Multimodal Sensing for Robust and Energy-Efficient Context Detection with Smart Mobile Devices

Valentin Radu

Doctor of Philosophy

Institute of Computing Systems Architecture
School of Informatics
University of Edinburgh

2017

Abstract

Adoption of smart mobile devices (smartphones, wearables, etc.) is rapidly growing. There are already over 2 billion smartphone users worldwide [1] and the percentage of smartphone users is expected to be over 50% in the next five years [2]. These devices feature rich sensing capabilities which allow inferences about mobile device user's surroundings and behavior. Multiple and diverse sensors common on such mobile devices facilitate observing the environment from different perspectives, which helps to increase robustness of inferences and enables more complex context detection tasks. Though a larger number of sensing modalities can be beneficial for more accurate and wider mobile context detection, integrating these sensor streams is non-trivial.

This thesis presents how multimodal sensor data can be integrated to facilitate robust and energy efficient mobile context detection, considering three important and challenging detection tasks: indoor localization, indoor-outdoor detection and human activity recognition. This thesis presents three methods for multimodal sensor integration, each applied for a different type of context detection task considered in this thesis. These are gradually decreasing in design complexity, starting with a solution based on an engineering approach decomposing context detection to simpler tasks and integrating these with a particle filter for indoor localization. This is followed by manual extraction of features from different sensors and using an adaptive machine learning technique called semi-supervised learning for indoor-outdoor detection. Finally, a method using deep neural networks capable of extracting non-intuitive features directly from raw sensor data is used for human activity recognition; this method also provides higher degree of generalization to other context detection tasks.

Energy efficiency is an important consideration in general for battery powered mobile devices and context detection is no exception. In the various context detection tasks and solutions presented in this thesis, particular attention is paid to this issue by relying largely on sensors that consume low energy and on lightweight computations. Overall, the solutions presented improve on the state of the art in terms of accuracy and robustness while keeping the energy consumption low, making them practical for use on mobile devices.

Acknowledgements

I had the pleasure of meeting many wonderful people during my PhD journey, who influenced my research in various ways. This great learning experience would not have been the same without them and I owe a great deal of this achievement to them.

Firstly, I would like to extend my deepest gratitude to my supervisor Mahesh K. Marina for all the support, guidance and patience I received throughout my PhD. I learned from him to strive for quality of research with the potential to make an impact. His unfailing positive personality made this an enjoyable experience and I appreciate his good disposition in many social events. I am forever indebted for the valuable lessons learned from him, which will serve me all my life. I am very thankful to my other supervisors, Hugh Leather and Tristan Henderson for very insightful discussions and critical analysis of my approach, steering my research in the right direction. Working closely with Panagiota Katsikouli and Rik Sarkar was a real pleasure and I am very grateful for this fruitful collaboration.

I am very grateful for the opportunity to work with Nicholas Lane, who guided me through a new research topic and was always available to offer support. I am also very grateful to Cecilia Mascolo who kindly welcomed me into her research lab at Cambridge, making me feel part of the group and for providing me with resources and advice throughout my stay there. Many thanks to Sourav Bhattacharya for the guided discussions on neural networks. Their great experience in top level research pushed me to improve my work.

Many thanks to Myungjin Lee for giving me access to his computer cluster and to Charles Sutton for insightful discussions on machine learning techniques. I am thankful to my thesis examiners, Hamed Haddadi and Michael O'Boyle, for accepting to evaluate this thesis on such short notice and for their valuable feedback to improve the presentation of this work.

I feel fortunate to have worked next to some amazing colleagues over the years, creating a productive and enjoyable atmosphere in the lab, Lito Kriara, Arsham Farshad, Saravana Rathinakumar, Galini Tsoukaneri, Mohamed Kassem, Konstantinos Balampekos, Jiwei Li, Emilian Radoi, Daniel Duma and Cosmin Dumitrache. Also, special thanks to Imke van Ettinger for putting up with me in the more stressful moments.

Not least, I want to thank my family for their love and continuous support. Their encouragements has been invaluable throughout this journey. I thank my sister, Daniela, for taking more responsibilities in the family to give me space to concentrate on this work. I am very lucky to have such a loving family.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Valentin Radu*)

Table of Contents

Chapter 1

Introduction

With the advancement of electronics and processor miniaturization, a new generation of smart mobile devices has emerged for personal computing (i.e., monitoring and private data processing). As representative of these devices, smartphones now represent the dominant computing platform around us. Their sensing capabilities (e.g., light sensor, accelerometer, gyroscope) allow them to collect data from their immediate environment, while their compute power enables them to interpret all this data. A common characteristic across these devices is their rich set of embedded sensors facilitating a more diverse perception of their surroundings. Traditionally, these sensors have been used independently for the specific role they were introduced for (e.g., radio interfaces for communication and accelerometer for screen orientation), though increasingly these sensors are being used for other tasks such as the accelerometer for games control and for step counting. However, beyond their current utility as just independent sensors with limited purpose, the presence of these simple sensors in significant numbers and diversity creates the opportunity to build more complex context inferences (e.g., indoor localization, indoor-outdoor detection). These new forms of context detection will be essential to the mobile applications of the future, offering awareness and adaptability to user surroundings. Everywhere -sensing and -computing to adapt environments to user needs have long been the vision of ubiquitous computing [3]. These smart mobile devices are in the best position to facilitate this vision becoming a reality through mobile context sensing.

Through their diverse set of sensors, smartphones and smart watches perceive the environment from different perspectives. These sensors span a wide range from inertial sensors (e.g., accelerometer, gyroscope, electronic compass), pressure sensors (e.g., barometer), radio frequency receivers (e.g., GPS, WiFi interface, cellular interface), light sensors, proximity sensors, audio sensors (e.g., microphone), thermometers and others. Perceiving the surrounding environment from so many perspectives is beneficial for context detection, thus it is vital to exploit the multitude of sensing modalities available. But this is far from trivial. This is due especially to their intrinsic differences and sensing characteristics (e.g., sampling rate, data generation model, triggering method, etc.). Traditionally, sensor fusion is achieved with filters (e.g., Kalman Filter [4]) and more recently with machine learning techniques relying on features extraction for synthesizing sensor signals into observable trends. Feature extraction and feature selection are typically driven by data analyst's intuition and experience, directly impacting the quality of detection and implicitly the success of their context driven applications. Adding to the difficulty is that each context detection task has its own unique needs and challenges, thus requiring specialized solutions to perform inferences.

This work proposes solutions to a set of highly representative context detection tasks (indoor localization, indoor-outdoor detection, human activity recognition) by integrating multiple sensing modalities, each designed specifically to address the difficulties of its context detection task. These solutions rely on low energy consuming sensors, progressing gradually in design complexity as follows. The first solution is based on a typical engineering approach, decomposing the context detection task into constructing blocks (each could be seen as a simpler form of context detection on their own), though more powerful in combination to achieve complex inferences. In particular, we consider indoor localization task, which can be composed with the elementary tasks of detecting the orientation of a device, estimating traveled distance and comparing radio signal signatures, and results fused with a particle filter. The second solution, considering the indoor-outdoor detection task, is based on hand crafted features and machine learning to perform context detection. And finally, eliminating any preliminary decomposition and prior analysis, this work presents a deep learning based solution that can be employed for general mobile context detection tasks by enabling the extraction of relevant information directly from sensor signals. Even though the work presented here can be applied even to wearable devices (smart watches, in-pocket devices, on-clothes and in-fabrics, etc.), for practicality of demonstration, all context detection tasks will be showcased as applications for smartphones.

Two domain specific metrics are considered in designing the proposed solutions: from a Data Analytics perspective, accuracy and robustness of context detection are important; and from the angle of Mobile Systems, energy efficiency is essential for any battery powered device.

This chapter continues with an overview of the domain where this research falls, namely ubiquitous mobile context sensing. Following this is a description of the specific yet challenging context detection tasks explored in this work and their limitations, each with different needs addressed by the proposed solutions. These context detection tasks are: indoor localization that is addressed with decomposition of the context detection task and fusion with a particle filter; indoor-outdoor detection addressed using a common machine learning approach with hand crafted features; and human activity recognition facilitated by integration of sensing modalities with deep neural networks. A brief description of the energy consumption considerations of these mobile systems is then presented before stating the contributions this thesis makes.

1.1 Mobile Context Sensing

Context sensing or detection is a key component of mobile and ubiquitous computing systems for enabling context-aware applications [5]. The term “context” encompasses a variety of aspects of a mobile user including location, time, environment, device and activity. Some of these aspects such as time are straightforward to identify whereas others are relatively more challenging to detect. The emergence of smartphones and their rapid adoption have created great interest in context-aware mobile applications. At the same time, the many sensors built into modern smartphones aid in the context detection task. For example, the accelerometer on a smartphone is used for sensing the device orientation and accordingly aligning the screen to switch between portrait and landscape modes. In recent years, there has been considerable research on context sensing with smartphones, mostly focused around (indoor) location tracking [6–9] but also looking at other aspects of context such as activity recognition [10] and transportation mode [11].

Advances in other research fields like Natural Language Processing has enabled mobile assistants (like Apple’s Siri and others) to be featured on mobile devices. Many other systems, built on top of Amazon’s Alexa[1] are growing in popularity because of their flexibility to integrate with Internet of Things (IoT) devices, thus accelerating home automation systems. Currently these rely on sound and voice recognition alone, though their easy integration with a multitude of devices makes it possible to employ ubiquitous mobile context sensing, to increase awareness in interactions with their

[1] http://alexa.amazon.co.uk/

users for contextual relevance.

Growing privacy concerns from uploading user sensor data to the cloud for inferences (e.g., uploading voice data for voice recognition with Alexa), encourage more of these inferences to be performed locally, on user's mobile devices. Most of the computation requirements are already satisfied by the majority of mobile devices, while judicious selection of sensors and algorithms aid in managing with the limited energy budget. This alternative assures users their sensor data is not exposed to any unnecessary risks through uploading data over the Internet.

There are different forms of context that can be detected with sensors available on smartphones, each with its own characteristics and challenges, from human activity recognition to transportation mode detection and beyond. Accurate detection of these context detection tasks is challenging, so they require specialized algorithms to suit their individual characteristics. The following sections introduce three such challenging and highly important context detection tasks for which this work proposes solutions: indoor localization, indoor-outdoor detection and human activity recognition.

1.2 Challenging Context Detection Tasks

Three forms of context detection with mobile devices are explored in greater detail in this thesis which drive the investigation of different approaches proposed for integrating multimodal sensor data. Though each of these three has been explored before in the literature, previous solutions are limited in addressing their fundamental challenges of attaining the best integration of sensor data to facilitate accurate, robust and energy-efficient inferences. In view of their uniqueness and difficulty, custom solutions of sensor fusion for each detection task are developed and presented in this work. This section presents an overview of proposed solutions for each of the three context detection tasks.

1.2.1 Indoor Localization

Indoor mobile phone localization is a popular research topic due to the increasing number of location-based services and applications that require accurate positioning or continuous tracking inside buildings. These applications can span from indoor navigation [12] to monitoring different aspects of the environment like the WiFi coverage [13] and can be used in many indoor spaces like offices, shopping malls and airports.

Dead reckoning and WiFi fingerprinting are well known approaches for indoor localization but each has its own advantages and limitations. While dead reckoning based schemes naturally enable continuous location tracking, error accrual over time is a major concern; moreover, dead reckoning in indoor environments with complex movement patterns is relatively more challenging. A WiFi fingerprinting based localization approach is an attractive alternative as it can leverage the smartphone WiFi interface to take advantage of existing WiFi infrastructure, nowadays commonplace in most indoor environments. However, WiFi fingerprinting has its own disadvantages like not being suitable for continuous location tracking due to heavy energy cost of performing WiFi scans on mobile devices. Also the applicability and effectiveness of WiFi fingerprinting is dependent on a number of factors including WiFi Access Point (AP) density, spatial differentiability and temporal stability of the radio environment.

In view of the above, we propose HiMLoc, a novel solution that synergistically uses Pedestrian Dead Reckoning (PDR) and WiFi fingerprinting, exploiting their positive aspects while limiting the impact of their negative aspects. Specifically, HiMLoc combines location tracking and activity recognition using inertial sensors on mobile devices with location-specific weighted assistance from a crowd-sourced WiFi fingerprinting system via a particle filter. HiMLoc relies on the most common sensors available on the large majority of smartphones: accelerometer, compass, and WiFi interface.

This novel integration of dead-reckoning with WiFi fingerprinting is based on the observation that some spaces in a building are more accurately localizable with WiFi fingerprinting than others. This is a consequence of the radio environment being more stable and having unique signatures due to building structure and radio signal propagation effects. This observation is exploited by associating a weight for the WiFi fingerprinting component in a particle filter to control its impact in the hybrid system. This weight is inversely proportional to similarity area metric computed by comparing a run-time WiFi fingerprint with fingerprint database – smaller similarity area results in a higher weight and vice versa.

To ease deployment, HiMLoc requires just a small set of parameters specific to each new building, like position of stairs, position of elevators, position of main entrances and height of each floor. The WiFi fingerprinting component is driven by crowd-sourcing. Unlike other particle filter systems that require a detailed knowledge of the building layout to restrain the particles, HiMLoc uses distances to known reference points (corner, stairs, elevators and WiFi estimations) to determine the weights of particles and opportunistic location calibration.

Experimental evaluation of HiMLoc using Android phones shows that median location accuracy of under 3 meters is achievable even with complex movement within a building (e.g., going between floors using stairs and elevators).

1.2.2 Indoor-Outdoor Detection

Indoor-Outdoor (IO) detection is an environment related aspect of user context that is important for enabling context-aware applications. While the IO distinction is important in determining context, it is also a subtle and challenging problem. Intuitively, several physical quantities differ between the two contexts, but inferring this difference using sensors is not straightforward and highly sensitive to environment. For example, light intensity is likely to change as one moves from inside a building to outside, but the nature of the change will be different depending on time of day, location, weather, and other parameters. Environment can similarly affect sound intensity, temperature and other quantities. WiFi and cellular signal strengths also vary from place to place, and are affected by different local attenuation characteristics and multipath effects; sound, light and temperature can be affected by the phone position with respect to the user (e.g., in a bag/pocket, hand); sensors and their calibration can vary between phones. Thus a context detection system has to have the ability to adapt to new environments and encountered situations.

A novel solution for IO detection is presented in this thesis. This explores the problem from a machine learning point of view, proposing a semi-supervised machine learning approach to infer the context on smartphone sensor data. In comparison, the most closely related piece of work, IODetector [14] has an accuracy in the range of 55-70% and GPS based technique gives an accuracy around 70% to 80%, as shown in Chapter 4. The fundamental issue is that when a user encounters a new environment or the classifier is applied on a device different from the one used in its training, the model needs to adapt to the new environment and/or device. Doing so naively by collecting additional labeled training data with ground truth information requires user involvement, which is intrusive and impractical. The semi-supervised approach presented here is superior because it can adapt to new environments by generating new labels according to how similar they are to past experiences.

1.2.3 Human Activity Recognition

Among the sensors available on smartphones, the accelerometer has gained much popularity in Human Activity Recognition (HAR) as it allows recognition of a wide variety of human activities, while having relatively low energy consumption compared to other sensors [15]. Many HAR systems use the accelerometer for detection in different environments, smart homes [16, 17], health care [18], daily activity tracking [19] and fall detection of elderly people [20].

More recent adoption of gyroscopes as embedded sensor to smartphones offers the opportunity to complement the accelerometer in performing inferences. This work expands the research presented in [21], using the two modalities for HAR with other forms of machine learning techniques. Building on recent developments in deep learning, this work evaluates new solutions for sensor fusion with deep neural networks. The advantage of these new architectures is that features are extracted automatically during training from raw data, thus replacing the initial process of feature extraction and feature selection.

1.2.3.1 Deep Learning for Context Detection

The simple and numerous sensors available on smartphones provide the opportunity to help with more complex inference tasks by combining capabilities across complementary modalities. But due to their intrinsic nature and sensing characteristics (e.g., sampling rate and statistical properties) integrating sensor streams is often very challenging.

So the aim is to investigate the ability for deep learning to advance the state of the art in multimodal sensing on mobile and embedded devices, considering human activity recognition task as a representative case.

Deep learning [22] is an area of machine learning that is revolutionizing several domains from computer vision to speech recognition and many others. This fast growing area of research has the potential to influence key topics like sensor data fusion, with study of this learning paradigm applied to mobile devices only recently begun [23, 24].

One attractive characteristic of deep learning is the ability to transform close to raw sensor data into a dense representation of features through different activation patterns of artificial neurons (i.e. units) within a deep neural network. This network is used to perform inferences (e.g., estimating the activity class) directed by the activation pattern of neurons in the network, and often achieves higher accuracy than classic modeling methods.

With evidence from deep architectures on dual modalities like text mixed with images [25] and audio linked with video [26, 27], similar impressive gains should be attainable with other combinations of modalities; for example, in the case of human activity recognition, cheap sensors like accelerometer and gyroscope present on mobile and wearable devices. The aim is to provide the initial answers to whether these algorithms can increase the accuracy of ubiquitous tasks (e.g., activity recognition) using sensor data from wearable devices, which is not well explored in the literature. This exploration is conducted using a multimodal Restricted Boltzmann Machine (RBM) architecture (a promising deep learning algorithm), with resource requirements make this architecture viable to resource constrained computation units like mobile and wearable devices.

1.3 Energy Efficiency

As previously mentioned, this work uses smartphones as the primary evaluation platform, though these observations can be easily translated to other wearable devices (e.g., smart watches, VR headsets and other wearable devices) as impacted by the same energy constrains. To ensure limited battery resources are efficiently managed while performing useful context detection tasks, algorithms need to be optimized to have low energy footprint as well as using the lower energy consuming sensors. Energy consumption is thus a key optimization criteria throughout the work presented in this thesis.

In order to accurately measure power consumption of individual sensors and different solutions presented in this thesis, the following methodology is used throughout. This essentially involves removing the phone battery and having it instead powered through the commonly used Monsoon Power monitor, which allows the exact power consumption to be measured. This reflects the exact power consumption associated with an application observed on the whole system (compute, memory, operating system calls, internal communication, etc.). Any external variables were eliminated by restricting communication and stopping all other running applications for the time of experiments. With this experiment set-up only the impact of the proposed solution affect the energy consumption. Energy consumption due to a sensor is the area under the power consumption curve over the duration of that sensor's sampling interval. The same approach extends to measuring energy consumption from a *set* of sensors.

Majority of sensors consume uniform power throughout continuous sampling, so both energy consumption and instantaneous power consumption can characterize impact of longer-term sampling on device battery.

1.4 Contributions

This thesis proposes three different approaches for integrating multimodal sensor data from mobile devices to infer important and challenging contexts (indoor localization, indoor-outdoor detection and human activity recognition). Design of the proposed approaches was aimed at optimizing accuracy/robustness and energy-efficiency. This work is timely due to emerging smart and adaptable applications requiring many forms of complex context detection to facilitate better services to their users in accordance with observed conditions in the environment.

The core contribution of this work is to present how apparently complex contexts can be detected by efficient combinations of multimodal sensor data, starting with an engineering approach of decomposition, followed by machine learning as core to adaptive sensing systems and finally using deep neural networks for more general mobile context detection. These contributions are outlined next with respect to the context detection task where they are introduced.

1.4.1 Indoor Localization

In the space of indoor localization, this work makes the following contributions:

- Presents the design of a hybrid indoor mobile phone localization mechanism called HiMLoc that combines the best aspects of two well-known localization techniques, pedestrian dead reckoning (PDR) and WiFi fingerprinting, neither of which is sufficient – location error accumulates over time with PDR especially when based on smartphone sensors, whereas WiFi fingerprinting does not work when there is no WiFi coverage. A key feature of HiMLoc is that it exploits the locations deemed to be accurately localizable via WiFi fingerprinting for correcting PDR based location estimates. Experiments show that HiMLoc achieves a median location accuracy below 3 meters in a building of approximately 12,000 m^2 spread over 5 floors.

- Showing that HiMLoc can be used as a localization platform for crowdsourced mobile applications that monitor the interior radio environment, through the Pazl system. This usecase has the following components: (1) an Android application for collecting WiFi, cellular, bluetooth and sensor (accelerometer and compass) measurements from each mobile crowdsensing participant's smartphone; (2) a cloud application based on the Google App Engine to localize the measurements from different phones and to merge, store, visualize and analyze for various monitoring related aspects (e.g., coverage holes, channel usage distribution, complex interference patterns resulting from exceptionally long range of some APs as seen from certain locations). For indoor WiFi monitoring Pazl provides similar results to the state of the art Ekahau Mobile Survey tool [28] but in a significantly more automated manner by drastically reducing the manual point-and-click location determination used in the Ekahau approach.

This work was reported in the following publications:

1. **Valentin Radu and Mahesh K. Marina**, "HiMLoc: Indoor Smartphone Localization via Activity Aware Pedestrian Dead Reckoning with Selective Crowdsourced WiFi Fingerprinting". In Proceedings of IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2013.

2. **Valentin Radu, Lito Kriara and Mahesh K. Marina**, "Pazl: A Mobile Crowdsensing based Indoor WiFi Monitoring System". In Proceedings of IEEE International Conference on Network and Service Management (CNSM), 2013.

3. **Valentin Radu, Jiwei Li, Lito Kriara, Mahesh K. Marina, Richard Mortier**, "A hybrid approach for indoor mobile phone localization". Poster in ACM International Conference on Mobile Systems, Applications, and Services (MobiSys), 2012.

1.4.2 Indoor-Outdoor Detection

With respect to this form of context detection, the main contributions are:

- Shows the limitations of previous systems performing indoor-outdoor detection measured from both energy consumption and robustness/accuracy of inference. This exploration highlights the sub-optimal performance of linear classifiers relying on thresholds for detection.

- Proposes the use of a *semi-supervised learning approach* that can continuously learn in new environments, and adapt to them for indoor-outdoor detection, without user involvement. Existing works on learning from mobile sensing data have largely utilized offline methods where all the data is available for analysis. This method instead works *online* – learning in real time – and on the device itself, at a modest computational cost.

- This work explores three different semi-supervised learning methods, namely clustering, self-training and co-training. A well-designed co-training model is found to be most effective providing greater than 90% accuracy across diverse and previously unseen environments. A choice of Naive Bayes classifiers gives the highest accuracy in this adaptive setting.

- The aforementioned co-training based indoor-outdoor detection system has several attractive properties. Naive Bayes classifiers can be designed to update online at negligible computation and memory costs, thus it can update and learn on the mobile device itself without communication costs and delay. This also makes it privacy preserving. The method is stateless: it does not need temporal history and can be run on-demand; thus the sensors can sleep except when responding to a query. The approach is lightweight and uses only low power sensors. A single state estimation costs only about 0.73 Joules, and experiments show this to be significantly more efficient than other methods. The system is also presented through an application designed to avoid wasteful WiFi scans while outdoors, achieving a 63% energy saving.

This work was reported in the following publications:

1. **Valentin Radu, Panagiota Katsikouli, Rik Sarkar and Mahesh K. Marina**, "A Semi-Supervised Learning Approach for Robust Indoor-Outdoor Detection with Smartphones". In Proceedings of ACM Conference on Embedded Networked Sensor Systems (SenSys), 2014.

2. **Valentin Radu, Panagiota Katsikouli, Rik Sarkar and Mahesh K. Marina**, "Am I Indoor or Outdoor?". Poster in ACM Annual International Conference on Mobile Computing and Networking (MobiCom), 2014.

1.4.3 Deep Learning for Human Activity Recognition and Other Context Detection

Leveraging a diverse set of sensors strongly aids in the inference of activities and context in mobile systems. However, building models that can fully leverage the information contained within – *and across* – each sensor is challenging largely due to intrinsic differences between sensor data.

With this motivation, this thesis presents a new approach to modeling of multimodal mobile data, which incorporates two recent deep learning innovations previously unseen for mobile and wearable devices. First, it adopts a new variant of a Restricted Boltzmann Machine (RBM) that supports a learning architecture that mixes isolated sensor specific layers, and shared cross-modality layers [25, 29, 30]. Second, the training of this deep architecture is performed with an autoencoder inspired technique [27, 29, 31] for multimodal settings to tolerate noisy data.

The contributions of this research are:

- The development of a RBM-based multimodal deep learning model designed specifically for mobile sensor streams, and common behavior and context inferences needed by wearable and mobile devices.

- An evaluation of this deep modeling approach on diverse datasets (for activity recognition, sleep stage detection, indoor-outdoor detection) capturing a variety of sensing modalities. Results indicate that this general purpose sensor model is able to outperform recently published state-of-the-art purpose designed techniques, as well as a range of shallow learning algorithms.

- A system resource feasibility study that verifies the overhead of the RBM-based multimodal model. Implementations under one mobile/wearable processor shows that the memory, battery and computational footprint of the model is manageable by a mobile device.

Publications related to this work are listed below:

1. **Valentin Radu, Nicholas D. Lane, Sourav Bhattacharya, Cecilia Mascolo, Mahesh K. Marina, Fahim Kawsar**, “Sensor Fusion using Multimodal Representational Learning for Improved Accuracy on Wearable and Mobile Devices”. Submitted to ACM Journal on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT), currently under review.

2. **Valentin Radu, Nicholas D. Lane, Sourav Bhattacharya, Cecilia Mascolo, Mahesh K. Marina, Fahim Kawsar**, "Towards Multimodal Deep Learning for Activity Recognition on Mobile Devices". In Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp/ISWC'16) Adjunct, 2016.

1.5 Structure

The rest of this thesis is structured as follows.

Chapter 2 gives a background on the concepts and techniques used in later chapters, and introduces the tools used for classification, Weka - Machine Learning toolkit, deep learning models and Torch. For each of the context detection tasks considered, related work is presented to discuss previous approaches and their limitations.

Chapter 3 presents the novel system proposed for indoor localization combining information collected from multiple sensors in stages. The first step is to determine the type of mobility action based on acceleration signals, using this to estimate the displacement with Pedestrian Dead Reckoning. In parallel, using WiFi scans to estimate locations based on the radio signatures. These estimations are combined using a particle filter to determine the phone location. To demonstrate the practical utility, a mobile application collecting wireless network coverage has been deployed and evaluated.

Chapter 4 explores different approaches to combine sensor features to create an adaptive system that is not restricted by hard-coded thresholds for this type of inference. A special class of semi-supervised learning methods, called Co-training achieves this goal by running two classifiers in parallel and independently observing different angles of the environment to assist each other in learning the characteristics of new environments over time.

Chapter 5 explores the opportunity to integrate sensor streams directly from raw data through a deep neural network, thus avoiding the tedious task of determining sensor features. This facilitates a faster deployment and was demonstrated through a set of applications, in particular human activity recognition.

Chapter 6 summarizes the work presented in this thesis and reiterates the contributions made by this work. Some interesting future work is suggested as basis for future development in the space of sensing with mobile devices.

Chapter 2

Background and Related Work

The first part of this chapter presents current solutions used for each of the three context detection tasks covered in this thesis: indoor localization, indoor-outdoor detection and activity recognition. Background information about the underlying techniques of the proposed solutions in this thesis is presented in the second part of this chapter.

2.1 Previous Work on Mobile Context Detection

2.1.1 Indoor Localization

While outdoor location tracking can be done easily using GPS based systems [32] even for long-term tracking of subjects with custom mobile devices [33, 34], indoor localization is harder due to the lack of GPS signals penetrating through wall. Indoor localization with smartphones is a well explored researched topic, with many systems aiming to tackle this task, though none has emerged as the dominant solution, which This also shows the difficulty of designing mobile systems to perform such a complex task. Majority of the available solutions follow a few established methods: Pedestrian Dead Reckoning, WiFi Fingerprinting and combinations of these two. These methods are presented in following sections, also highlighting some of their representative systems.

2.1.1.1 Pedestrian Dead Reckoning

The presence of inertial sensors has enabled the emergence of a new class of location tracking systems performing dead reckoning on mobile phones. These systems have the advantage that very little physical infrastructure is required for them to function.

Pedestrian Dead Reckoning (PDR) technique works by estimating successive positions starting from a known location, based on a method of estimating the traveled distance and the direction of walking. A solution to determine the traveled distance is to count the number of steps and estimate their length. Most typical step detection implementations are based on analyzing the acceleration data [35], [36], [37], but data from other sensors have also been tried, like angular velocity [38], [39], [40] and magnetometer data [41], or combination of these [42]. Using the acceleration magnitude, steps detection is performed through techniques like peak detection, which looks for peaks in the acceleration magnitude caused by the leg carrying the sensor touching the floor [43]; zero crossing, which monitors the acceleration value zero crossings [44]; and auto-correlation, by taking advantage of the repetitiveness of human walking [45]. The traveled distance can also be estimated, either by observing the rotation of the hip [46], or by estimating the length of the step. Probably the easiest way to estimate the step length is to appreciate it as a linear function of the frequency of stepping [47].

The other important component of the PDR is direction, which can be obtained by a compass or a gyroscope. The presence of a compass on a smartphone is more common than having a gyroscope. But compass indications are subject to magnetic interference inside buildings. Afzal et al. showed that these interferences can sometimes result in a direction deviation from the compass of up to 100^o [48]. However, our experience was that under the normal conditions of human walking not too close to walls or other metal structures along the way, magnetic interference is typically isolated and tolerable.

Common presence of sensors such as accelerometer and compass in smartphones have made PDR an attractive technique for mobile phone localization [49]. While most systems use PDR for outdoor tracking in conjunction with a map [50], others such as GAC [8] combine it with occasional GPS correction for energy-efficient location tracking on roads. A well-known limitation of PDR schemes is that error can get accumulated over time unless it is corrected occasionally.

The steady increase in performance of inertial sensors opened the opportunity for their use inside buildings with smartphones [51], [12], [50]. All of these systems have an increasing error accumulation if they are not periodically adjusted. Assisting the system with corrections from beacons has been experimented in [12]. For an easier deployment, activity recognition together with some knowledge of the building layout can provide some error correction points [51].

2.1.1.2 WiFi Fingerprinting

WiFi fingerprinting is a well-known localization technique that can exploit the presence of WiFi interfaces now common on smartphones. WiFi infrastructure is also prevalent these days in many indoor environments. Early WiFi fingerprinting systems such as RADAR [52] and Horus [53] rely on an initial training phase to construct fingerprint database for use as a reference in the positioning phase later but training phase can be quite time consuming and expensive. More recent WiFi fingerprinting systems make this training phase automated via crowdsourcing using mechanisms of increasing sophistication (e.g., Redpin [54], OIL [55], WiFi-SLAM [56], Zee [7]).

While these systems work well with a sufficient number of samples, it is still a challenge to know which runtime fingerprints stand a good chance to provide a more accurate location estimation than others. Using just one fingerprint on the go requires a way to rapidly determine the value brought by each scan.

WiFi fingerprinting can be quite expensive from an energy consumption perspective if solely relied on for continuous location tracking. Another more obvious disadvantage of WiFi fingerprinting is that it works only where there is WiFi coverage. There are however usually some areas inside buildings not generally considered for Internet connectivity requirements like the stairs, toilets and some corridors. Despite this, WiFi fingerprinting can offer the needed correction for a PDR based system where available and if used judiciously as shown with HiMLoc.

2.1.1.3 Hybrid Localization Solutions

Hybrid localization approaches that combine PDR with WiFi fingerprinting try to avoid the disadvantages of either of those two individual approaches: PDR have enough correction instances to reduce the error accumulation in the navigational component and there always is a location estimation no matter whether is WiFi signal coverage or not.

Combining PDR with WiFi fingerprinting has been considered in [6] and [57]. The UnLoc system [6] combines the use of inertial sensors (accelerometer, compass, gyroscope) with the notion of natural and organic landmarks that are learnt over time for indoor navigation. While UnLoc looks to find WiFi landmarks based on the set of APs it sees, in [57] the use of WiFi fingerprinting is used only in the location where maximum signal strength is seen, to correct PDR at those points. While both [6] and [57] use basic PDR scheme, HiMLoc incorporates a more sophisticated version with activity recognition capability that would be needed in more complex environments (e.g., multi-floor buildings with elevators and stairs to move between floors). Moreover, unlike [6] and [57], HiMLoc uses only accelerometer and compass for the PDR which are present in almost every smartphone, thus achieving greater applicability. HiMLoc is presented at a high level in its initial form in [13] in the context of Pazl mobile crowd-sensing based indoor WiFi monitoring system. The current paper provides a detailed design and evaluation of HiMLoc.

WiFi-SLAM [56] is a pioneer in bringing the robotics technique of SLAM (Simultaneous Localization and Mapping) into PDR. By using a detailed model of the building layout, their PDR implementation can track a person inside the building and collect WiFi scans to build the radio map at the same time. Their high accuracy is achieved by using specialized hardware. Similarly, Zee [7] learns the WiFi environment by using a PDR assisted by particle filter, in a crowd-sourcing manner. Unlike Zee and WiFi-SLAM, HiMLoc does not need a very detailed building model (the exact location of each wall); instead a few natural landmarks (position of elevators, stairs and corners) and some parameters of the building (height of each floor) are sufficient for HiMLoc to obtain a good level of localization accuracy. Another approach presented by Faragher et al. [58] was to use smartphones to collect acceleration data in order to estimate the movements using a Distributed Particle Filter Simultaneous Localization and Mapping (DPFSLAM). They relied on WiFi signal opportunistically, just to identify those places where the user has been before. Their experiment setup consisted of a single floor in an office building, with no intention of using landmarks like elevators and stairs and movements between floors.

HiMLoc builds on these modern solutions and takes them one step closer towards an easily deployable and widely applicable indoor localization system.

2.1.2 Indoor-outdoor Detection

Several systems rely on low GPS confidence or inability to get a fix as a hint to infer that the user is indoors. In [59], the authors use such a GPS based indoor vs. outdoor hint in a wireless protocol architecture that adapts to different user contexts based on sensor hints. In [6] and [60], similar approach is used to bootstrap indoor localization systems. IODetector [14] takes a different approach, relying instead on light, magnetic and cell based sensor features. It includes an intermediate semi-outdoor state that is subjective and tricky to interpret/use in practice but has the positive effect of making the IO detection problem somewhat easier on suitably labeled data from a single environment. More crucially, IODetector uses fixed thresholds for sensor features to distinguish between indoors, outdoors and semi-outdoors, which as shown in the earlier sections can lead to inaccurate estimations when used across different environment and device types. UPCASE [61] is a context detection system that uses on-body sensors connected to the phone via Bluetooth, somewhat similar to [10]. It does activity recognition using a classifier based on various sensor features, also like [10]. From an IO detection perspective, UPCASE allows distinguishing between user walking (running) inside and outside using accelerometer and temperature sensors. In contrast to the above techniques, the proposed solution here is a semi-supervised learning approach for robust and adaptive IO detection across different environments and devices. This is the first time semi-supervised learning methods are used for context detection with smartphones. Closest other setting where semi-supervised learning has been applied before is for co-localization of sensors and access points in a wireless sensor network [62].

2.1.3 Human Activity Recognition

Human Activity Recognition (HAR) with mobile devices is a broad topic, which has developed with the aim to detect user behaviour that allows computer systems to proactively assist users with their tasks. Applications of HAR range from very specific (such as in assisting with indoor localization) to general recognition in unconstrained daily life, limited only by desired classes and samples in the training set. Though very different in purpose, HAR applications generally build on acceleration signals predominantly and increasingly on gyroscope signals.

In the commercial space, Google provides an activity recognition service for the Android operating system, which can be used by any application to identify when the user is walking, running, still, cycling and in a vehicle. They make use of the accelerometer and more recently of the Bluetooth to reduce the latency of detected activities[1]. In similarity to this, Apple has a motion API for their smart watches, inferring activities like: stationary, walking, running, automotive, cycling and unknown[2]. These frameworks are not flexible to define other activity classes like climbing up stairs or climbing down stairs relevant and small scale events like entering/exiting a room and elevator movement, all relevant for localization or general purpose household activities

[1] https://www.youtube.com/watch?v=S8sugXgUVEI

[2] https://developer.apple.com/reference/coremotion/cmmotionactivity

like vacuuming, washing dishes or watching TV. The small set of activities detected by these APIs makes them suitable for limited purposes only, while for more complex detection tasks, developers need to construct their own solutions.

Gusenbauer et. al, introduced Pedestrian Dead Reckoning with Activity Classification, designed to navigate a person in an underground parking lot in [51]. Thus, they only consider the case of a person walking with the phone in hand and ahead of the user, not exploring other cases of carrying the phone and assuming no WiFi coverage in those environments. Ftrack [63] also uses an activity classifier to perform floor detection, having just a limited number of activities that can recognize, like movements on stairs and in elevator.

Other works explore HAR with more than one sensing modalities, considering both accelerometer and gyroscope signals from smartphones [64], [21] and smart watches [21].

Generally, the difficulty of detecting user activities comes from not capturing the full detail of user motion from sensor signals (e.g., acceleration) due to inappropriate feature selection, which can also be a very difficult task in itself. That is why solutions taking advantage of the full of information in raw format are preferable, as explored in Chapter 5 of this thesis.

2.1.4 Other Contexts

The focus of outdoor location tracking research on the other hand has been to rely on GPS but to use it sparingly. As with indoor localization, various proposals take advantage of other phone sensors (e.g., accelerometer, compass, cellular interface) [8, 9, 65]. Systems like Sensloc [66] aim to go beyond raw physical location, in the spirit of SurroundSense mentioned above, to provide information about places visited and paths traveled via combined and energy-efficient use of GPS, WiFi interface and accelerometer on phones. A related issue is dwelling detection, i.e., identifying when user is in a confined area (e.g., home, shop, office) but not necessarily stationary. Brouwers and Woehrle [67] present a study of dwelling patterns of users based on three different sensors (GPS, WiFi and phone's location service)

There has also been work on sensing other aspects of context with smartphones beyond location. Some research considers detection of device position (whether in pocket, handbag etc.). For example, [68] uses combination of light and proximity sensors on the phone to infer if it is in pocket, in bag or neither. For the same inference, a previous work [69] has considered different set of phone sensors and a machine learning based classification approach. Activity recognition is another issue that has received fair amount of attention. In [10], the authors present a system that leverages on-body sensors and user interface of smartphone for reliably detecting various daily user activities (e.g., walking, reading, working, eating). In an earlier work, Wang et al. [70] presented a hierarchical sensor management strategy for energy efficient sensing of mobile phone user activities. Somewhat related to activity recognition is the issue of detecting user's transportation mode (walking, traveling on bike, train, car, etc.). In [11], the authors present a system that fuses phone GPS and accelerometer data with GIS information to infer the user's transportation mode. More recently, a more energy-efficient approach that relies only on accelerometer data is presented in [71].

2.2 Multimodal Sensing and Learning Approaches

Popular methods for learning from multimodal data can be clustered into two groups differing by which stage of processing the fusion of information from each sensor is attempted. These are *feature concatenation* (FC) and *classifier merger* (CM). Under FC, data from each sensor is merged immediately into a single vector presented to classification stages. The solutions proposed for indoor-outdoor detection and human activity recognition fall in this category. In contrast, CM will train separate classifiers for each sensor or for a small group of sensors and delays merger until each classifier has reached a decision. This is the case for the indoor localization solution presented here constructing preliminary classifications like estimating distance traveled, walking direction, activity recognition and radio map matching.

FC is highly reliant on the use of not only features that discriminate inference classes based on single sensor types; but, also it demands the discovery of additional cross-sensor features. The degree to which multi-modal information is maximized is dependent on the quality of these hand-crafted features. Often feature selection in concert with the extraction of a large number of candidate features for each sensor type is attempted to automate, to a degree, this process (a technique adopted, for example, by the MSP [72]). However, this approach is bounded by the quality of features used and can easily overlook inter-sensor relationships – with the number of feature combinations explored limited by factors like the curse of dimensionality [73].

Like FC, variations of CM are also commonly adopted in multimodal activity models [74–76]. One key attraction is that existing classifier designs (i.e. combinations of features and models) that are tested and verified can be adopted for each sensor type available. Essentially merging sensor-specific classifier results enables the evidence of each data type to be considered before a final inference is drawn.

2.2.1 Machine learning with hand-crafted features

This section presents a short introduction to the tools used in the later investigation of machine learning for indoor-outdoor context detection.

Implementation of experiments was performed on top of WEKA [77], the open source machine learning software suite. WEKA includes several classification libraries categorized into eight types: *Bayes, Functions, Lazy, Meta, Mi, Misc, Rules, Trees*.

Based on popularity in applied machine learning and best performance seen with our datasets, we focus on a smaller set of classifiers in our analysis. The set of classifiers considered are: *J48 decision tree, Naive Bayes, BayesNet, Locally Weighted Learning (LWL)* and *Sequential Minimal Optimization (SMO)*. Among these classifiers, decision trees, Naive Bayes and BayesNet are popular classifiers used in machine learning and classification tasks, because they are simple to understand and use, and in practice often outperform more complex methods.

J48 decision tree works with a sorting of features by importance, and thus works well where some features are more discriminative than others. Naive Bayes assumes that features of a data instance contribute independently to determine the class of the instance, and performs well where this holds. Note that for sensor data on mobile phones, both these can be expected to hold to some extent. BayesNet classifier represents information as a probabilistic network of dependencies in an acyclic graph.

As per other methods, LWL uses an instance-based algorithm to assign instance weights and then performs classification with the use of Naive Bayes, and is effective in filtering noise. Finally, SMO is a training method for support vector machines, which are effective in binary classification tasks for datasets with unknown distribution or non-regular distribution, as is the case with our datasets. These latter methods however are computationally more expensive.

2.2.2 Particle Filter

A Particle Filter is a numerical approximation to a Bayesian filter [78]. It has a number of ’particles’, each representing a virtual position with its own weight to describe the likelihood of the user having that position. Particle filters are usually used in PDR system to incorporate maps in the system. Particles move independently on the floor plan and when they cross a wall they are eliminated, assigning higher weights to the other particles following the constraints imposed by the floor plan [38]. The only problem with this way of using Particle Filter is that a very detailed model of the building is required at deployment time, which is hard to obtain. In our case, the particle filter has the role of fusing activity classification and PDR estimation from inertial sensors with an independent location estimation from the WiFi fingerprinting positioning component.

2.2.3 Deep Learning

Overcoming the shortcomings of shallow classifiers can be achieved with modeling multimodal context data can be overcome through the use of deep learning; and have been successfully applied, for example, to images and text for image captioning [30] or speech and text for machine translation [79]. Such deep algorithms (e.g., Convolutional Neural Networks, Restricted Boltzmann Machines) learn a number of hierarchical layers of dense feature representations tied to the discriminative task at hand rather than relying on domain-specific features.

Deep learning is an area of rapid machine learning innovation that is causing disruptive leaps in accuracy across numerous applications, including the recognition of words [80], objects [81] and faces [82]. One of the defining characteristics of this approach is its ability to learn dense hierarchical networks that transform relatively raw forms of data into inferences (e.g., an activity class). This network merges the roles of feature extraction and classification stages of shallow modeling methods (e.g., SVMs [73]); it also replaces the need for hand-engineered task-specific features with layers of data representation that act as features and are learned directly from data.

Building evidence from deep architectures and algorithms designed for, and successful in, dual modality settings suggest these methods may also help current bottlenecks in learning cross-sensor features for multi-modal activity models. New training methods that leverage variation in information [30], multi-view representations [83], or modified autoencoders [27, 29] are showing an ability to fuse highly heterogeneous pairs of data types, such as: text mixed with images [25] and audio linked with video [26, 27]. The resulting bi-modality deep models offer considerable accuracy gains in tasks like image captioning [30] and emotion recognition [31, 84, 85] (merging facial expressions and sound). The goal of this work is to build initial answers as to how these emerging deep learning techniques can address the challenges of multimodal activity models (such as certain sensing tasks and data types) that are not yet well explored in the literature.

Chapter 3

A Hybrid Particle Filter based Approach for Indoor Localization

This chapter presents my work on indoor localization with smartphones. Using the sensors available on smartphones it is possible to estimate the position of these devices carried by people inside a building with a good accuracy to enable a range of applications. The novel indoor localization system presented here, HiMLoc, is a hybrid solution combining two classic approaches for indoor localization, Pedestrian Dead Reckoning and WiFi Fingerprinting.

This work has been presented in Proceedings of the IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2013 [86] where the Technical Committee nominated our paper for the Best Paper Award. An earlier version of this work was presented in the Proceedings of the ACM International Conference on Mobile Systems, Applications, and Services (MobiSys) [87]. The application built on top of this localization system to monitor wireless networks coverage was presented in the Proceedings of the International Conference on Network and Service Management (CNSM) [13]. This chapter is built on the work presented in these publications. The wireless monitoring application was developed with the help from Dr. Lito Kriara who created the visual interface to visualize the wireless maps.

HiMLoc was started during my MSc project, where I explored an indoor localization system using Pedestrian Dead Reckoning [88] for horizontal movements and estimating transitions between floors. That earlier version of the system relied on activity recognition for distinguishing between walking, being stationary, climbing stairs, taking an elevator and entering/exiting door, to perform different routines for each type of movement. This earlier work however aimed at obtaining a single deterministic estimation of a mobile device location each time, as it moves. Here I significantly increase the robustness by introducing a Particle Filter to perform multiple location estimations simultaneously considering different movements of a mobile device as well as incorporating WiFi fingerprinting based estimation. This new paradigm requires probabilistic models for each constituent component, which was achieved by modeling the error of measuring the walking distance, modeling the error in heading observations and modeling the confidence in localizing using WiFi fingerprinting. Also note that the process of transitioning between floors and the activity recognition component are covered in my earlier work so are left out of this dissertation.

This chapter begins by presenting a system overview of HiMLoc, and going into details of how each component works as part of the integrating particle filter. The evaluation of HiMLoc is described next. A crowdsourced wireless monitoring application, Pazl, is built on top of HiMLoc to showcase the utility of this indoor localization system and presented toward the end of this chapter.

3.1 Design and Implementation

This section describes the proposed hybrid indoor localization system, HiMLoc, and goes into details for each of the fundamental components of this system.

3.1.1 HiMLoc Hybrid Localization Mechanism Overview

HiMLoc system components are presented in Figure 3.1. Phone's sensors (accelerometer, compass and WiFi card) collect sensor data (acceleration, orientation and WiFi scans) to be used as direct input to HiMLoc. The Activity Classification component determines what activity the user is performing within a short interval of time by sampling the Acceleration data. If the estimated activity can be performed in just a very limited number of places inside a building, like going up and down the stairs or taking an elevator, then Map Knowledge can assist in determining these possible locations. Acceleration and Orientation are used in the Pedestrian Dead-Reckoning (PDR) component to track the continuous movement. Finally, if a WiFi Scan is available, it is used to extract a runtime WiFi fingerprint. Such a fingerprint is compared with those in a fingerprint database (created via crowd–sourcing). Estimations of these components are merged by the Particle Filter to obtain a single location estimation. At the end of this process, if WiFi Scan information is available, it is annotated with the estimated location and used to update the fingerprint database.

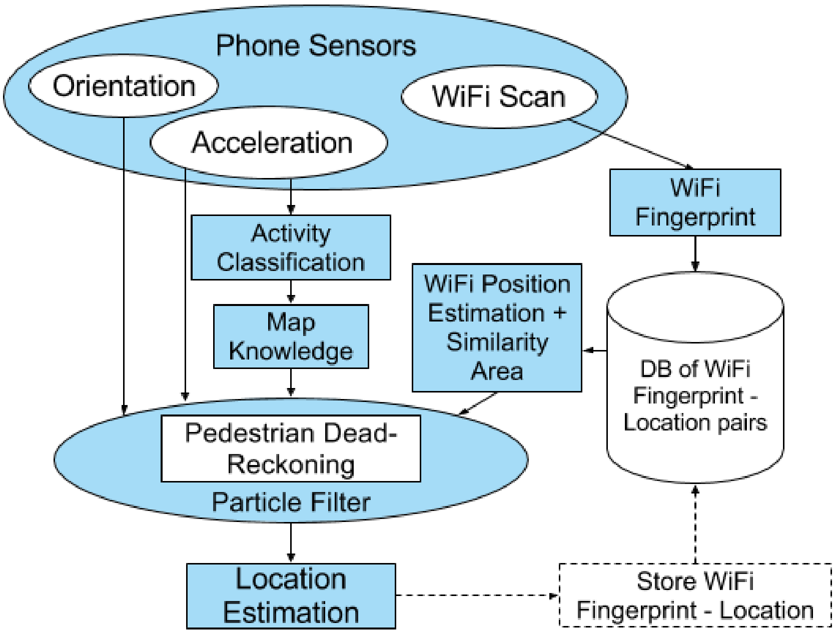

Phone Sensors
Orientation
WiFi Scan
Acceleration
WiFi Fingerprint
Activity Classification
WiFi Position Estimation + Similarity Area
Map Knowledge
DB of WiFi Fingerprint - Location pairs
Pedestrian Dead-Reckoning
Particle Filter
Location Estimation
Store WiFi Fingerprint - Location

Figure 3.1: Schematic of HiMLoc hybrid localization mechanism.

The main two components of HiMLoc are presented next: (1) the Pedestrian Dead Reckoning driven by Activity Classification for continuous tracking; and (2) the WiFi fingerprinting component.

3.1.2 Pedestrian Dead Reckoning Component

The PDR estimates successive positions of a moving pedestrian starting from a known position through estimations of traveled distance and direction of movement. HiMLoc uses this method to track the position of a person when walking. However, in order to know when the person is walking, HiMLoc involves an activity detection phase performed by the Activity Classification component.

Based on the detected activity, the system chooses how to interpret user's movements. HiMLoc needs this component to distinguish between vertical movements (going up/down stairs and elevators) and horizontal movements (walking). With the help of Map Knowledge, activity recognition can provide even more information about the user's location. Certain activities like going up or down stairs or taking an elevator can be performed only at a limited set of known locations inside a building. Getting the activity right has the effect of providing the needed periodic correction to the PDR in order to reduce the accumulation of error caused by noisy sensors and other interference on long tracks.

The most suitable sensor for activity recognition is the accelerometer as it is an inertial sensor permitting energy-efficient sampling at a high rate for continuous tracking. Most activities are performed similarly every time and their acceleration patterns can make them recognizable. All smartphones sense the acceleration on three axes orthogonal to each other. Considering that the sensitivity of the accelerometer is the same on all three axes, the acceleration magnitude will always indicate the same values, no matter how the phone is oriented:

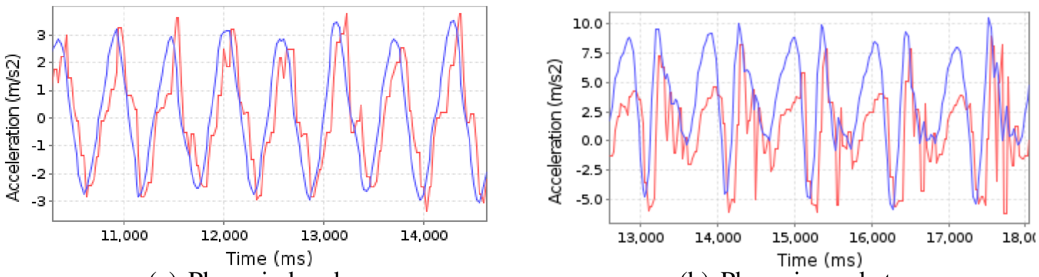
$$a = \sqrt{a_x^2 + a_y^2 + a_z^2} - g \tag{3.1}$$

where g is the Earth gravity, ax, ay and az represent the acceleration perceived on the Cartesian axes Ox, Oy and Oz respectively in the phone's frame.

HiMLoc was designed to permit two ways of carrying the phone: in pocket and in hand. For the case with the phone in pocket only the front of trousers was considered. In the case of carrying the phone in hand this represents the scenario of the user holding the phone straight in front and in direction of movement. A common aspect between these two cases is that the phone can be considered static relative to the user's body.

The system was trained to recognize the following activities: stationary, walking, elevator going up, elevator going down, going up on stairs, going down on stairs, opening and closing doors. Each of these were considered in the two scenarios mentioned before: carried in hand and carried in pocket.

Horizontal movements



(a) Phone in hand.

(b) Phone in pocket.

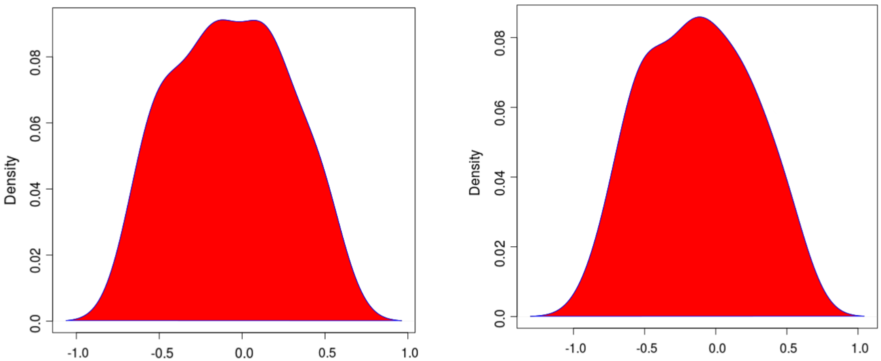
Figure 3.2: Acceleration pattern (raw acceleration with red and filtered acceleration with blue) when walking.

If the activity performed by the user is determined to be walking, either with the phone in pocket or with the phone in hand, the user's movement is tracked on a horizontal plane, using traveled distance estimation and direction of walking. Next, this presents how these estimations are obtained.

3.1.2.1 Distance Estimation

Figure 3.2(a) presents the acceleration magnitude pattern of walking with the phone in hand. The red curve indicates the raw acceleration and the blue curve is the same acceleration after adding a weighted average smoothing filter. Each step leaves the signature of a high spike in acceleration, caused by the heel touching the ground, followed by a deceleration. To estimate the traveled distance, HiMLoc first smooths the acceleration to eliminate some of the noise, then applies a zero crossing method to count the number of steps. In the case of walking with the phone in pocket, the same technique of counting the number of steps is used, but because the vibrations are more intensive when holding the phone in pocket, a low-pass filter is also used.

Step length is computed as a linear function of stepping frequency [49]. HiMLoc computes the traveled distance as the sum of each step's length. This solution gives good results, but has its limitations. To evaluate the efficiency of this method of distance estimation on a window size 3.2 seconds of uniform walking I designed an experiment. By doing several walks at different speeds it was observed deviations of the distance estimation from the actual traveled distance. The density of these deviations is represented in Figures 3.3(a) and 3.3(b). Errors of up to 15% were observed that can have negative effect on the accuracy of the system. This solution enforces the particle filter to correct for this deviations from the exact distance, as it will be presented in later subsections (Particle Filter).



(a) Phone in hand.

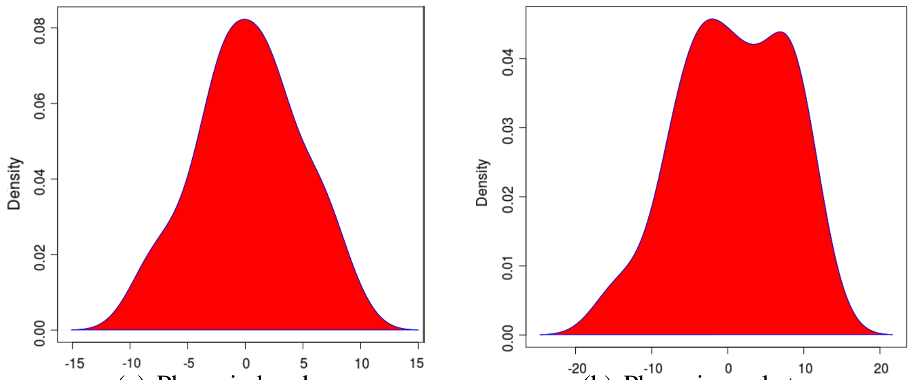
(b) Phone in pocket.

Figure 3.3: Deviations of the estimated distance from the real traveled distance

3.1.2.2 Direction Estimation

The direction of movement also needs to be estimated. Considering that each smartphone has a compass, this can be a good indicator of the direction of movement in the Earth frame. It is true that compasses are sometimes affected by magnetic interference inside a building caused by the building structure and electric equipment, but our observations indicate this interference to be just isolated and not very disturbing when the person is moving at normal walking speed. Using a time frame to average the compass indications can eliminate some of the local interference.

Evaluating the compass sensor on a long walk, it stands out that the human body has a slight rotation in motion which is detected by the compass. Choosing a good size window to average the compass data can overcome this rotation in order to provide a more reliable direction of movement. A window size of 3.2 seconds usually captures 6 steps of movement at average walking speed, which allows for every two consecutive steps to cancel each others rotations. This can be observed from Figures 3.4(a) and 3.4(b), where the compass indication is averaged over the time window and compared to the true direction of movement.



(a) Phone in hand.

(b) Phone in pocket.

Figure 3.4: Deviations of window averaged direction from the true direction of movement

HiMLoc considers the phone to have a static position relative to the body throughout the movement. To compensate any deviation of the phone from the user's frame orientation, a correction angle is determine after the initial few steps on the corridor, when the information of the corridor orientation is known from the Map Knowledge, or after two landmarks where the position of each landmark on the map is known by assuming the walking movement to be in a straight line.

The distance and direction corrections are considered in the Particle Filter when choosing a distance and direction for each particle to progress the PDR.

If the compass deviation suddenly gets close to a right angle, the system infers that the user has left the corridor, either to go into a room or made a turn to another corridor. This event is considered as encountering a landmark and the position of the closest one is used to correct the system as it will be described in the following subsections.

Vertical movements

Elevator movements present a specific pattern, with significant accelerations when the elevator starts and stops. Figure 3.5 presents these two events of the elevator denoted by the two large spikes in opposing directions. The number of floors ascended or descended by the elevator can be determined from the difference of times between the start and the stop of the elevator movement. In both cases of carrying the phone (pocket and hand), the elevator acceleration presents similar patterns. The movement between floors is more broadly covered in my Master thesis [88], presenting the principles of movement in the PDR component.

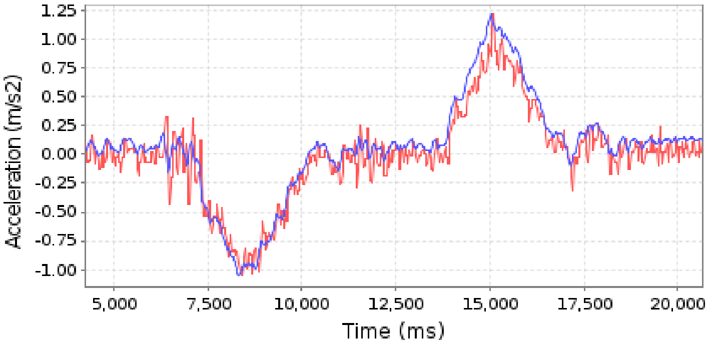


Figure 3.5: Elevator acceleration showing a large spike at start followed by an opposing spike when stopping.

For the activities of going up and down the stairs, a similar method with walking of step detection is used. By counting the number of stairs ascended or descended, the new level can be accurately determined as shown in [88]. Figure 3.6 presents the acceleration magnitude for the activity of going down on stairs.

Classification performance

Weka[1] machine learning library was used to classify the acceleration samples into activities. The training set consisted of 176 instances of activities from two participants manually annotated with the right activity, each activity having at least 6 in-

[1] http://www.cs.waikato.ac.nz/ml/weka/

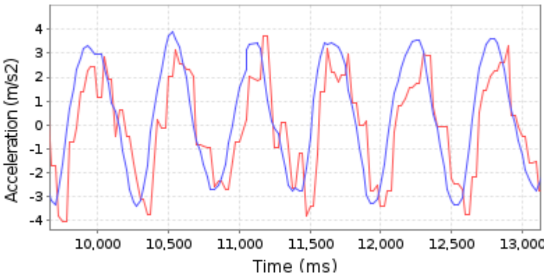


Figure 3.6: Going down the stairs with the phone in hand.

stances. These activities were: stationary, walking, going up on stairs, going down on stairs, going up by elevator, going down by elevator, opening and closing doors. All these activities were considered for both cases with the phone in pocket and with the phone in hand. Features were selected from the time domain (mean, variance, standard deviation, first integral (velocity), second integral (distance) and interquartile range) and from the frequency domain (energy and entropy) of the acceleration magnitude. A wider description of this activity classification is presented in my Master thesis [88], which was also the subject of that work for assisting the PDR system to detect movements between floors. That investigation showed above 85% accuracy for activity recognition.

3.1.3 WiFi Fingerprinting Component

This on its own can be seen as a stand alone indoor localization method, but in HiMLoc is used to complement the PDR estimation by providing additional fix points to reduce the accumulating error in PDR, integrated by a particle filter.

At run time, the vector of top five strongest APs and their signal strength values are selected and compared to the fingerprints in the database. The closest matching fingerprints are selected using Euclidean distance in the signal space (as in [89]). Fingerprints are stored in the database in groups representing cells. Each cell is a square with the sides of 1 meter and together they form the grid covering a floor plan. To support continuous update of the training set of WiFi fingerprints, all fingerprint are annotated with the time when they were collected. Newer fingerprints get a higher priority in fingerprint selection, thus creating a simple solution to adapt for wireless infrastructure changes or other changes over time. The centroid of the closest three fingerprints gives the location estimation of the component.

A crucial observation underlying this use of WiFi fingerprinting is that accuracy of WiFi fingerprinting varies in space – some areas provide a higher accuracy of localization than others. Figure 3.7 illustrates this point. Here accuracy is based on the distance between ground-truth position and the estimated position with WiFi fingerprinting. Positions with highest accuracy are shown in green whereas those with lowest accuracy are colored red.

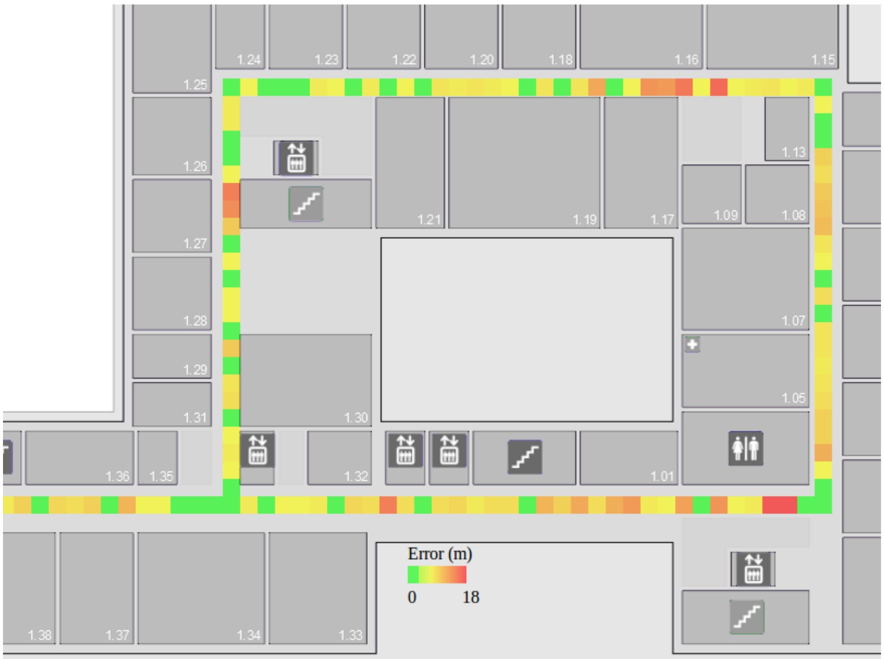


Figure 3.7: Spatial distribution of WiFi fingerprinting based location estimation errors on the floor plan.

In order to know when a WiFi location estimation is reliable, we introduced the notion of *similarity area of a WiFi fingerprint*, which is the area described by all points in the fingerprint database with a fingerprint *close* to the one at runtime. A threshold for the Euclidean distance in the signal space between the runtime fingerprint and each fingerprint in the database is used to define closeness. We set this threshold empirically to 12.5 in our implementation. The area spanning all *close points* determines the similarity area. Figure 3.8 shows the correlation between the estimation error and the similarity area.

It was observed that errors of estimation are much lower when the similarity area is small. While the errors are not necessarily larger when the similarity area is higher, they are more variable than to the left of the chart, so the solution was to consider the estimations with a low similarity area as offering a higher certainty of their indication. In fact, having a small similarity area is an indicator that the fingerprint is well distinguishable from other fingerprints and similar fingerprints can be found in just a

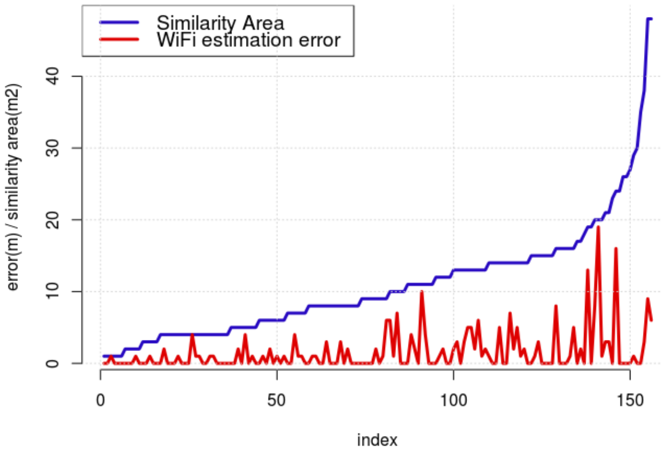


Figure 3.8: Overlaying the estimation error and the similarity area indicates a smaller error rate where similarity area is small, while more larger errors are seen for larger similarity areas.

small area in the building. HiMLoc assigns higher weights to the estimations with a low similarity area as they are considered to be more accurate.

3.1.4 Fusion via Particle Filter

HiMLoc uses a Particle Filter to integrate all estimations from Activity Classifier, Map Knowledge, WiFi positioning component and PDR's variables (distance and direction). The role of the particle filter is to correct these estimations that are possibly affected by noise. This is done by investigating all possible activities based on their probability, determining the possible distance deviation and compass deviation in each time window.

Each particle has its own PDR component where it chooses an activity for each time window based on the probabilities provided by the Activity Classifier for each activity, a distance deviation for walking in the time window and a compass deviation. The compass deviation at the window level (Figures 3.4(a) and 3.4(b)) and the distance deviation (Figures 3.3(a) and 3.3(b)) can be tightly fitted by a normal distribution. Based on their observed behavior in practice, the probability of encountering any deviation is:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/2\sigma^2} \tag{3.2}$$

where, x is the chosen deviation, μ is the mean and σ^2 the variance of observed model.

Based on the probability, each particle selects its own correction values to compensate for the estimated value. In turn, this probability will affect the weight of the particle. The activity recognition variable gets its probability from the classification confidence of the Activity Classifier.

The other purpose of the Particle Filter is to prevent the system from getting lost when the PDR starts accumulating errors. When there is an external assistance, for instance a position is indicated by the Map Knowledge (e.g. because of a corner), particles weights are updated inversely proportional to the distance between the particle's position and the assistance indicated position. In the case of the WiFi component estimations, the confidence of the estimation is determined based on the similarity area. As it can be observed from Figure 3.8, when similarity area is small, the errors of WiFi location estimation tends to be small, so these estimations should be assigned a higher confidence. An exponential model provides the confidence of WiFi location estimations by indicating high confidence when the similarity area is small and low confidence when the similarity area is high. The weight of each particle is updated based on WiFi estimation confidence and on the distance between the position of the particle and the WiFi estimation.

So, the weight of a particle is updated as a sum of all the weights of the probabilistic variables:

$$w_i = w_0 \oplus w_a \oplus w_o \oplus w_d \oplus w_f \tag{3.3}$$

where w_i is the final weight, w_0 is the initial weight of the particle and w_a, w_o, w_d, w_f are the weights computed for the particle's variables (activity selection, orientation, distance and WiFi fingerprinting based fix assistance if available) based on their likelihoods.

The life cycle of the Particle Filter begins with all the particles having the same weight at the starting point. There are three steps repeated by the Particle Filter in a loop:

- Selection of particles. At the start of an iteration, some particles are sampled to progress and create the new group of particles. This selection is done based on their weight.

- Weight update based on the variables selected by the PDR. Each particle randomly creates its own set of variables and progresses the particle, updating its weight accordingly.

- Observations about the environment update the particles' weight. If there is an external contributor like the Map Knowledge or the WiFi positioning, particles closest to the specific positions get higher weights.
- Weight normalization. The weight of all particles are normalized so that they sum up to one.

3.2 HiMLoc Evaluation

The performance of HiMLoc was evaluated in three different scenarios. First scenario was designed to evaluate the performance of the localization system on one floor of an office environment where frequent landmarks were present, corners and WiFi assistance, with a large training set of WiFi fingerprint-location pairs. The second scenario was to evaluate HiMLoc performance for movements that span multiple floors. Lastly, the benefits of hybrid approach taken in HiMLoc are assessed in comparison with the underlying approaches (WiFi fingerprinting and PDR) on which it is based.

Single floor of an office building

For this an experiment was conducted in the Informatics Forum, which is a modern office building. Before the experiment, the WiFi fingerprinting component was trained by collecting multiple fingerprints on the first floor annotating them with their precise location. This was done in a crowd-sourced manner, data being collected my multiple users to be joined in a single database on the server side application. There are already solutions available that can automate this process much faster, like WiFi-SLAM [56], but we chose this approach to avoid the complexity of other systems and to have a higher confidence on the training set for the WiFi localization component that would serve all the other participants. Similarly, the activity classifier was trained with the sample acceleration patterns from two participants and the classifier was used to classify activities of all the other participants.

The experimental evaluation of the system involved 5 participants, all with Nexus One phones running Android 2.3. To evaluate the accuracy of the localization solution, the following experiment setup was used. A track of about 100m was chosen on the corridors with multiple points (20 points), representing entrances to offices adjacent to a corridor, selected to offer the ground truth of our evaluation. Three participants walked on the track with the phone in hand and two with the phone in pocket. At the beginning of the track their time was synchronized with a clock and for every encounter of a ground truth position, the time was recorded. Location estimation errors were computed for each ground truth location as the Euclidean distance to HiMLoc's location estimation.

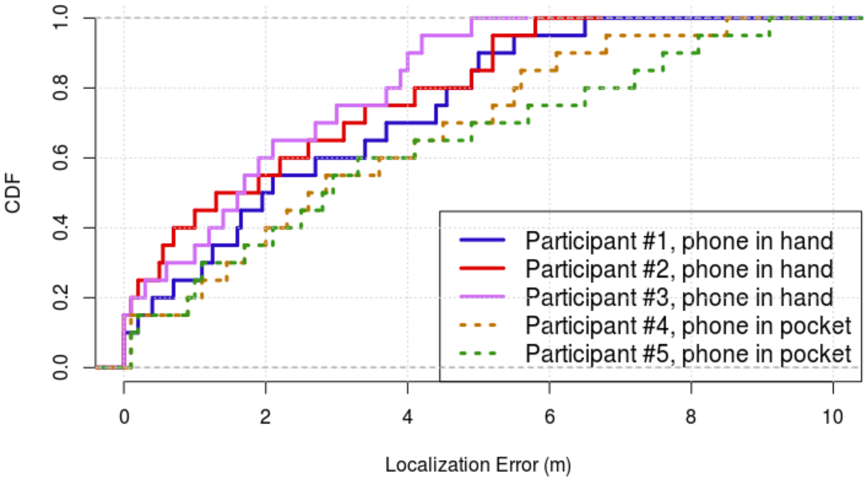


Figure 3.9: CDF of location estimation errors.

The localization error of HiMLoc is presented in Figure 3.9. It can be seen that the accuracy for the case of carrying the phone in pocket tends to be lower than the case with the phone in hand. This is because counting the number of steps with the phone in pocket is relatively a harder task.

For the following experiments only the case of carrying the phone in hand was considered.

Moving between floors

For the second experiment, the second floor of the same building was also added to the experiment to span the movement over two floors. Starting from the same starting point on the first floor, the route went up the stairs and followed the second floor corridor similar to the first floor track. This experiment was designed to evaluate the training set of the WiFi component when moving between floors in addition to the normal walking conditions. In the first instance all the WiFi fingerprints from the entire building were in a single training set. The effect of this was a lot of confusion in the WiFi component of HiMLoc, making mistakes between floors (Figure 3.10). As a consequence, we decided to rely on the PDR to estimate the floor and use only the fingerprints from the same floor as training set for the WiFi component. An extended evaluation of floor detection when moving between floors is provided in my Master thesis [88] as part of the proposed PDR system in there. That indicated a very high accuracy performance which is transferred to HiMLoc.

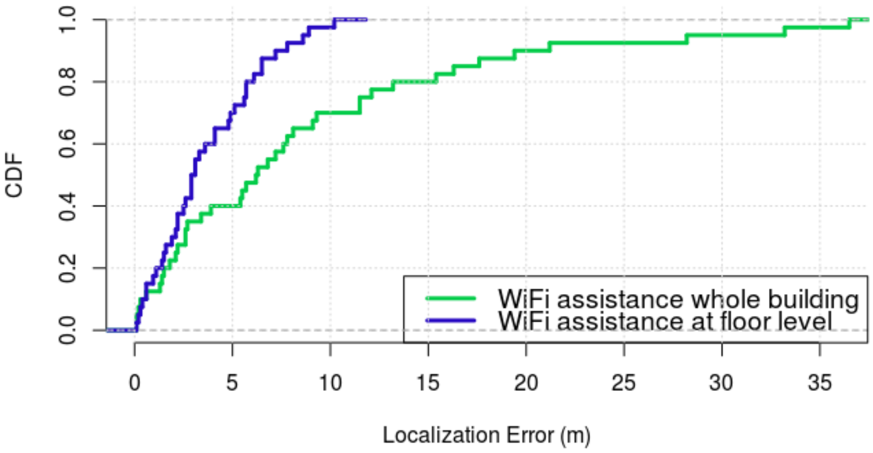


Figure 3.10: CDF of localization errors moving between two floors.

Benefit of hybrid approach

The next evaluation was to compare the performance of the hybrid approach HiMLoc with each of its two underlying localization components: PDR and WiFi fingerprinting. WiFi fingerprinting alone does not work where there is no proper WiFi coverage and continuous scanning has negative implications on the battery life. Figure 3.11 presents the power consumption for two different smartphones. While the accelerometer is about 5mW and the magnetic about 60mW, the power consumption for performing WiFi scans is an order of magnitude higher than the magnetic sensor and two orders of magnitude greater than the accelerometer. HiMLoc uses the cheaper sensors (compass and accelerometer) for continuous sensing and occasionally WiFi scans, with the effect of reducing the power consumption of the system.

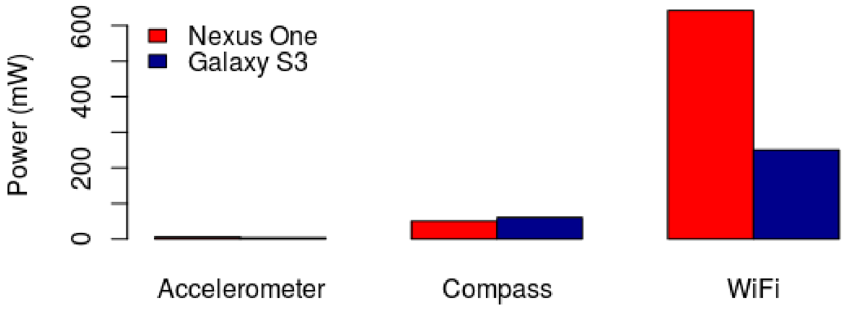


Figure 3.11: Power consumption comparison between the three sensors on two devices, Nexus One and Galaxy S3.

To evaluate the improvement of HiMLoc over just PDR, the same scenario over two floors was used, as presented in the previous section. The route involved walking on the corridor at the first floor, going up on stairs to the second floor, walking on the corridor at the second floor, walking in a large open space, resting on the couch, walking on the corridor again, taking the elevator back to the first floor and walking back to the starting point. Using this track, we compared the performance of the PDR (with activity recognition) with the performance of HiMLoc (Fig. 3.12). We can see that HiMLoc as a whole performs better as median error (improving from 3.4 meters to 2.2 meters) but also having lower errors overall, due to occasional assistance from WiFi fingerprinting when there are long periods of no assistance from Map Knowledge in the PDR, with the 90-percentile improving from 9 meters to 4.4 meters.

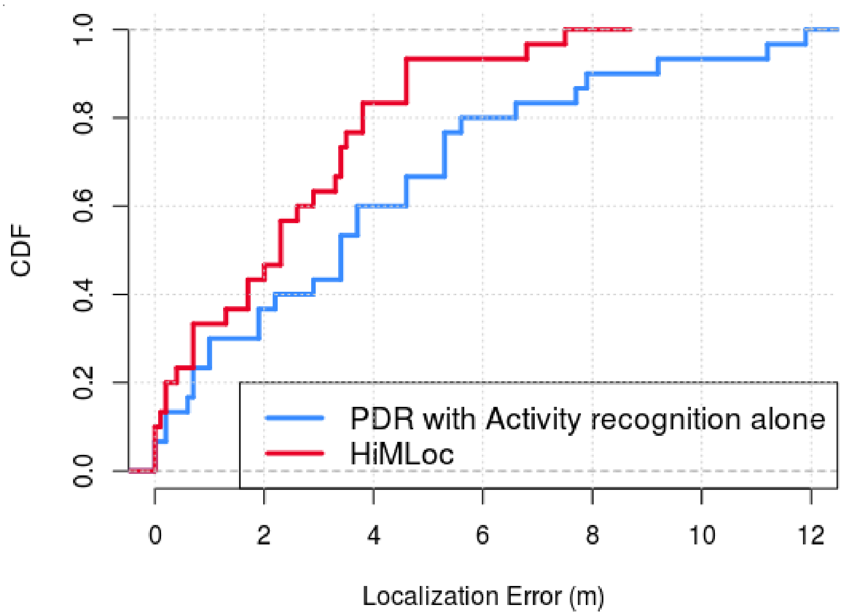


Figure 3.12: Comparison between PDR alone (without the WiFi component) and complete HiMLoc.

The effect of the WiFi component is to penalize particles that deviate from the proper direction of movement (when they choose higher deviation from the value indicated by the compass) and reinforce those who are moving closer to the corridor line.

3.3 Pazl: Using HiMLoc for Crowdsourced Indoor Wireless Network Monitoring

This section presents an application built on top of HiMLoc we call Pazl that can generate radio coverage maps through crowdsourcing wireless measurements from many people inside a common space. Figure 3.13 shows the high-level system architecture of Pazl. Measurement and sensor data is collected on phones and then uploaded to the cloud where, annotated with location is stored for later use. In the merging phase measurement data from different phones is aggregated to create reports and render the required coverage maps.

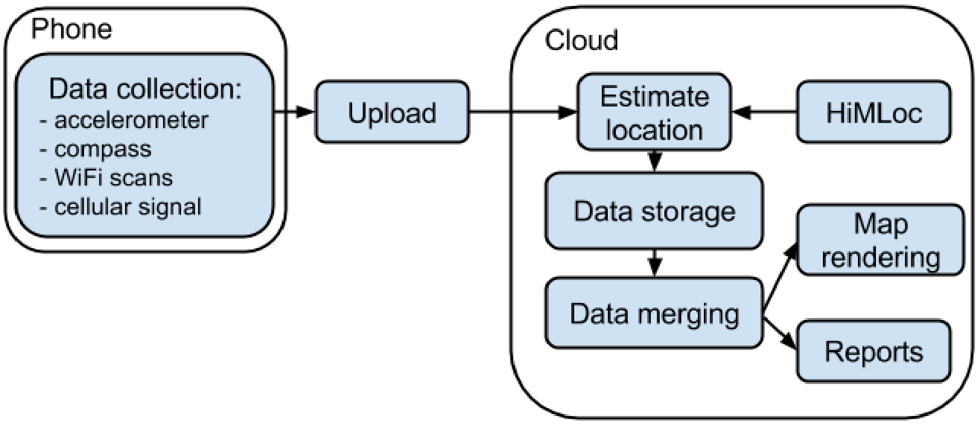


Figure 3.13: Pazl system components.

3.3.1 Pazl System Components

Pazl implementation consists of two parts: a mobile application that collects data from the phone's sensors and wireless measurement data (e.g., WiFi scans, cellular signal strength) and a server application that receives the data to be processed for estimating the locations of wireless measurements following the hybrid mechanism described above as well as for associated map visualization and analysis. The phone application is developed to run on a large variety of Android phones. To provide increased availability and concurrent access, the server application is developed to run on the cloud (Google App Engine in our implementation).

On the phone, acceleration, orientation and WiFi scans are collected only when the user is moving. When the phone is stationary, the compass and the radio interface are not used to save energy. Only the accelerometer is left on to run at a lower frequency just to sense when the user is moving again.

The frequency of WiFi scans was chosen to be one scan every 20 seconds, which is a compromise between keeping the energy consumption low, with each WiFi scan imposing an extra energy consumption on the phones, but also gather enough data to assist the PDR estimation more often.

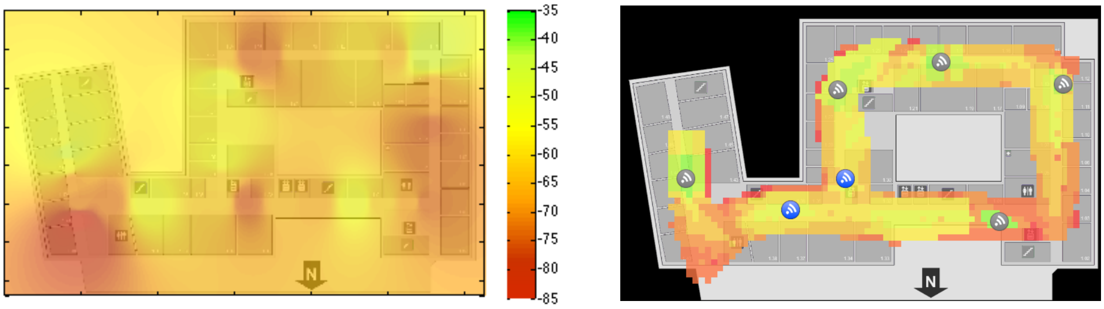
Data Upload

Data can be uploaded on demand, at a specified frequency, or it can be dependent on available connectivity options. If the phone is connected to a 3G network, data upload can be delayed until a connection to a WiFi network is established. Other upload policies can include other statuses of the phone like if the phone is charging and others. In the current implementation we considered only upload on demand.

Merging data and Map rendering

When the wireless measurement samples are obtained from different devices, they are stored in the cloud backend together with the time and estimated location of where they were collected (via HiMLoc). For visualization of the WiFi/cellular coverage maps, we use the Inverse Distance Weight based spatial interpolation [90]. Data is aggregated at a cell level of size 1 m^2, by the median value if there are more measurements collected within the same cell. Selecting just a small set of wireless measurement samples based on the time when they were collected, dynamic reports can be generated, like the behavior of the network in a particular time period over several days or between different times within a day.

3.3.2 Case Study 1: Indoor WiFi Monitoring

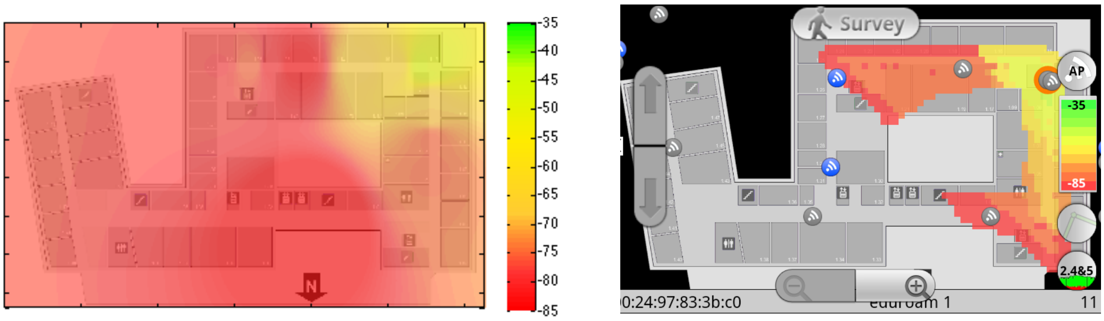


(a) With Pazl

(b) With Ekahau Mobile Survey

Figure 3.14: WiFi coverage on a floor in dBm.

Next, Pazl was evaluated in a small scale experiment that reflects crowd–sourced indoor WiFi site survey. The experiment was performed during a full working day (from 10am to 6pm), with 5 participants. They were asked to carry their Nexus One phones with them while moving freely during the day inside the building. No specific training was required beforehand other than just installing the Android app. This analysis focuses only on the first floor, even though participants were allowed to move freely in the rest of the building using elevators and stairs as demanded by their day tasks.



(a) With Pazl

(b) With Ekahau Mobile Survey

Figure 3.15: Coverage of an AP in dBm.

The WiFi coverage maps obtained with Pazl are compared with the ones from using Ekahau Mobile Survey tool [28] in Figures 3.14 and 3.15. The Ekahau application shows the signal coverage only near the locations where measurements were collected, indicated with distinctive colors, representing different values of the Received Signal Strength (RSS). For the coverage representation using Pazl, we tried to keep the same color scheme as Ekahau to allow comparison between the two systems. Pazl estimates an extended coverage map via spatial interpolation for the entire floor plan, even for areas with no measurements. In the coloring scheme green indicates very good RSS, red indicates poor RSS, and other values of RSS are represented with a mixture of the two colors. Comparison between the two systems can be done through color correlation or values comparisons in areas where they could both estimate the coverage, in particular on the corridors.

The coverage for the floor is shown in Figure 3.14 and as it can be observed, the poor RSS was identified by both systems in the bottom left (Pazl indicated -78dBm, while Ekahau indicated -75dBm) and bottom right sides of the floor plan (Pazl indicated -66dBm, whereas Ekahau indicated -70dBm). We can also observe that both systems detect stronger RSS in almost the same places, in vicinity of APs. As for differences, Pazl estimated a region with low signal strength in the middle of the corridor, near the elevator, indicating -75dBm, whereas Ekahau recorded the signal strength in that area to be -65dBm.[2]

[2] Based on manual wireless site survey in that area, we observed that the max signal strength varies

The corresponding results for a single AP is presented in Figure 3.15, with very close match observed between the two systems. A good coverage of the AP is detected by both systems closer to where the AP is located and also on the corridor going top to bottom in the figures. Coverage is relatively worse along the other corridor going from right to left which we believe is because AP does not have a clear view of that corridor as it is occluded somewhat by the corner where the two corridors intersect.

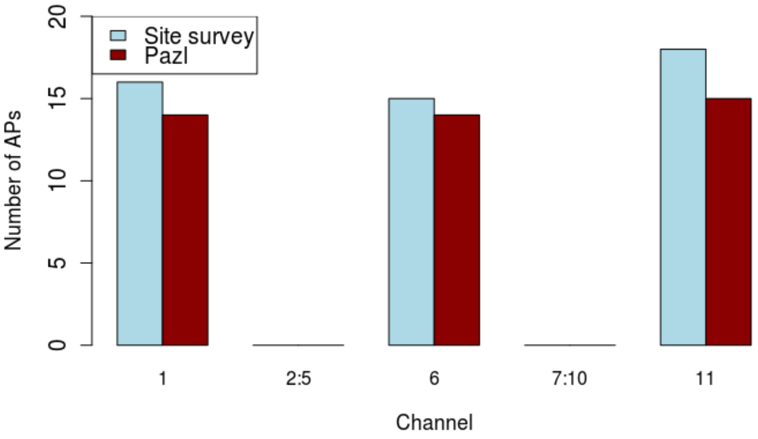


Figure 3.16: Channel usage distribution.

Channel usage distribution obtained with Pazl for a floor was compared with a manual site survey (Figure 3.16). The difference is between 1 and 3 APs per channel. This maybe because some of the APs located in other parts of the building can be sensed only in specific areas, which might not have been reached by any of our participants over the period of the experiment.

Pazl was provided with physical location of campus WLAN APs in the building to estimate the coverage range (maximum distance of propagation) of APs that are seen from a floor (Figure 3.17). These are not the exact maximum coverage range of APs because samples are limited to the areas traversed by participants. Still this experiment demonstrates that Pazl can detect problematic scenarios such as the APs that have unusual coverage. Analyzing a particular case, an AP located at the fifth floor was sensed at the first floor, over 55 meters away. This is due to the layout of the building which is mostly glass inside and has a large open area in the center. Observing this in addition to the fact that this fifth floor AP shares one of the heavily used 2.4GHz channel with other APs on the first floor, it is obvious that channel allocation is poorly done risking interference related performance degradation from a user perspective.

between -71 and -76dBm and that the nearest AP is shadowed by the corner of the wall. With only one run, this area may have witnessed direct line of sight when surveyed using the Ekahau tool.

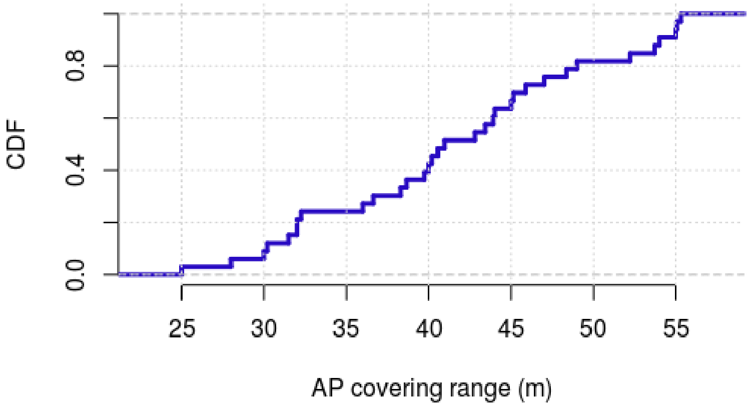


Figure 3.17: Coverage range of APs on a single floor obtained with Pazl.

3.3.3 Case Study 2: Indoor Cellular Coverage Measurement

The functionality of Pazl was extended to collect measurements of the cellular signal strength as well. Cellular signal strength was collected in an experiment involving three participants in similar conditions as presented in the case of WiFi mapping. The phones used in this experiment were two Nexus 4 devices (collecting 85% of the measurements) and a Samsung Galaxy S3 (15% of the number of measurements). The mapping is presented in Figure 3.18.

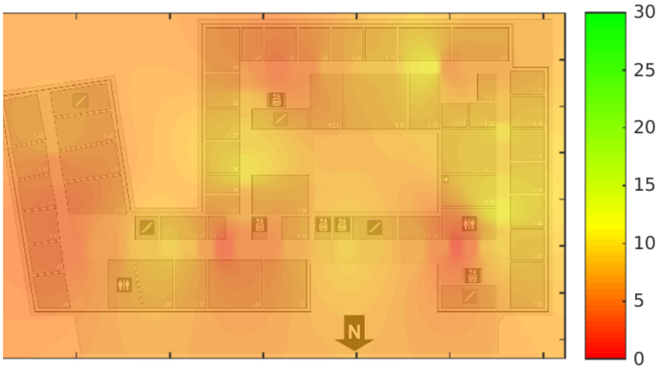


Figure 3.18: Cellular coverage estimated by Pazl in ASU.

To validate the results we did a direct comparison between the cellular signal indicated by Pazl at a location and the actual observable value at that location. Figure 3.19 presents the CDF of errors in ASU[3] (Arbitrary Strength Unit) between Pazl and a manual measurement of ground truth cellular coverage for 20 distinct locations. As can be seen, the median error is 0 ASU, while the 90-percentile is 2 ASU.

[3] conversion from ASU to dBm: dBm = ASU – 116

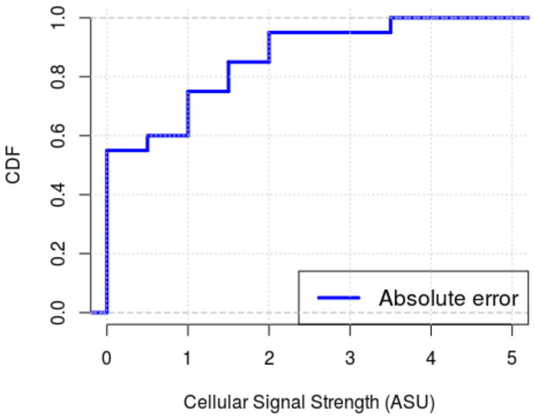


Figure 3.19: Error in cellular estimation by Pazl compared with the ground-truth.

An important observation is that the position of the phone in relation to the human body affects the signal strength in a significant amount. For example rotating 360 degrees with the phone in hand indicates differences in signal strength up to 4 ASU. In mapping the cellular signal strength the exact position of the phone in relation to the human body was not considered. By measuring the ground-truth, multiple measurements were collected in different directions and using an average to factor out this effect. Thus, the close match seen in signal strength measurements with automatically annotated locations using Pazl and manual ground truth at those estimated locations indirectly validates the effectiveness of HiMLoc localization mechanism employed in Pazl.

3.4 Discussion

One important subsystem directly affecting the overall performance of HiMLoc is the activity recognition component. Although explored in greater detail in my earlier work [88], it is worth discussing the impact of this component and the approach taken in this exploration with regards to activity recognition.

The activity recognition component of HiMLoc was trained using training data collected from two participants, both males in their twenties. Although a small number of participants contributed data to train the system, the rest of the experiments were conducted with users reflecting very similar physical construction, thus the impact on activity recognition across users being limited as also identified in other works [6]. Nevertheless, activity patterns vary substantially between different demographics [49], and the importance of training on a large population cannot be neglected for a production system that needs to operate across a larger diverse population. Chapter 5 also highlights the importance and need to customise the estimator to specific users, presenting differences of up to 10% which can be observed between a general population trained classifier and a specialized classifier for each individual. This aspect of existing differences between user patterns is also highlighted by the potential to perform identity detection simply from gait acceleration patterns [91], indicating subtle variations in how different people perform these actions.

3.5 Summary

This section presented a new hybrid indoor localization system that combines in stages the information inferred from multiple sensors. Such, the acceleration signal is used to estimate the user activity, and traveled distance. The orientation of the phone is estimated with assistance from an electronic compass. WiFi scans are used to estimate a location which is assessed for accuracy based on previous observations. These are integrated through a particles filter to estimate a single location. Achieved accuracy of location estimation is within 3 meters on average.

An application was created based on this indoor localization system to generate radio maps from participants moving freely through a building. Comparing with commercial applications this achieves similar performances for WiFi networks and it extends to cellular coverage as well.

Chapter 4

A Semi-Supervised Learning based Approach for Indoor-Outdoor Detection

The environmental context of a mobile device determines how it is used and how the device can optimize operations for greater efficiency and usability. This chapter explored the problem of detecting if a device is indoors or outdoors by employing semi-supervised machine learning methods and using only the lightweight sensors on a smartphone. Among the methods explored in this chapter, a particular semi-supervised learning method called co-training is observed to perform best on this task. It is able to automatically learn characteristics of new environments and devices, and thereby provides a detection accuracy exceeding 90% even in unfamiliar circumstances. It can learn and adapt online, in real time, at modest computational costs. Thus the method is suitable for on-device learning. Implementation of the indoor-outdoor detection service based on this method is lightweight in energy use. It is shown to outperform existing indoor-outdoor detection techniques that rely on static algorithms or GPS, in terms of both accuracy and energy-efficiency. It uses fast classification and incremental learning techniques which can be run entirely on the phone, thus preserving user privacy and saving communication costs.

This research was a joint work with Panagiota Katsikouli, Rik Sarkar and Mahesh K. Marina and was presented in the Proceedings of the ACM International Conference on Embedded Networked Sensor Systems (SenSys) [76], with an earlier version appearing as a poster in Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom) [92]. As part of this collaboration I led the development and experiments and unless stated otherwise, the work presented here was undertaken by myself. Significant contribution by Panagiota Katsikouli is reflected in assessing the feasibility of using machine learning for indoor-outdoor detection and characterizing the collected dataset.

The chapter begins by presenting the limitations of previous methods for indoor-outdoor detection and showing the opportunity for machine learning to bridge this gap. After presenting the collected dataset, a set of machine learning techniques are evaluated for their performance on our task, including unsupervised learning and semi-supervised learning. It was observed that Co-training performs the best on this task and details of how this was achieved is presented further in this chapter. Taking these observations, co-training is employed in a real-world application running on a smartphone to enable and disable the WiFi interface based on context (switch off outdoors and switch on indoors). The energy savings of 60% clearly show the benefit of using this method.

4.1 Preliminary Exploration and Critique of Existing Indoor-Outdoor Detection Techniques

This section presents the characteristics of sensor signals in indoor and outdoor environments, and how existing approaches perform with respect to detecting those environments. As previously mentioned, the most common methods are detection using GPS based methods (as in [6, 59]) and using IODetector [14]. I developed an Android application that records continuously or on demand the values of sensors available on the smartphone, and through a graphical interface users can indicate the ground-truth state (indoor or outdoor) as they transition from one to the other in different environments, which is also recorded. This initial exploration is intended to show the limitation of the two most commonly used methods for distinguishing the contexts. A small dataset was collected in the campus area across 5 university buildings, capturing the transition between indoors and outdoors, which is the basis for this initial analysis.

4.1.1 GPS based IO Detection

GPS signals are usually available outdoors where the sky is directly visible, and are often weak or unavailable indoors where the sky is obscured by ceilings and walls. Thus, the estimated accuracy of GPS localization can be used to infer if a user has moved from outdoors to indoors or vice versa [6].

The main drawback of GPS is its high power consumption – it is the most power hungry sensor on a smartphone. This was observed using a custom mobile application developed to enable and disable each sensors and measure their power consumption on a Samsung Galaxy S3 phone. Using the Monsoon Power Monitor[1] device the power consumption was measured precisely the by bypassing the battery input to the device with the measurement tool as a power source as shown in Figure 4.1.

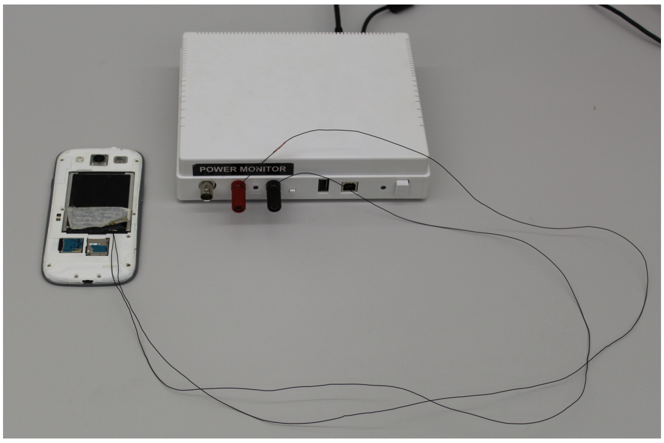


Figure 4.1: The setup for measuring the power consumption from a Samsung Galaxy S3 devices, replacing the battery input with the Monsoon Power Monitor device.

This experiment shows that GPS consumes 370mW in operation (Figure 4.2) – much higher than all other sensors. Note that power consumption for WiFi interface in Figure 4.2 refers to the average over one WiFi scan while the power consumption for the cellular interface is for obtaining a passive signal strength measurement. Repeating the experiment with other smartphones shows similar behavior.

Evaluating the GPS-based localization as a method to detect indoor/outdoor state indicates that it is not particularly reliable. Figure 4.3 shows how GPS locations can be estimated inside of a building frequently with high confidence, similar to the outdoor behavior. GPS can sometimes get a satellite fix indoors, for example when the user is close to a door or window, which can be beneficial in localization [60], but reduces its reliability as an indoor-outdoor classifier. Based on these experiments, it was observed that GPS often continues to report fixes for up to 10-35 seconds after the transition from outdoors to indoors has happened; this is illustrated in Figure 4.3. Consequently, as shown in the following sections, this only gets an accuracy in range of 70% to 80%

[1] https://www.msoon.com/LabEquipment/PowerMonitor

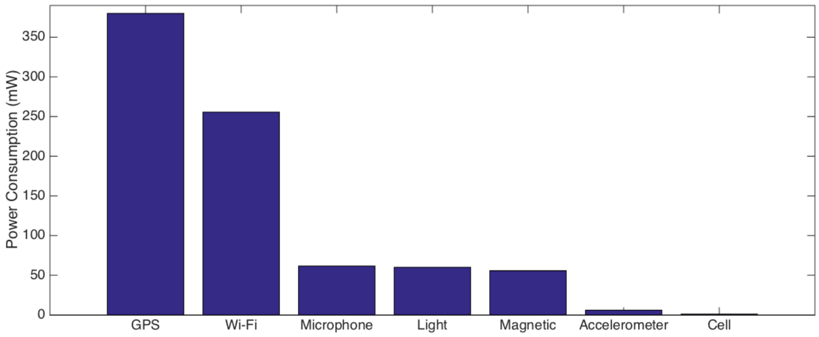


Figure 4.2: Power consumption of various sensors on a modern smartphone (Samsung Galaxy S3).

when using GPS localization inaccuracy (in comparison with an appropriately chosen threshold) as an indicator to detect indoor/outdoor state.

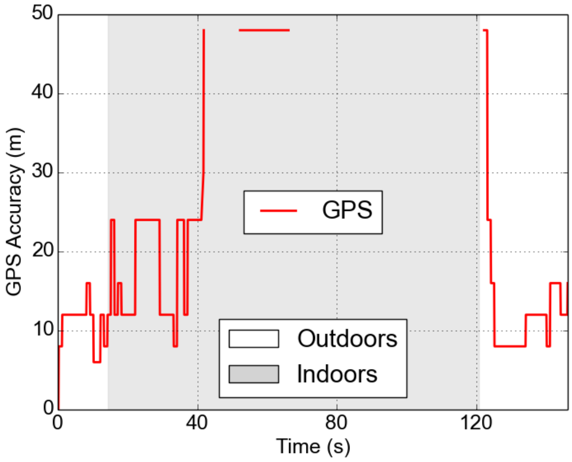


Figure 4.3: GPS accuracy at outdoor-indoor transition: GPS continues to provide fixes indoors; localization accuracy worsens gradually. Gaps indicate that GPS fix was not obtained at those instances.

4.1.2 IODetector

IODetector [14] is a recent work using primarily the cell, light and magnetic field sensors to determine indoor/outdoor state. Based on experimental data, IODetector establishes some characteristics of these quantities from empirical observation: (1) In daytime, in outdoors, light intensity is typically much higher than indoors; (2) When the user's context changes from outdoors to indoors, the cell signal strength drops rapidly due to attenuation from walls and ceilings; and (3) Magnetic field intensity measured on phone tends to fluctuate rapidly when the user is moving indoors due to appliances, electric currents and metallic objects nearby, compared to open spaces outdoors. IODetector correspondingly runs 3 primary detectors which provide their individual estimates for three environment states (indoor, outdoor, semi-outdoor), and corresponding confidence in those estimates. Then IODetector aggregates these results together. The state that receives the most overall confidence in estimations, is output as the current state.

The semi-outdoor state in IODetector is intended to cover the situation when a user is close to a building but still outside, or is in a semi-open environment, and the signals from the sensors do not easily distinguish between indoor and outdoor. However, for this work we decided to concentrate on the basic states, which are indoor and outdoor, since these are the ones most relevant to context adaptive applications. An uncertain state like semi-outdoor is difficult for many applications to interpret since the environment characteristics there are not defined. Indoor/outdoor transitions are relatively objectively defined, by crossing a threshold such as a door, but the determination of a state to be semi-outdoor is subjective, since in absence of a precise definition, any state can be treated as semi-outdoor. This makes it difficult to obtain meaningful ground-truths from users to evaluate the accuracy of a method using semi-outdoor state. As shown here, even though more challenging, it is possible to design a system that produces accurate indoor/outdoor detection without relying on uncertain intermediate states.

I evaluated the individual primary detectors as well as combined IODetector on our dataset. This data was collected in the ideal way for IODetector, with the phone in hand and in front of the user, exposed to the light and electromagnetic signals. Note that IODetector paper also describes a *stateful* detector based on a Hidden Markov Model. As the stateful scheme was shown to provide only a marginal improvement in accuracy over the simpler stateless scheme outlined above, so the focus in the detailed examination below is only on the latter.

Light Detector. Broadly speaking, the light component of IODetector operates using two thresholds. If it is daytime then it checks for light intensity $L > 2,000$Lux, in which case it outputs Outdoor, else it outputs Indoor, with high confidence. At night time, it checks for $L < 50$Lux to produce a low confidence output of Outdoor, else it outputs Indoor.

Evaluating the light detector on our dataset it was found that even the high confidence results do not always hold due to differences in climate and weather conditions. As observed from these experiments, in 4 out of 5 cases the light intensity did not go beyond 2,000Lux even in the plain open area outdoors. This behavior is closely tied to the weather: in a day with heavy clouds, the value of 2,000Lux is never reached. The observation was tested on multiple phones with the same outcome. This discrepancy is clearly a result of testing in a different place and in different weather conditions compared to the original development conditions for IODetector.

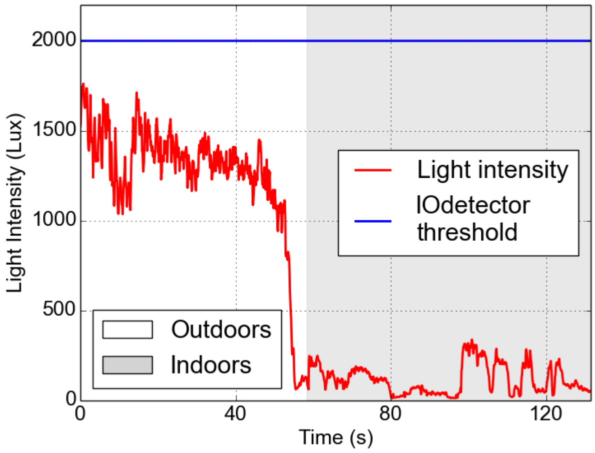
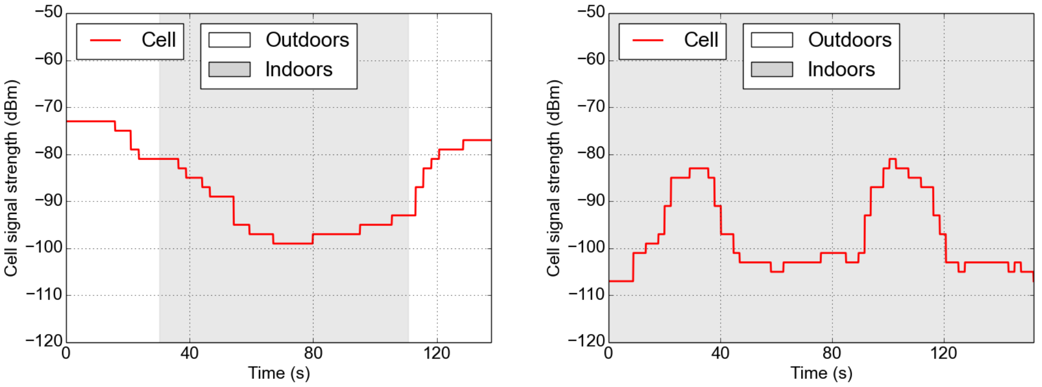


Figure 4.4: Light intensity at outdoor-indoor transition: light intensity drops on move to indoors, but outdoor intensity can be lower than IODetector threshold to detect outdoor state.

Figure 4.4 shows variation in light intensity at an outdoor to indoor transition, where a different threshold could have easily detected the state change. This suggests that light intensity is in fact a good feature to consider, provided the threshold can be determined suitably. But light sensor has the drawback that it is easily obstructed. If the phone is in a pocket or handbag, light does not help. IODetector uses the proximity sensor to detect when the phone may be in pocket or bag, and thus disregards the light sensor readings at those times.

Cell Detector. The cell detector component of IODetector looks for change of cellular signal strengths by 15 dBm (7.5 ASU) in an interval of 10 seconds, to detect transitions between indoor and outdoor. IODetector uses aggregate signal strengths of multiple towers, which on Android will require the phone to operate in GSM mode disrupting its normal use. Thus for these experiments the signal strength from the cell tower the phone is associated with at the time of measurement was used as cellular information.

In many cases, transitions between contexts do not have this slope. Figure 4.5(a) shows such a case where the change in signal strength is slower than the threshold at the transitions. On the other hand, when the user is moving from room to room inside a building, the presence of walls can cause the signal to change rapidly (Fig. 4.5(b)).



(a) Cell signal at transition

(b) Cell signal variations indoor

Figure 4.5: (a) At a transition, cell signal can change by less than 15dBm in 10 seconds. (b) During movements inside a building, cell signals can change faster than 15dBm in 10 seconds.

Derivatives (slopes, rate of change etc) of signals, while useful in principle, are sensitive quantities susceptible to noise, and as a result, can produce erroneous results as shown in Figure 4.5. Further, such quantities can only detect transitions; they cannot detect the state when the user is static. To make use of the cell signal derivative, the detection system has to be running continuously, since it cannot provide any information until a transition happens. Thus it cannot be used for a power efficient detector that can be activated on demand.

Magnetic Detector. The magnetic detector component of IODetector works by inspecting the variance of magnetic field strength measured in μT in a time window of 10 seconds. If this variance is above 18 then the environment state is determined to be indoors, otherwise the component outputs an outdoor state.

The finding here is that this component has the lowest accuracy of all at 40% or lower. One example is shown in Figure 4.6, indicating that outdoors the magnetic variance is usually below the threshold but indoors there are very few situations when the variance in a 10 second time window goes above the threshold of 18.

IODetector with all components. The results of all components combined can be seen in Table 4.1, for data from university campus with the user entering and leaving 5 different buildings. The overall accuracy is about 71.30%. The data was collected in partially cloudy weather, and included approximately equal volumes of indoor and outdoor samples. This table shows results for a single device (a Galaxy S3 phone),

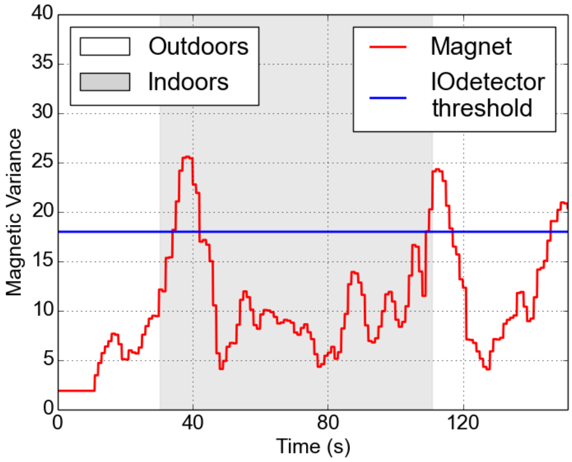


Figure 4.6: Magnetic variance fails to detect an indoor state with given threshold.

results are in the same range with those obtained by other devices (Nexus 4 and Nexus 5).

Building	**IODetector Accuracy(%)**	**IODetector – without light sensor(%)**
Building1	78.32	66.2
Building2	85.87	41.12
Building3	57.19	41.67
Building4	60.11	88.6
Building5	75.02	39.68
Average	**71.30**	**55.45**

Table 4.1: Accuracy of IODetector inside/outside 5 different buildings in our campus.

Since light sensor tends to be often unavailable due to the phone being inside a pocket or handbag, Table 4.1 shows results for detection without light information. The overall accuracy falls to 55.45%, which suggests that IODetector is in fact heavily dependent on light for accurate detection.

Also, Figure 4.7 shows the results for a specific time slice, where the IODetector is first confused by the Light Detector that the environment state is indoors for the first 30 seconds, whereas the Cell detector reacts only at around 110 seconds mark to detect the outdoor transition.

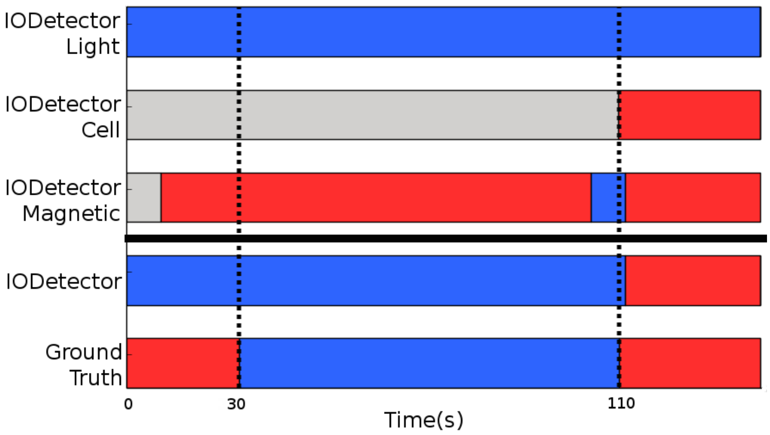


Figure 4.7: Decisions of all components, IODetector, and ground truth in a specific case. Blue: Indoors, Red: Outdoors, Gray: Undetermined. Cell signal derivative fails to produce results until the second transition.

4.1.3 Summary of preliminary investigation

To summarize these findings, the GPS based method is impractical due to its high power requirements, and is also not very accurate. IODetector is lightweight, but the plots shown above suggest that a difference in environment between where IODetector was designed and where our experiments were made causes it to produce poor results. While the essential trends utilized by IODetector were clearly present, the hard-coded thresholds it uses to estimate the indoor/outdoor states do not hold. The use of different phones is also a contributor to the poor performance observed with IODetector.

4.2 Opportunity for Indoor-Outdoor Detection with Machine Learning

Indoor-outdoor detection is essentially a classification type problem in machine learning: given a tuple of features based on measured sensor values, this can be classified to either Indoor or Outdoor. The most common classification technique – called supervised classification – works as follows. It is first provided some feature tuples with associated class labels (i.e., in our case, ground truth environment state — indoor or outdoor), from which a classifier is built or *trained*. This classifier essentially encodes the pattern of classes found in the labeled data. Afterwards, this encoded pattern or classifier can be used to infer labels of new data with unknown class. This works, provided the fresh data follows similar patterns as the training labeled data used to

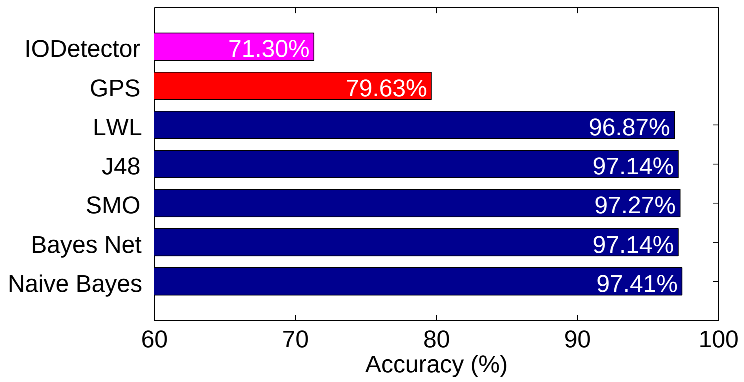


Figure 4.8: Accuracy of IODetector and GPS compared with several supervised learning techniques on the original smaller dataset. Classifiers used the primary set of features similar to IODetector, while the threshold for GPS was determined using the J48 classifier on another dataset. Supervised classification easily outperform the two existing methods

build the classifier. Supervised classification can deduce relatively complex relations between different features (attributes) in the labeled data. Thus classifiers are more general and powerful than methods that treat features separately (e.g., [14]).

The set of features (sensor data) considered for this investigation of supervised learning are:

1. **Primary features:** Light intensity, Cellular signal strength and Magnetic variance. (This is analogous to IOdetector, but we use cell signal strength instead of its derivative.)

2. **Extended feature set:** light intensity, sound intensity from microphone, temperature from battery thermometer, magnetic variance, cellular signal strength and proximity sensor value.

The extended set of features intuitively contains elements to detect important physical variations expected between indoor and outdoor – light, sound, electromagnetic signal in different bands etc. The primary set of sensors allows a direct comparison with IODetector for reference and to gain better understanding of benefits by using multiple sensors.

Using the dataset presented in Section 4.1 for direct comparison between classifiers with the primary set of features and IODetector, we can observe that supervised classifiers clearly outperform IODetector by identifying better patterns between features. This evaluation was performed with a 10-folds cross-validation for classifiers using the university campus (5 buildings) dataset. To compare with the GPS based method, a threshold for the GPS inaccuracy values was determined from the larger dataset to be beyond 8m, or 1 minute without a GPS fix for the indoors state.

Results presented in Figure 4.8 show that supervised learning classifiers produce better detection of indoor/outdoor state than GPS based method and IODetector. While existing methods detect the environment with an accuracy of at most 80%, the use of popular supervised classifiers give results with accuracy greater than 95%.

These observations clearly indicate the opportunity for using machine learning to distinguish between indoor and outdoor spaces better than previous solutions. The following section extends the study in this section using a larger dataset comprising measurements collected from *different* environments.

4.3 Large Experiment across Multiple Environments

After identifying the limitations of existing systems for IO detection and the opportunity for using Machine Learning to improve accuracy, this section expands the evaluation to a wider dataset covering multiple environments.

4.3.1 Experimental Setup

Data collection. We collected indoor and outdoor data from several different types of environments such as university campus, city center and residential areas, with two different types of phones (Nexus 5 and Galaxy S3). Two participants took part in this experiment and they were asked to move freely in those environments with normal use of the phone, including putting them in pocket or handbag when not in use. The only constraint was to input their transition between indoor and outdoor, which is necessary for having ground truth to generate labeled training data as well as for assessing the accuracy of different classification techniques.

Data consisted of sensor readings from the set of sensors available on smartphones (light, proximity, magnetic, microphone, cell, WiFi, GPS, battery thermometer). The type of environment where data was collected was also recorded:

- Campus area – buildings of university, concentrated in a small area inside the city.
- City center – downtown area with public buildings (like shopping centers, train stations, restaurants etc.) situated in the city center.

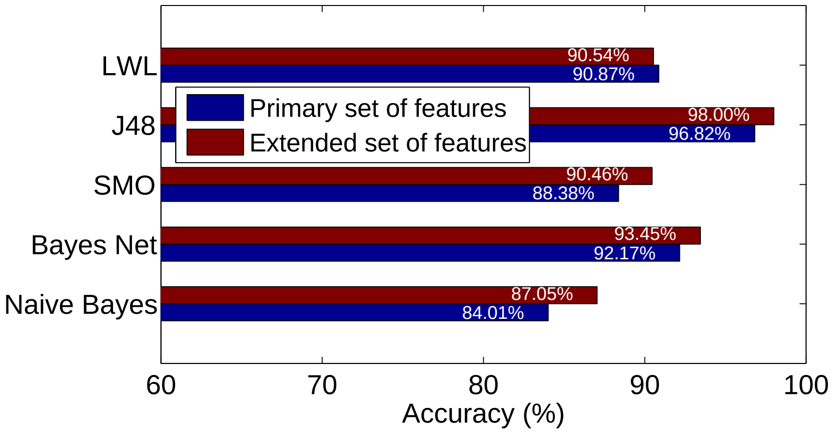


Figure 4.9: Accuracy with supervised classification on the larger and diverse dataset but *using labeled data for training from all environments*.

- Residential area – private buildings in residential area, i.e. homes of participants and friends.

Dataset	**Campus Area**	**City Center**	**Residential Area**
Dataset_1	1,259	1,337	1,271

Table 4.2: Number of instances in each environment collected as part of the larger dataset for the machine learning approaches.

Distribution of the collected data across environments is presented in Table 4.2. This larger dataset was collected with a Google Nexus 5 phone and in time this was two months later than the earlier described dataset used for the characterization of the two previous methods for IO detection.

4.3.2 Baseline with Supervised Learning

Using this larger dataset, supervised classifiers with a 10-fold cross-validation over the entire dataset achieve accuracy typically over 90% (Fig. 4.9).

The results in Figure 4.9 are deceptive though, as the use of 10-fold cross validation implicitly means that labeled data for training the classifiers spans all the environments across which they are tested. It is impractical to ensure labeled data from all possible environments that a mobile user may encounter. What the results suggest, however, is that sensor data from mobile phones contains sufficient information such that good detectors based on classification are possible, provided the training data is representative of the overall dataset. They also show that extended feature set is beneficial in most cases, though marginally.

The important questions is whether a classifier trained on labeled data from a subset of environments is effective when used for IO detection in a new previously unseen type of environment. To emulate this, the larger dataset was split into the three broad environments (campus, city center and residential area), with the classifiers trained on one of the three environments and evaluating them on the other two for classifying indoor/outdoor states. These experiments performed by my colleague, Panagiota Katsikouli, indicate that supervised classifiers fail to transfer to unfamiliar environments, giving results well below 90%. Results for each of the three scenarios (training on one environment and evaluating on the other two) are presented in the paper [76], as well as explanations for why this is the case from an environment diversity perspective.

Main conclusions from supervised classification:

1. Learning based classification produces substantially better results (with over 90% accuracy) than static detection algorithms.

2. Supervised learning on one environment does not translate to unfamiliar environments, as observed by my colleague Panagiota Katsikouli in [76].

These results are promising: they show that sensor signal data contains enough information to effectively discriminate indoor vs. outdoor. But they also imply that a more adaptive method to automatically learn the properties of new environments continuously. Next sections investigate such methods for continuous learning in new environments.

4.4 Robust Indoor-Outdoor Detection with Semi-Supervised Learning

This section presents the exploration of machine learning methods to continuously improve the system learning process while the phone is used across different environments. This requires a way to continue learning in a new environment without the need for involvement from users to gather ground-truth indoor/outdoor state information (for labeling training data).

Semi-supervised learning [93] offers a good solution for this problem: using the available “unlabeled” data to improve classification tasks when labeled data is scarce or expensive, as it is the case for the IO detection problem.

The three different solutions to learn from unlabeled data considered here are: (1) *clustering* which tries to group completely unlabeled data, then associate class labels to groups using small amount of labeled data; (2) *self-training* where a classifier built from some labeled data, tries to learn subsequently from its own outputs on unlabeled data; and (3) *co-training* where multiple classifiers learn from each other's outputs. These methods fall under the category of *semi-supervised* learning since they make use of both labeled and unlabeled data, and we show in the following that in fact these methods work well for indoor-outdoor detection. See [94] for a survey of semi-supervised learning techniques.

4.4.1 Cluster-then-Label

Clustering methods (also sometimes referred to as unsupervised learning) group input data points into subsets of similar items. Clustering methods do not need any labeled data, and are thus useful in uncovering unsuspected pattern/structure in the data. See [95] for more discussion of clustering. However, when clustering is applied for classification problems, the absence of any supervision might result in wrong association between clusters and classes. This is where some (even if small) amount of labeled data can help. This semi-supervised learning method is called "Cluster-then-Label" [93] and works in two steps. In the first step, all available data instances are clustered using a clustering technique (e.g., K-Means). In the second step, labeled data instances within each cluster are used to train a supervised classifier (e.g., Naive Bayes) which is then used for inferring the class labels of remaining instances of each cluster. In other words, classification is done using a set of supervised classifiers (one per each cluster).

This cluster-then-label method was explored considering two different clustering algorithms (K-Means and Expectation Maximization (EM)) and three different supervised classification techniques (Naive Bayes, LWL and J48 decision tree). For this evaluation the larger collected dataset is used. Specifically, this dataset contains more than 3,000 data instances and spanning three different environments (campus, city, home); of these, 300 instances taken from one of the environments (campus) make up the labeled data. Results from this investigation are presented in Figure 4.10 which shows detection accuracy in the region of 70%. This result, while confirming the earlier observation from supervised classification that there is some information in the data to aid classification, also indicates that the cluster-then-label method cannot effectively learn across environments.

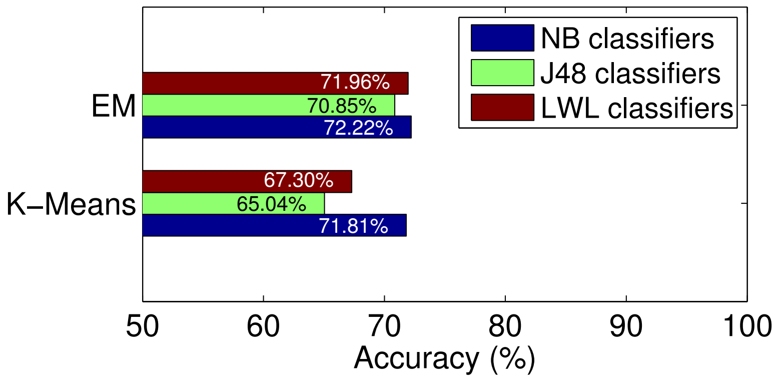


Figure 4.10: Indoor-outdoor detection accuracy with cluster-then-label method. Performance not good enough for our problem.

4.4.2 Self-Training

Another semi-supervised learning method worth exploring is self-training. In this method, a classifier is first built with the available labeled data using a standard supervised learning technique (e.g., Naive Bayes, decision tree). Afterwards, as the system generates class labels for new unlabeled input, these output labels are used to re-train the classifier. The idea is illustrated in Figure 4.11. The classifier thus attempts to learn over time as it incorporates more data into its model. More detailed discussion can be found in [94, 96].

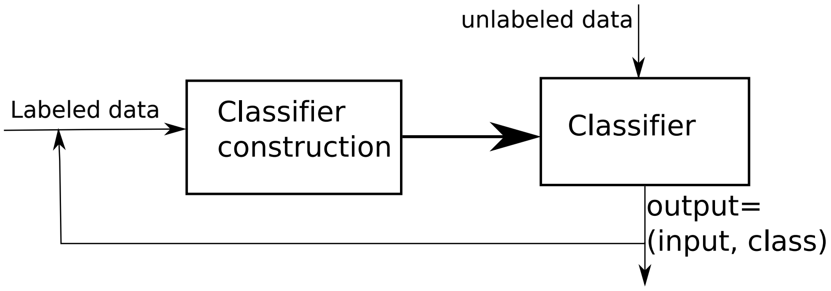


Figure 4.11: Self training. The output of the classifier is treated as labeled data to build improved classifier.

This strategy is evaluated with different classifiers and the extended feature set on the larger dataset. In each case, the initial classifier is trained on labeled data from one particular environment, then it is left to self-train with unlabeled data from the

Classifier	**Environment**	**Accuracy(%)**	**Accuracy(%)**	**Accuracy(%)**	**Accuracy(%)**	**Accuracy(%)**
	Unlabeled data	100	200	300	400	500
Naive Bayes	home+city	76.6	76.83	76.91	79.16	82
	campus+city	75.58	75.58	75.58	77.91	80.58
	campus+home	89.4	92.3	92.5	91.5	92.3
J48	home+city	77.16	77.16	77.16	79.5	82.16
	campus+city	75.58	75.58	75.58	77.9	80.58
	campus+home	79.16	80.25	81.91	79.16	80.58

Table 4.3: Self-training accuracy performance with the indicated environments being the ones from which the unlabeled data come from, while labeled data was taken from the other (third) environment. The number of labeled data is 300 in all cases, the number of unlabeled data varies as shown, and a separate set of 1,200 instances (drawn equally from all environments) are used for evaluation in all cases. Self training makes some improvement, but not enough.

unfamiliar environments. Evaluation was done on distinctive instances in the dataset, different than the ones used for the initial training, all from unfamiliar environments in the dataset.

As seen from Table 4.3, the accuracy of self-training is better than the previous clustering based method by about 5% in most cases. The table presents the accuracy of self-training by varying the number of instances used for additional training with unlabeled data before the evaluation. The accuracy is generally below 90%, with a few exceptions going above this. As observed from these results, self-training is not good enough to our task as some instances have poor performance for a reliable system.

4.4.2.1 Online learning with a ground truth provider

In order to understand if the failure of self-training to learn from unlabeled data is simply due to its own created labels being *unreliable* or due to some more complex reason related to the nature of the inaccuracy, this section explores the impact of additional training labels accuracy to the performance of self-training.

This is explored using an *Unreliable Ground-truth Provider (UGP)*; it is built using the manually obtained ground-truth state information in our datasets[2]. UGP works as follows: it returns the correct label with a specified probability p, and with probability $1-p$ it returns the incorrect label. After building the initial classifier with labeled data, the system continues to train on a small set of data labeled correctly by the UGP with the probability p. The results are shown in Figure 4.12 as a function of the probability

[2] This is unrealistic in practice. This ground-truth information is used simply to understand the shortcomings of self-training.

p.

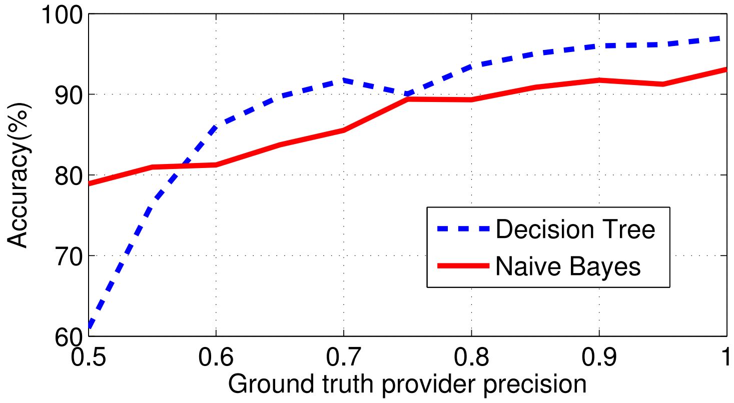


Figure 4.12: Accuracy using unreliable ground truth provider (UGP) on dataset_1. 300 correct labels for initial training from a training environment, 1,000 unreliable labels with probability p from other environments, and rest for evaluation; averaged over all training environments. Results show very good performance even with small values of $p \sim 0.65$ to 0.80.

Even with a very unreliable ground truth provider – one that gives correct labels only 65 to 80 percent of the time, the results are very good at about 90% or more. But self training fails to get comparable results with an underlying classifier that produces the similar quality of data.

The quality of ground truth provider output that distinguishes it from the classifier output is that its unreliability is completely random, and therefore *unbiased*, while the classifier output suffers from biases. Therefore, the failure of self-training does not stem simply from the data being unreliable, it is likely due to specific properties of the classifiers and their outputs and self-training itself, reinforcing its own errors and not learning suitably.

Thus, if we can find a source of probabilistically labeled data independent from the classifier's own output, we may be able to get better results as that source and the classifier would not then have same biases. This observation naturally corresponds with the next semi-supervised learning method, co-training, explored in the next subsection.

4.4.3 Co-Training

Co-training [97] is a semi-supervised learning method using 2 classifiers in parallel to improve predictions. The classifiers work with different features (sensors) to gain different perspectives and uncover different patterns. Each data instance is classified by the two different classifiers and the result with higher confidence is used to retrain and improve both classifiers[3]. The idea is shown in schematically in Figure 4.13. See [94, 96, 97] for more details on co-training.

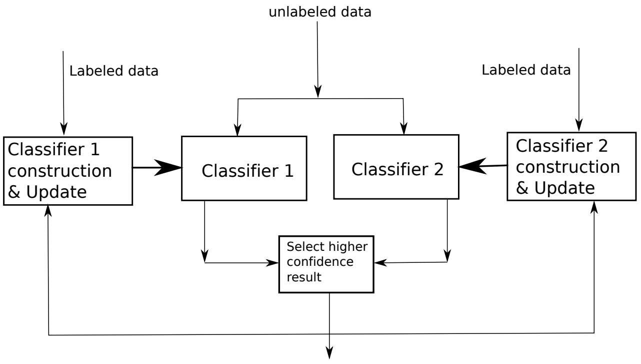

unlabeled data
Labeled data
Labeled data
Classifier 1 construction & Update
Classifier 1
Classifier 2
Classifier 2 construction & Update
Select higher confidence result

Figure 4.13: Co-training with 2 classifiers operating with different feature sets. The higher confidence classification for each data is used as the training label to improve classification.

As concluded in the previous subsection, IO classification methods can do well even with erroneous input, provided the error in the input does not have the same bias as the classifier itself. Co-training is a natural choice for such an extensions, since classifiers working with different sets of features (sensors) are able to complement each other in online training of indoor-outdoor classification.

4.4.3.1 Feature ranking and selection for co-training

In building the two classifiers for co-training, it is important to balance the feature sets in terms of quality. Some features like cellular signal or light are clearly good predictors of indoor/outdoor state, while features like magnetic variance and proximity sensor value are not so effective. Each classifier needs to have its fair share of effective features to produce meaningful results. These features are analyzed from an effectiveness ranking perspective in conjunction with machine learning techniques. This was done using tools provided in WEKA for attribute selection based on different classification techniques. The rankings are shown in Table 4.4.

These features were then split into disjoint pairs of sets considering their ranking, and assign them to underlying classifiers of co-training method as shown in Table 4.5. Note that for each ranking of features (Naive Bayes or SVM), there is a a different distribution of features to classifiers, which is comparatively evaluated shortly.

[3] Co-training refers to the general idea of two classifiers learning from each other. The implementation can have many variations. See the cited references.

Rank	Naive Bayes precision	SVM Attribute Evaluation
1	light intensity	cell signal strength
2	sound amplitude	battery temperature
3	time of day	light intensity
4	proximity	sound amplitude
5	cell signal strength	time of day
6	battery temperature	proximity
7	magnetic variance	magnetic variance

Table 4.4: Ranking of features by their importance with different methods.

Naive Bayes based selection	
Classifier 1	Classifier 2
light intensity, time of the day, proximity value, battery temperature	sound amplitude, cell signal strength, magnetic variance
SVM based selection	
Classifier 1	Classifier 2
cell signal strength, light intensity, time of day, proximity value	battery temperature, sound amplitude, magnetic variance

Table 4.5: Assignment of features (sensors) to co-training classifiers with the two different feature ranking methods.

4.4.3.2 Evaluation of co-training

Using the large dataset presented in Section 4.3 again, the co-training method described above is evaluated as follows. First, 300 labeled instances are chosen from the campus environment for initial training of the two underlying classifiers. Then 1,000 unlabeled instances are taken from the other two unfamiliar environments. Each such unlabeled instance is classified using the two classifiers and the higher confidence classification of the two is associated to the data instance as the "label" for online automatic re-training of both classifiers. This process is repeated for each unlabeled instance. The classifiers system so built are then evaluated using a separate set of 1,200 instances in the dataset with equal representation from all environments.

Figure 4.6 shows the results of this approach. Clearly, co-training performs better than self-training with the right choice of classifiers. Naive Bayes and J48 decision tree outperform the others, with Naive Bayes providing more than 90% accurate detections with both distributions of features, and better accuracy with SVM based feature ranking.

4.4.4 Learning Curve

To get a better insight on the process of learning with co-training that was found to be effective, the learning curve was also explored showing the improvement of classifier performance with increasing labeled/unlabeled data. The focus was set on Naive Bayes classifiers and SVM based feature ranking, the combination that provided the best accuracy results overall in the previous experiment.

First investigation was for the impact of unlabeled data. This was done by taking the two classifiers working in co-training which are initially trained with 300 labeled instances from campus environment, then varying the number of unlabeled instances taken from the three environments in the following order: first 500 instances from home environment, next 500 instances from campus and the last 500 instances from city environment. Figure 4.14 shows the resulting learning curve from using a separate but identical 1,200 instances taken from all three environments for each data point (number of unlabeled instances) on the Ox-axis. This clearly shows the learning of the co-training model in action, especially in the final third of data points. With unlabeled data from home environment (initial part), learning is modest as most of the data from this environment is from indoors. The middle part of the graph shows a steady flat learning curve as there is not much more to learn from unlabeled campus

Features Distribution	Classifier 1	Classifier 2	Performance (%) Classifier 1	Performance (%) Classifier 2	Performance (%) Co-training
Naive Bayes based	J48	J48	67.58	83.50	83.0
	LWL	LWL	78.17	93.16	78.17
	Naive Bayes	Naive Bayes	80.00	91.25	**91.66**
	J48	Naive Bayes	67.66	86.67	78.5
	Naive Bayes	J48	68.41	88.91	89.33
	Naive Bayes	LWL	74.25	93.16	74.25
	LWL	Naive Bayes	78.17	86.67	78.17
	J48	LWL	67.66	93.16	78.0
	LWL	J48	78.17	83.33	86.0
SVM Attribute ranking based	J48	J48	85.58	79.25	86.67
	LWL	LWL	78.16	81.91	78.16
	Naive Bayes	Naive Bayes	94.08	87.33	**93.33**
	J48	Naive Bayes	89.16	87.16	90.25
	Naive Bayes	J48	82.83	79.66	81.16
	Naive Bayes	LWL	77.16	81.91	77.83
	LWL	Naive Bayes	78.16	86.91	86.33
	J48	LWL	78.16	81.91	78.16
	LWL	J48	91.16	77.58	78.25

Table 4.6: Co-training of two classifiers working with different sets of features. Supervised training was done with 300 data items from campus environment, co-training on 1,000 random items from the other two environments, and tested on 1,200 items equally from all environments. Produces better performance than self-training, with Nave Bayes performing best – over 90%.

data beyond what is already learned from labeled campus data used for initial training. Ultimately, the co-training model achieves an accuracy over 90% after encountering sufficient unlabeled data from all different environments.

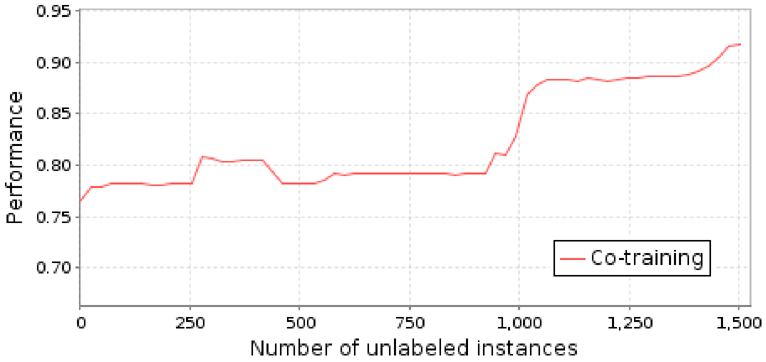


Figure 4.14: Learning curve of co-training as a function of number of unlabeled instances.

Also important is the impact of labeled data on the accuracy performance of Co-training. Figure 4.15 presents this by varying the number of labeled instances (from campus environment) and for each data point (on the Ox-axis in Figure 4.15) it uses the same 1,000 unlabeled instances from the other two environments and 1,200 instances evaluation set as in section 4.4.3.2. It is obvious that after about only around 50 labeled instances, Co-training model accuracy improves rapidly and stabilizes to peak levels. This suggests that a fairly small amount of labeled training data is needed up-front for the Co-training method to function effectively.

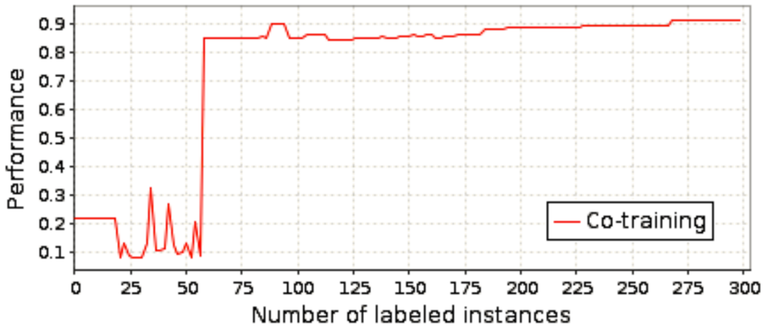


Figure 4.15: Learning curve of co-training as a function of number of labeled instances.

4.4.5 Learning across Devices

The evaluation so far was performed on the larger dataset collected with the Nexus 5 phone as indicated in Section 4.3. In practice it is desirable to have a method that operates well across different phone device types. For example, a developer can train and deploy a detection system using their own development devices, while users may run this on their personal device which is different from that used for the initial training and possibly with different sensor characteristics. For that reason we collected a second dataset resembling the dataset collected with the Nexus 5 device, but this time using the Samsung Galaxy S3 device.

To evaluate co-training across devices, the following experiment setup was developed. Labeled data collected with the same device in one environment (city) was used for the initial training of classifiers. Similar to the process presented in previous subsections, additional co-training of classifiers was performed with instances from the remaining environments, but this time the data was collected with a second device and different in characteristics. The results presented in Figure 4.16 show that co-training can successfully learn across devices, with a small drop in performance (compared to learning on the same device) due to different make and qualities of sensors on different phones.

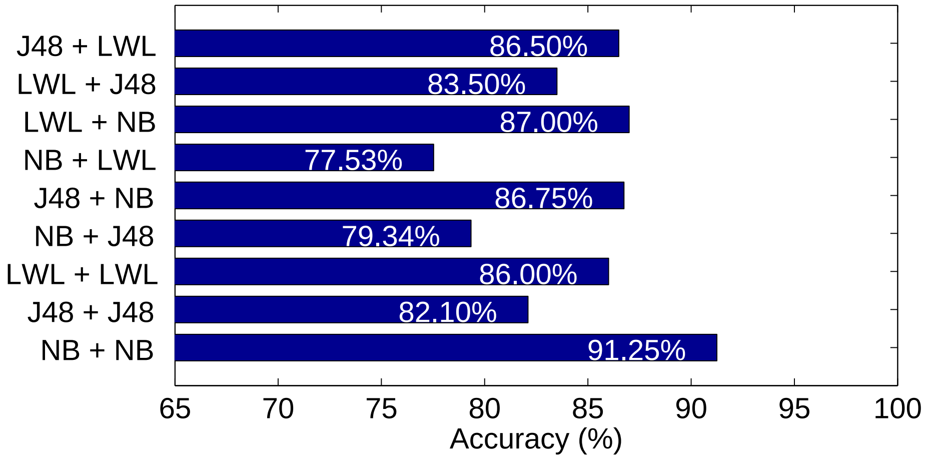


Figure 4.16: Performance of co-training across devices, using the SVM based feature ranking. Naive Bayes classifiers again provide the best accuracy.

An important observation to make here is that important differences may exist between phone sensors, which in turn can impact portability of context detection solutions between devices. Although in these experiments the Nexus 5 and Galaxy S3 have very similar sensor characteristics, clear differences can be observed when comparing with previous generation of smartphones like the Nexus One, which reports the light sensor values in discrete values with large gaps between levels, in contrast to most modern smartphones reporting continuous values. The difficulty of transferring observations between sensors should not be underestimated. Recent work shows that even phones of the same model have unique sensor patterns, which despite being minute, are enough to identify each phone in a large population [98]. This in fact highlights the utility of having an adaptive learning system to specialize trained models for each device sensing characteristics, which is what our Co-training based solution does.

4.4.6 Discussion for Semi-supervised Learning

Semi-supervised learning encompasses a set of solutions taking advantage of the more abundant unlabeled data with just a small set of initial labeled instances. In addition, the incremental nature of some of these solutions assures an adaptive characteristic of the system, which is important for our task.

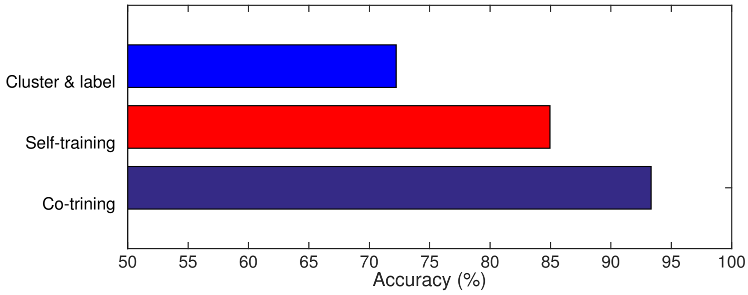


Figure 4.17: Direct comparison between the best performing three semi-supervised methods across all three type of environments.

The three semi-supervised methods considered here, Clustering and Label, Self-training and Co-training were built around two classifiers, Naive Bayes and Decision Tree (J48), due to their light overhead and computational efficiency ideal for running on mobile devices. For the same reasons these two classifiers are the preferred option for many mobile sensing applications [21]. A direct comparison of the different semi-supervised learning methods considered is presented in Figure 4.17. This shows that the best performing solution across all environments is Co-training, taking advantage of two independent classifiers assisting each other to label new samples. Unguided clustering with subsequent cluster labeling shows that: 1) there is enough inherent difference in sensor features to separate between the two sensors with above 70% accuracy without any preliminary guidance; and 2) gradual labeling building on observed confidence of previous models is a better strategy. This is what Self-training is aiming for by incremental expansion of training set through labeling unseen samples based on own classifier expectations. However a single classifier falls into its own bias as shown by the results for Self-training. So the Co-training solution overcomes this issue by having two independent classifiers operating in parallel to select the estimation with the highest confidence. This solution outperforms all other approaches with greater than 90% accuracy.

The other observation is that Naive Bayes classifier performs best as part of these semi-supervised learning solutions, due to its robustness to uncertain observations, fact also validated by the online learning ground truth provider presented above.

Next section presents a direct comparison between the best performing semi-supervised method, Co-training, and the other solutions discussed earlier for indoor-outdoor detection.

4.5 Implementation and Evaluation of Mobile Application using Co-Training for IO Detector

A context detection service needs to have fast response time, and needs to be lightweight – both in terms of sensing energy requirements and computational needs. It also needs to have fair degree of accuracy. While perfect accuracy may not be possible, we would like its output to correspond to our expectations most of the time. As previously seen, a major challenge in accurate context detection is variability in sensor signal characteristics across environments, and thus context detection needs to be adaptive and continue to learn in newly experienced environments.

The results from previous sections show that such an efficient, accurate and adaptive IO detection system can be built based on semi-supervised learning, and co-training in particular. Co-training produces excellent results without using expensive sensors like GPS and WiFi. Using only the lightweight sensors makes the system energy efficient. It is stateless and also does not require derivative based transition detections such as rate of drop of cell signal strength at the transition from outdoor to indoor. As discussed before, derivatives tend to be susceptible to noises and the need to detect transitions force services to run continuously. Instead, by using only current sensor values to detect purely the states and not transitions, the proposed system can return results on demand[4]. The IO detection service based on this proposed approach can thus turn off the sensors and sleep most of the time. When some other application requests indoor-outdoor context information, the service can wake up and go back to

[4] Magnetic variance is the only feature that needs measurements over several seconds. But note that it is the least influential of features (Table 4.4), and in experiments its removal does not change the results of Figure 4.6 in any significant measure.

sleep immediately after returning results.

4.5.1 Efficient Implementation via Incremental Learning

As seen in the evaluatin of Co-training (Table 4.6), the use of Naive Bayes classifiers with features partitioned according to the SVM ranking is the most effective configuration of Co-training. As it turns out, this is also an ideal option from an implementation perspective. Bayesian classification is extremely efficient and can be done in constant time once the classifier has been trained (since there is a constant number of sensors). In classical bayesian classifiers for discrete parameters [95], it is trivial to design an incremental version that updates the probabilities of the variables on each input. For real valued data such as sensor readings, it is possible to discretize the values into suitably sized bins and apply Naive Bayes as usual. Alternatively, a gaussian distributions can be maintained for each sensor with each class and obtain probabilities from these distributions [99]. Since both mean and variance can be maintained efficiently for streaming data, this method can keep the parameters up to date using no additional storage and at constant computational cost per update.

Implementation of Co-training based IO detector for Android smartphones was done using the WEKA libraries and in particular the updateable Naive Bayes, which uses gaussian distributions. This was successfully used to develop a mobile app which was evaluated on the Samsung Galaxy S3 phone. This obtained 92.33% accuracy for the same setup as presented in section 4.4.3.2, but this time in online mode using the Android implementation. This result is compared with alternative methods in Figure 4.18. In the figure, IODetector (old thresholds) corresponds to the IODetector with thresholds provided by authors in [14]; the new thresholds variant is using updated thresholds by inferring these the same 300 labeled data instances used in section 4.4.3.2 for the initial training of classifiers, thresholds obtained using a decision tree classifier for each of IODetector features, just as we did for obtaining inaccuracy threshold for GPS based method (see section 4.2). This evaluation is updated from the version presented in our original paper [76] to make it more favorable to IODetector by compensating indecisive scenarios scenarios to indoors as this is the more common context. As seen here, re-tuning the IODetector thresholds helps but not much as any one set of thresholds do not guarantee good performance across diverse environments. Overall, we observe that the proposed co-training provides the highest accuracy detection in comparison with existing methods including supervised classifiers.

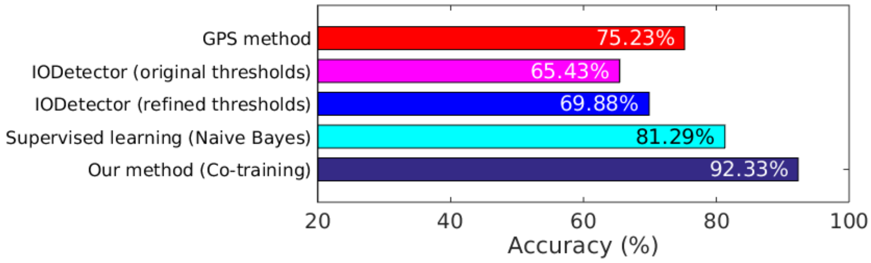


Figure 4.18: Accuracy comparison of co-training implementation on phone with alternative IO detection approaches.

Observe that co-training learns new environments quite rapidly and automatically without user involvement – using only a few hundred unlabeled data instances. For example, in Table 4.6, it learns 2 new environments from 1,000 unlabeled data points. Section 4.4.4 shows similar results. This implies that in general, we do not need to retrain the classifiers whenever a new unlabeled data instance becomes available. It will generally suffice to randomly record a small number of points to boost the classifier sufficiently for any environment where the user spends more time. The feature of learning from few inputs further helps the energy efficiency of the algorithm. Since our method is capable of learning across devices, it is easier to deploy it – the developer can ship the software with the supervised training done on her device, while the software once installed on the user's device can continue to learn new environments through co-training.

4.5.2 Power Consumption

This subsection compares the power consumption of the proposed Co-training implementation to the GPS based method and IODetector. Power consumption measurements were obtained using Galaxy S3 phone and with the help of Monsoon Power Monitor in the setup presented in Figure 4.1. The first evaluation of these three different IO detection methods is for a one-time use, followed by the evaluation of running these over a longer period of time (30 minutes).

IODetector power use. IODetector's power use sums up to about 121mW counting light sensor, the cellular interface and the magnetometer. The computation costs are negligible in comparison. IO detector keeps the sensors active continuously.

GPS Energy use per fix. During experiments it was observed that the phone required between 5 seconds to 45 seconds to obtain a GPS fix outdoors, whereas indoors and close to the windows between 15 seconds and 1 minute if it can obtain one. On a set of 20 random outdoor measurements, the GPS obtains a fix in a median time of 12 seconds, whereas for the indoor case in 25 seconds. The GPS uses 379.94mW for continuous scans, therefore obtaining a GPS fix outdoor for the median case would require 4559.28 mJoules. In the optimist view that the GPS obtains a fix indoors, the energy required for the median case is 9498.5 mJoules. Computation costs are negligible – simply comparing the measured location inaccuracy against a threshold.

Co-training energy use per estimation. For our implementation with Co-training the power consumption of all sensors sampling for one second is one average 136mW (light, microphone, cell, proximity sensors and battery thermometer), whereas for the magnetic sensor which samples for 10 seconds is 60mW. Thus, cost of sampling the sensors is 736mJoules.

Co-training method requires additional costs for computation. Inferring the state consumes 192mW for 0.01 seconds, increasing the cost of estimating a state by 1.92 mJoules. Note that the energy consumed for this operation is dominated by preparing the measured sensor values (features) in a format required by the WEKA library (as our implementation of co-training reuses WEKA code for updateable Naive Bayes); this could be drastically reduced by a clean-slate implementation. Updating the classifiers incurs marginal energy consumption, in essence changing just a few variables in the model of the two classifiers, the means and the standard deviations. In total, the energy consumption for estimating a single state is 738mJoules

Also, the evaluation considered the one time preprocessing cost of the initial training of the classifier. This takes about 11.4 seconds to train the classifiers at an average cost of 915.76mW, thus taking 10.35 Joules. This covers the costs of reading the training file, parsing it, initializing the classifiers and training them, and is dominated by the first two costs.

Continuous use over 30 minutes. IODetector needs to operate continuously to detect the state, since its cellular component detects only transitions. The other two methods (using the GPS inaccuracy and our implementation with Co-training) are stateless, meaning they can operate just when they are needed. For continuous estimation, stateless services can be activated periodically. The overall energy use will depend on the periodicity of this sampling.

Based on the energy requirements measured as described above, Figure 4.19 presents the energy consumption of these three methods over a 30 minute period for different

IO detection sampling intervals. IO detector runs continuously at 121mW, thus consumes a fixed 217.8 Joules over 30 minutes. The energy use of GPS and co-training decreases with the increase in interval between invocations of IO detection service. For a sampling interval of 10 minutes, the energy consumption of the GPS is 13.6 Joules outdoors (and 28.5 Joules for the indoors case), while for the co-training it is lower at 12.56 Joules. From here we observe that the Co-training method is energy-efficient compared to the other methods for any practical sampling interval.

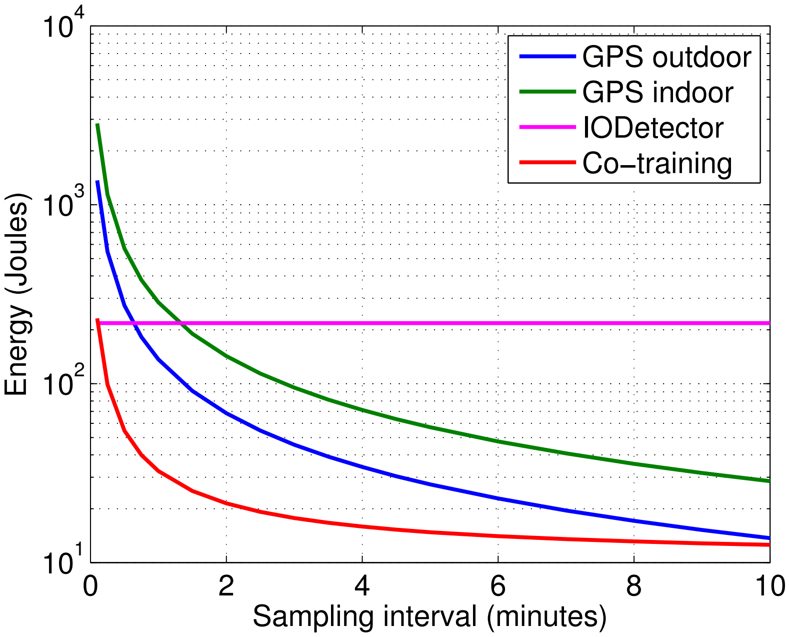


Figure 4.19: Energy consumption comparison between co-training and other methods for various intervals of IO detection service invocation. GPS based method is represented separately for outdoor and indoor (near window).

4.6 Case Study - Mobile App to Control WiFi Interface

A natural example of indoor-outdoor context aware power management is to reduce wasteful WiFi access point scanning automatically performed by the device. Mobile phones regularly scan the WiFi spectrum for available access points when disconnected from a network, even when the user is outdoor or traveling with no possibility of connecting to a WiFi network; this is possibly the most significant contributor to battery drain if the user is not making any active use of the phone.

The power consumption of a Samsung Galaxy S3 while trying to associate to a network is shown in Figure 4.20. The power consumption for each scan is about 250 mW for approximately 3.3 seconds, repeated every 18 seconds (which is pre-set and unchangeable). To put this in perspective, in comparison with GPS power consumption this is as follows. The power consumption of WiFi card to keep scanning for a network for 5 minutes is approximately the same as what GPS consumes when it tries to get a location fix continuously for a minute.

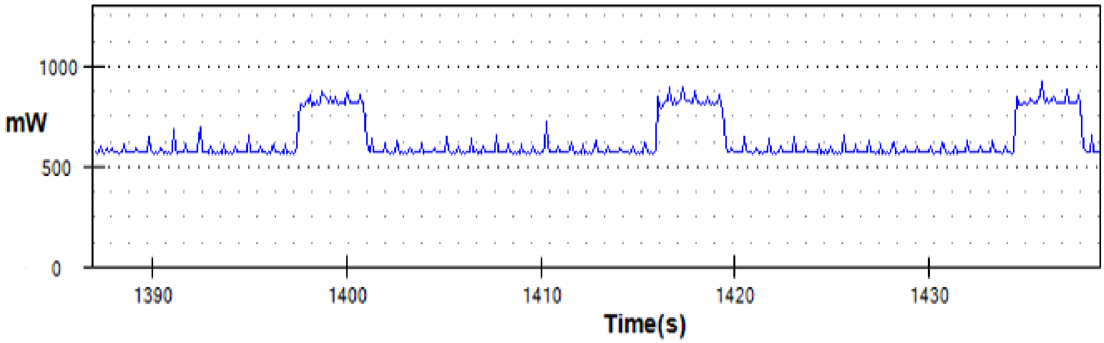


Figure 4.20: Power consumption of phone WiFi interface when searching to find a network to connect to by scanning the spectrum every 18 seconds.

Thus, switching off the WiFi interface while the user is outdoors and switching it back on when indoors can lead to significant power savings.

Evaluation. The scenario considered here is of a user traveling from her residence to the university campus during the regular hours of commute (9am and 5pm). The journey time between the two reference points was on average about 25 minutes and route spanned three different environments (residential, city and campus areas). Having learned about these environments on first exposure to them, the co-training system was able to reliably detect indoor-outdoor state in both of these environments.

The detection service scanned the sensors once every 2 minutes. The magnetic sensor needs to run for 10 seconds to obtain variance results, while other sensors run for 1 second or less. The sensing power consumption with these scanning characteristics for an entire travel period, including the CPU energy consumption, is about 25.8 Joules. On the other hand, when the WiFi is on, it scans once every 18 seconds, consuming a total of 69.3 joules. Thus, by simply disabling the WiFi interface and scanning lower power sensors to detect an indoor environment, we make an energy saving of about 63% for this application scenario.

4.7 Summary

This chapter explored the problem of determining whether a user is indoors or outdoors using low power sensors readily available on modern smartphones. For this IO detection problem, existing solutions were shown to be too energy hungry or fail to provide accurate results across a range of different environments typically encounter in practice, due to the use of fixed and environment agnostic thresholds in the underlying estimation schemes. Observations that further improvements can be achieved by viewing the IO detection as a machine learning classification problem with 2 classes (indoor, outdoor) guided this research to adapt classifier models to new environments and devices in order to achieve robust and accurate detection across diverse settings.

To address the fundamental issue of model adaptation on-the-fly and transparent to the user, a semi-supervised learning framework was adopted as the ultimate solution approach. Through the investigation of different commonly used semi-supervised learning methods, co-training method was found to yield most accurate results across a range of environments and different devices. An implementation of this co-training method on Android platforms was presented in this chapter, using an incremental version of Naive Bayes classifier. It was shown that this approach outperforms other alternative methods in terms of both accuracy and energy efficiency. Also, this implementation does not incur any communication overhead (as it does not need to communicate with a backend/cloud) and is privacy preserving. The use case application of switching off the WiFi interface when the user is outdoors was shown to save considerable power, thus extending the battery life in usual conditions.

Chapter 5

Multimodal Deep Learning for Versatile Mobile Context Sensing

This chapter presents the benefits of using deep learning to integrate multiple sensor data streams for increased performance on a variety of context detection tasks. The hypothesis is that neural networks can identify non-intuitive features much better than hand-crafted features, leading to more accurate estimations. In this chapter, the focus is on a promising Deep Learning representative, Restricted Boltzmann Machine (RBM) that is able to learn hierarchical representations (i.e., features) of multimodal data incorporating difficult to find non-linear cross-sensor correlations and relationships which are the key to maximizing inference accuracy. Experiments show a general-purpose multimodal RBM model is able to outperform conventional machine learning classifiers for a wide-range of sensor types and inference tasks. Evaluation is done on a range of context detection tasks: human activity recognition, sleep stage detection, indoor-outdoor detection and landmark differentiation.

An earlier exploration of Multimodal RBM for human activity recogniton was first published in the Proceedings of ACM International Joint Conference on Pervasive and Ubiquitous Computing, 2016 [100]. The extended version of this work covering additional use-cases with the same architecture for mobile context detection is currently under review at ACM Journal on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT). The last use-case in this chapter on using deep leaning for WiFi and magnetic fingerprinting was presented in the Cyberphysical Systems Seminars at the University of Oxford as an invited talk with the title ”Smartphone-based indoor localization with multimodal sensing”. This last use-case compares the accuracy of traditional WiFi fingerprint matching algorithms to that of multimodal RBM integrating magnetic field signal and radio signal (WiFi). All these use-cases demonstrate the generalization aspect of this architecture across many context detection tasks.

This work was developed in collaboration with researchers from Bell-Labs and the University of Cambridge as part of a HiPEAC research collaboration visit.

5.1 Deep Learning for Multimodal Sensing

The solutions presented in previous chapters as well as the vast majority of currently available solutions to integrate multiple sensor signals for context detection are *shallow*. This term contrasts them with recent developments in data science, *deep learning* algorithms [22, 101], which can also benefit context sensing, following the success of other research areas (Computer Vision [102], Natural Language Processing [103] and others). Presented in this section is the variant of deep learning architecture used for this investigation, showing its ability to learn – *purely from data* – multiple layers of feature representation which allows this to overcome challenges in using heterogeneous sensor data. To demonstrate this, activity recognition is used as representative example of context detection with mobile devices.

The multi-layer feature representation of deep-learning architectures allows them to extract more complex information than readily used shallow methods. Common shallow methods require a preliminary stage to select a set of hand crafted features which are used to distill sensor data in pre-processing. The quality of these handcrafted features impact system performance (accuracy and speed) directly.

To understand the utility of deep learning for activity recognition under systems with multiple sensors, this work studies the use of a multimodal version of Restricted Boltzmann Machines [27] (RBMs) (presented in Figure 5.1). In previous work, this variety of RBMs have been used to fuse pairs of text, video and audio data for the purpose of image captioning [25, 30], and speech [27] or emotion recognition [31]. The objective here is to empirically validated if these RBMs are still suited for new sensing tasks in the space of wearable devices and if their characterizing requirements are met by these devices.

As shown in Figure 5.1, the architecture used in this work has a separate input for each sensing modality, allowing initial intra-sensor features extraction through the first layers of the network. The alternative of uni-modal deep architectures do not have separate layers per sensor, but concatenating all data streams into one input, which prevents the network first learning sensor-specific information before these represen-

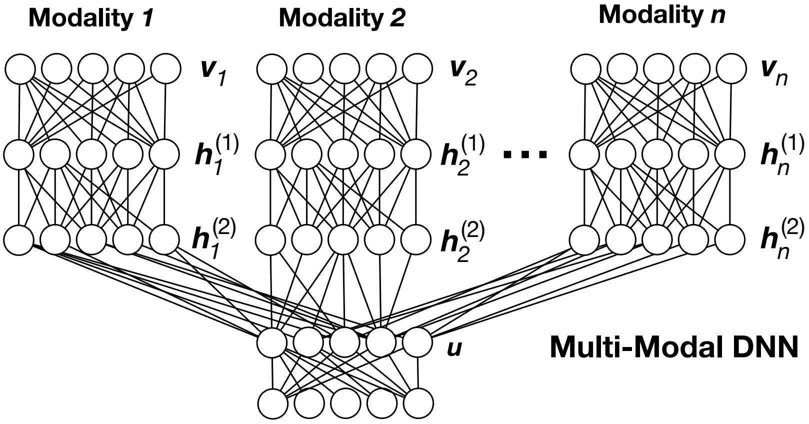


Figure 5.1: RBM-specific deep multimodal learning architecture.

tations are unified across all sensors. Previous work has shown this intra-sensor relationship to be much stronger than inter-sensor counterpart [30].

The Inference Process

Each sensor data stream is provided as time windows or static representations on each branch of the architecture. Parameters on each feed-forward layer determines how this information from the initial layer progress from layer to layer until the final/output layer. Inference ends once the output layer has activated, and an activity or context has been decided for the input sensor data.

Formalizing the inference process of RBMs: the state ($\mathcal{A}_i^{L+1}$) of each individual RBM unit (x_i^{L+1}) within a layer ($L+1$) is dependent on the unit weights connecting the j^{th} node in layer L to the i^{th} node in layer $L+1$. In this fully connected approach there is a connection between each (x_i^{L+1}) node to all nodes (x_j^L) on layer L, weighted as w_{ij}^{L+1}. Specifically this relationship is computed as:

$$\mathcal{A}_i^{L+1} = \frac{1}{1+\exp(-\sum_j w_{ij}^{L+1} x_j^L)} \tag{5.1}$$

As shown in Figure 5.1, separate branches (M_k) exist on the architecture for each sensing modality (sensor type), allowing relevant information for each sensing modality to be extracted. There operate independently on each branch until unified later in the architecture (U_l) to extract global features as composition of the individual branch distilled observations. As effect, all layers bring their contribution to learning a joint representation of all sensor modalities. This aspect is expressed as:

$$P(\mathbf{v},\mathbf{h};\Theta) = \frac{1}{\mathcal{Z}(\Theta)}\exp(-\mathcal{E}(\mathbf{v},\mathbf{h};\Theta)) \tag{5.2}$$

where $\mathbf{v}$ represents the visible units (input modalities), $\mathbf{h}$ represents the hidden units inside the network, $\mathcal{Z}(\Theta)$ is the normalizing function, $\mathcal{E}$ is the cumulative state of the final layer and $\Theta = \{\mathbf{a}, \mathbf{W}\}$ represent the set of RBM parameters ($\mathbf{a}$ are the biases for the hidden layers).

In essence, feature learning is performed at the level of network parameters, represented by the weights between the nodes and network depth. Training is performed by back-propagation, running several times over the training set, gradually optimizing the output with gradient descent by adjusting the network parameters to match the expectation provided as label to each instance of the training set.

Model Training

Conventional RBM training, using unsupervised learning as pre-training followed by fine-tuning with backpropagation with labeled data [101], had to be adapted to suit the described multimodal learning architecture. The new training process facilitates preliminary intra-sensor features learning followed by identifying cross-sensor relationships.

The approach was to adapt a new form of denoising autoencoder training [27, 29, 31]. This multiple phases training procedure aims to reach a joint probability distribution over all the sensor modalities, more formally:

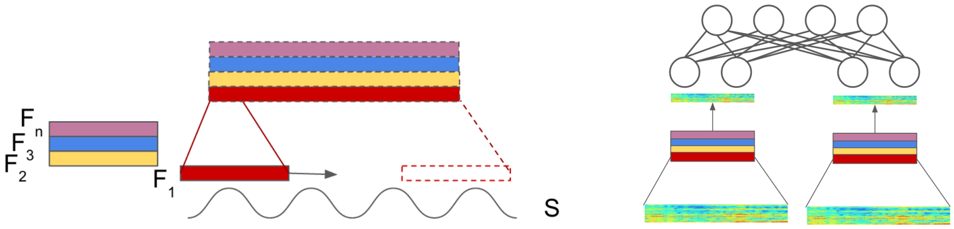
$$\begin{aligned} P(\mathbf{v}_1, \ldots, \mathbf{v}_n; \Theta) \quad &= \sum_{\mathbf{h}_\mathbf{1}^{(2)}, \ldots, \mathbf{h}_\mathbf{n}^{(2)}, \mathbf{u}} P(\mathbf{h}_\mathbf{1}^{(2)}, \ldots, \mathbf{h}_\mathbf{n}^{(2)}, \mathbf{u}) \\ &\prod_{i=1}^{n} (\sum_{h_i^{(1)}} P(\mathbf{v}_i, \mathbf{h}_\mathbf{i}^{(1)}, \mathbf{h}_\mathbf{i}^{(2)})) \end{aligned} \tag{5.3}$$

The initial step is to construct per-modality individual modules that allow the hidden units of the sensor-specific architecture branches to be set through the conventional unsupervised RBM pre-training approach. This process is repeated to build up the number of hidden layers determined for each sensor modality based on a standard hyper-parameter search. Next, collectively each individual network is joined to initialize the first shared hidden layer (based on the values of each contributing network). The newly proposed Dropout mechanism presented in [104] was used alongside autoencoders to increase robustness to noise by having clean data reconstructed successfully from a noisy input. Using Dropout has the effect of integrating many virtual neural networks in the same architecture, thus increasing the robustness to noisy data. Besides Dropout, interleaved in the architecture are normalizing layers, which guarantee a healthy update to weights by keeping their distribution in balance.

5.1.1 Learning representations with Convolutional Neural Networks

Convolutional Neural Networks (CNN) are one of the most successful variant of deep learning architectures, impacting performance in many research areas [22]. This is widely seen as the dominant representative of deep learning architectures, so we explore this as a baseline for the MM-RBM architecture considered here. The approach for using CNNs with multimodal streams is the same as for the MM-RBM, each network branch (sensing modality) enjoying a dedicated CNN for features extraction before unifying the information with fully connected layers in the higher levels of the architecture. The training process and performing inferences are similar to the ones described for MM-RBM before, so model constructions using CNNs work well with exploited framework.

Though very similar to standard RBMs, CNNs have a completely different approach to representing learned information internally. Differences stem from the use of many convolution filters to slide over input data, each being sensitive to different patterns in the signal. A good number of convolution filters assures that an architecture is sensitive to many distinguishable patterns. Figure 5.2(a) illustrate the convolution process, sliding filter F_1 across the signal S. Each filter F_1 to F_n is activated to different patterns in the signal, forming a new representation of the original signal. All of these representations are flattened to a single dimension with a MaxPooling layer by maximum signal across filters. Figure 5.2(a) presents just the part of the network dedicated to extracting features from each modality using a CNN.



(a) The convolution process of filter F_1 across the signal S. Each filter generates a new representation based on their sensitivity to different patterns, which are flattened with a MaxPooling layer.

(b) A simplistic view of the architecture with CNNs on each sensing modality generating new representations. These are merged with fully-connected layers.

Figure 5.2: Using Convolutional Neural Networks to generate new representations on each sensing modality.

CNNs have the advantage of recognizing signal features no matter where these are in the sensor stream due to their sliding filters by checking for these features in each position of the time window. Combining these easily identifiable features creates stronger and more reliable higher level features.

5.2 Implementation

Torch[1], a well known open-source platform for deep learning development, was used to implement this multimodal RBM architecture. Torch has the advantage of being a mature platform, widely used in industry and academia with a growing community to maintain and to contribute to its development. Facebook, Google, Twitter and New York University are just a few of the organizations contributing to supporting this platform – themselves using it for their in-house projects.

Torch uses the BLAS (Basic Linear Algebra Subprograms) library, which is highly optimized for mathematical computations. Even though development for Torch is done in Lua programming language, this is interpreted and compiled for C, making Torch very efficient on any platform, including embedded systems of wearable devices.

Development in Torch is made easy by the numerous layers available through the standard packages (e.g., Linear layer, Convolution layer, Batch Normalization layer, etc.). These can be used as constructing blocks to assemble different architectures. Similarly to these layers, for this work I implemented a multimodal layer accepting as input a set of inputs for each modality, and performing the usual operations (forward pass, backward and update).

To speed up the training process, GPUs were used by taking advantage of CUDA libraries, also part of the standard Torch packages. This speedup brought by GPUs was beneficial to trying many combinations of hyperparameters in order to find the best architecture.

The assembling process of the multimodal RBM architecture starts with the newly developed layer forming a container for other basic layers and facilitating parallel computations for each modality. Following the parallel branches are a sequences of linear layers at the top of the architecture, for sensor fusion, ending the network with a SoftMax layer to boost the class with the highest likelihood. Throughout the network Linear layers were separated by Dropout layers, non-linearity (transformation) layers and Batch Normalization layers. These added extra hyperparameters to evaluate with:

[1] http://torch.ch/

values for Dropout were tested in the range [0.2, 0.8] and determined best performing to be 0.4, while between non-linearities (ReLU, Sigmoind and Tanh) the best performance was achieved consistently with ReLU. Other hyperparameters controlled during experiments were learning rate (in [0.001, 0.5]), momentum, batch size, number of layers and size of layers. A stochastic approach was taken to evaluate as many of these combinations, going over more than 100 iterations, each spanning 500 epochs.

5.3 Evaluation

This section presents the performance of bespoke multimodal deep learning architectures across a range of typical inference tasks specific to wearable devices. First of these detection tasks is Human Activity Recognition, now growing in popularity for detection with smartphones. The following section compares the performance of deep learning models to that of shallow classifiers and explores the case of specializing the neural network to each individual user. This is followed by a validation section employing the same algorithms for indoor-outdoor detection, sleep stage detection and landmark matching, demonstrating a good generalization characteristic of the neural network architecture.

5.3.1 Human Activity Recognition

This evaluation is performed on a publicly available dataset [21], capturing accelerometer and gyroscope data collected from a group of 9 participants, each performing a set of 6 typical activities (sitting, standing, walking, climbing stairs, descending stairs, biking). Data was collected with a variety of devices (6 commercially available smartphones, each with different hardware specifications), showing the increased complexity of training and detection on this dataset.

As mentioned in previous sections, one advantage of using deep learning architectures is that no predefined features are required as the network is capable of extracting discriminating features from raw signals on its own. This characteristic is exploited in this evaluation by reducing data pre-processing to minimum. The only intervention is imposed by the nature of generating sensor data with Android devices. As in many other aspects, Android offers a best-effort policy for sensing, meaning that samples are generated as an event – such as when the sensor perceives changes in signal value, for instance when the value of acceleration changes. To guarantee a rigid frame (time window) as input to the neural network, sampling normalization is performed on the sensor signals as a pre-processing stage. This is done with a simple low-pass filter to assure that all data points are equally distant in time and anchored in timestamps generated by the Android API.

Three deep neural network classifiers were explored: a simple RBM with concatenated multimodal input, the multimodal RBM discussed above and a multimodal CNN. For this human activity recognition dataset there are two sensing modalities, accelerometer and gyroscope, each with three data streams representing movement perceived on the three orthogonal axes (Ox, Oy and Oz). Input to the first classifier (concatenated modalities) is obtained by concatenating all these sensor streams described above. For the second classifier, the model is constructed from independent network units on each sensing modality and combined further through a joining unit to generate the final class estimation. We call this network a multimodal RBM (MM-RBM). Input time windows are the same size as the ones used in previous works, 2 seconds [21], to capture the general periodicity in human activities. The same description applies for the multimodal CNN as well.

5.3.1.1 Baseline with shallow classifiers

Accuracy of user activity detection with the most commonly used two classifiers in ubiquitous sensing (Decision Tree and Random Forest) is presented in Table 5.1. These are generic shallow classifiers because of their lower dimension, feature extraction requirement and training process, representing a good baseline of comparison for the deep neural network architectures. These results are obtained by averaging the performance in a leave-one-user-out evaluation method, which exploits the diversity in performing activities across users. For this evaluation, the entire dataset was split into training data capturing all users but one, who is left aside to validate the accuracy of training solution. This process is repeated with permutation of each user as validation subject.

Complexity of this dataset is reflected in the low accuracy (F1 score[2]) of shallow classifiers as seen in Table 5.1. Though using a substantial and complex set of features, these classifiers are limited in their capability to capture strong and discriminative observations essential for detection on this large and diverse dataset (same training features were used as in [21]).

[2] F1 score: https://en.wikipedia.org/wiki/F1_score

Dataset	C4.5(%)	RandomForest(%)
Human Activity Recognition [21]	67	74.5
Indoor-Outdoor Detection [76]	54.87	58.92
Sleep Stages [105]	45.76	58.44

Table 5.1: Average performance (expressed using F1 score) of shallow classifiers on three datasets: IO-dataset - training on two environments, evaluating on the third, AR-dataset evaluation with leave one user out method, trained on all but one user; SS-dataset - cross validation across the entire set of 20 volunteers.

5.3.1.2 Baseline with Classifier Merger (CM)

A simple strategy to combine information from multiple sensors is by Classifiers Merger. This section presents the evaluation of classifiers performing inferences for activity recognition independently on each sensing modality (acceleration and angular velocity) and merging their outputs for a single final estimation. The chosen classifiers were two basic RBM architectures, which are more capable than the shallow ones presented in the previous baseline. There two classifiers were trained on each of the two sensor signals, independently. Based on the statistical performance of these two classifiers, the final output of CM is weighted toward the classifier with the best performance following a voting method.

Table 5.2 shows the performance of RBMs trained on each of the two modalities. The F1 score of RBM on acceleration signal is greater on average than that of RBM on gyroscope signal. Since there are just two modalities, the performance of CM can be imposed by the best performing classifier across all users, or individually for each user if CM is considered specialized for individual user, selecting the best classifier of the two in the second case. In a general scenario without specialization, the performance of a CM will replicate the performance of the RBM on acceleration signal as this has empirically better accuracy overall compared to that of the gyroscope. However, if CM is user sensitive, the performance of CM is influenced by the best performing of the two classifiers per user, as presented in Table 5.2, bottom row.

5.3.1.3 Multimodal Deep Neural Networks – RBM

Training the Multimodal RBM (MM-RBM) is done directly from raw data (with no hand-crafted features) and achieves better performance compared to shallow classifiers

User	a	b	c	d	e	f	g	h	i	Average
F1 score RBM on accel. (%)	69.95	79.1	66.72	76.75	75.54	63.48	65.06	68.27	68.64	70.39
F1 score RBM on gyro. (%)	70.3	74.52	60.84	72.02	73.57	71.34	62.33	78.03	54.2	68.57
F1 score Ensemble (CM) (%)	70.3	79.1	66.72	76.75	75.54	71.34	65.06	78.03	68.64	72.39

Table 5.2: Per user performance of RBM on each of the two modalities. If user specific, the performance of CM will be the performance of the best performing of the two classifiers.

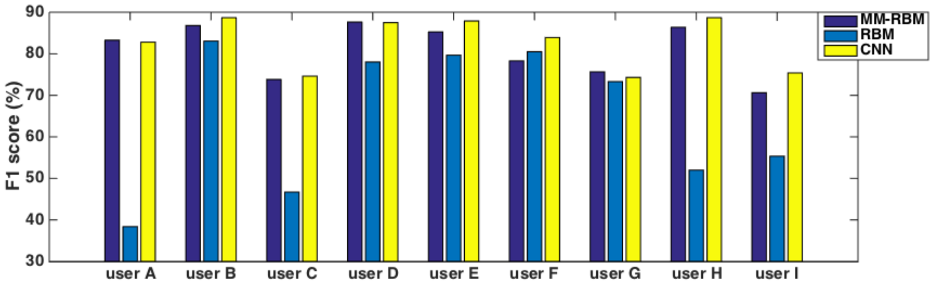
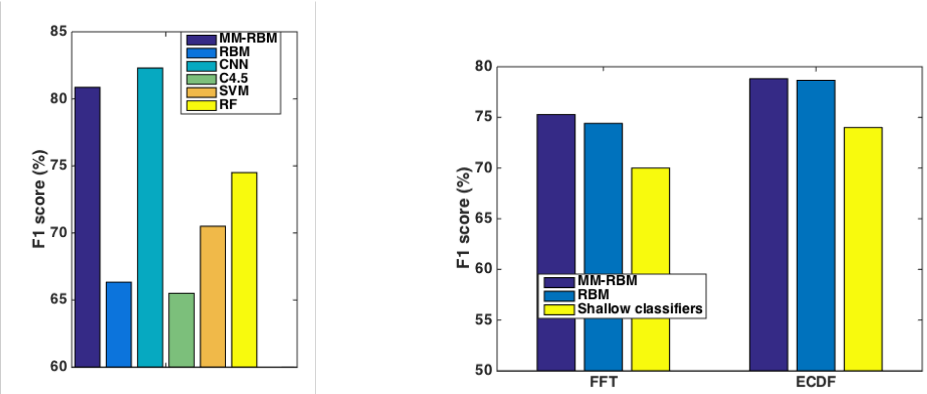


Figure 5.3: Per user comparison of performance between RBM, MM-RBM, CNN on the activity recognition task. With these classifiers training is performed directly on raw data – no preliminary features extraction.

(Figure 5.4(b)). Figure 5.3 shows the performance of RBM classifiers in a per-user evaluation mode (i.e. leave one user out evaluation method). In this evaluation, the size of hidden layers in RBM are slightly smaller than those of MM-RBM to maintain the same computation ratio, though both architectures were trained with identical hyper-parameters. This explains the lower performance of RBM compared to MM-RBM.

Figure 5.4(a) shows the performance of the MM-RBM, along with that of RBM using concatenated multi-modal inputs (referred to as RBM in the figure), multimodal CNN architecture (similar to MM-RBM) and the best performing shallow classifiers evaluated on a leave one user out approach [21]. This clearly shows that deep neural network approaches outperforming shallow classifiers, as well as the CM ensemble presented in Table 5.2. What is remarkable about these deep neural network constructions is that their performance is achieved without any hand selected features, skipping a required process in traditional inference systems.

To put the time domain performance of RBM classifiers into perspective, the evaluation was extended to using transformed signals as input (1) in frequency domain by using the Fast Fourier Transform and (2) by extracting equally spaced components of the ECDF (Empirical Cumulative Distribution Function). These new features are not specific just to human activity recognition, thus being generally useful for any type of inference on any other dataset. Figure 5.4(b) demonstrates the impact of pre-designed features, showing that if instead of raw data (as used in previous experiments), extracting signal transformed features actually decreases the accuracy of RBMs. This shows that raw signals carry substantially greater information, which is usually lost through features extraction. Though their performance is just below that of signals in time domain, RBMs still outperform shallow classifiers even with this features extraction process, showing that traditional classifiers cannot capture the full complexity of signals from pre-selected features.



(a) Comparative performance of the proposed deep-learning architecture (**MM-RBM**), a simple RBM architecture with concatenated sensor streams as input, a multimodal CNN and three shallow classifiers (C4.5, SVM and Random Forest). Deep learning solutions outperform these other traditional solutions for activity recognition with signals from multiple sensors.

(b) Classifier performance using general features: FFT (Fast Fourier Transform) and ECDF (Empirical Cumulative Distribution Function) – not particularly specific to human activity recognition. Evaluation performed the same as before, using the leave one user out method.

Figure 5.4: Accuracy of deep neural network architectures on human activity recognition dataset (a) Sensor streams in time domain; (b) Transforming sensor streams in frequency domain (FFT) and extracting equidistant ECDF features.

5.3.1.4 Using CNNs for feature extraction

An exploration of deep learning solutions for sensor fusion is not complete without considering the best performing architecture proven with so many other tasks, Convolutional Neural Networks (CNNs) [22]. Similar to how convolutions over images extract non-localized discriminative features in small granularity from first layers combining these to more complex features in the upper layers, patterns in sensor signals (inertial sensor streams) can also be captured by CNNs. Intuitively, these non-localized features would improve performance since they can be easily identified with CNNs irrespective to their temporary position in the sensor signal.

Evaluation of multimodal CNNs is performed in the same setup as for MM-RBM described in Section 5.3.1.3. CNNs are constructed using Temporal Convolution layers and Temporal Max Pooling layers for each sensing modality branch and combined with fully connected linear layers at the top of the network.

Training in the same conditions as indicated in Section 5.3.1.3, the F1 score of CNNs is greater by 1.5% than that of fully-connected based MM-RBM (Figure 5.4(a)). However, severe downsides to CNNs are training time (15 times slower training than MM-RBM) and more importantly for running on mobile devices, inference time (CNNs are almost 3 times slower). Results indicate that a fully-connected layer is preferred if training continues on user devices, as motivated in the following section. A shorter training time and inference time saves important energy resources on mobile devices. For this reason, the preferred solutions was chosen to be the MM-RBM architecture constructed with fully-connected layers.

Why fully-connected layers work well? In this situation, fully-connected layers work well because of the highly repetitive nature of signal patterns (inertial sensors - accelerometer and gyroscope), which means that a fix size input window of sensor data is more likely to match other randomly selected windows. For instance, assuming the signal has a sinusoidal shape, this separation between windows is done in starting phase, which is a limited range for highly-repetitive discretized signals. The second aspect is the high number of training samples, which almost guarantees that some windows overlap in patterns, this having very similar encounters in the training set (per our abstraction, matching in phase of a sinusoidal signal). Random selection of starting phase for training time windows and high density of training samples on signals resemble the convolution process of CNNs. This is how RBMs can learn as easily as the CNNs, having similar results, though with much less computation costs.

5.3.1.5 Incremental training

This experiment shows the power of proposed multimodal architecture to specialize on a single device by continuing the training with a small set of labeled data from one specific user.

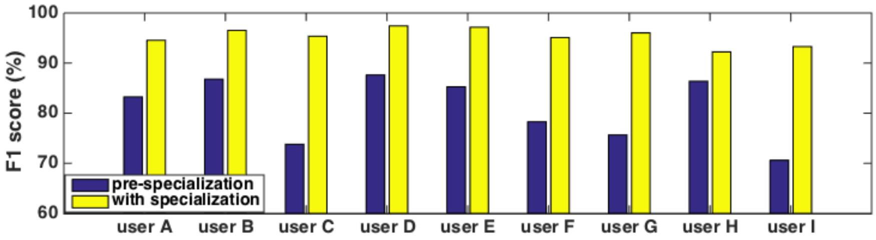


Figure 5.5: Specializing the network architecture to one individual patterns, by continuing the training of a general model on a small number of labeled instances from that user.

In this experiment, the MM-RBM architecture was trained with data from all participants except a single user, as presented in the previous section (see Figure 5.3). This time though, training of the MM-RBM architecture was extended to train on an additional small set of labeled data collected from the participant left out from the first training and used to evaluate the network performance (about 5 minutes of user activity recording data). Thus, performance of the network increases substantially with just a small amount of volunteer data (Table 5.5).

This demonstrates that a model can be trained with a general purpose, followed by fast and efficient specialization on each user and their device to achieve even greater performance. This can be seen as a general trained model shipped to each device and with a small amount of volunteered ground truth labels, the performance of inferences can be substantially improved (to above 95% on average for activity recognition).

5.3.2 Energy Efficiency and Mobile Hardware Feasibility

Using the same deep learning architecture evaluated before (MM-RBM, with two hidden layers on each of the two sensing modalities and joined through another hidden layer before the output), this section presents the energy consumption of this network on a typical hardware specific to wearable devices. Our Torch implementation is interpreted to C and compiled for the platform.

This experiment was conducted on the Qualcomm Snapdragon 410c development board (Figure 5.6). The same processor is found in many smartwatches currently on the market (e.g. LG GWatch R [106]) and includes a quad-core 1.4 GHz CPU and 1 GB of RAM. The key finding is that the MM-RBM is practical for this platform, and consumes a low enough amount of resources (see Table 5.3), making it feasible for wearable and mobile use.

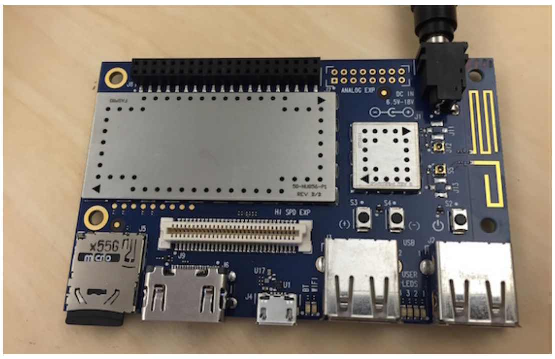


Figure 5.6: Qualcomm Snapdragon 410c. This development board runs a processor common to many smartwatches. We measure the performance of our algorithm on this processor to assess the performance on typical wearable device.

Power consumption was measured with the Monsoon Power Monitor following the same experiment set-up as presented in previous chapters for smartphones.

Metric (unit)	Value
Latency (ms)	50
Memory (MB)	2.75
Energy (mJ)	97

Table 5.3: Resource requirements of the MM-RBM. The low resource demands of MM-RBM makes the model feasible for constrained devices.
Time and energy consumption are indicated per one inference.

To understand the full extend of the energy budget, a comparison was made with the battery capacity of a smartphone. The energy consumption of sensing and performing inferences is negligible, on a device with a typical smartphone battery of 5.55 Wh, at just over 1% with a high sampling rate (1 sample every 2 seconds). By introducing further constrains of batching inferences or reducing sampling frequency, the impact on battery life drops considerably as presented in Figure 5.7.

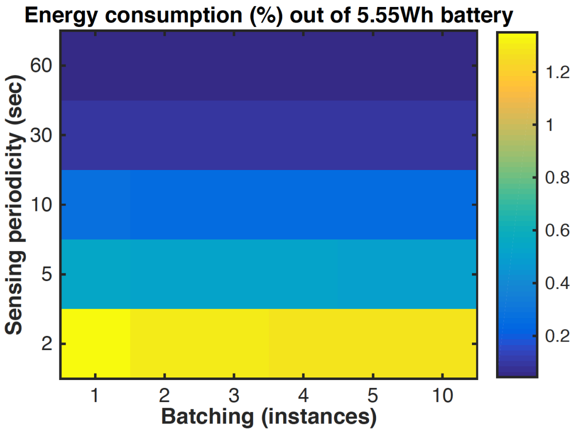


Figure 5.7: Percentage of a 5.55Wh battery consumed by sensing and detecting with the human activity recognition MM-RBM.

5.4 Detection of Other Contexts with Multimodal Deep Learning

To validate the generalization aspect of the deep neural network architecture, this section presents the performance of similar MM-RBM architectures on two other datasets for context sensing.

Sleep Stage (SS) dataset. This contains physiological sensor data [105] (EOG, two EEG and submental-EMG) collected from 20 patients with sleeping disorder. Data was annotated with exact sleep stage by specialists at every 10 seconds. The complexity of this dataset is increased due to patients suffering from sleep disorder, sometimes encountering different sleep stages in a short interval of time, which makes patterns difficult to observe.

Indoor-Outdoor (IO) dataset. This is the larger dataset introduced in the Indoor-Outdoor Detection section [76], which we are already familiar with. To reiterate, this contains smartphone sensor data collected in three different environments: university campus, city and residential area, with labels indicating indoors or outdoors context.

A baseline evaluation with shallow classifiers in presented in Table 5.1. This captures the more difficult cases of training across a large population with different behaviors or having diverse environments like in the case of IO detection – thus the lower performance of shallow classifiers.

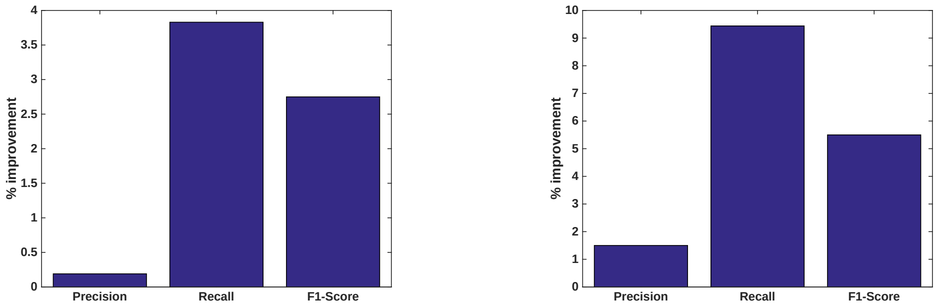
5.4.1 Sleep Stage Detection

The two networks (RBM and MM-RBM) are evaluated on the SS-dataset using a 10-fold cross validation method. In the baseline with shallow classifiers, F1 score is 45.76% and 58.44% respectively for the two simple classifiers. On the other hand, MM-RBM achieves a higher performance as shown in Table 5.4.

	Precision(%)	Recall(%)	F1Score(%)
Wake	69.76	62.56	65.96
Stage1	19.48	59.34	29.36
Stage2	86.28	79.21	82.59
Stage3	74.5	67.71	70.94
REM	63.47	58.55	60.91
Average	76.14	71.36	73.21

Table 5.4: Performance of using MM-RBM with the Sleep Stage detection dataset using a cross validation method.

Figure 5.8(b) shows the performance different between the two deep learning solutions. This is reflected in percentage gain for MM-RBM over the conventional RBM on the three key metrics.



(a) The percentage gain over the conventional RBM that MM-RBM achieves on the IO-dataset.

(b) The percentage gain over the conventional RBM that MM-RBM achieves on the Sleep dataset.

Figure 5.8: Some observations

5.4.2 Indoor-Outdoor Detection

Being able to generalized across environments was observed as a harder problem than cross folds validation, as presented in previous chapter, so the aim here is to validate features extracted in some environment generalize across the other unseen environment.

Table 5.5 presents the performance of this evaluation on a set of classifiers – shallow and deep using leave one environment out evaluation method. It is clear that deep neural network solutions outperform the shallow classifiers on context detection with this dataset too.

Training set	Test set	J48(%)	RF(%)	RBM(%)	MM-RBM(%)
env2 + env3	env1	77.05	68.45	84.5	87.75
env1 + env3	env2	26.6	38.7	57.62	65.44
env1 + env2	env3	60.95	69.63	90.19	92.64

Table 5.5: F1 score for cross environment evaluation on the IO-dataset. Two of the environments were used for training while the third was kept for test.

	Precision(%)	Recall(%)	F1Score(%)
Indoor	65.14	85.2	73.83
Outdoor	95.92	88.4	92.01
Average	89.68	87.75	88.32

Table 5.6: Statistic measures on Multi-Modal RBM running on IO-dataset.

Table 5.6 presents the performance of MM-RBM on the IO-dataset, and the gain over the RBM is presented in Figure 5.8(a). The same observations are applicable here, multimodal RBM outperforming shallow classifiers and RBM throughout the evaluation. This shows the strong generalization aspect of MM-RBM outperforming evaluated solutions across three very distinct datasets.

5.5 Landmarks Discrimination

Taking the positive results for these experiments and the good generalization aspect of MM-RBM across different datasets, it is useful to determine what impact this solution can have on components used for indoor localization. An relevant direction for indoor localization systems is to become more adaptive, instead of continuously using energy-hungry sensors (WiFi or GPS) in places where they may not bring any value, it is important to signal their demand in situations like landmark differentiation, where cheaper sensors are not as efficient. This section demonstrates how MM-RBM architectures can be used to discriminate between landmarks. The importance of this task is justify by systematic recalibration to reduce error accumulation in PDR (as presented in Chapter 3), as well as for on demand positioning. This section demonstrates that WiFi scans and magnetic field signatures are enough to discriminate between very close reference points, using the MM-RBM architecture.

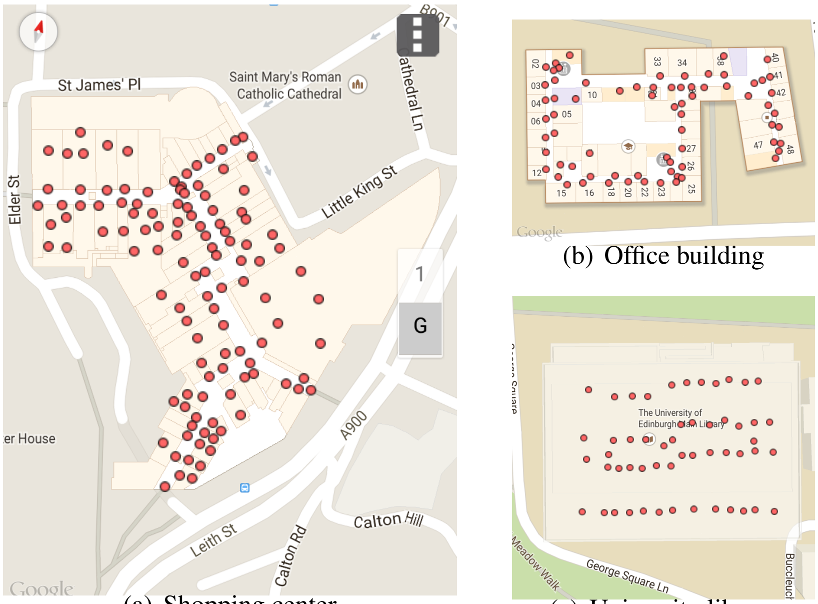
5.5.1 Landmarks and Dataset

Some places inside a building impact the regular mobility pattern in very obvious ways (e.g., stairs, elevators, corners, doors, etc.). Because these elements are static and practically permanent, determining their encouter is essential for indoor localization systems. In literature, landmarks are already essential to many indoor localization systems [6].

For instance, changing the locomotion pattern is observable when the user is transitioning from walking to climbing stairs, or opening and closing a door. These activities show clear patterns on acceleration signal and on gyroscope signal. Identifying these landmarks is trivial, using human activity recognition (using the MM-RBM as presented before for human activity recognition). However, more challenging is matching the position of these landmarks (reference points) to specific points on the map – determined through crowdsourcing as presented in Chapter 3 and other previous research [6, 7].

Continuous tracking will have to rely more on PDR due to the cheaper cost of continuous sensing with inertial sensors and less so on WiFi scans. Ideally, WiFi scans should be triggered when observing landmarks to increase confidence in the system (e.g., to differentiate between the exact door entered in a cluster of near-by doors). In this situation, accurately matching locations and environment observations become critical. While engineering approaches to match fingerprints are suboptimal (e.g., Euclidean distance in signal space [52]), the proposed MM-RBM can facilitate this task by extracting essential observations from training sets. This uses WiFi scans and magnetic field samples as input modalities to the network to facilitate inference.

To assess this opportunity, a dataset of sensor signals at different Landmarks was collected with a custom mobile app. Landmarks were considered to be encountered at different locations in a building where the continuous monotonic movement (walking) is replaced with another dynamic patterns of movement. thus being easy to identify with inertial sensors. Such changes of mobility pattern can be caused by stairs, elevators, doors and corners on corridor or caused by others obstacles in the environment. For visualization of where these landmarks were collected Figure 5.9 presents red dots the landmark position in a subset of the explored buildings.



(b) Office building

(a) Shopping center

(c) University library

Figure 5.9: Distribution of Landmarks in three of the explored buildings, reflecting inflection points on the continuous mobility pattern (stairs, elevator, doors and corners).

5.5.2 Differentiability Between Landmarks

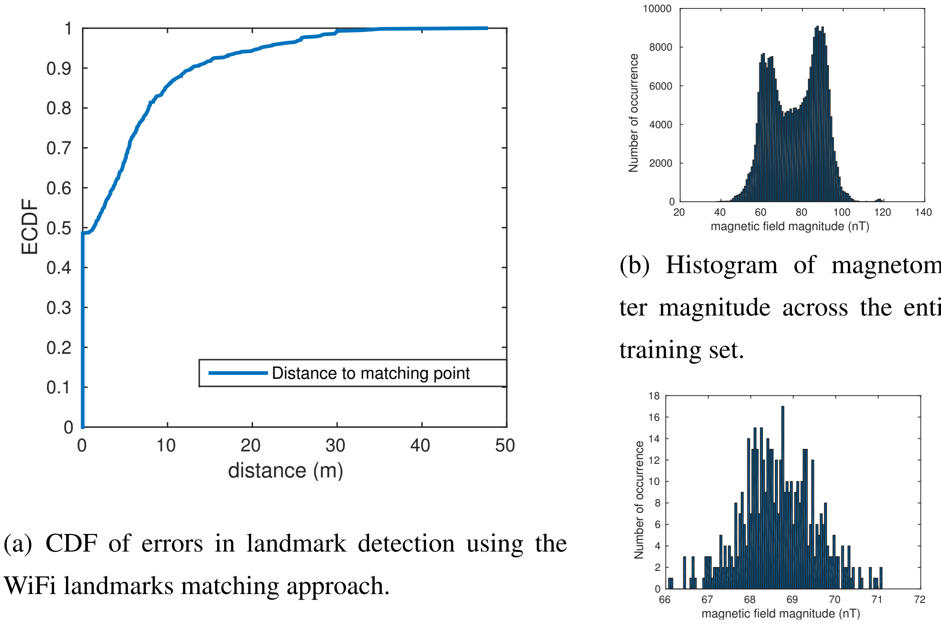
As mentioned before, classic approaches for location matching are far from optimal. The traditional approach of measuring Euclidean Distance in signal space [52] to match the closest reference point in training sets is evaluated here.

$$dist(r_t, r) = \sqrt{\frac{\sum_{i=1}^{N}(r_{t,i} - r_i)^2}{N}} \tag{5.4}$$

where r_t is the RSS of the tested WiFi fingerprint, composed of N number of APs and their signal strength $r_{t,i}$. Each of these are compared to the signal value r_i of the same

APs contained in a previous WiFi fingerprint. If the AP is not present in the other fingerprint, its signal strength values is considered to be -100dBm, equivalent to being too far to have a stable signal.

Figure 5.10(a) shows the euclidean distance in signal space for matching landmarks is just surpassing 50% of accuracy for exact matches, while errors are observed to reach even 20 meters. This is because WiFi signals can be very similar over larger distances for instance with open floor plans. This method is clearly not enough to identify exact matches for landmarks.



(a) CDF of errors in landmark detection using the WiFi landmarks matching approach.

(b) Histogram of magnetometer magnitude across the entire training set.

(c) Histogram of magnetometer magnitude collected at just one landmark location.

Figure 5.10: Observation in the two sensing modalities – WiFi fingerprinting displaying a low matching accuracy on its own and the diversity in magnetometer samples across the training set.

Figures 5.10(b) and 5.10(c) show the magnetic field value over one building and over many buildings respectively. These indicate the difficulty of relying on magnetic field alone for landmarks matching, due to its non-discriminative characteristic at a location and overlapping with many other locations. This while across many buildings values are observed to stay within a small space distribution, again, not very useful to distinguish landmarks purely on this feature.

5.5.3 Multimodal RBM for Landmark Differentiation

The two sensing modalities (WiFi and magnetometer) can be used a input signals to the neural network. As such, providing as one WiFi fingerprint in a WiFi signal vector as presented before, and the magnetic field observations in a 2-second time window and training over the entire dataset of landmarks and location on 16 floor plans, it is observed that the MM-RBM has a better performance (accuracy of exact landmark matches). The MM-RBM outperforms in this task both RBM solutions and Euclidean Distance approach for landmark matching. This shows that using both WiFi and magnetic field observations have more differentiation characteristics than using them individually and the MM-RBM is the best solution to integrate their observations.

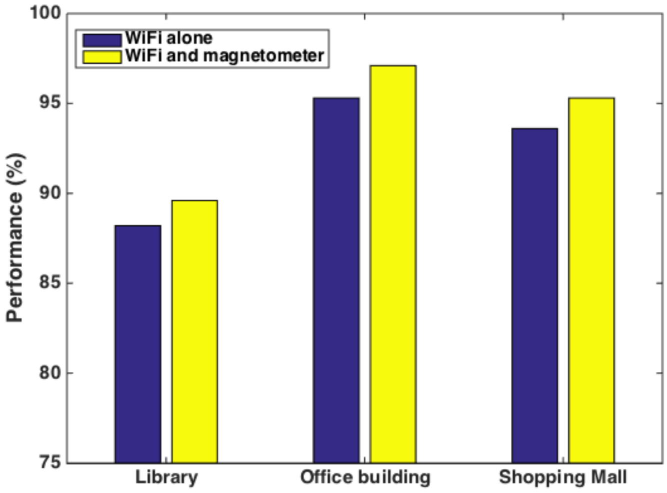


Figure 5.11: The accuracy of predicting exact landmark using the DNN approach over different buildings.

This shows the opportunity to use landmarks to facilitate continuous long-term tracking of mobile devices and benefiting from precise landmark recalibration options, while consuming less energy. By continuously observing the environment with energy-cheaper sensing modalities (accelerometer), more expensive sensing modalities (WiFi and GPS) can be triggered only at relevant moments to discriminate between landmarks. This creates the opportunity for sensor adaptive localization systems that can manage an energy budge and location accuracy trade-off much better, while being still accurate enough to provide quality location estimations.

These results are promising and indicate the potential for more deep neural network integration into indoor localization systems. The generalization property of neural networks can be using for many other components inside localization systems, or even provide localization using neural networks form end-to-end.

5.6 Summary

This chapter showed a variant of Deep Learning, Restricted Boltzman Machine (RBM) with multi-modal input, able to extract more decisive features compared to traditional, shallow classifiers, resulting in higher accuracy of detection for a range of tasks like activity recognition, sleep stages detection and indoor-outdoor detection. While comparable in performance with Convolutional Neural Networks, RBM architectures are computationally lightweight, making them suitable for inferences on mobile devices.

Generalization of this solution was demonstrated with a range of context detection tasks: activity recognition, sleep stage detection, indoor-outdoor detection and landmarks discrimination. Using MM-RBM with WiFi scans and magnetic field variation as sensing modalities for landmarks discrimination is proven to be an efficient recalibration method for indoor localization systems. Their use can potentially be triggered by lower-energy sensors (accelerometer) only when relevant, thus creating adaptive localization systems to reduce energy consumption even further.

Chapter 6

Conclusions and Future work

6.1 Conclusions

Ubiquitous computing is advancing with the expansion of mobile wearable devices carried by humans. Their many sensors can be re-purposed to make interesting observations about users and their surroundings (e.g., location, activity, sleep stage, transportation mode, emotional state, etc.). These forms of context enable more adaptive and intelligent mobile applications to understand their users' needs and over time even to take action for them to control surrounding appliances and adapt environments – the vision of ubiquitous computing.

Though smartphone come with a variate range of sensors, very different in their purpose and operation, this thesis has shown that cross-sensor observations can be made to facilitate better integration, specific to each context detection task. In this work three different approaches have been taken to integrate multimodal sensor data: composition – by combining simpler forms of detection (e.g., step counting, direction estimation, activity recognition, radio map matching), machine learning based solutions to facilitate continuous learning in new environments, and multimodal deep learning to perform inferences directly from raw data.

These three approaches to sensor data integration were developed to satisfy the complexity of three forms of context detection tasks: indoor localization, indoor-outdoor detection and human activity recognition. Important attention was given to accuracy of estimations and energy-efficiency since these are designed for operation on mobile devices. Design, assessment and experimental evaluation for these three forms of context detection tasks are the main contributions of this thesis. High level conclusions for each of these works are presented in the following subsections. Though insightful for how such systems can be created, this work has an exploratory character, with some limitations, which are highlighted in the final section together with proposed solutions to address these in future work.

6.1.1 Indoor Localization

This thesis presented HiMLoc, a hybrid indoor location tracking solution that integrates Pedestrian Dead Reckoning with indoor landmarks detection by activity recognition and WiFi fingerprinting. The main advantage of this solution is that it offers easy deployment due to its simple requirements of only a small set of building parameters (e.g., location of elevators, stairs, main doors, corners and distance between floors) and can provide good estimation for most smartphones by using just three of the most common sensors present on smartphones: accelerometer, compass and WiFi card. This integration of PDR with WiFi fingerprinting based estimations is performed by a particle filter and the introduced concept of similarity area for WiFi fingerprints. Very distinct fingerprints over a small area tend to provide very good location estimation accuracy as do fingerprints obtained from the same floor. Evaluations show that HiMLoc achieves median location error below 3 meters in most cases, recognizing two states of carrying the phone, with the phone in hand and with the phone in pocket.

6.1.2 Indoor-Outdoor Detection

Another problems explored in this thesis is determining whether a user is indoors or outdoors using low power sensors readily available on modern smartphones. For this IO detection problem, existing solutions were shown to be too energy hungry or fail to provide accurate results across a range of different environments typically encounter in practice, due to the use of fixed and environment agnostic thresholds in the underlying estimation schemes. Observations that further improvements can be achieved by viewing the IO detection as a machine learning classification problem with 2 classes (indoor, outdoor) guided this research to adapt classifier models to new environments and devices in order to achieve robust and accurate detection across diverse settings.

To address this fundamental condition of model adaptation on-the-fly, a semi-supervised learning framework was adopted as the ultimate solution approach. Through the investigation of different commonly used semi-supervised learning methods, co-training method was found to yield most accurate results across a range of environments and different devices. An implementation of this co-training method on Android platforms was presented in this thesis, using an incremental version of Naive Bayes classifier. It was shown that this approach outperforms other alternative methods in terms of both accuracy and energy efficiency. Also, this implementation does not incur any communication overhead (as it does not need to communicate with a backend/cloud) and is privacy preserving. The use case application of switching off the WiFi interface when the user is outdoors was shown to save considerable power, thus extending the battery life in usual conditions.

6.1.3 Context Detection with Deep Learning

The final solutions presented in this work, using deep neural networks, combines the elegance of performing analysis directly on the raw data, without preliminary feature extraction and the agility to run efficiently on mobile devices with contained resources. This approach is designed to increase the information captured in the numerous simple low-energy sensors found in mobile devices (e.g., light, magnetometer, accelerometer, barometer, heart-rate, proximity). Using human activity recognition as a use case with accelerometer and gyroscope signals, experiments show a significant gain in accuracy for the deep learning approach over traditional classifiers. Continuing to specialize a pre-trained neural network for activity recognition with just a small amount of labeled data from final user, increases the accuracy of estimations to above 95%. Experiments that span a wide range of: sensor types; competing multimodal learning algorithms; and, activities and context targets – collectively show the proposed general-purpose deep approach to multimodal mobile sensor modeling is broadly applicable and it exceeds the performance of even task-specific features based models.

This exploration is facilitate by a proof-of-concept implementations that is used to measure the overhead of such modeling techniques. Results for user activity recognition clearly indicates that battery life impact, memory requirements and inference time are minimal, allowing this to run continuously on the phone, despite the complexity of deep learning methods.

6.2 Discussion and Opportunities for Future Work

As highlighted in the previous section, this work advances our understanding of what is possible to achieve in mobile computing by sensing with mobile and wearable devices in a number of directions. However it is important to be mindful of assumptions, biases and limitations related to these findings. This section provides a reflective discussion on the presented work, lessons learned and the opportunity to improve this in future work.

The work presented in this thesis has an emphasis on experimentation throughout, with proposed systems undergoing design, development and experimental validation a very appreciated scientific method but comes with certain limitations. The cost of developing experiments is a major issue with this approach, requiring participants, devices and access to different environments for evaluation, while system iterations may not experience the same setup characteristics, due to changes in the experimentation environment over longer periods of time (e.g., WiFi APs variation, crowds, weather and other factors) and sensing characteristics (e.g., noisy sensor samples, device temperature and other processes running irregularly in parallel on the phone). That is also the reason why competitions bringing together researchers to experimentally evaluate their systems in the same place at the same time have emerged, such as the Microsoft Indoor Localization Competition [107].

The work presented in this thesis is also affected by these common problems impacting experimentation, these being reflected in the reduced number of participants (below 10 in most experiments), with a reduced number of devices and across a small number of experimental conditions which is not always representative of the many possible experimental conditions. For the indoor localization work, samples from only two young and healthy participants with similar body features were used to train the activity recognition component, the distance error model and the heading deviation model, while only one participant collected WiFi samples in indoor spaces to characterize the similarity area, everything using just one mobile device. For the indoor-outdoor work only two participants collected the training sets, using just 4 different devices, with experiments spanning less than a month, reflecting the weather conditions in spring, in just one city, Edinburgh. For the activity recognition, there were only 9 participants, far from being representative for the entire population, using just 6 different devices. Though results are decisive using these limited resources, future work should scale up these experiments to assess if these observations still hold across a larger population and in other environments.

To facilitate evaluation at a larger scale, new frameworks to easily collect larger amounts of experimental data will need to be developed in future work. Relevant here is the potential bias in my work stemming from how ground truth coordinates were collected while experimenting with HiMLoc. This was done by a researcher walking behind participants and signaling when a reference point was encountered by participant on the path to associate the reference point location with participant ground truth location. Ideally, there should be an automated process to collect these ground truth information, extremely relevant for training any context detection models. One option would be to deploy fixed infrastructure to collect ground truth information automatically such as cameras mounted in buildings to collect exact coordinates of pedestrians through computer vision. Besides location, cameras can also assist with activity recognition training and direction detection. For the indoor-outdoor detection system limitations were observed when using the system out of the box at different latitudes, requiring a few extra ground truth labels to stabilize, which also indicates the need for more training data.

Another interesting question to answer in future work is if we can find strong correlations between different forms of context, such as for example activity recognition and localization inside a building. With some contexts being easier to infer on raw sensor data than others, strong associations between contexts will allow these to be trained in coordination (or subordination), using inferred labels between them. This provides additional weakly-labeled data to expand training sets of context detection tasks for which obtaining training data is expensive. The bias in labeling quality will need to be investigated as well in future work.

Running over long periods of time with potentially erroneous inferences, the proposed indoor-outdoor detection system can accumulate signals that can eventually affect its performance as there are currently no safeguards against this happening. For analogy with HiMLoc, the Co-training mechanism functions similar to the PDR component, estimating states following previous observations in sequential order. To avoid accumulation of erroneous observations over time, a calibration mechanism can be useful, just like WiFi fingerprinting or landmarks detection used in HiMLoc. These solutions may built on correlations with other context detection schemes (e.g., elevator movements will always indicate the user is inside a building), or by opportunistically integration with the GPS or Bluetooth when confidence is low from both classifiers and, thus bring a external form of correction when needed and available from other sensors with higher confidence. Modeling the confidence region for the other anchor sensors can also be explored in the future.

Simulation is also a valid approach to confirm hypothesis. During this work, simulation was used to evaluate different versions of the algorithms on prerecorded data. This was the case for indoor localization, sensor data collected from users walks being analyzed offline for better understanding and used to guide different variations of the implementation. For instance, the similarity area threshold was determined offline by assessing its impact on location estimation accuracy. For indoor-outdoor detection readily collected datasets were used to simulate the prediction accuracy of different machine learning techniques, while for the activity recognition exploration the same dataset was used to experiment with different neural network architectures. Nevertheless, there is further room for simulation based assessment that can be explored in future work, such a building a simulator that can account for many variations across users not available through experimental data collection, to train models for each component in the particle filter, simulate a variety of other environments and devices. This will have the effect of alleviating some of the issues related to experimentation in the large data collection required for validation and generalization.

And finally, in order to increase the amount of data available for training these systems, the ultimate aim is to have them adopted by many users and running them over long periods of time. Providing a good case for these systems is not easy when it comes to privacy and security. A few key points that all systems should comply with when collecting user data is to perform more of the computations locally to the devices, with little data leaving the device and taking great consideration for the battery life of the device. Performing more of the computations locally and consuming the inference locally will assure users their data is disposed of after the context is determined. Even so, occasional data will need to be uploaded to the cloud when the system identifies useful data to update generic common models in the cloud, for instance when classifier confidence is low or when identifying unexplored environments. In these situations, data needs to be anonymized and used only for the purpose of training the context detection classifier in the cloud. Many solutions can be explored of how privacy can be guarantied in training global models across many devices, such as splitting training between devices and the cloud [108, 109].

The availability of these larger datasets will facilitate training of deep neural networks for many context detection tasks. One such solution is to construct indoor localization with neural networks from end-to-end, thus avoiding individual training of sub-components (distance estimation, direction estimation, activity recognition, WiFi fingerprinting).

The wider purpose of context detection systems is to understand users in their environment and provide them useful services. A larger vision for context detection is to contribute to context-aware personal assistants, which will emerge from this understanding. These personal assistants can be used to control network connected devices such as lights, radio, thermostats and other appliances. Sharing intelligence between these devices and wearable devices (smartphones, smart watches) allows for a deeper understanding of user context. By learning user behavior over time in relation to observable contexts, these systems will be able to control devices in the environment to suit different needs without explicit intervention from their users. This will lead to truly smart environments, adapting to user actions, mood and intentions.

Bibliography

[1] “Number of smartphone users worldwide from 2014 to 2020,” https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/, 2016.

[2] “Global Mobile Data Traffic Forecast Update, 2016-2021 White Paper,” http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html, 2016.

[3] M. Weiser, “The computer for the 21st century,” in *ACM SIGMOBILE Mobile Computing and Communications Review*, 1999.

[4] G. Welch and G. Bishop, “An Introduction to the Kalman Filter,” in *SIGGRAPH Course Notes*, 2001.

[5] A. K. Dey, *Ubiquitous Computing Fundamentals*. CRC Press, 2010.

[6] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury, “UnLoc: No Need to War-Drive: Unsupervised Indoor Localization,” in *Proc. MobiSys*. ACM, 2012.

[7] A. Rai *et al.*, “Zee: Zero-Effort Crowdsourcing for Indoor Localization,” in *Proc. ACM MobiCom*, 2012.

[8] M. Youssef, M. A. Yosef, and M. El-Derini, “GAC: Energy-Efficient Hybrid GPS-Accelerometer-Compass GSM Localization,” in *Proc. IEEE GLOBECOM*, 2010.

[9] J. Paek, K.-H. Kim, J. P. Singh, and R. Govindan, “Energy-efficient positioning for smartphones using Cell-ID sequence matching,” in *Proc. MobiSys*. ACM, 2011.

[10] M. Keally, G. Zhou, G. Xing, J. Wu, and A. Pyles, "PBN: Towards Practical Activity Recognition Using Smartphone-based Body Sensor Networks," in *Proc. SenSys*. ACM, 2011.

[11] L. Stenneth, O. Wolfson, P. S. Yu, and B. Xu, "Transportation mode detection using mobile phones and gis information," in *Proc. GIS*. ACM, 2011.

[12] I. Constandache, X. Bao, M. Azizyan, and R. R. Choudhury, "Did you see Bob?: Human Localization using Mobile Phones," in *Proc. ACM MobiCom*, 2010.

[13] V. Radu, L. Kriara, and M. K. Marina, "Pazl: A Mobile Crowdsensing based Indoor WiFi Monitoring System," in *Proc. IEEE CNSM*, 2013.

[14] P. Zhou, Y. Zheng, Z. Li, M. Li, and G. Shen, "IODetector: A Generic Service for Indoor Outdoor Detection," in *Proc. SenSys*. ACM, 2012.

[15] S. Bhattacharya, H. Blunck, M. Kjrgaard, and P. Nurmi, "Robust and energy-efficient trajectory tracking for mobile devices," in *IEEE Trans. Mobile Computing (TMC)*, 2015.

[16] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in *In Proc. Conf. Pervasive Computing (Pervasive)*, 2004.

[17] B. Logan, J. Healey, M. Philipose, E. M. Tapia, and S. Intille, "A long-term evaluation of sensing modalities for activity recognition," in *In Proc. international conference on Ubiquitous computing (UbiComp)*, 2007.

[18] J. Lester, T. Choudhury, and G. Borriello, "A practical approach to recognizing physical activities," in *In Proc. Conf. Pervasive Computing (Pervasive)*, 2006.

[19] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," in *In ACM SIGKDD Explorations Newsletter*, 2011.

[20] J. Dai, X. Bai, Z. Yang, Z. Shen, and D. Xuan, "Mobile phone-based pervasive fall detection," in *In Proc. international conference on Ubiquitous computing (UbiComp)*, 2010.

[21] A. Stisen and et al., "Smart devices are different: Assessing and mitigating mobile sensing heterogeneities for activity recognition," in *In Proc. SenSys*, 2015.

[22] Y. Bengio, I. J. Goodfellow, and A. Courville, "Deep learning," 2015.

[23] T. P. N. Y. Hammerla, S. Halloran, “Deep, convolutional, and recurrent models for human activity recognition using wearables,” in *In Proc. IJCAI*, 2016.

[24] N. D. Lane and P. Georgiev, “Can Deep Learning Revolutionize Mobile Sensing?” in *In Proc. HotMobile*. ACM, 2015.

[25] N. Srivastava and R. R. Salakhutdinov, “Multimodal learning with deep boltzmann machines,” in *In Proc. NIPS*, 2012.

[26] D. S. Sachan, U. Tekwani, and A. Sethi, “Sports video classification from multimodal information using deep neural networks,” in *AAAI*, 2013.

[27] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, “Multimodal deep learning,” in *In Proc. ICML*, 2011.

[28] “Ekahau Mobile Survey,” http://www.ekahau.com/products/ekahau-mobile-survey/mobile-survey-overview.html.

[29] P. Wu, S. C. Hoi, H. Xia, P. Zhao, D. Wang, and C. Miao, “Online multimodal deep similarity learning with application to image retrieval,” in *Proceedings of the 21st ACM International Conference on Multimedia*, ser. MM ’13. New York, NY, USA: ACM, 2013, pp. 153–162. [Online]. Available: http://doi.acm.org/10.1145/2502081.2502112

[30] K. Sohn, W. Shang, and H. Lee, “Improved multimodal deep learning with variation of information,” in *In Proc. NIPS*, 2014.

[31] W. Liu, W. Zheng, and B. Lu, “Multimodal emotion recognition using multimodal deep learning,” *CoRR*, 2016.

[32] I. E. Radoi, J. Mann, and D. Arvind, “Tracking and monitoring horses in the wild using wireless sensor networks,” in *Proc. IEEE WiMob*, 2015.

[33] ——, “Performance Evaluation of the VB-TDMA Protocol for Long-term Tracking and Monitoring of Mobile Entities in the Outdoors,” in *Proc. ACM Symposium on QoS and Security for Wireless and Mobile Networks*, 2015.

[34] J. Mann, I. E. Radoi, and D. Arvind, “Prospeckz-5–A Wireless Sensor Platform for Tracking and Monitoring of Wild Horses,” in *Proc. Euromicro, Digital System Design (DSD)*, 2014.

[35] R. Stirling, J. Collin, K. Fyfe, and G. Lachapelle, “An Innovative Shoe-Mounted Pedestrian Navigation System,” in *GNSS*, 2003.

[36] B. Krach and P. Roberston, “Cascaded estimation archi- tecture for integration of foot-mounted inertial sensors,” in *Proc. IEEE Position Location and Navigation Symposium*, 2008.

[37] N. Castaneda and S. Lamy-Perbal, “An improved shoe- mounted inertial navigation system,” in *Proc. IEEE IPIN*, 2010.

[38] O. Woodman and R. Harle, “Pedestrian localisation for indoor environments,” in *Proc. ACM UbiComp*, 2008.

[39] F. Cavallo, A. Sabatini, and V. Genovese, “A step toward GPS/INS personal navigation systems: real-time assessment of gait by foot inertial sensing,” in *Proc. IEEE Conference on Intelligent Robots and Systems*, 2005.

[40] L. Ojeda and J. Borenstein, “Non-GPS navigation with the personal dead-reckoning system,” in *Proc. SPIE*, 2007.

[41] A. R. Jimenez, F. Seco, C. Prieto, and J. Guevara, “A comparison of Pedestrian Dead-Reckoning algorithms using a low-cost MEMS IMU,” in *IEEE International Symposium on Intelligent Signal Processing*, 2009.

[42] E. Foxlin, “Pedestrian Tracking with Shoe-Mounted Iner- tial Sensors,” in *IEEE Computer Graphics and Applications*, 2005.

[43] L. Fang, P. Antsaklis, L. Montestruque, M. McMickell, M. Lemmon, Y. Sun, H. Fang, I. Koutroulis, M. Haenggi, M. Xie, and X. Xie, “Design of a Wireless Assisted Pedestrian Dead Reckoning SystemThe NavMote Experience,” in *IEEE Trans. Instrum. Meas.*, 2005.

[44] P. Goyal, V. J. Ribeiro, H. Saran, and A. Kumar, “Strap-down Pedestrian Dead-Reckoning system,” in *Proc. IEEE IPIN*, 2011.

[45] H. Ying, C. Silex, A. Schnitzer, S. Leonhardt, M. Schiek, S. Leonhardt, T. Falck, P. Mahonen, and R. Magjarevic, “4th International Workshop on Wearable and Implantable Body Sensor Networks,” in *Springer Berlin Heidelberg*, 2007.

[46] H. WeinBerg, “AN-602: Using the ADXL202 in Pedometer and Personal Navigation Applications,” in *Analog Devices, Tech. Rep.*, 2002.

[47] S. Yang and Q. Li, "Ambulatory walking speed estimation under different step lengths and frequencies," in *Proc. IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2010.

[48] M. H. Afzal, V. Renaudin, and G. Lachapelle, "Assessment of indoor magnetic field anomalies using multiple magnetometers," in *Proc. ION GNSS*, 2010.

[49] R. Harle, "A Survey of Indoor Inertial Positioning Systems for Pedestrians," *IEEE Communications Surveys & Tutorials*, 2013.

[50] I. Constandache, R. R. Choudhury, and I. Rhee, "Towards Mobile Phone Localization without War-Driving," in *Proc. IEEE INFOCOM*, 2010.

[51] D. Gusenbauer, C. Isert, and J. Krsche, "Self-Contained Indoor Positioning on Off-the-Shelf Mobile Devices," in *Proc. IEEE IPIN*, 2010.

[52] P. Bahl and V. N. Padmanabhan, "RADAR: An In-Building RF-Based User Location and Tracking System," in *Proc. IEEE INFOCOM*, 2000.

[53] M. Youssef and A. K. Agrawala, "The Horus WLAN Location Determination System," in *Proc. MobiSys*, 2005.

[54] P. Bolliger, "Redpin – Adaptive, Zero-Configuration Indoor Localization through User Collaboration," in *Proc. ACM MobiCom MELT Workshop*, 2008.

[55] J. Park *et al.*, "Growing an Organic Indoor Location System," in *Proc. MobiSys*, 2010.

[56] B. Ferris, D. Fox, and N. Lawrence, "WiFi-SLAM Using Gaussian Process Latent Variable Models," in *Proc. IJCAI*, 2007.

[57] Y. Kim, H. Shin, Y. Chon, and H. Cha, "Smartphone-based Wi-Fi Tracking System Exploiting the RSS Peak to Overcome the RSS Variance Problem," *Elsevier Pervasive and Mobile Computing*, vol. 9, no. 3, Jun 2013.

[58] R. M. Faragher, C. Sarno, and M. New, "Opportunistic Radio SLAM for Indoor Navigation using Smartphone Sensors," in *Proc. IEEE Position Location and Navigation Symposium (PLANS)*, 2012.

[59] L. Ravindranath, C. Newport, H. Balakrishnan, and S. Madden, "Improving-Wireless Network Performance Using Sensor Hints," in *Proc. USENIX NSDI*, 2011.

[60] K. Chintalapudi, A. P. Iyer, and V. N. Padmanabhan, "Indoor Localization without the Pain," in *Proc. ACM MobiCom*, 2010.

[61] A. C. Santos, J. M. Cardoso, D. R. Ferreira, P. C. Diniz, and P. Chaínho, "Providing user context for mobile and social networking applications," *Pervasive and Mobile Computing*, vol. 6, no. 3, 2010.

[62] J. J. Pan, S. J. Pan, J. Yin, L. M. Ni, and Q. Yang, "Tracking Mobile Users in Wireless Networks via Semi-Supervised Co-Localization," *Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[63] H. Ye *et al.*, "FTrack: Infrastructure-free Floor Localization via Mobile Phone Sensing," in *Proc. IEEE PerCom*, 2012.

[64] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine," *Lecture Notes in Computer Science*, vol. 7657 LNCS, pp. 216–223, 2012.

[65] I. Constandache, S. Gaonkar, M. Sayler, R. Choudhury, and L. Cox, "Enloc: Energy-efficient localization for mobile phones," in *Proc. INFOCOM*. IEEE, 2009.

[66] D. H. Kim, Y. Kim, D. Estrin, and M. B. Srivastava, "SensLoc: Sensing Everyday Places and Paths Using Less Energy," in *Proc. SenSys*. ACM, 2010.

[67] N. Brouwers and M. Woehrle, "Dwelling in the canyons: Dwelling detection in urban environments using gps, wi-fi, and geolocation," *Pervasive and Mobile Computing*, vol. 9, no. 5, 2013.

[68] J. Yang, E. Munguia-Tapia, and S. Gibbs, "Efficient In-pocket Detection with Mobile Phones," in *Proc. UbiComp*. ACM, 2013.

[69] E. Miluzzo, M. Papandrea, N. D. Lane, H. Lu, and A. T. Campbell, "Pocket, Bag, Hand, etc. - Automatically Detecting Phone Context through Discovery," in *Proc. PhoneSense*. ACM, 2010.

[70] Y. Wang, J. Lin, M. Annavaram, Q. A. Jacobson, J. Hong, B. Krishnamachari, and N. Sadeh, "A Framework of Energy Efficient Mobile Sensing for Automatic User State Recognition," in *Proc. MobiSys*. ACM, 2009.

[71] S. Hemminki, P. Nurmi, and S. Tarkoma, "Accelerometer-based transportation mode detection on smartphones," in *Proc. SenSys.* ACM, 2013.

[72] T. Choudhury, G. Borriello, S. Consolvo, D. Haehnel, B. Harrison, B. Hemingway, J. Hightower, P. P. Klasnja, K. Koscher, A. LaMarca, J. A. Landay, L. LeGrand, J. Lester, A. Rahimi, A. Rea, and D. Wyatt, "The mobile sensing platform: An embedded activity recognition system," *IEEE Pervasive Computing*, vol. 7, no. 2, pp. 32–41, Apr. 2008. [Online]. Available: http://dx.doi.org/10.1109/MPRV.2008.39

[73] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics).* Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[74] A. Kapoor and R. W. Picard, "Multimodal affect recognition in learning environments," in *Proceedings of the 13th annual ACM international conference on Multimedia.* ACM, 2005, pp. 677–682.

[75] P. Zappi, T. Stiefmeier, E. Farella, D. Roggen, L. Benini, and G. Tröster, "Activity recognition from on-body sensors by classifier fusion: Sensor scalability and robustness," in *3rd Int. Conf. on Intelligent Sensors, Sensor Networks, and Information Processing (ISSNIP)*, 0 2007, pp. 281–286. [Online]. Available: http://www2.ife.ee.ethz.ch/~droggen/publications/wear/EDAS_ISSNIP.pdf

[76] V. Radu, P. Katsikouli, R. Sarkar, and M. K. Marina, "A Semi-Supervised Learning Approach for Robust Indoor-Outdoor Detection with Smartphones," in *Proc. Conference on Embedded Networked Sensor Systems (SenSys).* ACM, 2014.

[77] "Weka - Machine Learning Suite," http://www.cs.waikato.ac.nz/ml/weka/.

[78] J. Hightower and G. Borriello, "Particle Filters for Location Estimation in Ubiquitous Computing : A Case Study," in *Computing*, 2004.

[79] R. Kiros, R. Salakhutdinov, and R. S. Zemel, "Unifying visual-semantic embeddings with multimodal neural language models," *CoRR*, vol. abs/1411.2539, 2014. [Online]. Available: http://arxiv.org/abs/1411.2539

[80] A. Y. Hannun, C. Case, J. Casper, B. C. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, "Deep speech:

Scaling up end-to-end speech recognition," *CoRR*, vol. abs/1412.5567, 2014. [Online]. Available: http://arxiv.org/abs/1412.5567

[81] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *In Proc. NIPS*, 2012.

[82] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[83] W. Wang, R. Arora, K. Livescu, and J. Bilmes, "On deep multi-view representation learning," in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015, pp. 1083–1092.

[84] S. Ebrahimi Kahou, X. Bouthillier, P. Lamblin, Ç. Gülçehre, V. Michalski, K. R. Konda, S. Jean, P. Froumenty, Y. Dauphin, N. Boulanger-Lewandowski, R. Chandias Ferrari, M. Mirza, D. Warde-Farley, A. Courville, P. Vincent, R. Memisevic, C. Pal, and Y. Bengio, "Emonets: Multimodal deep learning approaches for emotion recognition in video," *Journal on Multimodal User Interfaces*, pp. 1–13, 2015. [Online]. Available: http://dx.doi.org/10.1007/s12193-015-0195-2

[85] Y. Kim, H. Lee, and E. Provost, "Deep learning for robust feature generation in audiovisual emotion recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, May 2013, pp. 3687–3691.

[86] V. Radu and M. Marina, "HiMLoc: Indoor Smartphone Localization via Activity Aware Pedestrian Dead Reckoning with Selective Crowdsourced WiFi Fingerprinting," in *Proc. IEEE IPIN*, 2013.

[87] V. Radu, J. Li, L. Kriara, M. K. Marina, and R. Mortier, "A hybrid approach for indoor mobile phone localization," in *Proc. ACM MobiSys*, 2012.

[88] V. Radu, "Indoor localization of mobile devices for wireless networks monitoring system based on Pedestrian Dead Reckoning," http://www.inf.ed.ac.uk/publications/thesis/online/IM111056.pdf, 2011, [Online; accessed 20-July-2016].

[89] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz, "Localization from Mere Connectivity," in *Proc. ACM MobiHoc*, 2003.

[90] D. Shepard, “A Two-Dimensional Interpolation Function for Irregularly-Spaced Data,” in *Proc. 23rd ACM National Conference*, 1968.

[91] A. Primo, V. V. Phoha, R. Kumar, and A. Serwadda, “Context-aware active authentication using smartphone accelerometer measurements,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2014.

[92] V. Radu, P. Katsikouli, R. Sarkar, and M. Marina, “Poster: Am I Indoor or Outdoor?” in *Proc. Conference on Mobile Computing and Networking (MobiCom)*. ACM, 2014.

[93] X. Zhu and A. B. Goldberg, *Introduction to Semi-Supervised Learning*. Morgan & Claypool Publishers, 2009.

[94] X. Zhu, “Semi-supervised learning with graphs,” Ph.D. dissertation, Carnegie Mellon University, 2005.

[95] K. Murphy, *Machine learning a probabilistic perspective*. MIT Press, 2012.

[96] O. Chapelle, *Semi-supervised learning*. Cambridge, Mass: MIT Press, 2006.

[97] A. Blum and T. Mitchell, “Combining labeled and unlabeled data with co-training,” in *Proc. COLT*. ACM, 1998.

[98] S. Dey, N. Roy, W. Xu, R. R. Choudhury, and S. Nelakuditi, “AccelPrint: Imperfections of Accelerometers Make Smartphones Trackable,” in *Proc. NDSS*, 2014.

[99] J. D. Rennie, L. Shih, J. Teevan, and D. Karger, “Tackling the poor assumptions of naive bayes text classifiers,” in *Proc. ICML*, 2003.

[100] V. Radu, N. D. Lane, S. Bhattacharya, C. Mascolo, M. K. Marina, and F. Kawsar, “Towards multimodal deep learning for activity recognition on mobile devices,” in *In Proc. Ubicomp*. ACM, 2016.

[101] L. Deng and D. Yu, “Deep learning: Methods and applications,” Tech. Rep. MSR-TR-2014-21, January 2014. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=209355

[102] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[103] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 6645–6649.

[104] A. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *JMLR*, vol. 15, no. 15, 2014.

[105] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, pp. e215–e220, 2000 (June 13), circulation Electronic Pages: http://circ.ahajournals.org/cgi/content/full/101/23/e215 PMID:1085218; doi: 10.1161/01.CIR.101.23.e215.

[106] "LG G Watch R," https://www.qualcomm.com/products/snapdragon/wearables/lg-g-watch-r, 2016.

[107] D. Lymberopoulos, J. Liu, X. Yang, R. R. Choudhury, S. Sen, and V. Handziski, "Microsoft indoor localization competition: Experiences and lessons learned," *GetMobile: Mobile Computing and Communications*, vol. 18, no. 4, pp. 24–31, Jan. 2015.

[108] J. Konecny, B. McMahan, and D. Ramage, "Federated optimization: Distributed optimization beyond the datacenter," in *arXiv:1511.03575*, 2015.

[109] S. A. Osia, A. Shahin Shamsabadi, A. Taheri, H. R. Rabiee, N. D. Lane, and H. Haddadi, "A hybrid deep learning architecture for privacy-preserving mobile analytics," in *arXiv:1703.02952*, 2017.