmogy, A. Atwan, and E. El-Daydamony, "Efficient Occupancy Detection System Based on Neutrosophic Weighted Sensors Data Fusion," *IEEE Access,* vol. 10, pp. 13400-13427, 2022, doi: 10.1109/ACCESS.2022.3146346.

[23] J. Lin, L. Zhu, W.-M. Chen, W.-C. Wang, C. Gan, and S. Han, *On-Device Training Under 256KB Memory*. *arXiv preprint* arXiv:2206.15472, 2022.

[24] S. J. Russell and P. Norvig, *Artificial intelligence : a modern approach*, Fourth edition. ed. (Pearson series in artificial intelligence). Hoboken: Pearson, 2021.

[25] W. Zhang, Y. Wu, and J. K. Calautit, "A review on occupancy prediction through machine learning for enhancing energy efficiency, air quality and thermal comfort in the built environment," *Renewable and Sustainable Energy Reviews,* vol. 167, p. 112704, 2022, doi: 10.1016/j.rser.2022.112704.

[26] Y. Bae *et al.*, "Sensor impacts on building and HVAC controls: A critical review for building energy performance," *Advances in Applied Energy,* vol. 4, p. 100068, 2021, doi: 10.1016/j.adapen.2021.100068.

[27] V. L. Erickson and A. E. Cerpa, "Thermovote," in *Proceedings of the Fourth ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings - BuildSys '12*, 2012: ACM Press, doi: 10.1145/2422531.2422534.

[28] C. Karmann, S. Schiavon, L. T. Graham, P. Raftery, and F. Bauman, "Comparing temperature and acoustic satisfaction in 60 radiant and all-air buildings," *Building and Environment,* vol. 126, pp. 431-441, 2017, doi: 10.1016/j.buildenv.2017.10.024.

[29] L. Zhang *et al.*, "Sensor impact evaluation and verification for fault detection and diagnostics in building energy systems: A review," *Advances in Applied Energy,* vol. 3, p. 100055, 2021, doi: 10.1016/j.adapen.2021.100055.

[30] R. Adeogun, I. Rodriguez, M. Razzaghpour, G. Berardinelli, P. H. Christensen, and P. E. Mogensen, "Indoor Occupancy Detection and Estimation using Machine Learning and Measurements from an IoT LoRa-based Monitoring System," in *2019 Global IoT Summit (GIoTS)*, 17-21 June 2019, pp. 1-5, doi: 10.1109/GIOTS.2019.8766374.

[31] D. Giri, S. Shreya, P. Kumari, and R. Yadav, "Indoor human occupancy detection using Machine Learning classification algorithms & their comparison," *IOP Conference Series: Materials Science and Engineering,* vol. 1110, no. 1, p. 012020, 2021, doi: 10.1088/1757-899X/1110/1/012020.

[32] M. Maaspuro, "Infrared occupancy detection technologies in building automation - A review," *ARPN Journal of Engineering and Applied Sciences,* vol. 13, pp. 8055-8068, 2018.

[33] B. Nastasi, N. Markovska, T. Puksec, N. Duić, and A. Foley, "Renewable and sustainable energy challenges to face for the achievement of Sustainable Development Goals," *Renewable and Sustainable Energy Reviews,* vol. 157, p. 112071, 2022, doi: 10.1016/j.rser.2022.112071.

[34] N. Cao, J. Ting, S. Sen, and A. Raychowdhury, "Smart Sensing for HVAC Control: Collaborative Intelligence in Optical and IR Cameras," *IEEE Transactions on Industrial Electronics,* vol. 65, no. 12, pp. 9785-9794, 2018, doi: 10.1109/TIE.2018.2818665.

[35] Z. Chen, M. K. Masood, and Y. C. Soh, "A fusion framework for occupancy estimation in office buildings based on environmental sensor data," *Energy and Buildings,* vol. 133, pp. 790-798, 2016, doi: 10.1016/j.enbuild.2016.10.030.

[36] T. Teixeira, G. Dublon, and A. Savvides, "A Survey of Human-Sensing: Methods for Detecting Presence, Count, Location, Track, and Identity," *ACM Computing Surveys,* vol. 5, p. 59, 2010.

[37] T. A. Nguyen and M. Aiello, "Energy intelligent buildings based on user activity: A survey," *Energy and Buildings,* vol. 56, pp. 244-257, 2013, doi: 10.1016/j.enbuild.2012.09.005.

[38] S. Y. Tan, M. Jacoby, H. Saha, A. Florita, G. Henze, and S. Sarkar, "Multimodal sensor fusion framework for residential building occupancy detection," *Energy and Buildings,* vol. 258, p. 111828, 2022, doi: 10.1016/j.enbuild.2021.111828.

[39] K. Lam *et al.*, "Occupancy detection through an extensive environmental sensor network in an open-plan office building," *IBPSA 2009 - International Building Performance Simulation Association 2009,* 2009.

[40] Z. Yang, N. Li, B. Becerik-Gerber, and M. Orosz, "A Multi-Sensor Based Occupancy Estimation Model for Supporting Demand Driven HVAC Operations," *Proceedings of the 2012 Symposium on Simulation for Architecture and Urban Design (SimAUD '12)*: Society for Computer Simulation International, San Diego, CA, USA, Article 2, 1–8.

[41] T. Ekwevugbe, N. Brown, and D. Fan, "A design model for building occupancy detection using sensor fusion," in *2012 6th IEEE International Conference on Digital Ecosystems and Technologies (DEST)*, 2012: IEEE, doi: 10.1109/dest.2012.6227924.

[42] J. Wang, J. Huang, Z. Feng, S.-J. Cao, and F. Haghighat, "Occupant-density-detection based energy efficient ventilation system: Prevention of infection transmission," *Energy and Buildings,* vol. 240, p. 110883, 2021, doi: 10.1016/j.enbuild.2021.110883.

[43] A. Hammad, "Improving Indoor Security Surveillance by Fusing Data from BIM, UWB and Video," *Proceedings of the 30th International Symposium on Automation and Robotics in Construction and Mining (ISARC 2013): Building the Future in Automation and Robotics*, 2013.

[44] Z. S. Sabri and Z. Li, "Low-cost intelligent surveillance system based on fast CNN," (in eng), *PeerJ Comput Sci,* vol. 7, p. e402, 2021, doi: 10.7717/peerj-cs.402.

[45] R. Tomastik, S. Narayanan, A. Banaszuk, and S. Meyn, "Model-Based Real-Time Estimation of Building Occupancy During Emergency Egress," Pedestrian and Evacuation Dynamics 2008. Springer, Berlin, Heidelberg. 2008, pp. 215-224.

[46] R. Tomastik, S. Narayanan, A. Banaszuk, and S. Meyn, "Model-Based Real-Time Estimation of Building Occupancy During Emergency Egress," in *Pedestrian and Evacuation Dynamics 2008*, Berlin, Heidelberg, W. W. F. Klingsch, C. Rogsch, A. Schadschneider, and M. Schreckenberg, Eds., 2010: Springer Berlin Heidelberg, pp. 215-224.

[47] S. J. Fong, C. M. Bhatt, D. G. Korzun, S. Yang, and L. Yang, "Internet of Breath (IoB): Integrative Indoor Gas Sensor Applications for Emergency Control and Occupancy Detection", in *Advances in Intelligent Systems and Computing, Advances in Intelligent Systems and Computing*, 2019, pp. 342–359. doi: 10.1007/978-3-319-91337-7_32.

[48] B. W. Hobson, D. Lowcay, H. B. Gunay, A. Ashouri, and G. R. Newsham, "Opportunistic occupancy-count estimation using sensor fusion: A case study," *Building and Environment,* vol. 159, p. 106154, 2019/07/15/ 2019, doi: 10.1016/j.buildenv.2019.05.032.

[49] Z. Yang, N. Li, B. Becerik-Gerber, and M. Orosz, "A systematic approach to occupancy modeling in ambient sensor-rich buildings," *SIMULATION,* vol. 90, no. 8, pp. 960-977, 2013, doi: 10.1177/0037549713489918.

[50] C. Duarte, K. Wymelenberg, and C. Rieger, "Revealing occupancy patterns in an office building through the use of occupancy sensor data," *Energy and Buildings,* vol. 67, pp. 587-595, 2013, doi: 10.1016/j.enbuild.2013.08.062.

[51] Y. Jin, D. Yan, A. Chong, B. Dong, and J. An, "Building occupancy forecasting: A systematical and critical review," *Energy and Buildings,* vol. 251, p. 111345, 2021, doi: 10.1016/j.enbuild.2021.111345.

[52] Z. Y. Wu, W. Lv, and K. Yu, "A Framework of Intelligent Evacuation Guidance System for Large Building," in *Proceedings of the 2016 5th International Conference on Civil, Architectural and Hydraulic Engineering (ICCAHE 2016)*, 2016: Atlantis Press, doi: 10.2991/iccahe-16.2016.111.

[53] D. K. Tiller, X. Guo, G. P. Henze, and C. E. Waters, "Validating the application of occupancy sensor networks for lighting control," *Lighting Research & Technology,* vol. 42, no. 4, pp. 399-414, 2010, doi: 10.1177/1477153510375524.

[54] D. Yang, B. Xu, K. Rao, and W. Sheng, "Passive Infrared (PIR)-Based Indoor Position Tracking for Smart Homes Using Accessibility Maps and A-Star Algorithm,", *Sensors (Basel),* vol. 18, no. 2, 2018, doi: 10.3390/s18020332.

[55] B. Gunay, Z. Nagy, C. Miller, M. Ouf, and B. Dong, "Using Occupant-Centric Control for Commercial HVAC Systems,", *ASHRAE Journal,* Article vol. 63, p. 30+, 2021 2021.

[56] C. E. Commission, "2022 Building Energy Efficiency Standards for Residential and Nonresidential Buildings: For the 2022 Building Energy Efficiency Standards Title 24, Part 6, and Associated Administrative Regulations in Part 1" (Building Energy Efficiency Standards - Title 24). 2022.

[57] C. International Code, "International building code," 2021. [Online]. Available: https://search.library.wisc.edu/catalog/999907647602121.

[58] *Standard for Health Care Facilities*, NFPA 99, N. F. P. Association, 2021.

[59] I. C. Council, *International Energy Conservation Code*. Country Club Hills, IL, 2021.

[60] B. George, H. Zangl, T. Bretterklieber, and G. Brasseur, "Seat Occupancy Detection Based on Capacitive Sensing," *IEEE Transactions on Instrumentation and Measurement,* vol. 58, no. 5, pp. 1487-1494, 2009, doi: 10.1109/tim.2009.2009411.

[61] Y. Zhang and K. Rasmussen, "Detection of Electromagnetic Interference Attacks on Sensor Systems," in *2020 IEEE Symposium on Security and Privacy (SP)*, 2020: IEEE, doi: 10.1109/sp40000.2020.00001.

[62] D. L. Tryon, "Bayes' Network and Smart Sensors – Occupancy Detection," Ph.D., The University of Nebraska - Lincoln, Ann Arbor, 28258302, 2020.

[63] A. P. Singh, V. Jain, S. Chaudhari, F. A. Kraemer, S. Werner, and V. Garg, "Machine Learning-Based Occupancy Estimation Using Multivariate Sensor Nodes," in *2018 IEEE Globecom Workshops (GC Wkshps)*, 9-13 Dec. 2018 2018, pp. 1-6, doi: 10.1109/GLOCOMW.2018.8644432.

[64] E. Hailemariam, R. Goldstein, R. Attar, and 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 (SimAUD '11)*. Society for Computer Simulation International, San Diego, CA, USA, 141–148.

[65] T. Ekwevugbe, N. Brown, V. Pakka, and D. Fan, "Real-time building occupancy sensing using neural-network based sensor network," in *2013 7th IEEE International Conference on Digital Ecosystems and Technologies (DEST)*, 2013: IEEE, doi: 10.1109/dest.2013.6611339.

[66] S. H. Kim and H. J. Moon, "Case study of an advanced integrated comfort control algorithm with cooling, ventilation, and humidification systems based on occupancy status," *Building and Environment,* vol. 133, pp. 246-264, 2018, doi: 10.1016/j.buildenv.2017.12.010.

[67] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing," *Proceedings of the IEEE,* vol. 107, no. 8, pp. 1738-1762, 2019, doi: 10.1109/JPROC.2019.2918951.

[68] S. Azizi, R. Rabiee, G. Nair, and T. Olofsson, "Effects of Positioning of Multi-Sensor Devices on Occupancy and Indoor Environmental Monitoring in Single-Occupant Offices," *Energies,* vol. 14, no. 19, p. 6296, 2021, doi: 10.3390/en14196296.

[69] A. Ebadat, G. Bottegal, D. Varagnolo, B. Wahlberg, H. Hjalmarsson, and K. H. Johansson, "Blind identification strategies for room occupancy estimation," in *2015 European Control Conference (ECC)*, 15-17 July 2015, pp. 1315-1320, doi: 10.1109/ECC.2015.7330720.

[70] V. Becker and W. Kleiminger, "Exploring zero-training algorithms for occupancy detection based on smart meter measurements," *Computer Science - Research and Development,* vol. 33, no. 1, pp. 25-36, 2018/02/01 2018, doi: 10.1007/s00450-017-0344-9.

[71] M. Jin, R. Jia, Z. Kang, I. Konstantakopoulos, and C. Spanos, "PresenceSense: Zero-training Algorithm for Individual Presence Detection based on Power Monitoring," *BuildSys 2014 - Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings* , 2014, doi: 10.1145/2674061.2674073.

[72] C. P. Filho *et al.*, "A Systematic Literature Review on Distributed Machine Learning in Edge Computing," *Sensors,* vol. 22, no. 7, p. 2665, 2022, doi: 10.3390/s22072665.

[73] H. Cai, C. Gan, L. Zhu, and S. Han, "TinyTL: Reduce Memory, Not Parameters for Efficient On-Device Learning," in *Neural Information Processing Systems*, vol. 33, pp. 11285-11297: Curran Associates Inc., Red Hook, NY, USA, 2020.

[74] T. Hayes and C. Kanan, "Online Continual Learning for Embedded Devices," *arXiv preprint* arXiv: 2203.10681, 2022.

[75] S. Petersen, T. H. Pedersen, K. U. Nielsen, and M. D. Knudsen, "Establishing an image-based ground truth for validation of sensor data-based room occupancy detection," *Energy and Buildings,* vol. 130, pp. 787-793, 2016, doi: 10.1016/j.enbuild.2016.09.009.

[76] S. Ali and N. Bouguila, "Towards scalable deployment of Hidden Markov models in occupancy estimation: A novel methodology applied to the study case of occupancy detection," *Energy and Buildings,* vol. 254, p. 111594, 2022, doi: 10.1016/j.enbuild.2021.111594.

[77] C. Geng, S.-J. Huang, and S. Chen, "Recent Advances in Open Set Recognition: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 43, no. 10, pp. 3614-3631, 2021, doi: 10.1109/tpami.2020.2981604.

[78] A. Jha, M. Dave, and S. Madan, "Comparison of Binary Class and Multi-Class Classifier Using Different Data Mining Classification Techniques," *SSRN Electronic Journal,* 2019, doi: 10.2139/ssrn.3464211.

[79] D. M. Sofos, "Saving Energy Nationwide in Structures with Occupancy Recognition," 2022. [Online]. Available: https://arpa-e.energy.gov/?q=arpa-e-programs/sensor

[80] C. Wang, W. Du, Z. Zhu, and Z. Yue, "The real-time big data processing method based on LSTM or GRU for the smart job shop production process," *Journal of Algorithms & Computational Technology,* vol. 14, 2020, doi: 10.1177/1748302620962390.

[81] J. Schmidhuber, "A fixed size storage O(n3) time complexity learning algorithm for fully recurrent continually running networks," *Neural Comput.,* vol. 4, no. 2, pp. 243–248, 1992, doi: 10.1162/neco.1992.4.2.243.

[82] D. K. Sari, D. P. Wulandari, and Y. K. Suprapto, "Training Performance of Recurrent Neural Network using RTRL and BPTT for Gamelan Onset Detection," *Journal of Physics: Conference Series,* vol. 1201, no. 1, 2019, doi: 10.1088/1742-6596/1201/1/012046.

[83] C. L. Giles and C. W. Omlin, "Pruning recurrent neural networks for improved generalization performance," *IEEE Transactions on Neural Networks,* vol. 5, no. 5, pp. 848-851, 1994, doi: 10.1109/72.317740.

[84] B. Jacob et al., "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," *arXiv preprint* arXiv: 1712.05877, 2017.

[85] Y. Lin, S. Han, H. Mao, Y. Wang, and W. Dally, "Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training," *arXiv preprint* arXiv:1712.01887, 2017.

[86] A. Kusupati, M. Singh, K. Bhatia, A. Kumar, P. Jain, and M. Varma, "FastGRNN: A Fast, Accurate, Stable and Tiny Kilobyte Sized Gated Recurrent Neural Network," In *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NIPS'18)*. Curran Associates Inc., Red Hook, NY, USA, 9031–9042.

[87] H. Cai *et al.*, "Enable Deep Learning on Mobile Devices: Methods, Systems, and Applications," *ACM Transactions on Design Automation of Electronic Systems,* vol. 27, no. 3, pp. 1-50, 2022-05-31 2022, doi: 10.1145/3486618.

[88] L. Wu, Z. Chen, and Y. Wang, "Occupancy Detection Using a Temperature-Sensitive Adaptive Algorithm," *IEEE Sensors Letters,* vol. 5, no. 12, pp. 1-10, 2021, doi: 10.1109/LSENS.2021.3126285.

[89] M. E. Zhangjie Chen, Libo Wu, Qijie Shen, Ya Wang, "Indoor Occupancy Estimation using Particle Filter and SLEEPIR Sensor System," *IEEE Sensors Journal,* 2022.

[90] B. Dong and B. Andrews, "Sensor-based occupancy behavioral pattern recognition for energy and comfort management in intelligent buildings," in *11th International IBPSA Conference - Building Simulation BS 2009*, 2009, pp. 1444-1451.

[91] L. M. Candanedo and V. Feldheim, "Accurate occupancy detection of an office room from light, temperature, humidity and CO2 measurements using statistical

learning models," *Energy and Buildings,* vol. 112, pp. 28-39, 2016/01/15/ 2016, doi: 10.1016/j.enbuild.2015.11.071.

[92] W. Wang, J. Chen, and T. Hong, "Occupancy prediction through machine learning and data fusion of environmental sensing and Wi-Fi sensing in buildings," *Automation in Construction,* vol. 94, pp. 233-243, 2018/10/01/ 2018, doi: 10.1016/j.autcon.2018.07.007.

[93] Z. Chen, Y. Wang, and H. Liu, "Unobtrusive Sensor-Based Occupancy Facing Direction Detection and Tracking Using Advanced Machine Learning Algorithms," *IEEE Sensors Journal,* vol. 18, no. 15, pp. 6360-6368, 2018, doi: 10.1109/JSEN.2018.2844252.

[94] Y. Yang, J. Hao, J. Luo, and S. J. Pan, "CeilingSee: Device-free occupancy inference through lighting infrastructure based LED sensing," in *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 13-17 March 2017 2017, pp. 247-256, doi: 10.1109/PERCOM.2017.7917871.

[95] M. Kotaru, K. Joshi, D. Bharadia, and S. Katti, "SpotFi," *ACM SIGCOMM Computer Communication Review,* vol. 45, pp. 269-282, 08/17 2015, doi: 10.1145/2829988.2787487.

[96] S. K. Trisal and A. Kaul, "K-RCC: A novel approach to reduce the computational complexity of KNN algorithm for detecting human behavior on social networks," *Journal of Intelligent & Fuzzy Systems,* vol. 36, pp. 5475-5497, 2019, doi: 10.3233/JIFS-181336.

[97] D. Marutho, S. H. Handaka, E. Wijaya, and Muljono, "The Determination of Cluster Number at k-Mean Using Elbow Method and Purity Evaluation on Headline News," in *2018 International Seminar on Application for Technology of Information and Communication*, 21-22 Sept. 2018 2018, pp. 533-538, doi: 10.1109/ISEMANTIC.2018.8549751.

[98] L. van der Maaten and G. Hinton, "Visualizing Data using t-SNE," *Journal of Machine Learning Research,* vol. 9, no. 86, pp. 2579-2605, 2008 2008.

[99] S. Zhang, K. Liu, Y. Zhang, and Y. Wang, "A Coarse Fingerprint-Assisted Multiple Target Indoor Device-Free Localization with Visible Light Sensing," *IEEE Sensors Journal,* pp. 1-1, 2021, doi: 10.1109/JSEN.2021.3130711.

[100] L. Wu and Y. Wang, "Compressive Sensing Based Indoor Occupancy Positioning Using a Single Thermopile Point Detector With a Coded Binary Mask," *IEEE Sensors Letters,* vol. 3, no. 12, pp. 1-4, 2019, doi: 10.1109/LSENS.2019.2955125.

[101] Z. Chen and Y. Wang, "Infrared–ultrasonic sensor fusion for support vector machine–based fall detection," *Journal of Intelligent Material Systems and Structures,* vol. 29, no. 9, pp. 2027-2039, 2018/05/01 2018, doi: 10.1177/1045389X18758183.

[102] R. Hua and Y. Wang, "Monitoring Insole (MONI): A Low Power Solution Toward Daily Gait Monitoring and Analysis," *IEEE Sensors Journal,* vol. 19, no. 15, pp. 6410-6420, 2019, doi: 10.1109/JSEN.2019.2910105.

[103] Z. Chen and Y. Wang, "Remote Recognition of In-Bed Postures Using a Thermopile Array Sensor With Machine Learning," *IEEE Sensors Journal,* vol. 21, no. 9, pp. 10428-10436, 2021, doi: 10.1109/JSEN.2021.3059681.

[104] J. Andrews, M. Kowsika, A. Vakil, and J. Li, "A Motion Induced Passive Infrared (PIR) Sensor for Stationary Human Occupancy Detection," in *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, 20-23 April 2020 2020, pp. 1295-1304, doi: 10.1109/PLANS46316.2020.9109909.

[105] A. D. Shetty, Disha, B. S, and K. S, "Detection and tracking of a human using the infrared thermopile array sensor — "Grid-EYE"," in *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, 6-7 July 2017 2017, pp. 1490-1495, doi: 10.1109/ICICICT1.2017.8342790.

[106] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," *arXiv preprint arXiv:1506.00019,* 2015.

[107] J. Suto, S. Oniga, and P. P. Sitar, "Feature analysis to human activity recognition," *International Journal of Computers Communications & Control,* vol. 12, no. 1, pp. 116-130, 2016.

[108] M. Z. Alom, A. T. Moody, N. Maruyama, B. C. Van Essen, and T. M. Taha, "Effective quantization approaches for recurrent neural networks," in *2018 International Joint Conference on Neural Networks (IJCNN)*, 2018: IEEE, pp. 1-8.

[109] N. Y. Hammerla, S. Halloran, and T. Plötz, "Deep, convolutional, and recurrent models for human activity recognition using wearables," *arXiv preprint arXiv:1604.08880,* 2016.

[110] J. Scott *et al.*, "PreHeat: controlling home heating using occupancy prediction," in *Proceedings of the 13th international conference on Ubiquitous computing*, 2011, pp. 281-290.

[111] A. Bhattacharyya, "On a measure of divergence between two statistical populations defined by their probability distributions," *Bull. Calcutta Math. Soc.,* vol. 35, pp. 99-109, 1943.

[112] L. Wu and Y. Wang, "True presence detection via passive infrared sensor network using liquid crystal infrared shutters," in *Proceedings of the ASME 2020 Conference on Smart Materials, Adaptive Structures and Intelligent Systems*, Virtual, Online. September 15, 2020.

[113] J. Schiff and K. Goldberg, "Automated intruder tracking using particle filtering and a network of binary motion sensors," in *2006 IEEE International Conference on Automation Science and Engineering*, 2006: IEEE, pp. 580-587.

[114] T. Miyazaki and Y. Kasama, "Multiple human tracking using binary infrared sensors," *Sensors,* vol. 15, no. 6, pp. 13459-13476, 2015.

[115] R. Hua and Y. Wang, "Robust Foot Motion Recognition Using Stride Detection and Weak Supervision-Based Fast Labeling," *IEEE Sensors Journal,* vol. 21, no. 14, pp. 16245-16255, 2021, doi: 10.1109/JSEN.2021.3075151.

[116] Z. Chen, "Bayesian Filtering: From Kalman Filters to Particle Filters, and Beyond," *Statistics,* vol. 182, 01/01 2003, doi: 10.1080/02331880309257.

[117] I. Bradley and R. L. Meek, *Matrices and society: matrix algebra and its applications in the social sciences*. Princeton University Press, 2014.

[118] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[119] N. Bellotto and H. Hu, "Computationally efficient solutions for tracking people with a mobile robot: an experimental evaluation of Bayesian filters," *Autonomous Robots,* vol. 28, no. 4, pp. 425-438, 2010/05/01 2010, doi: 10.1007/s10514-009-9167-2.

[120] M. F. Bugallo, S. Xu, and P. M. Djurić, "Performance comparison of EKF and particle filtering methods for maneuvering targets," *Digital Signal Processing,* vol. 17, no. 4, pp. 774-786, 2007/07/01/ 2007, doi: 10.1016/j.dsp.2006.10.001.

[121] C. H. Tong and T. D. Barfoot, "A Comparison of the EKF, SPKF, and the Bayes Filter for Landmark-Based Localization," in *2010 Canadian Conference on Computer and Robot Vision*, 31 May-2 June 2010 2010, pp. 199-206, doi: 10.1109/CRV.2010.33.

[122] M. Piernik and T. Morzy, "A study on using data clustering for feature extraction to improve the quality of classification," *Knowledge and Information Systems,* vol. 63, no. 7, pp. 1771-1805, 2021-07-01 2021, doi: 10.1007/s10115-021-01572-6.

[123] H. Wecker, A. Friedrich, and H. Adel, "ClusterDataSplit: Exploring Challenging Clustering-Based Data Splits for Model Performance Evaluation," Online, November 2020: Association for Computational Linguistics, in Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems, pp. 155-163, doi: 10.18653/v1/2020.eval4nlp-1.15

[124] D. L. Davies and D. W. Bouldin, "A Cluster Separation Measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. PAMI-1, no. 2, pp. 224-227, 1979, doi: 10.1109/TPAMI.1979.4766909.