CONTEXT-DRIVEN IMPLICIT INTERACTIONS

DOCTORAL THESIS

CMU-HCII-19-103

Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

GIERAD LAPUT
Human-Computer Interaction Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213

THESIS COMMITTEE

Chris Harrison (Chair), Carnegie Mellon University
Scott Hudson, Carnegie Mellon University
Niki Kittur, Carnegie Mellon University
Shwetak Patel, University of Washington

To my parents, for taking the risk; you are an inspiration to all.

To my six siblings— you are the fount of all my joy and energy.

And to Liz, for tolerating my high energy, and for teaching me to focus on what truly matters.

TABLE OF CONTENTS

PART I: THE ARM AS A CONTEXT SENSING SPRINGBOARD

PART II: GENERAL-PURPOSE ENVIRONMENT SENSING

PART III: REDUCING USER BURDEN

ABSTRACT

As computing proliferates into everyday life, systems that understand people's context of use are of paramount importance. Regardless of whether the platform is a mobile device, a wearable, or embedded in the environment, context offers an implicit dimension that will become highly important if we are to power more human-centric experiences. Context-driven sensing will become a foundational element for many high-impact applications, from specific domains such as elder care, health monitoring, and empowering people with disabilities, to much broader areas such as smart infrastructures, robotics, and novel interactive experiences for consumers.

In this dissertation, I discuss the construction and evaluation of sensing technologies that can be practically deployed and yet still greatly enhance contextual awareness, primarily drawing upon machine learning to unlock a wide range of applications. I attack this problem area on two fronts: 1) supporting sensing expressiveness via context-sensitive wearable devices, and 2) achieving general-purpose sensing through sparse environment instrumentation. Finally, I explore how such sensing schemes can become more practical, by reducing user burden through data-driven approaches. I discuss algorithms and pipelines that extract meaningful signals and patterns from sensor data to enable high-level abstraction and interaction. I also discuss system and human-centric challenges, and I conclude with a vision of how rich contextual awareness can enable more powerful experiences across broader domains.

1. INTRODUCTION

When people converse, explicit (*e.g.,* speech, pointing) and implicit cues (*e.g.,* gaze, body language, facial expressions) play key roles in creating nuanced and expressive communication. We can draw similar parallels for interactive systems. Although computers have rich output capabilities (*e.g.,* screens, sound, haptics), they lack input richness. Most systems are driven by a limited set of explicit input modalities (*e.g.,* typing, touch, voice). For example, mobile and wearable computers feature small screens, and smart speakers only offer linear voice commands that result in suboptimal expressiveness. One of the main threads of my research developed new ways to expand explicit input richness across different modalities [145, 140, 141, 143, 144], making key contributions in this area.

Furthermore, systems lack *implicit* inputs and situational awareness to foster nuanced and assistive human-computer interactions. Most interactive systems have no implicit input channel, primarily due to lack of contextual sensing and understanding. For example, wearables and Internet-of-Things (IoT) devices are oblivious to their physical context, despite being right there, strapped on a user's arm or sitting on a kitchen countertop— a missed opportunity.

In this thesis, I refer to the implicit input as the ability for devices to know *what is happening around them.* But if devices know what is happening around them, why would this be important? As new platforms proliferate across different facets of our everyday lives, increasing implicit input bandwidth will become highly important if we are to power more human-centric experiences. Digitizing the physical environment through context-awareness has many high-impact applications, from specific domains such as elder care, health monitoring, and empowering people with disabilities, to much broader applications such as smart infrastructures, robotics, and novel interactive experiences for consumers.

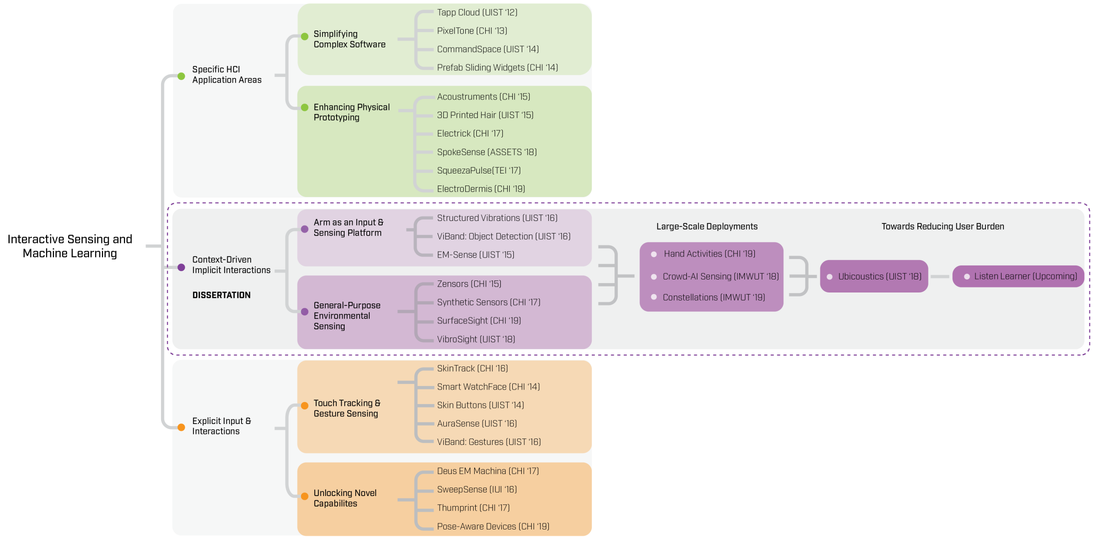

Interactive Sensing and Machine Learning
Specific HCI Application Areas
Simplifying Complex Software
Tapp Cloud (UIST '12)
PixelTone (CHI '13)
CommandSpace (UIST '14)
Prefab Sliding Widgets (CHI '14)
Enhancing Physical Prototyping
Acoustruments (CHI '15)
3D Printed Hair (UIST '15)
Electrick (CHI '17)
SpokeSense (ASSETS '18)
SqueezaPulse(TEI '17)
ElectroDermis (CHI '19)
Context-Driven Implicit Interactions
DISSERTATION
Arm as an Input & Sensing Platform
Structured Vibrations (UIST '16)
ViBand: Object Detection (UIST '16)
EM-Sense (UIST '15)
General-Purpose Environmental Sensing
Zensors (CHI '15)
Synthetic Sensors (CHI '17)
SurfaceSight (CHI '19)
VibroSight (UIST '18)
Large-Scale Deployments
Hand Activities (CHI '19)
Crowd-AI Sensing (IMWUT '18)
Constellations (IMWUT '19)
Towards Reducing User Burden
Ubicoustics (UIST '18)
Listen Learner (Upcoming)
Explicit Input & Interactions
Touch Tracking & Gesture Sensing
SkinTrack (CHI '16)
Smart WatchFace (CHI '14)
Skin Buttons (UIST '14)
AuraSense (UIST '16)
ViBand: Gestures (UIST '16)
Unlocking Novel Capabilites
Deus EM Machina (CHI '17)
SweepSense (IUI '16)
Thumprint (CHI '17)
Pose-Aware Devices (CHI '19)

Figure 1-1. An overview of the research avenues that I have pursued during my Ph.D, showing both breadth and depth. The middle track, context-driven implicit interactions, is the focus of this dissertation.

For example, a wearable that knows what the hands are doing could have many health-related applications, such as better understanding a user's typing behavior to prevent repetitive strain injury (RSI) [25]. Likewise, a smartwatch that can detect when a user coughs more frequently than normal could to track the onset of symptoms (*e.g.*, COPD [250], if monitored by a doctor), or nudge the user towards preventive measures such as alerting them to get a flu shot or offering to schedule a doctor's appointment. Eldercare monitoring systems [54, 211, 258] could also make use of this new and nuanced information source. Likewise, smart speakers like Amazon Alexa or Google Home being integrated into people's living spaces (*e.g.*, kitchen, living room, bathroom) could have more nuanced understanding of events in the environment. Contextual sensing can equip these platforms with unique interactive opportunities.

SENSING CHALLENGES

Achieving this vision requires the increasing mediation of *sensors*, which are crucial for creating a streamlined interface between physical and digital environments. However, several challenges exist in making this a reality.

Practicality

First, existing sensing approaches are costly, obtrusive, and special-purpose. Numerous approaches have been attempted and articulated, though none have reached widespread use to date. One option is for users to upgrade their environments with newly released "smart" devices (*e.g.,* light switches, kitchen appliances), many of which contain sensing functionality. However, this sensing is generally limited to the appliance itself (*e.g.,* a smart light switch knows if it is on or off) or when it serves its core function (*e.g.,* a thermostat sensing occupancy). Likewise, few smart devices are interoperable, thus forming silos of sensed data that thwarts a holistic experience. Instead of achieving a smart home, the best one can hope for—at least in the foreseeable future— are small islands of smartness. This approach also carries a significant upgrade cost, which so far has proven unpopular with consumers, who generally upgrade appliances piecemeal.

To sidestep this issue, we are now seeing aftermarket products (*e.g.*, [170, 190, 260]) and research systems (*e.g.*, [154, 256]) that allow users to distribute sensors around their environments to capture a variety of events and states. For example, Sen.se's Mother product [241] allows users to attach "universal" sensor tags to objects, from which basic states can be discerned and tracked over time (*e.g.*, a tag on a coffee machine can track how often coffee is made). This approach offers great flexibility, but at the cost of having to instrument every object of interest in an environment.

Fidelity, Semantics, and Privacy

A single room can have dozens of complex environmental facets worth sensing, ranging from "is the coffee brewed" to "is the tap dripping." A single home might have hundreds of such facets, and an office building could have thousands. The cost of hundreds of physical sensors is significant, not including the even greater cost of deployment and maintenance. Moreover, extensively instrumenting an environment in this fashion will almost certainly carry an aesthetic and social cost [29].

Equally challenging is that raw sensor output rarely matches human semantics. For example, an accelerometer provides coarse movement information, but rarely fine-grained human activity. Similarly, a door sensor may fall short in answering a user's true question: "*Are my children home from school?"* Unfortunately, these

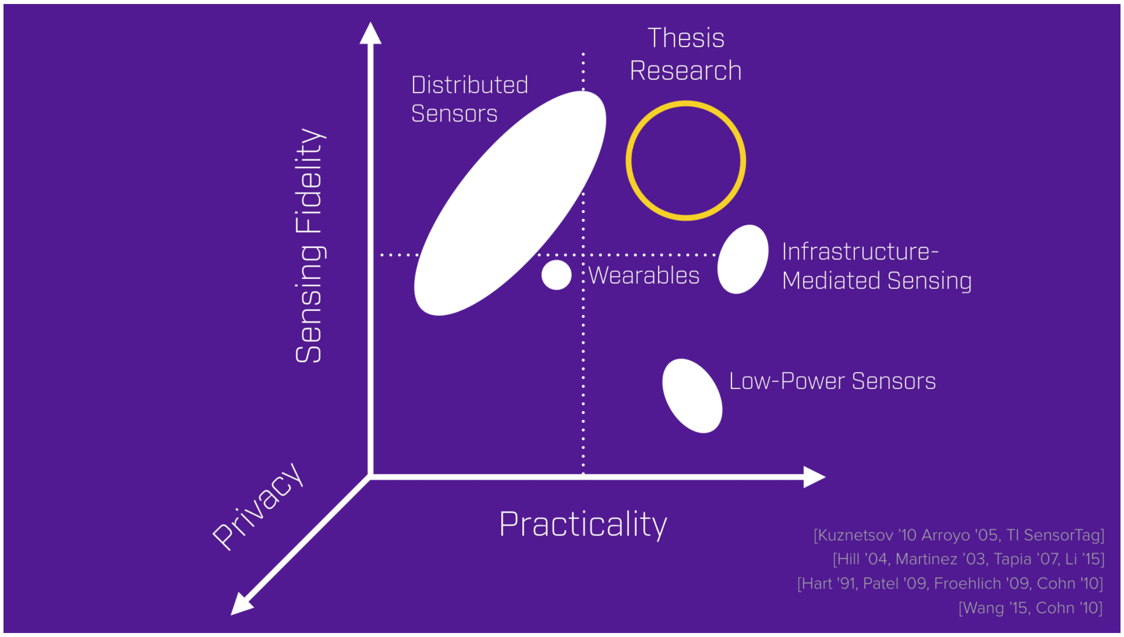


Figure 1-2. A taxonomy describing the different approaches for contextual sensing. They vary in their level of practicality and sensing fidelity. My thesis offers an approach that offers high sensing fidelity and high practicality (upper right), with privacy as part of the design from the beginning.

sophisticated, contextual, and multi-dimensional questions are not easily answered by the conventional sensors we deploy today. Systems that generalize across these broad contexts do not yet exist.

Finally, an ideal approach should also be designed with privacy in-mind from the very beginning. Researchers have long studied the inherent trade-off between sensing fidelity and user privacy [113]. For example, traditional computer vision (CV) activity recognition approaches affords higher resolution sensing, but the capture and transmission of live camera data, raises privacy concerns. An ideal sensing solution should provide a path where features are denatured (*i.e.,* no recoverable content) and data processing runs purely on-device.

CONTEXT-DRIVEN IMPLICIT INTERACTIONS

Therefore, the “holy grail” for achieving the promise of nuanced, highly ubiquitous, context-aware systems rests on exploring sensing modalities that exploit form factors that already exist (ideally popular) or do not require costly or invasive instrumentation (*i.e.*, “plug and play”). Such sensing systems should support “virtualization” of low-level raw signals into semantically-encoded, human-relevant representations, and they should be designed with privacy as a first-order primitive from the very beginning.

Motivated by this problem, my thesis focuses on enhancing context-awareness through ubiquitous and unobtrusive contextual sensing, primarily drawing upon machine learning to unlock a wide range of applications. I approach this problem space from two complementary perspectives: 1) supporting sensing expressiveness via *context-sensitive* wrist-worn devices, and 2) achieving *general-purpose* sensing through sparse environment instrumentation.

The Arm as a Sensing Springboard

Watches are unique among computing devices in that they are worn, offering great potential to transform arms and hands into expressive input and sensing platforms. For example, as people use their hands, tiny micro-vibrations propagate through the arm, carrying information about the objects they interact with and the activities they perform throughout the day. Smartwatches are ideally situated to capture these vibrations (see Chapter 5). In particular, people’s interactions with objects offer rich, contextual information closely reflecting one’s immediate activity (Chapter 4). Yet practical detection and recognition of object interactions remains an elusive research goal. For example, although RFIDs can provide object recognition capabilities, the technology requires all desired objects to be physically tagged and it is unknown if users are simply nearby or truly touching an object.

The first component of my thesis offers a set of techniques that could help make contextual sensing more practical, by taking advantage of the watch and its unique position in the body. But at the moment, smartwatches are nothing more than extensions of people’s phones. They sit right next to the user’s hands, but know nothing about what the hands are doing—a missed opportunity. As philosopher Immanuel Kant argued, *"The hand is the visible part of the brain."* Indeed, knowing what the hands are doing from the vantage point of a watch could serve to make computational experiences much more context-sensitive. This offers great potential for allowing a user’s arms and hands to be transformed into an expressive context-sensing platform— an *Arm v2.0.*

General-Purpose Sensing

In some cases, user instrumentation (via wearables) can create steep requirements that dampen adoption. For example, environments such as classrooms and cafés could benefit from context-aware sensing, but it is unrealistic to expect every user to wear a wrist-worn device. A complementary approach that I explore in my thesis is to *embed* sensors at key probe points within an environment. Substantial prior work exists in this area, which I have represented diagrammatically in Figure 1-2. Along the y-axis is the number of distinct sensed facets (*e.g.*, states and events), while the x-axis is the number of sensors needed to achieve this output.

A single room can have dozens of complex environmental facets worth sensing (Figure -11, y-axis), ranging from "*Is the coffee brewed?*" to "*Is the dishwasher done*?" A single home might have hundreds of such facets, and an office building could have thousands. The cost of hundreds of physical sensors is significant (Figure 1-2, top-left), not including the even greater cost of deployment and maintenance. Moreover, extensively instrumenting an environment in this fashion will almost certainly carry an aesthetic and social cost. Therefore, the ideal sensing approach occupies the top-right of Figure 1-2, wherein one sensor can enable many sensed facets, and more specifically, beyond any one single instrumented object. This *general-sensing* approach is challenging, as it must be inherently indirect to achieve this breadth.

A lightweight, general-purpose sensing approach could overcome many of these issues. Ideally, a handful of "super" sensors could blanket an entire environment – one per room or less. To be minimally obtrusive, these sensors should be capable of sensing environmental facets indirectly (*i.e.*, from afar) and be plug and play – forgoing batteries by using wall power, while still offering omniscience despite potential sub-optimal placement. Further, such a system should be able to answer questions of interest to users, abstracting raw sensor data (*e.g.*, z-axis acceleration) into actionable feeds, encapsulating human semantics (*e.g.*, a knock on the door), all while preserving occupant privacy.

DOCUMENT ORGANIZATION

In this dissertation, I introduce multiple enabling technologies for context-driven implicit sensing and interaction that offer high practicality, high-sensing fidelity, and are inherently privacy preserving. These systems have been deployed across long periods and multiple environments, the results of which show the versatility, accuracy and potential for practical ubiquitous context sensing. By combining novel sensing with machine learning, my work transforms raw signals into intelligent abstractions that can power rich, context-sensitive applications, unleashing the potential of next-generation computing platforms.

Part I of this dissertation will tackle projects on worn sensing (*EM-Sense, ViBand*), while Part II discusses projects on general-purpose sensing (*Zensors, Synthetic Sensors*). In Part III, I describe follow-up research that explores data-driven approaches for making the deployment of such systems even more practical, complementing the contributions outlined in Part I and Part II. Finally, I provide a of summary contributions and a section acknowledging the people and organizations that have made this dissertation possible.

2. BACKGROUND

In Mark Weiser's seminal *Scientific American* article on ubiquitous computing [286], he paints a compelling vision of profound technologies that "*weave themselves into everyday life until they are indistinguishable from it.*" His group at PARC prototyped and described classes of computing platforms (tabs, pads, and boards) that seamlessly adapt to user's needs and behavior that they could effectively "disappear" into the periphery. Key to this vision is the ability for devices to understand relevant aspects about the user and the environment. For example, "*if a computer merely knows what room it is in, it can adapt its behavior in significant ways without requiring even a hint of artificial intelligence.*"

Weiser's ubiquitous computing ("ubicomp") vision had three tenets: 1) cheap computing platforms, 2) a network that connects them together, and 3) systems implementing applications that can seamlessly adapt to user needs. Decades later, we have witnessed two of these requisite dimensions come to fruition. We now have capable, Internet-connected devices that fit in our pockets (smart phones), sit on our wrists (smart watches), or keep us company in our living spaces (smart assistants). However, these devices know very little about the user or the environment, even though they are situated right next to user's hands (*i.e.*, on a watch) or embedded within the user's physical space (*e.g.*, smart speakers). Two decades after Weiser's landmark article, Abowd characterized the state of "ubicomp" [3] as still lacking the tools and physical sensing capabilities to build applications that fully take advantage of these dynamic computing platforms.

Abowd's assessment is consistent with what researchers describe as critical human-computer interaction challenges in this space. Dey [60] distills three distinct problems: *input*, *understanding of input*, and *output*. In most cases, computers are excellent at providing feedback (output) through sounds and displays—communication channels where humans are well equipped to process and interpret. Thus, the more critical interaction problem lies in the other two areas: the sparse input vocabulary that humans provide to computers, and the computer's limited understanding of how to interpret that input. My thesis is situated between these two key problems.

More concretely, my work addresses what researchers in this space refer to as *implicit* input [4], or situational awareness. As I briefly mentioned in Chapter 1, when people talk to each other, implicit input or *context*—eye gazes, hand gestures, body language—helps ground the conversation [41] and increases communication bandwidth. Likewise, context increases the input bandwidth for human-computer interactions, especially in highly dynamic "ubicomp" platforms such as mobile devices, wearables, and smart environments.

DEFINING CONTEXT

In this thesis, I subscribe to the definition of context laid out by Dey's seminal work on context-awareness [60]. Although many definitions have been proposed (Schilit [236], Ryan [226], Pascoe [196]), Dey's definition encapsulates a more generalized notion of context, which includes the *who*, *where*, *when,* and *what* of relevant entities:

> *Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves [60].*

In this definition, context types can include *location*, *identity*, *time*, and *activity*. Acquiring location, time, and identity contexts has been an area of active research (see [4] for a survey), and most modern devices provide some level of support for these context types. My work makes a contribution in this space by proposing novel solutions that capture *activity* contexts, but in a more practical and unobtrusive manner.

CONTEXTUAL SENSING

The systems that I built are defined in the literature as *context-aware*— that is, *any system that uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task* [60]. Pascoe offers a useful taxonomy [196] that categorizes the different functions and capabilities that a context-aware system might have:

1) *Contextual sensing*— the ability to detect contextual information and present it to the user, augmenting the user's own sensory system (*e.g.*, GPS, illumination)
2) *Contextual adaptation*— the ability to exploit contextual knowledge and adapt seamlessly to the user's environment (*e.g.*, auto screen brightness reduction)
3) *Contextual resource discovery*— the ability to discover other resources that are of the same context as itself, and leverage those resources (*e.g.*, devices in the same location can reduce power consumption through task arbitration [56])
4) *Contextual augmentation*— the ability to use context to augment the environment (*e.g.*, tagging a place or object with data).

My thesis primarily focuses on enhancing *contextual sensing*, increasing the ability for devices to sense more nuanced contextual information about the user and the immediate environment. Further, the contributions of my work can be categorized as *application agnostic* [3], primarily driven by the inception of systems or techniques that advance core technical capabilities. As a result, my work is able to equip context-aware systems with contextual adaptation, contextual resource discovery, and contextual augmentation capabilities (which are largely application-centric).

SENSING APPROACHES

To more completely situate my thesis relative to prior activity-based, contextual-sensing systems, here I describe several illustrative research projects. These are categorized into two prevalent approaches—mobile & worn, and environment instrumentation. These categories fit with the "execution environments" described by Schilit, Adams, and Want [236].

Mobile & Worn

Mobile and wearable systems are unique in that they are always carried (or worn) by the user. Contextual sensing in these platforms capture contexts such as location, nearby people or objects, physical state of the user, or other relevant social situations.

Active Badge (Want et. al)

Many researchers refer to Active Badge [279] as arguably one of the first context-aware applications. Active Badge leverages user-worn IR transmitters that emit unique

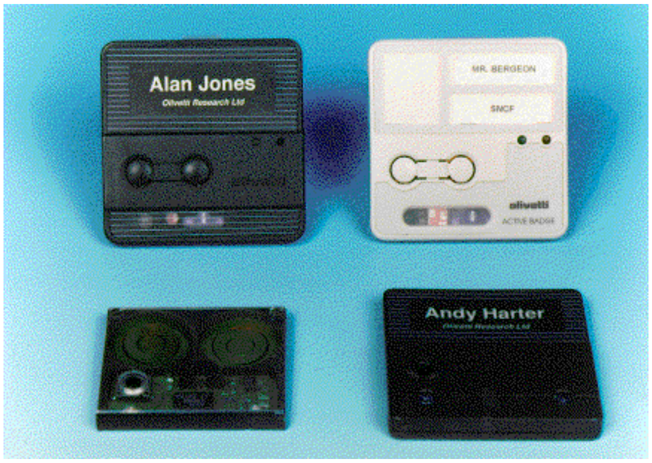


Figure 2-1. The Active Badge prototype, a pioneering context-aware system.

patterns, which can then be used to dynamically update user location. ActiveBadge then utilized location information to drive context-aware features such as contextual resource discovery and contextual adaptation. This simple yet clever prototype demonstrated one of the earliest instances of a location-based service, a feature that is now pervasive in modern phones.

The Mobile Sensing Platform (Choudhury et. al)

Whereas ActiveBadge enabled contextual sensing through a single channel (*i.e.*, IR), sensor fusion approaches bring a more nuanced understanding of context. The Mobile Sensing Platform (MSP) introduced by Choudhury and colleagues [40] brought an automatic activity recognition system via on-body sensors. This portfolio of sensors allowed the system to accurately track user activities such as walking, running, working on a computer, or using fitness equipment— an impressive set that unlocks a rich set of ubiquitous applications such as health monitoring and automatic fitness logging.

Body-Worn Activity Recognition (Ward et. al)

Researchers have also explored sensing activities beyond coarse locomotion, and towards finer-grained contexts such as object or tool use. These contexts could help provide a better understanding on what the user is doing, and could reveal the current state of her environment. Ward and colleagues [283] built on-body sensing platform (*i.e.*, using microphones and three-axis accelerometers mounted on the user's arms) for continuous recognition of workshop activities (sawing, hammering, filing, drilling,

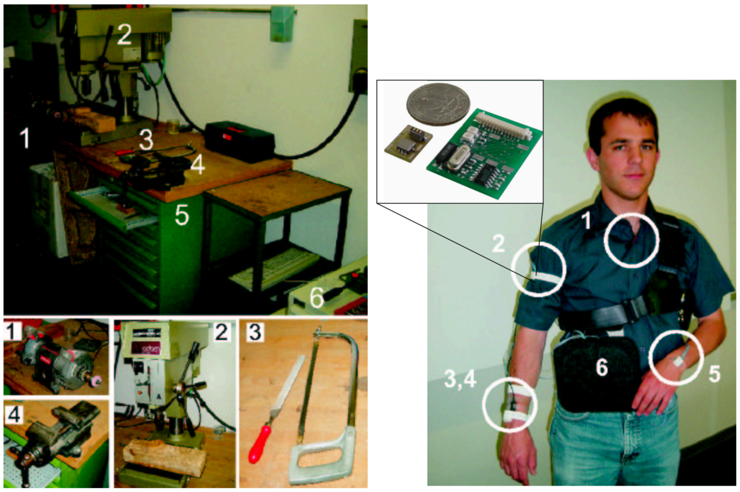


Figure 2-2. Ward's context-aware system, able to detect fine-grained activity context (*e.g.*, tool usage), with just a few, well-placed sensors on the body.

grinding, sanding, opening a drawer, tightening a vise, and turning a screwdriver). Ward's system showed that a few, well-placed sensors mounted on the body can offer surprisingly accurate results, echoing Weiser's view that "*no revolution in artificial intelligence—just the proper imbedding of computers into the everyday world.*"

Grasped Object Detection (Maekawa et. al)

Although Ward's context-ware system was impressive, its reliance on microphones and accelerometers meant it was susceptible to environmental noise and unintended hand motions. Therefore, the Holy Grail for enabling object activity recognition is to detect *when* an object is used, at the *moment* of touch, and knowing *what* that object is. Several researchers attempt to address this problem through clever use of sensors. For instance, Maekawa *et al.* [167, 168] utilized magnetic sensors and hand-worn coils to identify objects based on magnetic field changes, at the moment of touch. This is made possible since magnetic induction relies on proximate contact between objects and the sensing apparatus. My research on worn-based contextual sensing uses Maekawa's system as a baseline for understanding. Is it possible to reach the same level of sensing fidelity, yet with less instrumentation, more object types, and at better accuracies?

Environment Instrumentation

In some cases, user instrumentation (via wrist-worn devices) can create steep requirements that dampen adoption. For example, environments such as classrooms and cafés could benefit from context-aware sensing, but it is unrealistic to expect every user to wear a wrist-worn or mobile device. A complementary contextual sensing approach is to embed sensors at key probe points within an environment.

Special Purpose Sensing

The most intuitive and prevalent form of environmental sensing is to use a *single* sensor to monitor a *single* facet of an environment. For example, in UpStream [133] and WaterBot [13], a microphone is affixed to a faucet so that water consumption can be inferred (which in turn is used to power behavior-changing feedback). Similarly, efficient management of HVAC has been demonstrated through room-level temperature [129] and occupancy sensors [238]. Special-purpose sensors tend to be robust for well-defined, low-dimensional sensing problems, such as occupancy sensing and automatically opening doors. However, this relationship is inherently a one-sensor to one-sensed-facet relationship (*i.e.*, *one-to-one*; Figure 2-3, bottom left quadrant). For example, an occupancy sensor can only detect occupancy, and a door ajar sensor can

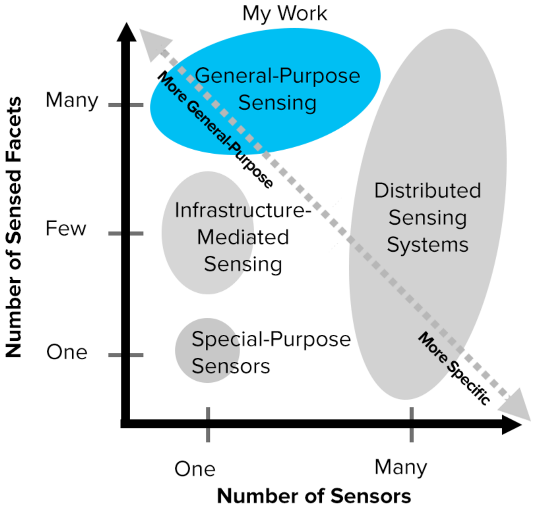


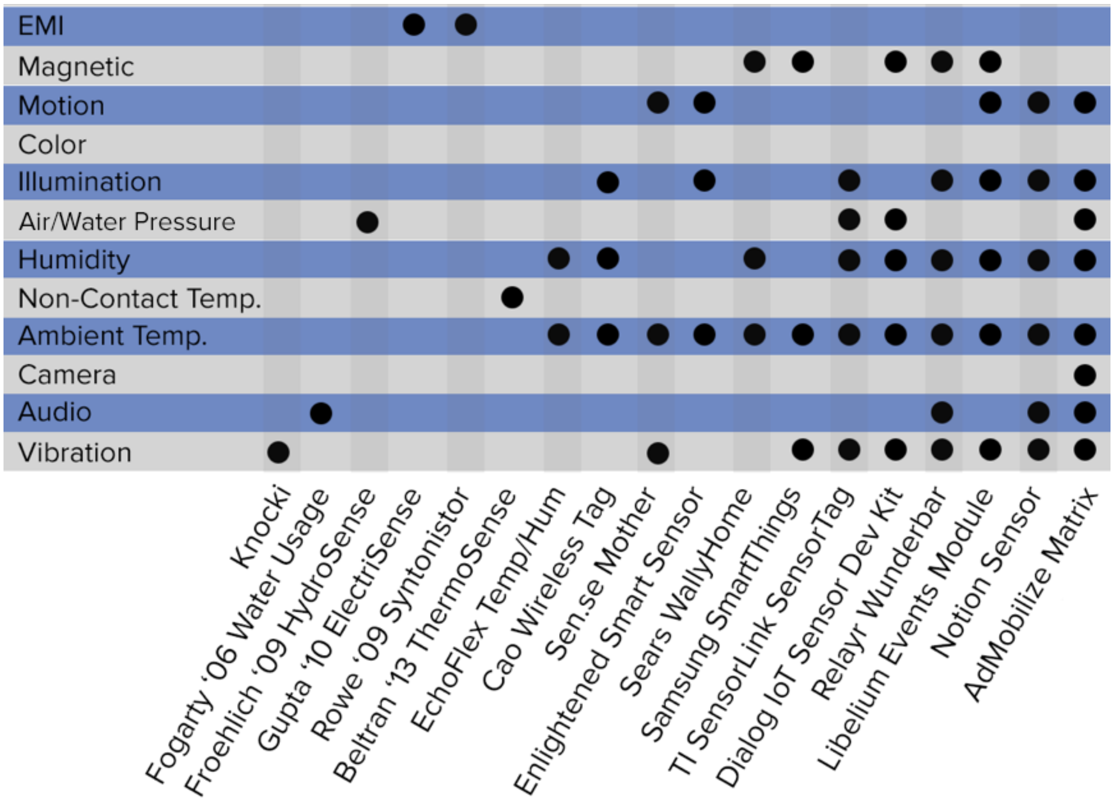
Figure 2-3. I created the Sensor Utility Taxonomy, which summarizes canonical approaches in environmental sensing.

only detect when a door is open. There is no notion of generality; each desired facet is monitored by a specific and independent sensor.

Distributed Sensing

It is also possible to deploy many sensors in an environment, which can be networked together, forming a *distributed sensing* system [107]. This approach can be used to enlarge the sensed area (*e.g.*, occupancy sensing across an entire warehouse) or increase sensing fidelity through complementary readings (*e.g.*, seismic events [22, 265]). The distributed sensors can be homogenous [73] (*e.g.*, an array of identical infrared occupancy sensors) or heterogeneous (*i.e.*, a mix of sensor types) [257, 290]. Also, the array can sense one facet (*e.g.*, fire detection) or many (*e.g.*, appliance use).

A home security system is a canonical example of a heterogeneous distributed system, where door sensors, window sensors, noise sensors, occupancy sensors and even



	Knocki	Fogarty '06 Water Usage	Froehlich '09 HydroSense	Gupta '10 ElectriSense	Rowe '09 Syntonistor	Beltran '13 ThermoSense	EchoFlex Temp/Hum	Cao Wireless Tag	Sen.se Mother	Enlightened Smart Sensor	Sears WallyHome	Samsung SmartThings	TI SensorLink SensorTag	Dialog IoT Sensor Dev Kit	Relayr Wunderbar	Libelium Events Module	Notion Sensor	AdMobilize Matrix
EMI				●	●													
Magnetic											●	●		●	●	●		
Motion									●	●						●	●	●
Color																		
Illumination								●		●			●		●	●	●	●
Air/Water Pressure			●										●	●				●
Humidity							●	●			●		●	●	●	●	●	●
Non-Contact Temp.						●												
Ambient Temp.							●	●	●	●	●	●	●	●	●	●	●	●
Camera																		●
Audio		●													●		●	●
Vibration	●								●			●	●	●	●	●	●	●

Table 2-1. An inventory of research and commercial approaches offering varying degrees of physical environment contextual-sensing.

cameras work together for a singular classification: is there an intruder in the home? This is a *many-to-one* scheme, and thus occupies the bottom right of our Figure 2-3 taxonomy. Conversely, for example, Tapia *et al.* [257] use a homogenous array of 77 magnetic sensors to detect object interactions throughout an entire house, and thus is a *many-to-many* scheme (upper right in Figure 2-3). Thus, distributed systems occupy the entire right side of our Figure 2-3 taxonomy.

A distributed sensing system, as one might expect, is highly dependent on the quality of its sensor distribution. Achieving the necessary sensor saturation often implies a sizable deployment, perhaps dozens of sensors for even a small context, like an office. This can be costly; with sensors often costing $30 or more, even small deployments can become unpalatable for consumers. Moreover, as the number of sensors grow, there is a danger of being invasive in sensitive contexts such as the home [28, 80, 113, 257].

Infrastructure Mediated Sensing

To reduce deployment cost and social intrusiveness, researchers have investigated the installation of sensors at strategic infrastructure probe points. For example, work by Abott [1], Hart [102, 101] and Gupta [93] used sensors coupled to a building's power lines to detect "events" caused by electrical appliances. Since home electrical lines are shared, a single sensor can observe activities across an entire house.

This *infrastructure-mediated sensing* approach has also been applied to *e.g.*, HVAC [197], plumbing [80, 85], natural gas lines [45] and electric lighting [92]. In all of these cases, a sensor was installed at a single probe point, enabling application scenarios that would otherwise require more costly distributed instrumentation of an environment. Although considerably more general purpose than the other approaches we have discussed, this approach is still constrained by the class of infrastructure it is coupled to. For example, a plumbing-attached sensor can detect sink, shower and toilet use, but not microwave use. Thus, we denote it as a *one-to-few* technique in our taxonomy.

General-Purpose Sensing

The *ideal* sensing approach occupies the top-left of our taxonomy, where *one* sensor can enable *many* sensed facets, and more specifically, beyond any one single instrumented object. This *one-to-many* property is challenging, as it must be inherently *indirect* to achieve this breadth. The ultimate embodiment of this approach would be a single, omniscient sensor capable of digitizing an entire building. The second part of my thesis examines this class of sensing approach, via a camera-based visual sensing platform, or through a portfolio of sensors packaged into a single wall-mounted board.

CONTEXT-DRIVEN IMPLICIT SENSING AND INTERACTIONS

Finally, in lieu of prior work, this dissertation discusses the construction and evaluation of sensing technologies that can be practically deployed and yet still greatly enhance contextual awareness, primarily drawing upon machine learning to unlock a wide range of applications. I approach this problem on two primary fronts: 1) supporting sensing expressiveness via *context-sensitive wearable devices*, and 2) achieving *general-purpose sensing* through sparse environment instrumentation. This thesis also explores how such sensing schemes can become more practical, by reducing user burden through data-driven approaches. I discuss algorithms and pipelines that extract meaningful signals and patterns from sensor data to enable high-level abstraction and interaction. I also discuss system and human-centric challenges, and I conclude with a vision of how rich contextual awareness can enable more powerful interactive experiences.

PART I:

THE ARM AS A CONTEXT-SENSING PLATFORM

3. WORN SENSING

"The hand is the visible part of the brain."
-Immanuel Kant

WEARABLES: A GATEWAY TO HUMAN ACTIVITIES

Critical to the success of smart environments is *implicit input*, that is, background knowledge about the user's context, including ongoing tasks such that we might wish to augment with computation and information. Human activity sensing has been an area of active research for several decades (see *e.g.*, [17, 33, 42, 79]). The advent of robust mobile sensing platforms (like the Intel Mobile Sensing Platform (MSP) [40]) and the ubiquity of smartphones has served to further accelerate research in this domain.

In just the past few years, wearables have emerged, affording researchers a beachhead on the body, offering improved fidelity and new sensed dimensions. Today, many consumer smartphones and smartwatches include built-in activity sensing capabilities

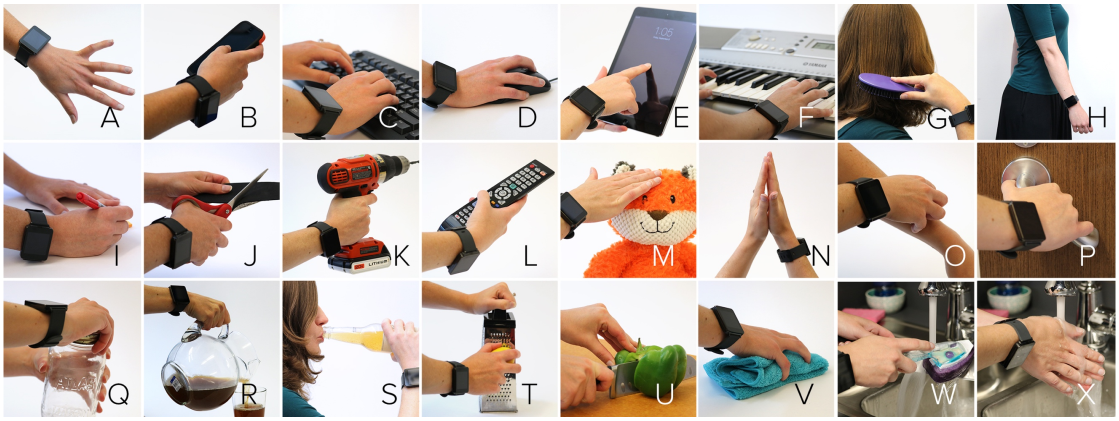


Figure 3-1. In this work, we explore the feasibility of sensing hand activities using commodity smartwatches, which are uniquely positioned to capture such fine grained activity.

that can distinguish between *e.g.*, walking, biking, driving and sleeping [79]. Most often, classified data is exposed to users for fitness and personal informatics applications, and thus is chiefly focuses on locomotion (or the lack thereof). However, this ignores a rich and expansive landscape of fine-grained human actions, especially those undertaken by the hands. I call these *hand activities*, and they are often independent of body activity. For example, one can *type on their smartphone* (hand activity) while *walking* (body activity); or take a *sip* of water from a bottle while *jogging*; or *flip through a book* while *lying* in bed. Wilson [291] offers an eloquent portrayal of these diverse uses:

> *"After tugging at the covers and sheets and rolling yourself into a more comfortable position, you realized that you really did have to get out of bed. Next came the circus routine of noisy bathroom antics: the twisting of faucet handles, opening and closing of cabinet and shower doors, putting the toilet seat back where it belongs. There were slippery things to play with: soap, brushes, tubes, and little jars with caps and lids to twist or flip open; […] Each morning begins with a ritual dash through our own private obstacle course – objects to be opened or closed, lifted or pushed, twisted or turned, pulled, twiddled, or tied. The hands move so ably over this terrain that we think nothing of the accomplishment. […] Our lives are so full of commonplace experience in which the hands are so skillfully and silently involved that we rarely consider how dependent upon them we actually are."*

If computing systems could know the activity of both the body *and* the hands, applications could be more context sensitive and assistive to immediate, ongoing tasks. State-of-the-art activity detection has been largely stuck at ambulatory states (walking, standing, sleeping, etc.) for decades. I envision smartwatches (slowly, but increasingly pervasive) as a unique beachhead on the body for capturing rich everyday actions. This could unlock many applications, ranging from personal informatics, health and skills assessment, and broadly, context-awareness. For example, a system that knows what your hands are doing can intelligently avoid interruptions. Hand activity detection can also be used to identify the onset of harmful patterns (e.g., repetitive strain injury or hand-arm vibration syndrome), or for building healthy habits (*e.g.*, regular brushing / hand washing).

RELATED SYSTEMS

The projects I tackle in this domain intersect with a range of HCI and sensing topics, most notably worn hand input, gesture sensing, and wearable-based object detection. Here, I discuss related systems to help position my thesis contributions.

Worn Hand Input and Gesture Recognition

Hand gestures offer expressive input modalities that complement existing interfaces and devices. A popular approach for hand gesture recognition takes advantage of optical sensors such as cameras [126] and IR sensors [87, 191, 192, 285]. It is also possible to sense hand gestures by approximating skin contours and deformations. For instance, armbands instrumented with IR sensors [191, 192] or pressure sensors [57, 122] can measure skin contact variations whenever particular gestures are performed. Despite being low-cost, these approaches are highly dependent on contact conditions, which are inherently sensitive to periodic armband removal, and equally susceptible to unintentional arm movements.

Hand gestures can likewise be modeled by examining the internal anatomical configuration of the user's arm. Approaches can be passive, such as electromyography [232, 233], where gestures are classified by measuring the electrical signals caused by muscle activation, or active [217], as in Touché [234] and Tomo [302], where a signal is injected into the body to detect hand gestures.

Finally, coarse and fine hand gestures indirectly induce arm motions which can be captured by inertial sensors *e.g.*, accelerometers and gyroscopes. Previous work introduced gloves equipped with accelerometers to model fine hand gestures. Likewise, several techniques take advantage of the inertial sensors present in contemporary smartwatches. Akl *et al.* [6] and Bernaerts *et al.* [23] utilize wearable accelerometers to recognize gross-motor or whole hand motions. Xu *et al.* [54] used inertial sensors attached to the arm, wrist, and finger to detect three types of gestures (arm, wrist, and finger movements, respectively), although their system was trained and tested on one user sitting in an armchair. Wen *et al.* [288] introduced finger gesture recognition using commodity accelerometers on a smartwatch, supporting a maximum of five gestures.

However, they caution that their technique is highly sensitive to arm orientation, and was never deployed in a real-time environment.

Object Recognition

Object recognition offers relevant information more closely matching a user's immediate context and environment [80, 164]. Most approaches rely on markers [32, 109, 136, 216, 219] or special-purpose tags [157, 305]. These offer robust recognition, but ultimately require every object to be instrumented. Further, these approaches approximate whether an object is nearby, and not when it is truly *grasped* or *handled.* Prior work has also leveraged acoustics to recognize objects [31, 43, 201]. For example, Ward *et al.* [282] built a worn necklace equipped with an accelerometer and a microphone to classify workshop tools, although the approach was susceptible to background noise.

Wearable devices are also increasingly being used for object sensing and recognition. Maekawa *et al.* [167, 168] utilized magnetic sensors and hand-worn coils to identify objects based on magnetic field changes. MagnifiSense [277] offered a similar approach, using three magneto-inductive sensors to identify objects during regular operation. Magnetic induction relies heavily on proximate contact between the sensor and the object, which is affected by posture, hand orientation, or even the inherent electromagnetic noise absorbed by the body. It is also possible to characteristically identify objects solely based on unintentionally emitted electromagnetic (EM) noise. EMISpy [303] detects different types of monitors and LCD screens by touching a screen surface while simultaneously holding a handheld EM sensor.

PART I: ROADMAP

In my thesis, I show that hand activity can be sensed robustly from a commodity, off-the-shelf smartwatch, without any external infrastructure or instrumentation of objects, opening a new and practical means for achieving practical ubiquitous context sensing. In addition to tracking coarse movement and orientation of the hand, the wrist is also the perfect vantage point to capture information produced as a byproduct of most hand activities (*e.g.*, typing, brushing teeth). These signals are inherently diverse, owing to user variance, innumerable tools and accessories, and differences in environment. To overcome these variations, I developed multiple systems that take advantage of a watch's unique vantage point. In this section, I provide an overview and a general roadmap of what the subsequent chapters will discuss. First, I discuss EM-Sense and ViBand in detail. I then discuss follow-up work that go beyond sensing gestures and object detection— *hand activities*— opening even more possibilities for responsive and context-sensitive applications.

Chapter 4: EM-Sense

In EM-Sense [148], I built a novel sensing approach that exploits the unintentional electromagnetic (EM) noise emitted by many everyday electrical and electromechanical objects, such as kitchen appliances, computing devices, power tools and automobiles. These signals tend to be highly characteristic to these objects owing to unique internal operations (*e.g.*, brushless motors, capacitive touchscreens) and different enclosure designs, material composition, and shielding. When a user makes physical contact with these objects, EM signals propagate through the user's body, as it is conductive. EM-Sense detects and classifies these signals in real time, enabling robust, on-touch object detection from a smartwatch form factor.

By learning EM "signatures", EM-Sense can discriminate between scores of objects, independent of wearer, time, and environment. In one of several user studies, EM-Sense achieved an average accuracy of 96.1% across nine objects, 12 users, two locations, and using data trained on a single independent user collected six weeks prior. This capability could power many context-based applications, ranging from passive monitoring of wearer's daily routines, to proactive suggestions (*e.g.*, recipe and routes) based on user-object interactions.

Chapter 5: ViBand

In addition to electromagnetic signals, device use can also be inferred from micro-vibrations transmitted into the arm. As people use their hands, tiny micro-vibrations propagate into and through the arm, carrying information about the objects they interact with and the activities they perform throughout the day. Smartwatches are ideally situated to capture these vibrations. Unfortunately, typical smartwatch accelerometers operate at around 200 Hz, which is sufficient for detecting screen orientation, step counting, and other coarse-grained interactions. By modifying the Android kernel, I was able to put a LG G smartwatch accelerometer into a rarely-used, high-speed sample mode: 4000 Hz. At this high sample rate, ViBand [147] can capture small compressive waves (*i.e.*, bio-acoustics) propagating through the user's arm. Unlike microphones, bio-acoustics are physically coupled to the body, making this technique naturally resistant to external noise.

Sensing micro-vibrations is only a preliminary step; the subsequent signal processing and machine learning capabilities are what allow this technique to unlock a wide range of applications. For example, ViBand can classify hand gestures such as flicks, claps, scratches, and taps, which combine with on-device motion tracking to create a wide range of expressive hand activities. Similar to EM-Sense, ViBand can be used to identify grasped objects for automatically launching context-relevant functionality or applications. Specifically, when a user operates a mechanical or motor-powered device, the object produces characteristic vibrations that transfer into the operator. My bio-acoustically-enhanced smartwatch can capture and classify these signals, allowing applications to better understand user context. In our study deployment, across 27 objects, 17 users, and using data that was trained on a single person four weeks prior, we obtained an overall object detection accuracy of 94.0%.

Chapter 6: Fine-Grained Hand Activities

Prior work has focused on detecting whole-body locomotion, such as walking, running and bicycling. However, capturing fine-grained hand activities could make computational experiences more powerful. Building on top of ViBand, I explore the feasibility of sensing fine-grained hand activities from a commodity smartwatch. As it turns out, important spatial-temporal relationships are encoded in the accelerometer’s three axes. For instance, when wiping a table, the Z-axis is mostly unperturbed (chiefly bio-acoustic noise resulting from friction, but little coarse motion), while the X and Y channels experience low frequency oscillations and the hand slides around the surface, often in a linear rubbing or circling motion. My results reveal that this approach can successfully support 25 fine-grained hand activities.

4. EM-SENSE: TOUCH DETECTION OF ELECTRICAL AND ELECTRO-MECHANICAL OBJECTS

INTRODUCTION

For years, intelligent systems have promised to improve people's lives by inferring context and activities in diverse environments. In particular, people's interactions with objects offer rich, contextual information closely reflecting one's immediate activity. Yet practical detection and recognition of object interactions remains an elusive research goal. For example, although RFIDs can provide object recognition capabilities, the technology requires all desired objects to be physically tagged and it is unknown if users are simply nearby or truly touching an object.

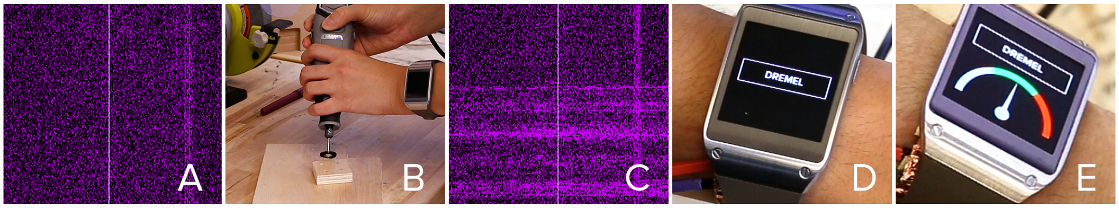


Figure 4-1. Spectrogram of ambient electromagnetic noise (A). When a user operates an electro-mechanical object, such as a Dremel (B), it emits EM noise (C), which we classify (D) and use to enable rich contextual applications (E).

I propose a novel sensing approach for object detection, triggered only when objects are *physically touched*. My approach exploits unintentional EM noise emitted by many everyday electrical and electromechanical objects, such as kitchen appliances, computing devices, power tools and automobiles. These signals tend to be highly characteristic (Figure 4-1), owing to unique internal operations (*e.g.,* brushless motors, capacitive touchscreens) and different enclosure designs, material composition and shielding. When a user makes physical contact with these objects, electrical signals propagate through the user's body, as it is conductive. By modifying a commodity software-defined radio receiver, I can detect and classify these signals in real time, enabling robust, on-touch object detection.

My approach, which I call *EM-Sense*, utilizes low-cost, commodity hardware and is small enough to be worn on the wrist or, in the near future, integrated into smartwatches. I draw inspiration from the sensing principles introduced in Humantenna [46], and move beyond environment localization and gesture recognition, to focus instead on context and activity sensing made possible through object interaction detection. Unlike existing approaches requiring object instrumentation (RFIDs, barcodes, BLE, etc.), EM-Sense can identify objects solely on their EM signatures.

BACKGROUND

Electronic devices, especially those driven by motors (*e.g.*, power drills) or switching power supplies (*e.g.*, LCD screens), produce significant levels of electromagnetic noise. These unwanted signals propagate as radio frequency (RF) waves and can disrupt nearby devices operating with similar frequency bands. Beginning in the late 1970s, the US Federal Communications Commission (FCC) chartered mandates to regulate the susceptibility of consumer devices to EM noise [193, 261]. These were also established to prevent EM noise from interfering with other electronics, utilities, and purposeful broadcasts, such as TV and radio.

Infrastructures and environments also produce EM noise. For example, AC electricity and devices connected to the power line contribute to majority of electrical noise at home. In general, EM noise propagates through conduction over circuits and power lines (1kHz - 30MHz) or through radiation in free space (30MHz to 10GHz).

Additionally, a few classes of non-electromechanical objects can have unique EM signatures. Most notable among these are large, metallic objects, such as structural members in buildings, doors, ladders, furniture, and window framework. These are sufficiently large that they act as antennas, capturing energy radiated by proximate noisy devices and wiring. The amalgamation of these signals, in our experiences, is fairly unique and particular to a location.

RELATED SYSTEMS

EM-Sense intersects with several bodies of research, including activity and object recognition, visual, acoustic, and RF-based approaches, and EMI-based sensing systems.

Handled Object Recognition

Traditional activity recognition systems infer user state based on temporal data of physical movement (*e.g.*, accelerometers). These require individuals to wear sensors or have a smartphone continuously monitoring data. Extensive prior work [94, 209] has demonstrated promising results for determining *e.g.*, running, walking and sitting. However, motion-driven approaches by themselves lack context to infer higher-level activities. For this reason, I pursue a complimentary approach that recognizes *handled objects*. This provides relevant information more closely reflecting a user's immediate environment and activity. Many approaches have been considered for object recognition, though most methods require objects to be instrumented with some form of marker or sensor [210, 247]. These can provide robust recognition, but as there are many objects in the world, installation and maintenance is troublesome and costly.

Recent work from Maekawa and colleagues cleverly utilized magnetic sensors [167] and hand-worn coils [168] to detect objects based on temporal changes in the magnetic field during an object's operation. Although related, magnetic induction relies on proximate contact between objects and the sensing apparatus, which means object detection is strongly affected by hand posture and inherent magnetic noise in the body, or even diamagnetic properties of hands and fingers. Conversely, as we will show, our approach is robust across users, time and hand/body posture.

Visual, Acoustic, and RF-Based Approaches

Early research into visual markers used 1D barcodes, and more recently, fiducial markers as unique identifiers. Further, there is considerable work in the computer vision domain for object recognition in natural scenes without artificial markers [48], as well as efforts that leverage crowd workers [150, 151]. These schemes require cameras, line of sight, and suitable lighting conditions.

Acoustic-based object recognition has also been explored extensively [31, 43]. For example, Acoustic Barcodes [100] described tags with sound-producing physical notches that resolve to a binary ID. More related to my approach are acoustic methods that attempt to recognize objects from vibro-acoustic information generated by operation of a device. For example, Ward *et al.* [282] used worn accelerometers and microphones to classify workshop tools.

Also popular are RFID-based approaches. Example systems include a wrist-worn, near-field RFID reading system that could identify objects affixed with tiny RFID tags[157, 205]. Similarly, Buettner *et al.* [211] used the Wireless Identification and Sensing Platform (WISP), which is a battery-free, long range RFID tag enhanced with an accelerometer to detect movement of a tagged object. Other object recognition efforts exist that use Wi-Fi [278], NFCs [90], BLE [66], and body-area networks [181].

EMI-Based Sensing

There are two main classes of EM-based sensing techniques: 1) infrastructure-mediated sensing and 2) using the human body as an antenna. The former instruments the infrastructure, while the second instruments the user.

Infrastructure-Mediated Sensing. Early work by Abott [1] and Hart [102, 101] in the 1980s used metering devices attached to a building's electrical lines to detect "events" caused by home appliances. Because the electrical lines in a house are shared infrastructure, a single sensor can observe activity across the entire home. These pioneering efforts inspired infrastructure-mediated sensing (IMS), *i.e.*, attaching probes to a variety of utility infrastructures, including HVACs [197], plumbing [85], natural gas lines [47], lighting [47] and electrical wiring [49, 92, 127, 198].

Using the Human Body as an Antenna. Because the human body is conductive, it has electrical properties that allow it to behave much like an antenna. Pioneering work in HCI has exploited this "body antenna effect." For example, in DiamondTouch [64], the human body is used as an electrical conductor, which allows the system to differentiate

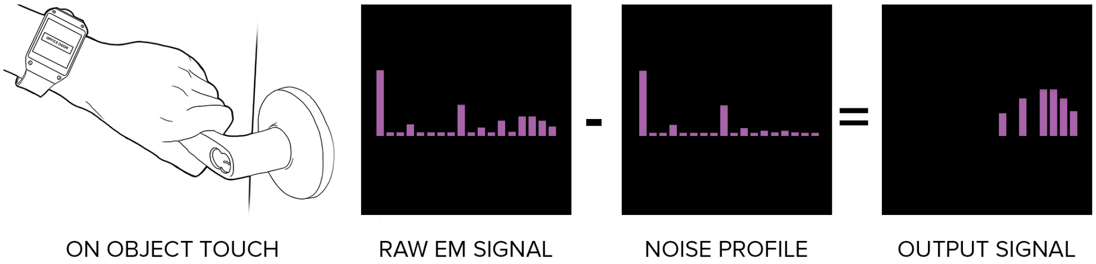


Figure 4-2. Spectrogram of ambient electromagnetic noise (A). When a user operates an electro-mechanical object, such as a Dremel (B), it emits EM noise (C), which we classify (D) and use to enable rich contextual applications (E).

touches between users. More recently, in “Your Noise is My Command,” Cohn *et al.* [48] utilize the human body as an antenna for detecting EMI signals in the home. A small electrode is attached behind the neck of the user and connected to a backpack-bounded A/D converter. As the user moves around the home, the system captures all recorded EM noise received by the human antenna. With this setup, they inferred user location within a home, as well as detect different gestures and continuous touch tracking along a wall. A later extension enabled free-space, whole body gestures by utilizing EM Doppler shifts [46]. Unlike infrastructure-mediated sensing, body-antenna EM sensing requires no instrumentation of the environment.

IMPLEMENTATION

For my proof-of-concept hardware implementation (Figure 4-4), I modified a software-defined radio receiver (RTL-SDR) to function as an inexpensive, but extremely fast A/D converter (Figure 4-3). Originally, RTL-SDRs use a tuning chipset to listen for frequencies covering FM bands and beyond (25 – 1766 MHz). However, useful EM emissions for most objects fall well below this operating range. To address this limitation, I modified the RTL-SDR’s circuitry by bypassing its tuner and routing raw signals directly into the chipset’s main A/D converter. It has an 8-bit resolution with ~2Vpp.

As a result, this modification re-adjusts the device’s sensing range to 1Hz – 28.8MHz, making it possible to detect low-band EM signals present in many electrical and

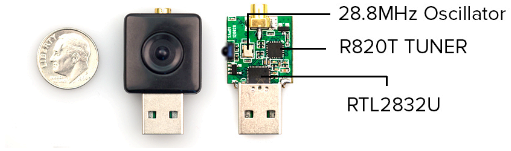


Figure 4-3. Software-defined radio (SDR) receiver used for the EM-Sense prototype. 8-bit resolution, 2Vpp

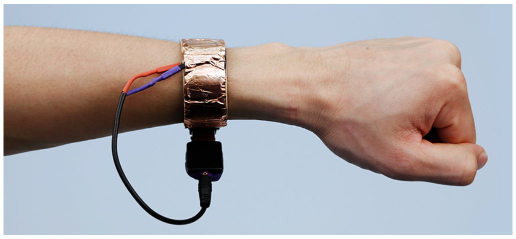


Figure 4-4. Wrist-worn prototype. Body-coupled EM signals captured through a modified SDR receiver.

electromechanical objects. Figure 4-3 details simple steps describing my modifications, which researchers and hobbyists can replicate. My sensing setup costs under $10, two orders of magnitude cheaper than previous EM sensing approaches.

To make the prototype wearable, I retrofitted the interior of an armband with copper tape, and connected it to the RTL-SDR's antenna terminal. Data received from the RTL-SDR is further processed through a software pipeline. First, I read from the RTL-SDR's input channel through a physical USB connection. At the time of research, no smartwatch on the market was capable of hosting a USB-OTG interface. In response, I offload USB-reading to a smartphone (Nexus 5), which is clipped to waist of the wearer, and uses an open source RTL-SDR driver (osmocom.org) we ported. In addition to reading the SDR, my smartwatch software also streams incoming data to a laptop computer over Wi-Fi, which in turn performs signal processing and live classification. With this setup, I can wirelessly stream data sampled at 1MHz with minimal packet loss. All of the physical components of EM-Sense fit into a wrist-worn apparatus, which could be easily integrated into future smartwatches.

Sensing Raw EM Signals

Whenever a user makes physical contact with an electrical or electromechanical object, its EM signal propagates through the body and is sensed by a conducting electrode worn on the user's wrist (Figures 4-2 and 4-4). Connected to the electrode is our modified software-defined radio, which converts this analog signal into digital data. I sample incoming signals at 1MHz; thus the theoretical Nyquist limit is 500kHz.

Note that neighboring objects and signals (*e.g.,* objects proximate to the user, but not in direct physical contact) can introduce interference through capacitive coupling and the body antenna effect. However, these signals are weaker compared to those transmitted by actual physical contact, and do not appear to affect detection.

Baseband Shift

To extend the effective bandwidth, I shift the SDR receiver's baseband frequency to 500kHz. Without shifting, the frequency spectrum is symmetric because it is a real signal. In this mode, the effective bandwidth for a signal sampled at 1Ms/s is -0.5MHz to 0.5MHz (*i.e.*, see Fig 7 raw FFT, where left half is redundant). Shifting to 500kHz moves the bandpass sampling window from 0 to 1MHz (0.5MHz and above will be undersampled, but still useful). As a result, the left-shifted spectrum contains no redundant information. I then apply a fast Fourier Transform (with an FFT size of 16384 bins *i.e.*, 61 Hz per band), and the resulting frequency domain values become the primary input for my sensing and classification pipeline.

Environmental Noise Rejection

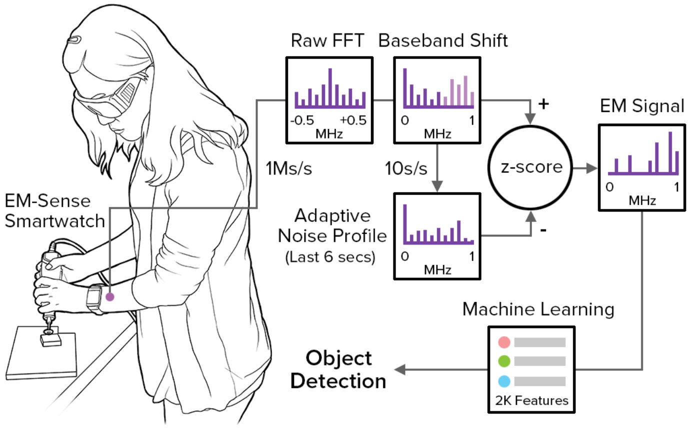


Figure 4-5. EM signals are captured at 1M samples per second, and baseband shifted to 500kHz. Next, it is compared against an adaptive noise profile using Z-score thresholding. The resulting signal is used object classification

To enable robust object detection, my sensing approach must differentiate between environmental EM *noise* and EM *signals* from objects. In addition to differences in amplitude (touched objects generally transmit more signal), I also take advantage of the fact that environmental EM noise tends to change at a slower rate, while EM signals change rapidly at the moment an object is touched or released (or the object is turned on/off). These events appear as high delta, “spiking” events in the signal.

I build a model of environmental EM noise using an adaptive background subtraction approach: an average frequency spectrum derived from a six-second rolling window, updated every 100ms (Figures 4-1 and 4-5). This provides a baseline “noise profile” from which I can subtract the live signal, amplifying transitions in touch state. In this particular implementation, if an object is held for a few seconds, its EM signal is integrated into the noise profile. The release of an object thus generates a large negative change, interpreted as a “touch up” event and signifying the object is no longer held.

Object Signal Extraction

To extract EM signals generated from object on-touch events, I perform real-time statistical analysis between the modeled noise profile and all incoming EM readings. I compute a baseline *threshold signal* based on the statistical *Z-score* of the individual frequency bands in the noise profile. Essentially, frequency bands whose values are above a specified *Z-score* (*e.g.*, 3.5 standard deviations above the noise profile) are amplified, while frequencies below the threshold are set to zero. A frequency band at index **n** of the extracted EM signal, ***S***, can be characterized as:

$$S_n = A \times \max(0, F_n - (G_n + z\sigma_n))$$

where ***F*** is the incoming EM reading, ***G*** is the noise profile, **σ** holds the standard deviations for each frequency band at index **n**, ***A*** denotes the amplification factor, and **z** is a constant that denotes a statistical z-score parameter. I use an amplification factor of 18 and a *z-score* of +3.5 (upper 0.1% of a normal distribution curve).

Live Object Classification

Once an object’s EM signature is decoupled from environmental noise (Figure 4-5), I use it as input for live object classification. First, I downsample the EM signature’s

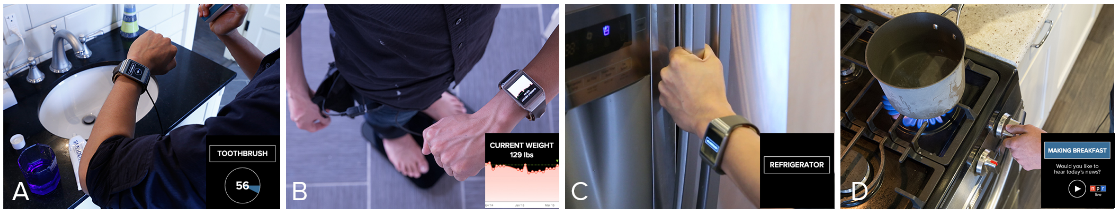


Figure 4-6. EM-Sense can augment activities in the home. For example, EM-Sense can launch a timer (inset) when the user is brushing his teeth (A), or display the user's data when stepping on a scale (B). Next, EM-Sense knows that the user is making breakfast and fetches the morning news (C and D).

FFT into 512 frequency bands. From this, I generate ~2K additional features based on: 1st and 2nd Derivatives (1021), min index, max index, RMS, center of mass, standard deviation, area under the curve, pair-wise band ratios (496), spectral kurtosis, crest factor, and 2nd order FFT (512).

These features are fed into a SMO-trained Support Vector Machine ($c=1.0$, $\epsilon=1^{-12}$, poly kernel) provided by the Weka Toolkit [19]. Feature selection analysis revealed that derivatives, band ratios, 2nd order FFTs, and max index serve as the important distinguishing features (providing 80% merit), but the remaining features nonetheless are important to fully capturing nuanced signal behaviors. Other machine learning techniques could potentially allow EM-Sense to scale to larger collections of objects. Object classification can be treated as an "information retrieval" problem, which means that techniques such as clustering, similarity metrics, and other methods are applicable.

EM-SENSE INTERACTIONS

To demonstrate how EM-Sense can augment activities across a wide range of contexts and environments, I built six interaction categories, which I describe briefly. These include five categories of objects: *home, office, workshop*, *fixed structures*, and *transportation* – a taxonomy we employ in our subsequent evaluation.

Object-Specific Applications. When a user handles objects with known EM signatures (Figure 4-7), EM-Sense can launch object-specific applications. For example, our electric toothbrush example launched a timer application.

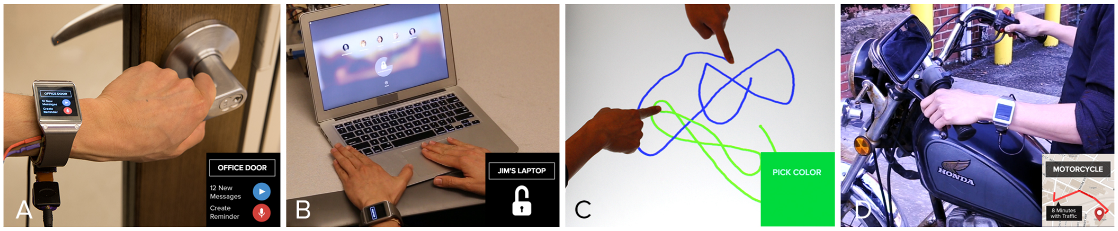


Figure 4-7. In the office, the use cases for EM-Sense are diverse. EM-Sense can be used for context-based communication (A), and for seamless authentication (B). In collaborative tasks, users with EM-Sense-capable devices enable user differentiation on touchscreens. When EM-Sense detects that the user is riding a motorcycle on the way home, a map is displayed (D).

Object Sequence Applications. It is also possible to launch applications based on sequences and patterns of object events (Figure 4-8). Combined with other readily accessible features, such as time of day and rough geospatial location, activity and context recognition is possible. For example, a pattern of activation in the kitchen suggesting dinner preparation can launch music, recipe, and other applications.

Object State Recognition. I can further extend object-specific applications by utilizing changes in an object's EM signature in different operational modes (Figure 4-1). We demonstrated this in our Dremel application depicting a "speedometer".

Authentication. A smartwatch with EM-Sense could allow users to authenticate across devices and applications, potentially without passwords. For example, to log in into a laptop, a user can simply touch the trackpad (Figure 4-7B). Because the watch knows that a trackpad is being touched, and the trackpad knows that it is being touched, a handshake mediated by the cloud could proceed (using *e.g.*, temporal co-occurrence). For added security, a confirmation button can be displayed on the owner's smartwatch.

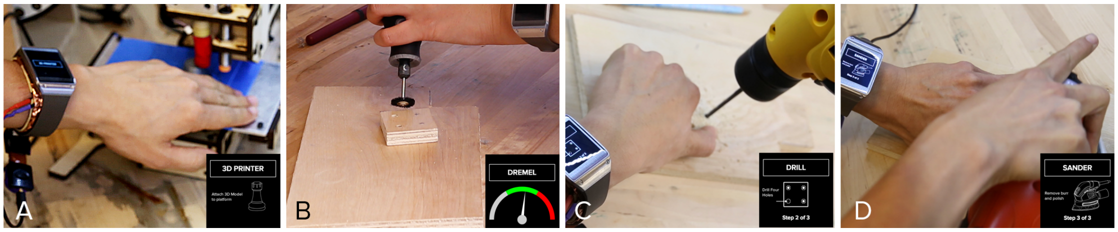


Figure 4-8. In the workshop, EM-Sense can assist in fabrication activities. Here, a tutorial is displayed on the watch. When the user operates a tool, the tutorial is automatically advanced (A, C, D). For some tools, EM-Sense can detect the operational state, such as the speed of the Dremel (B).

User Differentiation. Similar to the authentication interaction above, knowledge of touchscreen events provided by EM-Sense could be used to differentiate users in groupware applications (Figure 4-7C), which have many uses (see *e.g.*, [64, 110]). Specifically, a wearer's smartwatch knows the time of touch contact, which can be paired (e.g., in the cloud) to a touch event registered on a shared screen. Because the smartwatch knows its owner, touches can be attributed and parameterized to a specific user – in our example day, we used the watch display for a personalized color selector.

Object-Tagged Messaging. Knowledge of which objects are being handled also enables tagging of items with media (Figure 4-7A), such as text and voice messages. In our example day, Julia leaves a message for herself by tagging her office's door handle. By using names, it would also be possible to leave messages for particular people.

EVALUATION

I ran multiple studies evaluating several facets of EM-Sense. These studies serve several purposes: 1) to evaluate the accuracy and robustness of our sensing approach across different users, 2) to observe the longitudinal consistency of object EM signatures over time, and 3) to form a baseline understanding of the uniqueness of EM signatures across a wide range of objects. I also conducted several smaller supporting studies, which explore other important aspects of EM-Sense, including signal similarity across similar devices and object state recognition. Overall, EM-Sense enables an expansive range of applications (see Applications section), and in my evaluation, I endeavored to select objects reflecting diverse contexts and environments.

Accuracy and Longitudinal Consistency

This study aims to evaluate the sensing accuracy of EM-Sense across different users and determine whether object EM signatures are consistent over time. Because my sensing technique relies on the conductivity of the human body, my approach can be sensitive to differences in anatomy. Thus, I recruited 12 adult participants (5 female, age range 22 – 40, 1 left-handed), encompassing different statures and body types (mean height = 67 in., mean weight = 148 lbs., BMI range 20 – 28).

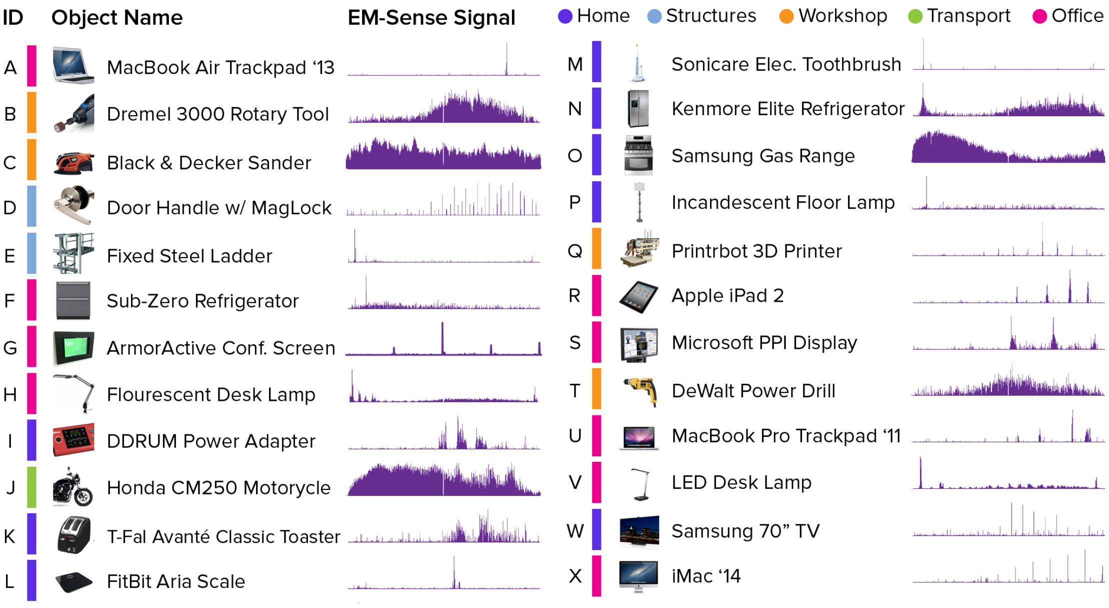


Figure 4-9. EM spectrums (0Hz - 1MHz) for the evaluation, captured by the

To further test sensing robustness, each study session was split across two different buildings, and we used data collected from *a single* user *six weeks* prior to the user study (*i.e.*, no per user training or calibration). Nine objects were evaluated, dispersed across our two locations: MacBook Air trackpad, mouse sander, door handle with an electromagnetic lock, fixed steel ladder, refrigerator, ArmorActive conference room touchscreen affixed to a wall, fluorescent desk lamp, power adapter, and a Dremel rotary tool.

For logistical convenience, all experiments started in the first location. Participants were asked to wear our prototype on their preferred arm (anecdotally, we noticed participants preferred to wear the device on their non-dominant arm, as is the norm for watches). For each trial, an experimenter announced an object name (e.g., "Dremel"), and participants were asked to touch, grasp, or operate the object. The experimenter recorded the *real-time* prediction made by EM-Sense. Objects were requested in a random order, appearing five times each in total. Participants were free to interact with objects with either or both hands. Each session took approximately 45 minutes to complete, and participants were paid $15 for their time.

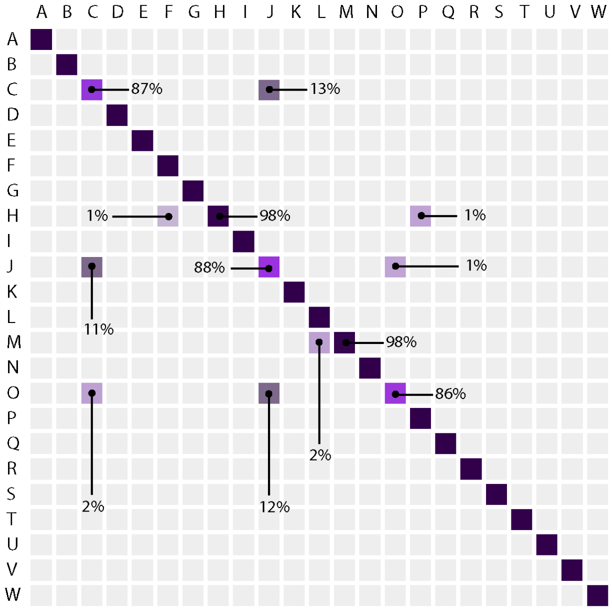


Figure 4-10. Object confusion matrix. Accuracy is 100%, unless indicated otherwise. Across 24 classes (including null class), average prediction accuracy was 97.9%. Figure 4-9 provides a key to the letters used on the axes.

Across nine objects, 12 users, two locations, and using data trained on one user collected six weeks prior, EM-Sense achieved an average overall accuracy of 96.1% (see Figure 4-10, STDEV=4.9%, chance 11%). This result is promising given the strict constraints imposed on the training data. Some objects achieved an accuracy of 100% (lowest is 85%, Sub-Zero Refrigerator). While not the focus of the study, I can report anecdotally that signal magnitudes appear stronger when the prototype is worn on the same arm as the hand touching the object (consistent with prior findings [48]). Overall, these results indicate that sensing is accurate and robust across different users and that object EM signatures are consistent over time.

Signal Uniqueness Across Objects

To more fully explore the uniqueness of EM Signatures across many objects, I ran a second study that collected data from 23 objects across four locations. This set was composed of our initial nine objects, plus fourteen new objects that spanned a wider range of contexts and environments, including the home, office, workshop, large structural features, and transportation (see Figure 4-7D). I also included *similar* objects (same category, but different models) to see if this caused classification confusion. Specifically, I include two refrigerators, two identical Apple laptops (same model), three lamps, and four devices where the LCD display is touched (ArmorActive conference room touchscreen, iPad, Samsung TV, and Microsoft PPI display).

Due to the large number of objects and locations, I performed an offline analysis. Three rounds of data were collected for each object, with at least 15 minutes between rounds. I utilized the first two rounds of data for training and the third and final round for testing. This procedure prevents the model from over-fitting on time-adjacent instances (*e.g.*, inherent similarities in touches when performed back-to-back). For each object, a round consisted of collecting 250 instances with various hand poses to aid classifier generality. In total, I collected 17,250 data points (23 objects x 3 rounds x 250 instances). I also added a null, "no object touched" class, increasing the set to 24. I then trained a single SVM model using the aforementioned features and parameters.

Across these 24 classes, the system achieved an overall accuracy of 97.9% (STDEV=4.3%, chance 4%), which suggests object EM signatures are reasonably unique and discriminable (see Figure 4-9). Note that the majority of objects (18 of the 24) reach an accuracy of 100%, while the lowest object accuracy is at 85.6% (Samsung Gas Range). These results are promising given the large number of classes and the low volume of training instances per object.

EM-Signatures of Similar Objects

As noted previously, I purposely included multiple objects of the same category, but different models, to see if *similar* devices would produce similar EM signals, and thus result in classification confusion. For our two refrigerators (Figure 4-9, F and N), two Apple laptops (A and U), and four LCD devices (G, R, S and W), there was 0% confusion. I found 1% confusion between the incandescent lamp (P) and the fluorescent lamp (H). These results strongly suggest that objects within a common category still have their own unique EM signatures.

EM-Signatures of Identical Objects

I ran a supplemental study to determine if EM signals are consistent across *identical* objects. For this, I used the ArmorActive touchscreens installed at four conference rooms in an office setting. I used the 24-object classifier from our second study, which was trained on *one* of these devices six weeks prior. I then evaluated real-time classification accuracy on all *four* units. I ran 10 randomized touch trials per device, for a total of 40 trials. My EM-Sense classifier correctly identified the object as the ArmorActive touchscreen 100% of the time (chance is 4%).

I used the same procedure for five iMac 2014 computers. I gathered training data on one machine, and ran 10 randomized classification trials on all five machines, for a total of 50 trials. Similarly, our classifier correctly classified these as iMacs with 98% accuracy (chance 4%). Overall, these results suggest that the EM signatures of identical devices are very similar, allowing for object recognition even when that *particular* instance of the object has never been touched before. This outcome is beneficial, as it means EM-Sense capable devices could be *preloaded* with an EM signature database of known objects (or *e.g.*, use a database in the cloud, which could grow overtime as users and companies add newly encountered objects).

Inferring Object State

For some objects, it is also possible to infer the operational state based on EM signature. For example, the magnitude of a power drill's EM signal is generally proportional to the rotational speed of its motor. In response, we ran another supplemental study to determine whether EM-Sense can exploit this phenomenon.

We trained an EM-Sense classifier to detect four operational speeds of a Dremel 3000 rotary tool: OFF, LOW, MID, and HIGH. A total of 200 instances were collected per state. Of note, we tweaked our EM-Sense noise-detection parameters (e.g., from 6s to 60s) to delay the system from integrating EM signals into its background noise profile. Across 40 trials (10 trials per state), our system achieved a real-time classification accuracy of 92.5% across the four speeds, suggesting that variations in EM signal can also reveal object state.

DISCUSSION AND LIMITATIONS

Because I perform adaptive background subtraction, the technique is location independent. In fact, most portable objects in the study (Dremel, laptop, iPad, etc.) were trained in one location (again, 6 weeks prior), and tested in another location without issue. Throughout piloting, we never observed a location effect. However, large passive objects, like the ladder, which are an amalgamation of EM signals from their respective local environments, would change if relocated.

My approach is passive, capturing noise, but not generating any signals itself. As discussed, this limits the technique to certain classes of objects. Indeed, most objects do not generate EM signals (*e.g.*, chairs, cups, books). Thus, the sensing scope is generally limited to electrical and electro-mechanical objects (and some large static objects, as discussed previously). Even still, not all of these objects are detectable, as the strength of EM signals is subject to the physical design of objects (*e.g.*, variations in electrical shielding and grounding). Moreover, some frequencies of noise may not be (faithfully) conducted through the human body and thus not reach the sensor.

Additionally, high fidelity analog sensing requires a stable and strong electrical ground as a reference. In my prototype, I tried to faithfully replicate the grounding conditions of a smartwatch, which contains a small battery. Additionally, the SDR receiver only provided 8-bit ADC resolution. With a superior ground reference and increased resolution (*i.e.*, a commercial-level implementation), EM-Sense may support even larger sets of objects.

Finally, I noticed in some cases that very strong environment noise (*e.g.*, transmitters that broadcast in overlapping bands of interest, or a microwave oven in operation) raised the noise floor and overpowered local EM signals. Even when noise subtraction is applied, high intensity noise can blanket subtle but discriminative EM signals. Additionally, because the sensor is worn on the arm, it is subject to frequent movements, which can cause unintended electrical effects (*e.g.*, Doppler shifts). Movement information from *e.g.*, accelerometers and gyroscopes could compensate for *e.g.*, sudden arm movements, or simply pause classification. Anecdotally, however, these effects appear to be minor.

CONCLUSION

EM-Sense is a novel sensing approach for on-touch object detection that exploits the unintentional electromagnetic noise generated by commonplace objects. By modifying a small, low-cost, embedded software-defined radio receiver, I can detect and classify EM signals in real time, enabling quick and robust detection of *when* an object is touched and *what* that object is. My experiments show that sensing can be accurate and robust. I also highlighted the wide variety of objects that EM-Sense can detect in several example contexts and environments, which point towards more powerful assistive, context sensing, and communication applications.

~

In the next chapter, I will discuss a complementary approach to EM-Sense, which I call *ViBand*. This system leverages the fact that in addition to electromagnetic noise, device use can also be inferred from micro-vibrations transmitted into the arm. As people use their hands, tiny micro-vibrations propagate into and through the arm, carrying information about the objects they interact with and the activities they perform throughout the day. Smartwatches are ideally situated to capture these vibrations.

5. VIBAND: BIO-ACOUSTIC SENSING USING COMMODITY WATCH ACCELEROMETERS

INTRODUCTION

As mentioned in the previous chapter, watches are unique among computing devices in that they are worn, offering great potential to transform arms and hands into expressive input and sensing platforms. As people use their hands, tiny micro-vibrations propagate through the arm, carrying information about the objects they interact with and the activities they perform throughout the day. Smartwatches are ideally situated to capture these vibrations (Figures 5-1 and 5-2).

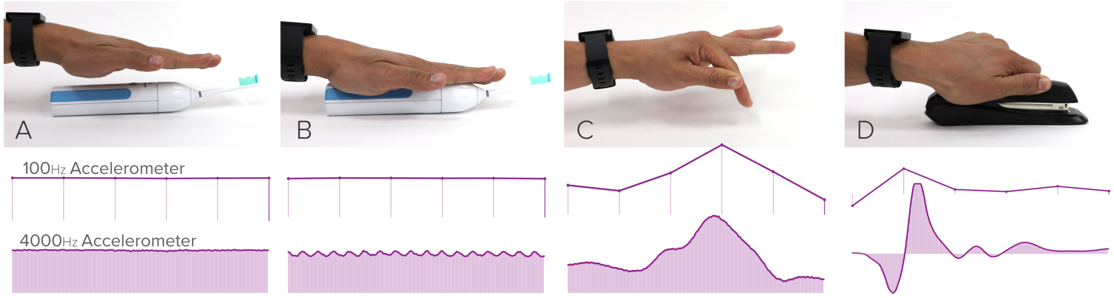


Figure 5-1. 100 Hz vs. 4000 Hz accelerometer signals. At steady state, both signals are identical (A). At, high frequency, micro-vibrations propagating through the arm are missed by the 100 Hz accelerometer (B). Vibrations can come from oscillating objects (B), hand gestures (C) and mechanical objects (D).

Although all modern smartwatches contain accelerometers, their APIs generally limit the sampling rate to around 100 Hz (Figure 5-1, top purple lines). This is sufficient for their main use: detecting the orientation of the watch (*e.g.,* to automatically activate the screen when raised). Some smartwatches also track step count (~2 Hz), which is also easily captured with 100 Hz sampling.

In this work, I use an off-the-shelf smartwatch with a modified OS kernel to capture accelerometer data at 4000 times per second (Figure 5-1, bottom purple lines). This fast sampling allows the smartwatch to not only capture coarse motions, but also rich bio-acoustic signals. For example, in Figure 5-1B, the sinusoidal oscillations of the toothbrush's motor are clearly visible. In Figure 5-1C (fingers rubbing) and 5-1D (pressing stapler), the 100 Hz signal captures the coarse impulse, but no useful spectral information is available.

Most smartwatches include microphones, which provide even higher sampling rates (typically 44.1 kHz). However, microphones are specifically designed to capture airborne vibrations, not contact vibrations, which means purposeful signals must be segmented from background environmental noise. In contrast, our bio-acoustic approach only captures signals that are physically coupled to the body (Figures 5-1A and B). This approach makes our technique naturally resistant to external environmental noise.

As I will discuss, my approach can be applied to a wide array of use domains; I selected three that I found to be particularly compelling. First, I use bio-acoustic data to classify hand gestures, which I combine with on-device motion tracking to enable a wide range of expressive input modalities. Second, I detect and classify vibrations of grasped mechanical or motor-powered objects, enabling un-instrumented object recognition. Finally, I explore structured vibrations and demonstrate reliable data transmission through the human body.

My evaluations show that my sensing technique is accurate, robust to noise, relatively consistent across users, and independent of location or environment. My system, which I call *ViBand*, makes the following contributions: 1) a system that performs bio-acoustic sensing using commodity accelerometers already present in modern smartwatches; 2) a set of example use domains enabled by our technique, including gesture detection, grasped object sensing, and data transmission; 3) a series of user studies evaluating the feasibility and accuracy of our sensing technique; and 4) a series of example applications for wrist-worn bio-acoustic sensing that illustrate the potential of my approach. Collectively, these bring to light novel and rich functionality for smartwatches, expanding their envelope of possible interactions.

RELATED SYSTEMS

ViBand intersects with a range of HCI and sensing topics I mentioned in the Background chapter, including worn hand input and gesture sensing, and object recognition. In this section, I will briefly discuss systems directly related to ViBand, which includes bio-acoustics, and through-body data transmission.

Bio-Acoustic Input and Sensing

Bio-acoustics has been studied in many fields [65, 86, 97, 175, 208, 245] including HCI. For instance, Amento *et al.* [9] placed contact microphones on the user's wrist to capture gross finger movement. Their work was first to demonstrate the use of on-body acoustic signals to passively recover finger gestures in one hand, although no formal evaluations were conducted. This became the direct inspiration to Hambone [62], which instrumented the user's limbs with piezo sensors to detect gestures (*e.g.*, finger flick, left foot rotate).

Likewise, Skinput [99] leveraged a similar technique, using an array of piezo sensors strapped onto the user's arm (above and below the elbow). Building on top of Hambone, Skinput's sensor placement further expanded touch-interaction onto the arm, palm, and fingers. The Sound of Touch [182] employed a similar technique, using transdermal propagation of ultrasound across the user's arm to recover finger gestures and discrete touch points. A signal-emitting ring emitted ultrasound when the arm was touched or tapped, while an array of transducers monitored the transmitted signal. These bio-acoustic sensing approaches rely heavily on special-purpose sensors, increasing their invasiveness and ultimately limiting their practicality.

Through-Body Data Transmission

Data transmission through the body has been successfully demonstrated with radio frequency (RF) waves, in the form of "personal area networks." Such networks can successfully transmit data at very high speeds amongst specially-equipped devices near

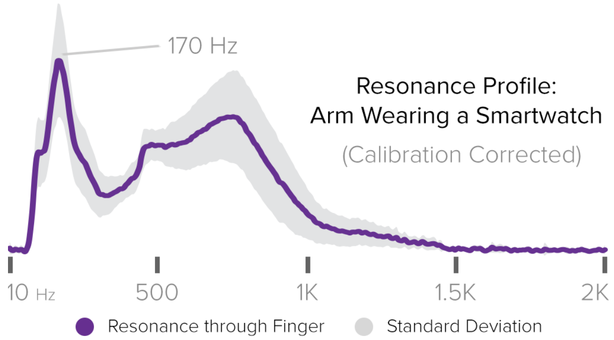


Figure 5-2. Bio-aoustic resonance profile of an arm wearing a watch, calibrated on the arm. Vibration frequencies between 20 Hz and 1 kHz transmit particularly well through the arm.

the body [309]. More related to my technical approach are systems that use vibroacoustics to transmit data. Ripple [225], using an accelerometer and vibration motor mounted to a cantilevered metal arm (to amplify vibrations), transmitted data at about 200 bits/sec. Ripple II [225] utilized audible frequencies (2-10 KHz) to transmit data between a vibrating finger ring and a microphone at the finger tip. AT&T Labs publicly demonstrated a system that transmitted bio-acoustic data using a piezoelectric buzzer [14], although the technical details have not been published. Finally, and most similar to ViBand, is OsteoConduct [307], which transmits data through bone conduction. This system successfully demonstrated a data transmission rate of “almost 5 bits/sec” between the wrist, ear and lower back.

THEORY OF OPERATION

Although most mobile devices (including smartwatches) contain accelerometers and other inertial measurement sensors, existing APIs generally limit accelerometer data access to about 100 Hz. This rate is sufficient for detecting coarse movements such as changes in screen orientation or gross interactions such as walking, sitting, or standing. However, these accelerometers often support significantly higher sample rates – up to thousands of Hz. At these faster sampling speeds, the smartwatch can listen to nuanced and fine-grained movements that are initiated or experienced by the human arm. Like water, the human body is a non-compressible medium, making it an excellent vibration carrier. For example, when sampling at 4000 Hz, vibrations oscillating up to 2000 Hz (*e.g.*, gestures, grasped objects) can be sensed and identified (per the Nyquist Theorem). This superior sensitivity transforms the smartwatch into a *bio-acoustic* sensor capable of detecting minute compressive waves propagating through the arm.

In my initial experiments, I sought to investigate whether the high-speed accelerometer signal was indeed bio-acoustic. To test this theory, I walked around our lab and performed a range of activities (*e.g.*, tapping on table, scratching hand, grasping power tools) while simultaneously extracting accelerometer signals obtained from our prototype. Each activity and object produces characteristic vibroacoustic signatures, and more critically, were only captured when in contact with the hand. These signals resemble those captured by a microphone, yet lack any audible external noise.

Like any medium, the human arm characteristically amplifies or attenuates vibrations at different frequencies. Therefore, I ran an experiment to identify the frequency transmission envelope of the human arm while wearing a smartwatch. First, I captured the resonance profile of an unworn smartwatch (LG G W100) placed directly on a transducer running a 0 to 2 kHz vibrational sweep. I then captured the resonance profile while the smartwatch was worn on the arm while pressing the transducer with the index finger. Figure 5-2 depicts the average resonance profile across three users. These results suggest that oscillations between 20 Hz to 1 kHz transmit particularly well through the arm, with salient peaks at ~170 Hz and ~750 Hz.

IMPLEMENTATION

My proof-of-concept system was developed on an LG G W100 smartwatch, which includes an InvenSense MPU6515 inertial measurement unit (IMU) capable of measuring acceleration at 4000 samples per second [116]. Of note, this is the same series of accelerometer used in many other popular smartwatches, including the Moto 360, LG Watch Urbane, Samsung Gear 2 and Gear Fit. However, the maximum rate obtainable through the Android Wear API [89] is 100 Hz. Therefore, my coauthors and I modified the Linux kernel on the device, replacing the existing accelerometer driver with our own custom driver.

Specifically, our kernel driver interfaces with the IMU via I^2C, configuring the IMU registers to enable its documented high-speed operation [117]. Notably, this requires us to use the IMU’s onboard 4096-byte FIFO to avoid excessively waking up the system CPU. However, this FIFO only stores 160 ms of data—each data sample consists of a 16-bit sample for each of the three axes. Thus, we configured the driver to poll the accelerometer in a dedicated kernel thread, which reads the accelerometer

FIFO into a larger buffer every 50 ms. Overall, this thread uses 9% of one of the watch’s four CPU cores.

I found that the accelerometer’s internal clock was not temperature-stabilized, resulting in higher sampling rates as the CPU temperature increased. I measured sampling rates varying between 3990 Hz (watch sleeping, off wrist) to 4080 Hz (on arm, high CPU activity). In response, my coauthors and I augmented our kernel driver to compute the rate at which samples were written into the MPU’s FIFO buffer using a nanosecond-precision kernel timestamp. For applications requiring precise sampling rates, such as resonance profiling and data transmission, we normalized the input data to 4000 Hz using a sinc-based interpolator capable of supporting continuously variable input sample rates [55].

For prototyping purposes, I configured the watch to transmit all captured accelerometer data via Bluetooth to a paired Android phone, which then relayed the data to a laptop for analysis. This enabled rapid testing, iteration and development of our bio-acoustic applications. However, I also implemented data transmission and object classification on the watch itself for fully self-contained operation. With this implementation, ViBand unlocks a wide variety of applications. Next, I describe how my technique enables novel interactions in three distinct application domains.

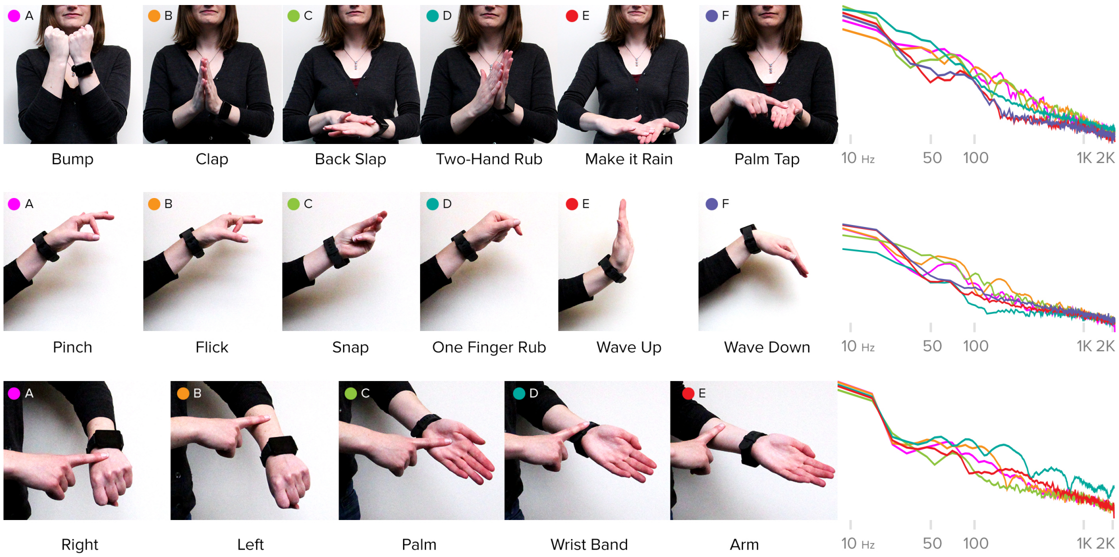


Figure 5-3. Example ViBand gesture sets: two-handed (top), one-handed (middle), and on-body touch input (bottom).

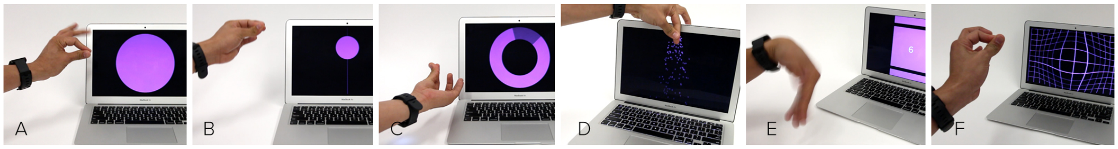


Figure 5-4. ViBand enables a wide range of interaction modalities when combined with coarse motion tracking information natively available on existing smartwatches. Modalities include binary buttons (A), linear sliders (B), radial knobs (C), counters (D), hierarchical navigation (E), and even relative spatial tracking (F). In these examples, a pinching action serves as a gesture clutch.

EXAMPLE DOMAIN 1: GESTURES

First, my technique can be used to classify unique hand gestures, such as flicks, claps, snaps, scratches and taps. These hand gestures create distinctive micro-vibrations that propagate through the arm. Depending on the location and type of gesture, different frequencies of vibrations are generated. Subsequently, various frequencies are attenuated during propagation (*e.g.*, anatomical features can act as passive vibroacoustic filters [99]). The resulting frequency profiles make many gestures uniquely identifiable. I explored three example gesture sets that offer distinctive bio-acoustic signals: one-handed gestures, two-handed gestures, and on-body touch input.

Once the bio-acoustic signals are captured on the watch, I perform several signal processing operations to detect and classify hand gestures in real-time. For each incoming signal frame t, I first compute the power spectra of the fast Fourier transform (FFT) on data from each accelerometer axis, producing three spectra $\mathbf{X}_t$, $\mathbf{Y}_t$, $\mathbf{Z}_t$. I use a Hamming window on the FFT to minimize spectral banding. To make sensing robust across hand orientations, I remove the DC component and combine the three FFTs into one by taking the max value across axes ($F_{t,i} = \max(X_{t,i}, Y_{t,i}, Z_{t,i})$)

Next, I compute the average of the w=20 past FFT spectra ($S_i = \mu(F_{t,i}, F_{t-1,i}, \ldots, F_{t-w+1,i})$) and extract statistical features from the averaged signal: mean, sum, min, max, 1st derivative, median, standard deviation, range, spectral band ratios, and the n highest peaks (n=5). These features form the input to an SVM (poly kernel, $\varepsilon=10^{-12}$, normalized) for real-time classification. From my experiments, I found that band ratios, peaks, mean, and standard deviation provide 90% of the bio-acoustic signal's discriminative power. The table below describes these features and their justifications.

Feature Set	**Operation**	**Justification**
Power spectrum	S_i	Specific frequency data
Statistical	μ_S , σ_S , Σ_S , max(**S**), min(**S**), centroid, peaks	Characterizes gross features of FFT signal
1st Derivative	$\frac{d}{dt}(\boldsymbol{S}_{t+1}) = \boldsymbol{S}_{t+1} - \boldsymbol{S}_t$	Encodes signal peaks and troughs
Band Ratios	$\boldsymbol{B}_{j,k} = \frac{\boldsymbol{S}_j}{\boldsymbol{S}_k}$	Describes overall FFT shape, power distribution

When hand gestures are combined with relative motion tracking (*e.g.*, native data from IMUs), ViBand uncovers a range of interaction modalities (see Figure 5-4). These include: buttons, sliders, radial knobs, counters, hierarchical navigation, and positional tracking. On top of these, ViBand can build applications that utilize these rich and expressive interaction modalities.

EXAMPLE DOMAIN 2: OBJECT DETECTION

ViBand can also be used to identify grasped objects in order to *e.g.,* launch context-relevant functionality or applications automatically. Specifically, when a user operates a mechanical or motor-powered device, the object produces characteristic vibrations, which transfer into the operator. My bio-acoustic smartwatch captures and classifies these signals, allowing interactive applications to better understand their user's context and further augment a wide range of everyday activities.

Worn microphones, which capture sounds produced by objects in operation, have been previously used for object recognition [282]. Because microphones are coupled through the air, they are particularly sensitive to ambient noise. Further, microphone-based techniques can only approximate when users are near to an active object, but not when they are truly interacting with an object. In contrast, my approach recognizes objects at the moment of touch, allowing us to generate *on-touch* and *on-release* events.

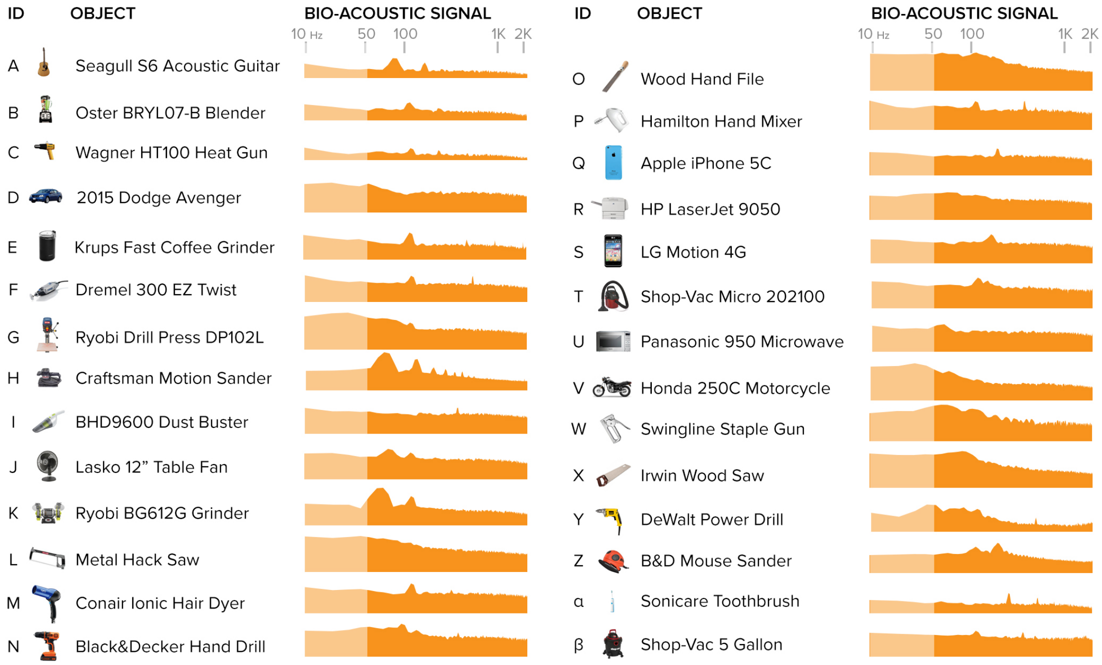


Figure 5-5. Example ViBand gesture sets: two-handed (top), one-handed (middle), and on-body touch input (bottom).

I utilize the same signal processing pipeline for both gestures and object detection, but with slightly tweaked parameters (w=15, n=15). I also apply a simple voting mechanism (size=10) to stabilize the recognition. My setup recognizes a wide range of objects (see Figure 5-5), complementing existing techniques (*e.g.*, [148, 167, 277]), and further expanding capabilities for rich, context-sensitive applications.

EXAMPLE DOMAIN 3: STRUCTURED VIBRATIONS

In addition to being able to capture "passive" vibrations from objects and hand motions, ViBand can also augment environments and objects with *structured vibrations*. I developed a "vibro-tag" consisting of a small (2.4 cm^3) SparkFun COM-10917 Bone Conductor Transducer, powered by a standard audio amplifier. When a user touches the transducer, modulated vibrations are transmitted bio-acoustically to the smartwatch, which decodes the acoustic packet and extracts a data payload (Figures 5-6 and 5-7). Such tags could be used much like RFID or QR Codes while employing a totally orthogonal signaling means (vibroacoustic). A unique benefit of ViBand is that

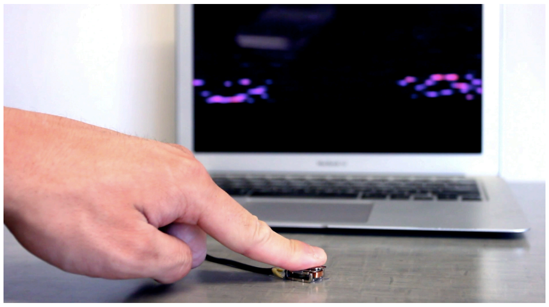


Figure 5-6. A “vibro-tag” transmitting FSK bio-acoustic data through the arm, and received by the watch.

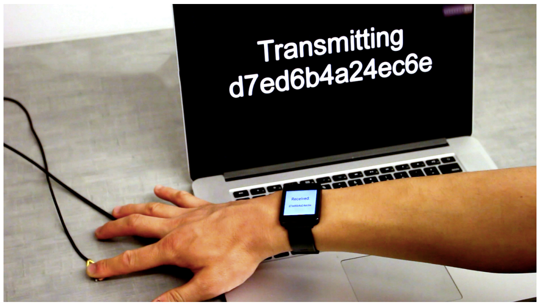


Figure 5-7. We implemented a range of encoding schemes to robustly transmit data-encoded vibrations.

it is only triggered upon physical touch (*i.e.*, not just proximity) and is immune to variations in *e.g.*, lighting condition.

Critically, I wanted to make the vibro tags *inaudible*, but still capable of transmitting data at high speed. Because the accelerometer can only sense frequencies up to 2 KHz, I cannot use ultrasound frequencies (*e.g.* frequencies above 16 kHz). I also ruled out frequencies above 300 Hz, as they would manifest as audible “buzzing” sounds. Using the transmission envelope data (Figure 5-2) and experiments with transmission frequencies, I selected 200 Hz as a suitable carrier frequency for transmission.

The data transmission system is a full stack signal pipeline, consisting of data packetization, error detection, error correction, and modulation layers. I first segment the input data stream into individually transmitted *data packets*. Applications are free to choose their own packet formats, but the recommended format consists of an 8-bit sequence number combined with a data payload. Packet size is constrained by the error detection and correction layers; in the current implementation, it can be up to 147 bits in length. In order to detect transmission errors and ensure that bad data is not accidentally accepted, I append an 8-bit *cyclic redundancy check* (CRC) to the message. In the present implementation, the CRC is computed by truncating the Adler-32 CRC of the message.

Next, error correction is applied. Although this stage also detects errors (like the CRC), its primary purpose is to mitigate the effects of minor transmission problems. We use a Reed-Solomon code [215] with 5 bits per symbol, allowing ViBand to have 31 symbols per message (a total of 155 bits). These parameters were chosen to allow a single message to be transmitted in approximately one second using common modulation parameters. The number of ECC symbols can be tuned to compensate for noisier transmission schemes; see the evaluation for more details. At this point, I transmit the full message+CRC+ECC, totaling 155 bits, as modulated vibrations. I experimented with four different classical modulation schemes [20], using binary Gray coding to encode bit strings as symbols:

- *Amplitude Shift Keying* (ASK): data is encoded by varying the amplitude of the carrier signal.
- *Frequency Shift Keying* (FSK): data is encoded by transmitting frequency multiples of the carrier signal (Figure 5-6, note spectrogram on laptop screen in background).
- *Phase Shift Keying* (PSK): adjusting the phase of the carrier signal, with respect to a fixed reference phase.
- *Quadrature Amplitude Modulation* (QAM): data encoded as variations in phase and amplitude, with symbols encoded according to a *constellation diagram* (Figure 5-10) mapping phase and amplitude to bit sequences.

I prefix the message with a short header sequence consisting of three 20 ms chirps at 100 Hz, 300 Hz, and 200 Hz. This sequence is readily recognized and quite unlikely to occur by accident. Furthermore, the presence of a 300 Hz chirp in the header prevents accidental detection in the middle of a transmission. Finally, the 200 Hz chirp provides a phase and amplitude reference for the ASK, PSK and QAM transmission schemes, eliminating the need for clock synchronization between sender and receiver.

Decoding is performed on the watch itself (Figure 5-7), using an optimized decoding routine written in C. The decoder continuously reads samples from the accelerometer, converts the samples to 6400 Hz (to simplify FFT computations), and continuously searches for the header sequence. When found, the decoder demodulates the signal (using the amplitude and phase of the 200 Hz header chirp), performs decoding, verifies the CRC, and reports the resulting message to an application (if successful).

SYSTEM EVALUATION

My user studies sought to address critical questions on feasibility, accuracy, and key operating parameters for bio-acoustic sensing across different application contexts. To push the limits of my system even further, I explored several questions relating to robustness and consistency: Are object vibration signatures consistent over time? How robust is the sensing accuracy when the watch is re-worn? Is sensing robust across different locations? Can sensing work on a model trained on a different device?

Participants

I recruited 18 participants (10 female, mean age 25.3, 17 right-handed) for a live user study. Participants were asked to perform a series of tasks while wearing my ViBand prototype. Users wore the prototype on whichever arm they preferred. Since variations in user anatomy could affect bio-acoustic signal propagation, I recorded user's body mass index (BMI, mean=22.3) to further explore the accuracy of my sensing technique.

The study had three distinct phases, which I discuss in detail subsequently, and lasted approximately 70 minutes in total; participants were paid $20. Of note, one user had to be dropped from the study because the smartwatch strap did not have a notch that allowed the smartwatch to be adequately secured to their thin arms. All subsequent discussion uses data from the remaining 17 participants.

Setup and Apparatus

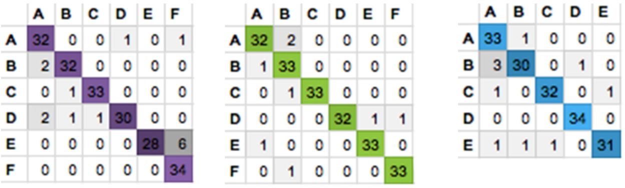
The entire study took place in a mixed office-workshop building with two floors and rooms of varying function. Although parts of the study involved accessing tools spread across different rooms and floors, participants were initially welcomed in the lobby. From there, participants were briefed and eventually asked to wear the LG G Watch. Since ViBand relies on physical coupling to the body, the system is susceptible to loose armband tightness. I instructed participants to wear the watch in a "comfortable but firm" manner. At the end of the study, I had participants complete two Likert-scale questions: participants reported a mean tightness of 4.2 (1=loose, 5=tight), with a mean comfort rating of 3.5 (1=uncomfortable, 5=comfortable).

To verify the robustness of the classifiers across devices, I ran the study using two different smartwatches of the same model (Watch A and Watch B), randomized per user. All machine learning models were trained on Watch A, but deployed and tested on both watches. Data from our watches was streamed to a laptop via a Bluetooth bridge for data recording and live classification.

Study 1: Gesture Recognition

Procedure. I trained different machine learning models for each gesture set (Figure 5-3). Each model was calibrated per-participant, *i.e.*, models were trained for each user. First, the presentation order of the three gesture sets was randomized, and participants were asked to perform a gesture within that set. I collected fifteen data instances per gesture and trained a model *in situ*. Once trained, the participant was asked to perform each gesture once (in random order). Once all gestures were performed for a given set, the participant was asked to remove the watch. After approximately five seconds, participants were asked to re-wear the watch. The participant then performed all of the gestures in that set again (random order), with the classifier output recoded. In total, participants performed two rounds, per gesture, per set.

Results. Across all 17 users and 17 gestures (in all three gesture sets), ViBand achieved a mean accuracy of 94.3% (SD=4.1%). Figure 5-8 offers a confusion matrix for each gesture set. I found no statistically significant differences between users and their BMI.



	A	B	C	D	E	F
A	32	0	0	1	0	1
B	2	32	0	0	0	0
C	0	1	33	0	0	0
D	2	1	1	30	0	0
E	0	0	0	0	28	6
F	0	0	0	0	0	34

	A	B	C	D	E	F
A	32	2	0	0	0	0
B	1	33	0	0	0	0
C	0	1	33	0	0	0
D	0	0	0	32	1	1
E	1	0	0	0	33	0
F	0	1	0	0	0	33

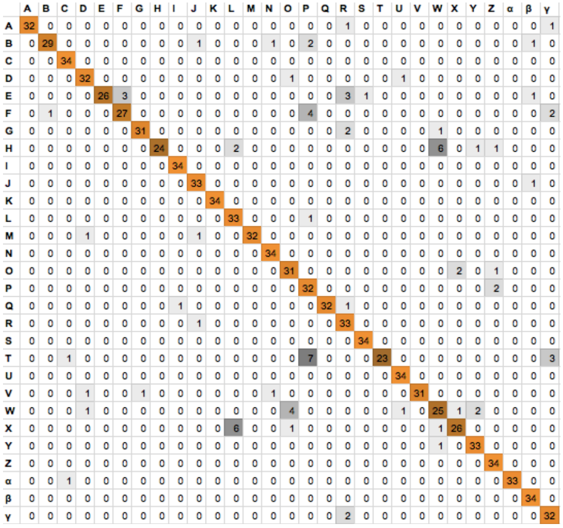
	A	B	C	D	E
A	33	1	0	0	0
B	3	30	0	1	0
C	1	0	32	0	1
D	0	0	0	34	0
E	1	1	1	0	31

Figure 5-8. Confusion matrices for one-handed gestures (left, purple), two-handed gestures (mid), and on-body touch input locations (right). Across all users and all gestures, average accuracy was 94%.

There was a slight decrease in accuracy before and after the watch was removed, but this was not statistically significant, and so the results have been combined.

Study 2: Object Detection

This study aimed to evaluate whether, 1) bio-acoustic signals could be used to classify grasped objects, 2) object vibrations are consistent over time, and 3) how well the



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	α	β	γ
A	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
B	0	29	0	0	0	0	0	0	0	1	0	0	0	1	0	2	0	0	0	0	0	0	0	0	0	0	0	1	0
C	0	0	34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	32	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
E	0	0	0	0	26	3	0	0	0	0	0	0	0	0	0	0	0	3	1	0	0	0	0	0	0	0	0	1	0
F	0	1	0	0	0	27	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	2
G	0	0	0	0	0	0	31	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	1	0	0	0	0	0	0
H	0	0	0	0	0	0	0	24	0	0	0	2	0	0	0	0	0	0	0	0	0	0	6	0	1	1	0	0	0
I	0	0	0	0	0	0	0	0	34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
J	0	0	0	0	0	0	0	0	0	33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
K	0	0	0	0	0	0	0	0	0	0	34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L	0	0	0	0	0	0	0	0	0	0	0	33	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
M	0	0	0	1	0	0	0	0	0	1	0	0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	0	34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
O	0	0	0	0	0	0	0	0	0	0	0	0	0	0	31	0	0	0	0	0	0	0	0	2	0	1	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	32	0	0	0	0	0	0	0	0	0	2	0	0	0
Q	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	32	1	0	0	0	0	0	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	33	0	0	0	0	0	0	0	0	0	0	0
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	34	0	0	0	0	0	0	0	0	0	0
T	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0	0	23	0	0	0	0	0	0	0	0	3
U	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	34	0	0	0	0	0	0	0	0
V	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	31	0	0	0	0	0	0	0
W	0	0	0	1	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	1	0	25	1	2	0	0	0	0
X	0	0	0	0	0	0	0	0	0	0	0	6	0	0	1	0	0	0	0	0	0	0	1	26	0	0	0	0	0
Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	33	0	0	0	0
Z	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	34	0	0	0
α	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	33	0	0
β	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	34	0
γ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	32

Figure 5-9. Object confusion matrix across 29 objects and 17 users. Results from both testing rounds (pre and post re-wearing of the watch) are combined, yielding 34 trials per object. Chance is 3%.

approach works across different users. I conducted a month-long study to explore this. First, I collected data from one user on 29 objects using a single ViBand prototype (Watch A). The collected data was then used to train a machine learning model. The example object set and their bio-acoustic signatures are shown in Figure 5-5.

Procedure. Four weeks later, the same model was used to perform real-time object classification for all 17 participants using the same 29 objects. Objects were spread across six locations to vary environmental conditions. These locations include: personal desk area, shared woodshop, office, kitchen and bathroom, public common area, and a parking space outside of the building. Further, all objects were tested in a location that was different from where it was trained (except the motorcycle). A single trial in our live object classification study involved a user interacting with one of our 29 objects. Participants were briefly shown how to operate the objects (for safety), but were free to grasp the object however they wished. Objects were randomized per location (rather than randomized globally). For each location, I performed two classification rounds per object (58 total trials for all 29 objects), with a quick break in between (*i.e.*, go to location one, test objects, break, test again). During each break, I asked the participant to remove the watch from their wrist, and wear it again after ten seconds. This routine is a more realistic measure of the system's accuracy, as users wear and re-wear smartwatches in the real world.

Results. Across 29 objects, 17 users, and using data that was trained on a single person four weeks prior, ViBand obtained an overall object detection accuracy of 91.5% (SD=4.3%). I found two outlier objects that were 3.5 standard deviations below the mean (using Iglewicz and Hoaglin's outlier test [114]). When these two outlier objects are removed, ViBand obtained an overall accuracy of 94.0% (27 objects). Figure 5-9 shows the confusion matrix for all 29 objects. Note that many objects achieve 100% accuracy, despite purposeful inclusion of experimental procedures that usually impact recognition accuracy, *e.g.,* no per-user calibration, significant time separation between train and test, and removal and replacement of the smartwatch *during* the experiment.

Additionally, I found no statistical differences between accuracies on the two watch prototypes. I also found no statistically significant differences between participant's body-mass index, object location, and whether before/after the watch was removed. Overall, these results suggest that object detection is indeed accurate and robust across users and environment, and object bio-acoustic signatures are consistent over time.

Study 3: Structured Vibration Data Transfer

I sought to quantify the fidelity of our structured vibrations through a data transmission study. I first ran a pilot study in which I tested several variations of ASK, PSK, FSK and QAM modulation schemes over multiple symbol rate and bits-per-symbol configurations. I rejected configurations that resulted in higher than 10% bit error rates. I chose the five schemes with the highest raw transmission rates: 4-FSK (2 bits per symbol, transmitting frequencies of 50, 100, 150 and 200 Hz), 4-PSK (2 bits per symbol), 8-PSK (3 bits per symbol), 8-QAM (3 bits per symbol, non-rectangular constellation), 16-QAM (4 bits per symbol, non-rectangular constellation). FSK ran at 50 symbols per second, while the other four ran at 100 symbols per second – higher symbol rates were found to be too unreliable.

Procedure. I collected four rounds of data from each user. In each round, the user placed either their outstretched index finger (F) or their whole palm (P) against the transducer, with the watch on the same arm as the contacting hand. Users were randomly assigned one of four possible round orderings: FPFP, FPPF, PFFP, or PFPF. Between the second and third round, the watch was removed and put back on again. I tested both hand and finger contacts in order to determine the error rate difference between these two postures. The hand is a larger contact area close to the wrist, so I anticipated lower error rates there compared to the finger. Each round consisted of 5 data transmission trials for each condition, for a total of 25 trials. Trial order was fully randomized. In each trial, the experiment system (running on a laptop connected to the transducer) transmitted a single packet using the one of the five modulation schemes (Table 2) and waited for 0.5 seconds for the packet to be demodulated. In total, this yielded 1700 trials (17 participants x 4 rounds x 5 conditions x 5 trials per condition)

Results. Out of the 1700 trials collected, no header could be detected in 23 trials (1.4%). These trials were excluded from further analysis. For all remaining trials, I computed the bit error rate by comparing the received, demodulated message with the original transmitted message. The results are summarized in Table 5-1. Raw bit transmission rate indicates the modulation method's data transmission speed, while bit error rate (BER) indicates the percentage of bits in the received message that were incorrect. The bit error distribution has a significant long tail across all conditions: most messages are received correctly, but a small number of messages are received with many errors. Therefore, I also computed the *80^{th} percentile BER* (BER_{80}), for parity with Ripple [225], to get a better sense of the distribution. This measurement has a practical impact on the choice of error correction parameter: if I choose an error correction scheme that can correct errors up to BER_{80}, then ViBand can successfully decode 80% of transmitted packets.

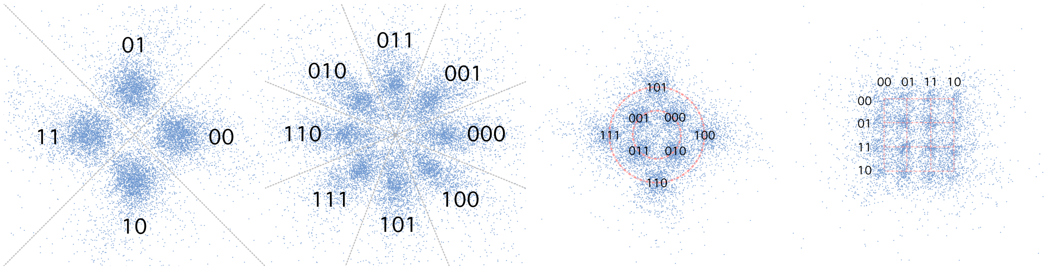


Figure 5-10. QAM constellation diagrams plotting all data recorded in the study. From left to right: 4-PSK (showing phase boundaries), 8-PSK (showing phase boundaries), non-rectangular 8-QAM (showing symbol construction), and rectangular 16-QAM (showing grid construction).

The results indicate that 4-PSK is the clear winner in terms of BER across all conditions, when considering the raw bit rate. With a BER_{80} of 0.6% (0.93 message bits), ViBand would need to add only 2 Reed-Solomon ECC symbols to our message in order to correct 80% of messages, leaving 137 bits for the payload. This payload takes 0.83 seconds to transmit (155 bits at 200 bits per second, plus header overhead), for an overall transmission rate of 165 bits per second (with a 20% packet loss rate), through the finger, hand and wrist. This significantly outperforms the most related prior work, OsteoConduct, which operated at "almost 5 bits/sec" [307]. In fact, this performance approaches that of Ripple [225], which obtained an effective bitrate of 196.6 bits per second (using correction up to the 80th BER percentile) transmitting though a cantilevered metal bar (which is obviously far superior to human tissue for transmitting mechanical vibrations).

	Bit Rate (bits/sec)	BER (hand)	BER (finger)	BER_{80} (hand)	BER_{80} (finger)
4-FSK	100	1.0%	2.9%	0.0%	0.3%
4-PSK	200	1.0%	3.1%	0.0%	0.6%
8-PSK	300	2.9%	5.8%	3.8%	7.1%
8-QAM	300	3.6%	7.9%	7.7%	15.3%
16-QAM	400	6.9%	8.6%	12.8%	16.0%

Table 5-1. Data Transmission Results

Study 4: False Positive Rate

In a system that takes advantage of accelerometers, it is critically important to reduce the detection of false positives (*i.e.*, an action that is unintentionally triggered). To validate the resistance of our sensing approach to false positives, I trained our classifier with a large set of *background* data (*i.e.*, negative training examples) and tested the system *live* with participants. Specifically, the 17 participants were asked to perform several mundane and physically rigorous activities in different locations. These activities included: walking for two minutes, jogging in place for 30 seconds, performing jumping jacks for 30 seconds, reading a magazine or book for one minute, and washing hands for 30 seconds. These five activities were randomly interspersed

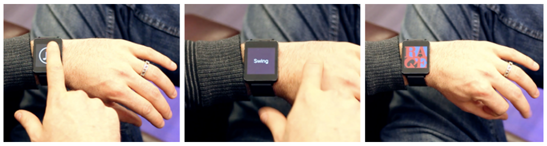


Figure 5-11. In a launcher, we can place navigation controls on the skin. Users can traverse back up through the hierarchy with a flick gesture.

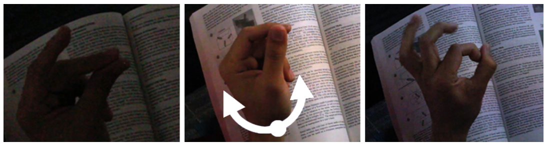


Figure 5-12. Snapping turns on the nearest light. A pinch followed by wrist rotation offers brightness control. A flick confirms the manipulation.

throughout the object detection study (*i.e.*, when users transitioned between each of the six building locations).

While participants performed these activities, I tallied the number of "false detections" triggered by the system (any prediction that was not "null" or "no object" was considered a false positive). Across 17 users, six random locations, and five activities, collectively spanning a total of 77 minutes, the system triggered a total of six false positive classifications. For 12 of 17 participants, the system triggered no false positives. These results suggest that false positives can be greatly reduced by exposing the machine-learning model to a large set of negative examples.

EXAMPLE APPLICATIONS

I created a series of example applications in the three use domains previously described: gestures, object detection, and data transmission. These functional applications leverage the use of real-time recognition of bio-acoustic signals.

Expanding Smartwatch Input. Hand gestures can be used to appropriate the area around the watch for input and sensing. For example, in a smartwatch launcher, we can place navigation controls on the skin (*e.g.,* left, right, select), as well as enable users to traverse back up through the hierarchy with a flick gesture (Figure 5-11).

Controlling Remote Devices. Likewise, gestures can be used to control remote devices. For example, a user can clap to turn on a proximate appliance, such as a TV; wave gestures navigate and snaps offer input confirmation. Flick gestures can be used to navigate up the menu hierarchy (Figure 5-12).

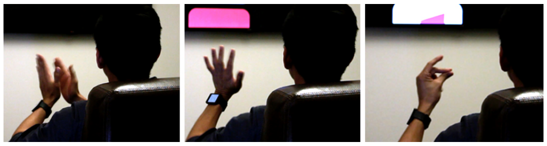


Figure 5-13. Gestures can control remote devices. In the left, the user claps to turn on the TV. Wave gestures navigate and snaps select.

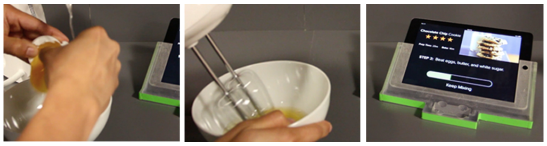


Figure 5-14. Object sensing enables rich, context-sensitive applications such as sensing equipment used in the preparation of a meal (left).

Input for Infrastructure. Gestures can also be used to control nearby infrastructure. For example, a user can snap his fingers to turn on the nearest light. A pinching gesture can be used as a clutch for continuous brightness adjustment, and a flick confirms the manipulation (Figure 5-13).

Object-Aware Applications. Because our sensing approach can also be used to identify objects, we offer applications the ability to better understand context and augment everyday activities. For example, we can augment the kitchen experience by sensing equipment used in the preparation of a meal and *e.g.*, offering a progress indicator for blending ingredients with an egg mixer (Figure 5-14).

Detecting Unpowered Objects. Our technique can also sense unpowered objects, such as an acoustic guitar. For example, we can detect the closest note whenever the guitar is grasped, and provide visual feedback to tune the instrument precisely (Figure 5-15). Detection happens *on touch*, which makes it robust to external noise in the environment.

Augmenting Analog Experiences. Through object sensing, we can also augment *analog* experiences with *digital* interactivity. For example, with a Nerf gun, we can detect the loading of a new ammo clip, and then track of the number of darts left (Figure 5-16).

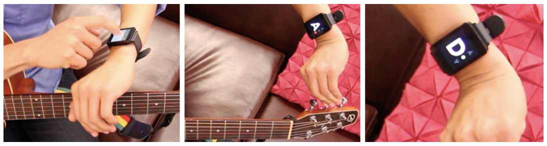


Figure 5-15. For an unpowered object *e.g.*, an acoustic guitar, we can detect the closest note to tune the instrument precisely.

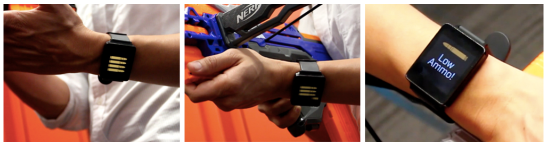


Figure 5-16. Augmenting analog experiences. In Nerf gun, we can detect users loading ammo clips (left) and firing darts (middle).

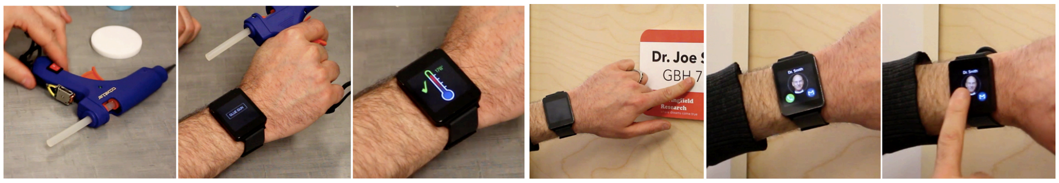


Figure 5-17. Vibro-tags emit structured vibrations that carry data. For example, the user can touch an office nameplate instrumented with a vibro tag (top), which transmits the person's contact details to the smartwatch. We can also instrument non-mechanical objects, e.g., a glue gun (bottom). It broadcasts an object's ID and metadata — in this case, the current temperature and ideal operating range.

Vibro-Tags. Many classes of objects do not emit characteristic vibrations, which means ViBand cannot detect them. However, we can instrument them with a *vibro-tag* that emits inaudible, structured vibrations containing data. For example, we can instrument a glue gun (non-mechanical but electrically powered) with a vibro tag. The tag broadcasts an object ID that enables the watch to know what object is being held. It also transmits metadata *e.g.*, its temperature and ideal operating range (Figure 5-17).

Tagging Infrastructure. Structured vibrations are also valuable for augmenting fixed infrastructure with dynamic data or interactivity. For example, in an office setting, a user can retrieve more information about an occupant by touching the room nameplate augmented with a vibro tag, which transmits *e.g.,* the person's contact details to the smartwatch (Figure 5-17, right).

DISCUSSION

I have demonstrated that high-speed accelerometer sampling in smartwatches offers new and compelling human interface possibilities, and my hope is that this research encourages manufacturers to expose this useful data source in future devices. Dedicated microcontrollers that sit between sensors and the application processor ("sensor hubs") are already employed in many devices to help improve power efficiency. Sensor hubs could easily be used to buffer and process high-speed accelerometer data, enabling low-power, always-on, bio-acoustic applications.

ViBand should be readily portable to most smartwatches, as modern IMUs have comparable specs. Indeed, InvenSense is one of the largest IMU vendors, and as mentioned previously, the same series of accelerometers we use is also used in many other popular smartwatches. I also saw no performance difference when training or testing on two different watches of the same model. However, I suspect that entirely different models of smartwatch would alter the physical coupling slightly. I ran some basic tests, and this effect appears to be minor compared to the *active* signal of the gesture, object or data transmitting transducer.

I note that unintended and competing oscillations (e.g., bus/walking) inherently decrease the signal-to-noise ratio (SNR). For human actions, like locomotion, I observed these chiefly happen within lower frequency bands (roughly 0-20 Hz), which are easily filtered. Overall, similar to many deployed technologies, noise robustness can be improved through *e.g.*, adaptive background subtraction or by incorporating diverse negative training examples.

Finally, applying structured vibrations to large objects or surfaces (*e.g.*, tables) can result in audible noise (by essentially turning the surface into a amplifying diaphragm). For this reason, I used a vibration transducer with a small active area (as opposed to a voice coil with a large diaphragm), so airborne emissions were limited. Additionally, I note that malicious interception of vibration-borne data might be possible, especially with *e.g.*, a high quality directional microphone or laser Doppler vibrometer. Whether all bits could be resolved is an open question.

CONCLUSION

In ViBand, I explored bio-acoustic sensing on commodity smartwatches, introducing a wide range of novel interaction modalities and use cases. More importantly, these contributions unlock user interface techniques that previously relied on special-purpose hardware. The applications I describe and demonstrate could be deployed to existing smartwatches with an over-the-air update. My evaluations show that ViBand can be accurate, robust to noise, and reliable across users. This has the potential to make smartwatches more useful, and complex interactions on them more practical. In the next chapter, I discuss follow-up work that builds on the findings from ViBand. I discuss techniques that allow the watch to detect fine-grained hand activities, which further unlocks a wide range of health and interactive sensing capabilities.

6. SENSING FINE-GRAINED HAND ACTIVITY WITH SMARTWATCHES

INTRODUCTION

As mentioned in the previous chapter, if computing systems could know the activity of both the body *and* the hands, applications could be more context sensitive and assistive to immediate, ongoing tasks. State-of-the-art activity detection has been largely stuck at ambulatory states (walking, standing, sleeping, etc.) for decades. I envision smartwatches (slowly becoming more pervasive) as a unique beachhead on the body for capturing rich everyday actions. This could unlock many applications, ranging from personal informatics, health and skills assessment, and broadly, context-awareness. For example, a system that knows what your hands are doing can intelligently avoid interruptions. Hand activity detection can also be used to identify the onset of harmful patterns (*e.g.,* repetitive strain injury or hand-arm vibration syndrome), or for building healthy habits (*e.g.,* regular hand washing).

In this work, I show that hand activity can be sensed robustly from a commodity, off-the-shelf smartwatch, without any external infrastructure or instrumentation of objects, opening a new and practical means for achieving this vision. In addition to tracking coarse movement and orientation of the hand, the wrist is also the perfect vantage point to capture bio-acoustic information produced as a byproduct of most hand activities (*e.g.,* typing, brushing teeth). Here, I define bio-acoustics as body-coupled micro-vibrations propagating through the user's arm (see Chapter 5). These signals are inherently diverse, owing to user variance, innumerable tools and accessories, and differences in environment. To overcome this, I developed a processing pipeline that demonstrates surprising robustness, underscoring the feasibility of my approach.

In this work, I also draw an important distinction between ***hand actions*** versus ***hand activities***. Specifically, I define a hand activity as a sustained series of related hand actions, typically lasting seconds or minutes. For example, a single clap would be a hand action, whereas a series of claps would be the activity of clapping. The decision to focus on hand activities was both *practical* (*e.g.,* more data from a continuous signal to enable robust classification) and *functional* (instantaneous hand events are rarely indicative of activity, and offer less opportunities for computational enhancement).

As I will discuss in detail, I started my investigations with a 50 participant, in-the-wild, experience sampling (ESM) study [52]. This yielded a trove of real-world data that informed my machine learning efforts. Secondly, it gave me a working set of how people employ their hands in the modern world, as there were no contemporary "hand ethnographies" to draw upon. I categorized participants' labels and selected 25 routine, yet interesting hand activities (Figures 3-1 and 6-2) to study for a second, in-lab feasibility evaluation. Employing an "obstacle course" methodology [17], I tested my full pipeline, which demonstrated 95.2% classification accuracy. I also ran a series of supplemental experiments to investigate specific questions, such as false positive rejection. Overall, this work demonstrates practical sensing of fine-grained hand activities using just a commodity smartwatch, opening new possibilities for responsive and context-sensitive applications.

RELATED WORK

Hands and their activities are the subject of study in many fields; here I concentrate on prior work directly related to my immediate efforts, with a focus on HCI literature.

Coarse Activity Recognition

Most activity recognition efforts have inferred user-state through worn sensors [17, 33, 209, 153]. These systems are generally constrained to a limited set of coarse, whole-body activities, such as walking, running and bicycling. In general, systems must strike a balance between signal richness and instrumentation unobtrusiveness. Activity recognition systems have seen some success in the market, including Nike+ shoe sensors, Apple Watch, FitBit, and Garmin armbands. These products sense human activity via a combination of inertial measurement units (IMU), heart rate sensors and

GPS. However, these products require explicit selection of pre-planned activities (*i.e.*, user selects a new run session from Nike+) to function reliably.

Fine-Grained Activity Recognition

One approach for fine-grained, human activity sensing is to deploy sensors and tags in the environment. Methods include acoustic monitoring [43, 201], computer vision [271], electromagnetic sensing [213, 299], and tagging objects of interest with markers [157, 205] and sensors [162]. Infrastructure-mediated [47, 85, 93] and general-purpose sensing approaches [149, 142] have attempted room- and building- scale activity recognition. Alternatively, activity sensing can be achieved through worn sensing systems (see [42] for a survey). Wearables with electromagnetic [148], magneto-inductive [277], inertial [178, 179, 180] and acoustic sensing [283] have all been used to recognize activity sensing, including recognition of tool and appliance use. Also, worn cameras, in glasses [251] or on wrists [126, 168, 191], have been used.

Hand Pose and Gesture Detection

Hand pose and motion sensing technologies can also be used to infer a user's context and activity (*e.g.*, typing, playing a musical instrument, grasping a cup). A wide variety of approaches have been demonstrated, including computer vision [126, 191, 285, 67], electromyograph [233], ultrasonics [284], bioacoustics [99], anatomical tomography [302], high-frequency radio [160, 301] and motion sensing [6, 23, 203, 288]. One of the major uses of such sensing is automatic sign language translation [243, 297].

PROOF-OF-CONCEPT SMARTWATCH

As previously noted, a wide array of methods have been considered for detecting hand actions and activities. In contrast to almost all prior work in hand sensing, I purposely selected a commodity platform for my explorations and studies. On one hand, this is constraining, limiting the worn locations and breadth of sensors I can bring to bear on this challenging problem. On the other hand, if successful, it offers an immediate and practical means to achieve my vision; manufacturers could deploy such sensing functionality with little more than a software update. As a proof-of-concept platform, I

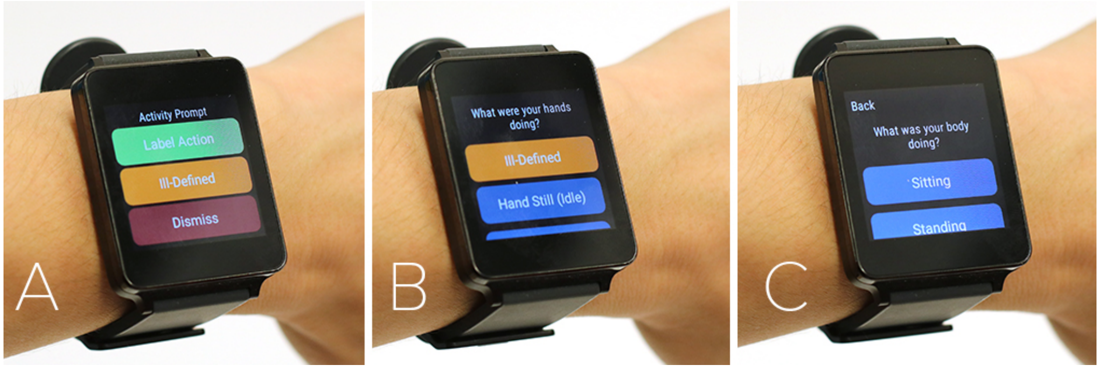


Figure 6-1. The experience sampling watch app. At random intervals, wearers are prompted for activity labels (A), after which they select a hand activities (B,C).

use the smartwatch and high-speed sampling mode identified in the previous Chapter [147]. This is a LG W100 smartwatch running Android Wear (Figure 6-1). By modifying the publicly available kernel [117], it is possible to configure the built-in MPU6515 IMU to stream three-axis accelerometer data at 4kHz [147]. This data stream captures coarse hand movement and orientation, as well as bio-acoustic data (2kHz Nyquist).

POWER CONSUMPTION

In worn systems, with small batteries, it is important to consider how changes in operation will affect battery life. The MPU6515 datasheet details power consumption rates. At 200Hz, power draw is 147µW, while at 4kHz sampling, power draw is 2719µW. While a ~18x difference is substantial, both are small values and it is important to consider it in context. The LG G watch contains a 1520mWh battery, which means the difference of 2572µW consumes <0.2% of battery life per hour.

Harder to estimate is total power load, which includes *e.g.,* waking the main application processor, moving data around in memory, and saving data to persistent storage. As a real-world test, I configured five LG G watches to *continuously* capture high-speed accelerometer. I gave these watches to five participants, who wore them all day, and charged them at night. Over the course of five days, I recorded battery statistics from when the watches were powered on to when they ran out of power.

Across five days and five devices, the average battery life was 7.1 hours (SD=2.5). Given that the application kept the main application processor awake, I believe all day battery life is immediately achievable in a commercial implementation. It is now standard practice in the mobile industry to use low-power coprocessors (*i.e.*, “sensor hubs”) for reading, buffering and processing continuous sensor data (for functions such as step counting, lift to unlock, and spoken keyword detection).

CONTEMPORARY HAND ACTIVITIES

Hands are central to the human experience, and as such, have been the focus of inquiry across many fields, including paleontology and anatomy [292], linguistics [174] and neuroscience [291], to name just a few. Ethnographic work has studied how hands are employed in everything from domestic life to industrial settings [8, 132]. Many hand taxonomies have been proposed, most commonly organized by grip or communication primitives [53], which roughly correlate to functional or expressive uses respectively (see *e.g.,* [68] for a survey of taxonomies). To add to this seminal body of work, I wished to know two key questions:

> 1) *What activities do humans perform with their hands in the modern world?* Armed with such a list, I hoped to focus my technical efforts and better understand how recognition of these activities could be valuable in a computationally-enhanced setting.
>
> 2) *Do different hand activities generate characteristic signals?* In other words, are hand activities distinct and separable? Does a commodity sensor in a smartwatch provide sufficient fidelity to enable robust classification?

Experience Sampling Study

To explore these questions, I sought to collect hand activity data, in the wild, from a random cross-section of participants going about their daily routines. Although retrospective data collection methods (*e.g.,* surveys, interviews) are relatively easy to deploy, they are also subject to self-selection and recall bias [131], especially for something as unexceptional as hand activities. I also considered observational methods, but this was impractical for the scale of deployment I wished to achieve. Instead, I

RANK	HAND ACTIVITY	CATEGORY	COUNT
1	Hands Still / Idle (a) • ‡	atomic	1797
2	Scrolling on Trackpad / Phone (b) • ‡	atomic	615
3	Typing on Keyboard (c) • ‡	atomic	480
4	Swaying (while locomoting) • ‡	atomic	346
5	Typing on Phone (e) ‡	atomic	281
6	Moving/Clicking Mouse (d) ‡	atomic	266
7	Eating ‡	compound	241
8	Gesturing (while speaking) ‡	compound	236
9	Carrying Object ‡	ambiguous	233
10	Writing (with implement) (i) • ‡	atomic	127
11	Drinking (s) • ‡	atomic	61
12	Cooking ‡	compound	53
13	Steering (while driving) ‡	atomic	38
14	Turning Pages	atomic	32
15	Smoking ‡	ambiguous	25
16	Washing Hands (x) •	atomic	23
17	Exercising (on elliptical) ‡	atomic	19
18	Brushing Teeth (y) • ‡	atomic	19
19	Stocking Items ‡	ambiguous	19
20	Using Hand Tools ‡	compound	9
21	Grasping Bicycle Exercise Machine ‡	atomic	8
22	Playing Piano (f) •	atomic	8
23	Operating Weight Machine ‡	ambiguous	8
24	Sign Language ‡	compound	7
25	Washing Dishes	compound	7
26	Putting on Clothes	compound	7
27	Showering	compound	5
28	Dancing	compound	5
29	Cleaning	compound	4
30	Putting Away Clothes	compound	4
31	Brushing Hair (g) •	atomic	4
32	Folding Napkins	ambiguous	4
33	Scratching (o) •	atomic	3
34	Doing Makeup	compound	3
35	Using Scissors (j) •	atomic	3
36	Pushing	ambiguous	2
37	Operating Microscope	compound	2
38	Petting (m) •	atomic	2
39	Drying Hair	ambiguous	2
40	Using Remote / Game Controller (l) •	atomic	2
41	Clapping (n) •	atomic	2
42	Folding Clothes	compound	2
43	Opening Door (p) •	ambiguous	2
44	Closing Door	ambiguous	2
45	Reaching for Object	ambiguous	2
46	Giving Massage	compound	2
47	Tying Shoes	atomic	2
48	Adjusting Watch	ambiguous	2
49	Kickboxing	compound	2
50	Operating Hand Drill (k) • ‡	atomic	2
51	Pilates	compound	2
52	Wiping (cleaning) (v) •	atomic	2
53	Selecting Clothes	compound	1
54	Exercising	compound	1
55	Shaving	ambiguous	1
56	Tying Hair	compound	1
57	Counting Cash	ambiguous	1
58	Holding Phone (on call)	atomic	1
59	Grating (food) (t) •	atomic	1
60	Chopping Vegetables (u) •	atomic	1
61	Using Spoon (eating)	atomic	1
62	Using Knife (eating)	atomic	1
63	Yoga	compound	1
64	Washing Utensils (w) •	atomic	1
65	Scrubbing Counter	atomic	1
66	Operating Vacuum	atomic	1
67	Putting on Lotion	ambiguous	1
68	Stretching	ambiguous	1
69	Searching Pocket	ambiguous	1
70	Screwing Bolt	atomic	1
71	Opening Bottle	ambiguous	1
72	Opening Jar (q) •	atomic	1
73	Operating Scanner	compound	1
74	Putting on Jacket	ambiguous	1
75	Grooming Beard	ambiguous	1
76	Shifting Gears (while driving)	atomic	1
77	Tapping Screen (e) •	atomic	1
78	Pouring Drink (r) •	atomic	1
79	Blowing Nose	ambiguous	1
80	Playing Tennis	compound	1
81	Sorting Paper	compound	1
82	Lifting Free Weights ‡	ambiguous	1
83	Putting on Chapstick / Lipstick	atomic	1
		Total Count	**5065**

Table 6-1. All hand activities captured and labeled during the experience sampling study (50 users, 950 worn hours). ‡ denotes activities that were pre-populated on the deployed watches (i.e., not entered manually by users). Bulleted (•) items were incorporated into the subsequent obstacle course user study (letter key also used in Figures 3-1 and 6-2).

employed an experience sampling method (ESM) [52], which reduces biases by collecting data in situ [36]. Using a fleet of ten smartwatches, I deployed a custom application to 50 participants over the course of two weeks. I used a participant pool drawn from the local population to cover a variety of ages, genders and professions (25 female, mean age of 26.3).

The smartwatches ran a custom background application that I developed (Figure 6-1). After a random sleep interval between 7 and 15 minutes, the application surreptitiously captures ten seconds of accelerometer data. The app then activates the screen and vibration motor to catch the wearer's attention. A simple labeling interface is displayed. The initial screen offers three options: ignore the prompt, mark the activity as ill-defined (*e.g.,* indistinct, between activities), or proceed to label the hand activity. Selecting either of the first two options causes the application to return to sleep. If "*label activity*" is selected, the next screen asks: "*what were your hands doing?"* A pre-populated list of activities is provided, as well as the ability to add custom labels (using a companion smartphone application for ease of typing), which are added to the list for future use. If no user input was received on any screen for more than 30 seconds, the application returns to sleep.

Before deployment, participants completed a one-hour setup and orientation. The pre-populated hand activity labels were reviewed for understanding. Participants could also add additional labels as they desired. Participants also specified when they did not wish to be disturbed by the experiment (*e.g.,* 10pm – 8am). Following this orientation, participants wore the smartwatch for two days on their dominant arm (removed at night for recharging). Participants were paid $10 per day, plus $0.25 per label, up to a maximum of $15 on top of the base pay (*i.e.*, max $25 per day). The study concluded with a 30-minute open-ended interview. Participants often elaborated on hand activities they noticed but were never captured by the watch's random sampling interval.

Results

Cumulatively, our watches were deployed for 100 days (950 worn hours), during which time they captured 5830 instances. Of these, 765 instances (13.1%) were labeled as ill-defined. The remaining 5065 instances contained 120 unique labels. To regularize participant labels, a pair of human coders used a consensus merging scheme. For example, "hand in pocket" and "hand on hips" were ultimately merged into a unified "hand still" label, which is the fundamental hand activity. This reduced the number of unique labels to 83, provided in Table 6-1.

The insights and implications from the experience sampling study were multifold. Foremost, it confirmed my assumption that human hands engage in an incredible diversity of activities. However, a few activities dominate: 35% of the labels are of the hands still or idle, and the next 4 most frequent labels are more common than the remaining 78 hand activities combined. I believe this bodes well for hand activity sensing, as detecting a small class of common actions could encompass most hand activities over the course of a day (an easier classification problem). However, it is also apparent there is an extraordinarily long tail of less frequent activities. Some of these may be rare, but others may be common and are simply short in duration, so as to be infrequently captured by the random sampling method.

I also found many labels that participants did not decompose into *atomic* hand activities. By atomic, I mean events that cannot be broken down into distinct stages. For example, eating and cooking were common labels, but these are *compound* activities that encompass a variety of atomic hand activities (*e.g.,* washing, chopping, mixing). The coders also encountered labels that were *ambiguous.* For example, "open bottle" might mean twisting a cap or using a bottle opener, which I view as distinct activities (though with a similar goal). These categorizations are seen in Table 6-1.

Obviously, this result is just a small window onto the diverse landscape of hand activities, and much future work remains to be done in both HCI and beyond. Nonetheless, this result was sufficient to ground my assumptions and guide subsequent technical efforts.

HAND ACTIVITY CLASSIFICATION

Informed by the findings from the experience sampling study, I proceeded to build a hand activity sensing pipeline for evaluation. This is comprised of three key stages: sensing, signal processing, and machine learning.

SENSING

As mentioned earlier, my software for the LG G watch captures both gross orientation and movement of the hands, as well as higher-fidelity, bio-acoustic information resulting from hand activities. A dedicated background process reads IMU data and fills three, 256-length circular buffers with accelerometer readings (X, Y and Z axes) at 4kHz. These buffers are sent to a laptop over Bluetooth at ~16 FPS, which maintains an even larger buffer and performs additional processing operations.

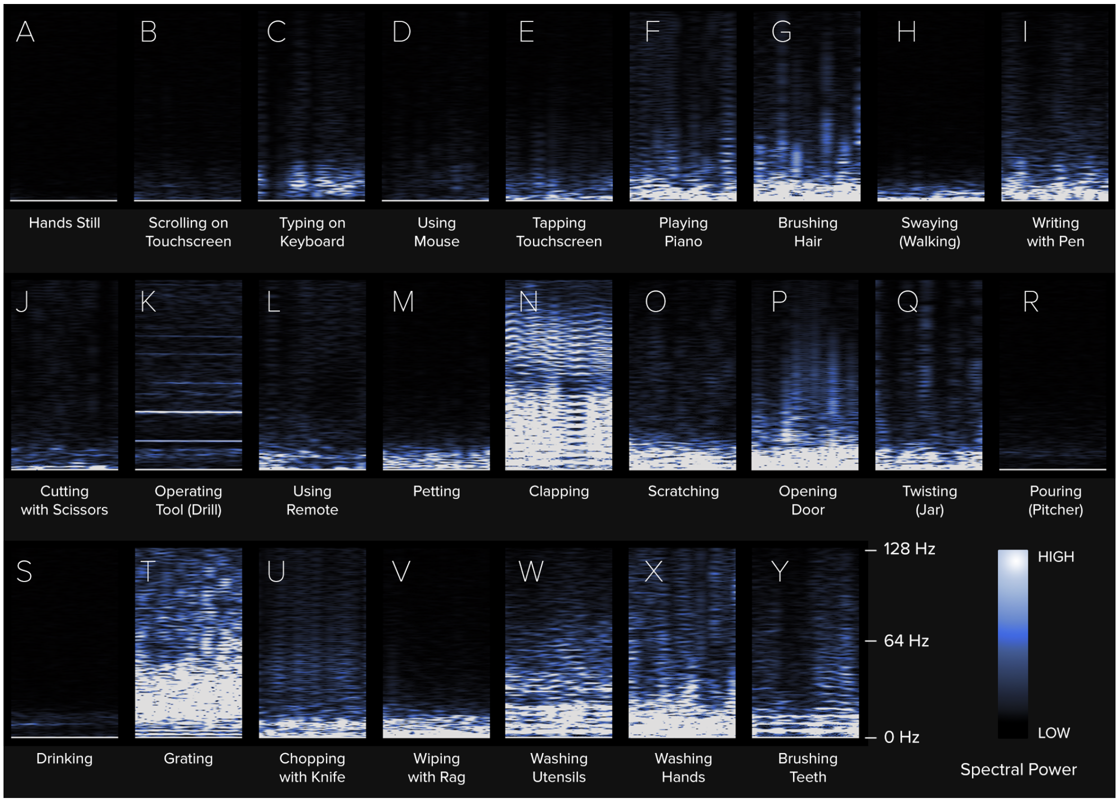


Figure 6-2. Example spectrograms of the 25 hand activities used in the obstacle course study (max of accelerometer axes shown). Y-axis is spectral power from 0 to 128 Hz. X-axis is time (3 seconds). Photos of these hand activities are shown in Figure 3-1, while Table 6-1 offers an estimation of activity frequency.

SIGNAL PROCESSING

A sampling rate of 4kHz in combination with a large buffer (8192 samples) allows our system to compute very high resolution Fourier transforms (4096 bins with a 0.5Hz resolution) within a short period – just over two seconds worth of data, which is about how fast hands transition to new activities. We utilize only the lower 256 FFT bins representing frequencies from 0-128Hz, which we found best characterized most human activities in our ESM study. Finally, these 256 bins are saved into a 48-frame rolling spectrogram, representing ~3 seconds of data (Figure 6-2). These spectrograms are maintained for each of the three accelerometer axes.

MACHINE LEARNING

The next stage of the pipeline is extracting and modeling patterns from the signal. From my experiments, I noticed that important spatial-temporal relationships are encoded in the accelerometer's three axes. For instance, when wiping a table, the Z-axis is mostly unperturbed (chiefly bio-acoustic noise resulting from friction, but little coarse motion), while the X and Y channels experience low frequency oscillations as the hand slides on the surface, often in a linear or circling motion. Indeed, I found many hand activities generated similarly distinctive activation patterns, which can be automatically learned with sufficient data.

To learn from the data, I leverage a convolutional neural network (CNN) architecture [7]. Specifically, I use a variant of VGG16 [246] with modified fully connected layers (Figure 6-3, last two layer sizes set to 2000 and 500). CNNs have been widely used for visual datasets (*i.e.*, width × height × color channel), and in my case, I represent hand activities as spatio-temporal patches of bio-acoustic data. Specifically, I stack accelerometer spectrograms as 256 frequency bins × 48 frames × 3 orientation channels, which serves as input to the CNN. Because of the strongly coupled nature of these channels, this setup forces the architecture to learn cross-axis relationships.

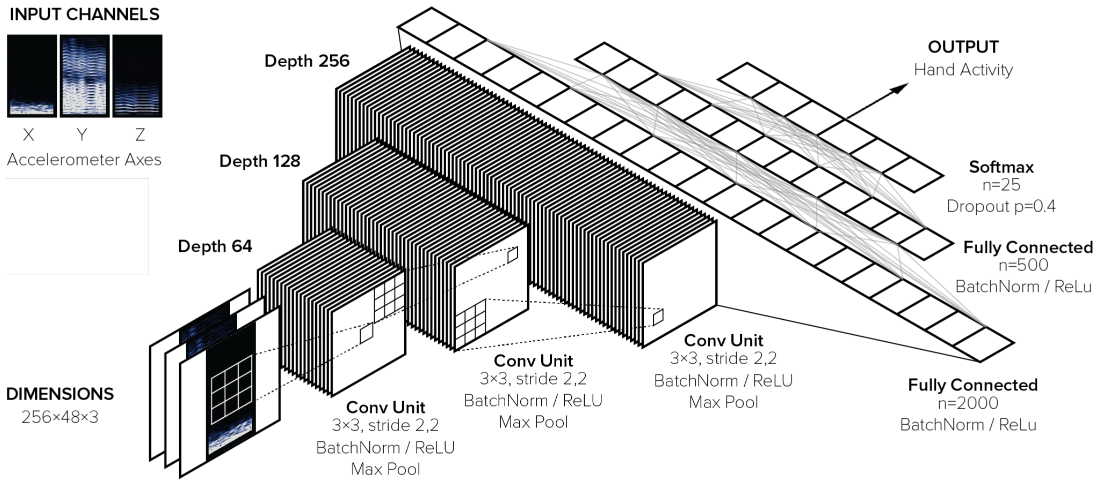


Figure 6-3. Convolutional neural network (CNN) architecture, comprised of several convolutional units (three shown here), two fully connected layers, a dropout layer, and a softmax. I also apply batch normalization between non-linear layers.

Each convolutional unit in the VGG16 architecture is comprised of four sub-layers: 1) the convolutional operator, 2) a batch normalization layer [118], 3) a rectified linear unit (ReLu) activation layer [186], and 4) a pooling layer [235]. I also added a dropout layer [246] to the output of the second fully connected layer (p=0.4) to mitigate overfitting. An illustration of the network architecture is offered in Figure 6-3 (showing 3 of 5 convolutional units). The network was implemented using TensorFlow (tensorflow.org) and Keras (keras.io).

EVALUATION

To quantify the feasibility and robustness of my hand activity classifier, I conducted a second user study. To properly validate the system, a reliable ground truth was needed. Because of the unsupervised nature of our earlier experience sampling study, it was not possible to use that dataset for an accuracy evaluation (though I use it to study false positives, described later). Instead, I employed an "obstacle course" methodology [17] – a technique that has been reliably used in past research to provide ground truth data collection, while emulating natural activities and settings. For this, I selected 25 atomic hand activities (Figures 3-1 and 6-2) from classes identified in my experience sampling study (Table 6-1, bulleted items). I dropped several frequent hand activities that were impractical to capture experimentally, including as eating, cooking, and steering a vehicle. I integrated the final hand activity set into a series of physical tasks that participants completed while wearing the smartwatches.

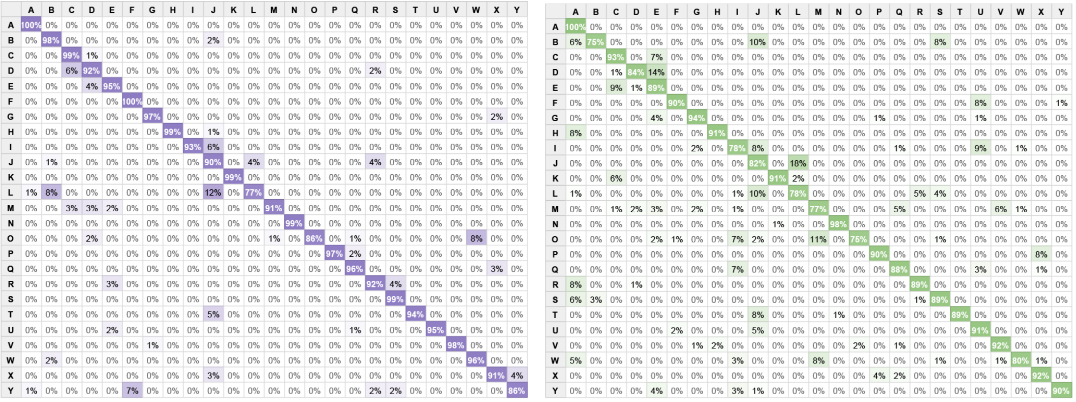
I recruited 12 people from a public participant pool (9 female, mean age 26.6), who were compensated $20 for the 90-minute study. Participants were asked to wear the smartwatch on their dominant arm. Once comfortable, the “obstacle course” began. Each “lap” of the course consisted of visiting four stations with physical activities that incorporated the 25 hand activities (random order). Participants performed each hand activity for at least 15 seconds, and they were free to perform them however they saw fit, capturing natural user variation.

In total, participants completed four laps of our course, with three-minute breaks in between. This ensured temporal separation between data collection rounds. Additionally, in between laps three and four, participants were asked to remove and then re-wear the smartwatch, again to capture variation and to mitigate overfitting (common in worn sensing systems). A trained observer labeled data using a laptop interface immediately after each hand activity was performed. This process yielded 2500 labeled instances per session, per user, resulting in a total of 120K instances.

RESULTS

Per-User Accuracy

To assess whether accelerometers provide sufficient information power to distinguish between dozens of hand actions, I trained a model using data from laps one and two, and tested it with data collected from lap three. Across all participants and 25 hand activities, the system achieved a mean per-user accuracy of 95.2% (SD=4.1, max=98.8%, chance=4%). See Figure 6-4 (left) for the confusion matrix.



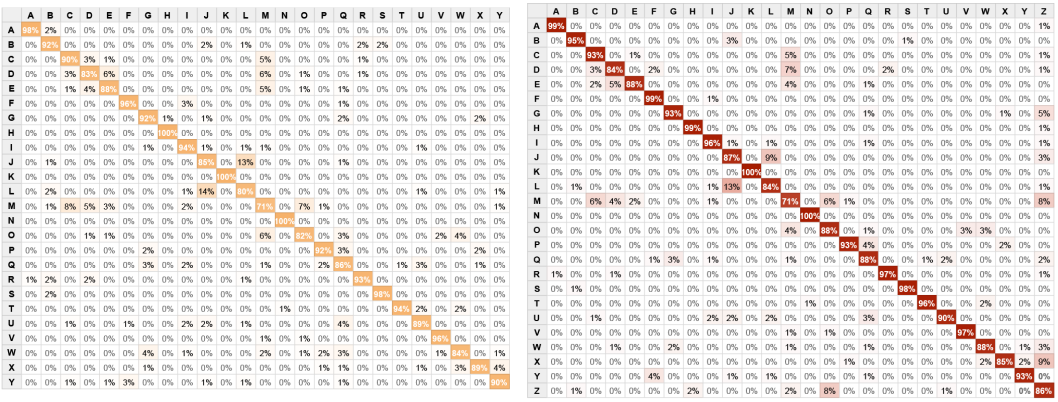
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
A	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
B	0%	98%	0%	0%	0%	0%	0%	0%	0%	2%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
C	0%	0%	99%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
D	0%	0%	6%	92%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	2%	0%	0%	0%	0%	0%	0%	0%
E	0%	0%	0%	4%	95%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
F	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
G	0%	0%	0%	0%	0%	0%	97%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	2%	0%
H	0%	0%	0%	0%	0%	0%	0%	99%	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
I	0%	0%	0%	0%	0%	0%	0%	0%	93%	6%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
J	0%	1%	0%	0%	0%	0%	0%	0%	0%	90%	0%	4%	0%	0%	0%	0%	0%	4%	0%	0%	0%	0%	0%	0%	0%
K	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	99%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
L	1%	8%	0%	0%	0%	0%	0%	0%	0%	12%	0%	77%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
M	0%	0%	3%	3%	2%	0%	0%	0%	0%	0%	0%	0%	91%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
N	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	99%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
O	0%	0%	0%	2%	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	86%	0%	1%	0%	0%	0%	0%	0%	8%	0%	0%
P	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	97%	2%	0%	0%	0%	0%	0%	0%	0%	0%
Q	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	96%	0%	0%	0%	0%	0%	0%	3%	0%
R	0%	0%	0%	0%	3%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	92%	4%	0%	0%	0%	0%	0%	0%
S	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	99%	0%	0%	0%	0%	0%	0%
T	0%	0%	0%	0%	0%	0%	0%	0%	0%	5%	0%	0%	0%	0%	0%	0%	0%	0%	0%	94%	0%	0%	0%	0%	0%
U	0%	0%	0%	0%	2%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	95%	0%	0%	0%	0%
V	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	98%	0%	0%	0%
W	0%	2%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	96%	0%	0%
X	0%	0%	0%	0%	0%	0%	0%	0%	0%	3%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	91%	4%
Y	1%	0%	0%	0%	0%	7%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	2%	2%	0%	0%	0%	0%	0%	86%

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
A	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
B	6%	75%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	8%	0%	0%	0%	0%	0%	0%
C	0%	0%	93%	0%	7%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
D	0%	0%	1%	84%	14%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
E	0%	0%	9%	1%	89%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
F	0%	0%	0%	0%	0%	90%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	8%	0%	0%	0%	1%
G	0%	0%	0%	0%	4%	0%	94%	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	0%	1%	0%	0%	0%	0%
H	8%	0%	0%	0%	0%	0%	0%	91%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
I	0%	0%	0%	0%	0%	0%	2%	0%	78%	8%	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	9%	0%	1%	0%	0%
J	0%	0%	0%	0%	0%	0%	0%	0%	0%	82%	0%	18%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
K	0%	0%	6%	0%	0%	0%	0%	0%	0%	0%	91%	2%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
L	1%	0%	0%	0%	0%	0%	0%	0%	1%	10%	0%	78%	0%	0%	0%	0%	0%	5%	4%	0%	0%	0%	0%	0%	0%
M	0%	0%	1%	2%	3%	0%	2%	0%	1%	0%	0%	0%	77%	0%	0%	0%	5%	0%	0%	0%	0%	6%	1%	0%	0%
N	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	0%	98%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
O	0%	0%	0%	0%	2%	1%	0%	0%	7%	2%	0%	0%	11%	0%	75%	0%	0%	0%	1%	0%	0%	0%	0%	0%	0%
P	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	90%	0%	0%	0%	0%	0%	0%	0%	8%	0%
Q	0%	0%	0%	0%	0%	0%	0%	0%	7%	0%	0%	0%	0%	0%	0%	0%	88%	0%	0%	0%	3%	0%	0%	1%	0%
R	8%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	89%	0%	0%	0%	0%	0%	0%	0%
S	6%	3%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%	89%	0%	0%	0%	0%	0%	0%
T	0%	0%	0%	0%	0%	0%	0%	0%	0%	8%	0%	0%	0%	1%	0%	0%	0%	0%	0%	89%	0%	0%	0%	0%	0%
U	0%	0%	0%	0%	0%	2%	0%	0%	0%	5%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	91%	0%	0%	0%	0%
V	0%	0%	0%	0%	0%	0%	1%	2%	0%	0%	0%	0%	0%	0%	2%	0%	1%	0%	0%	0%	0%	92%	0%	0%	0%
W	5%	0%	0%	0%	0%	0%	0%	0%	3%	0%	0%	0%	8%	0%	0%	0%	0%	0%	1%	0%	0%	1%	80%	1%	0%
X	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	4%	2%	0%	0%	0%	0%	0%	0%	92%	0%
Y	0%	0%	0%	0%	4%	0%	0%	0%	3%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	90%

Figure 6-4. Left: Per-user-trained model confusion matrix. Mean accuracy is 95.2% across 25 activities and 12 users. Right: Post-watch removal confusion matrix. Mean accuracy is 88.3% across 25 activities and 12 users.

Accuracy Post-Removal

Too often, worn sensing systems are trained (or calibrated) and then tested having never been removed from the user. This is artificial, as most wearables are removed daily. Owing to placement sensitivity for most worn sensors, it also tends to lead to artificially



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
A	98%	2%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
B	0%	92%	0%	0%	0%	0%	0%	0%	0%	2%	0%	1%	0%	0%	0%	0%	0%	2%	2%	0%	0%	0%	0%	0%	0%
C	0%	0%	90%	3%	1%	0%	0%	0%	0%	0%	0%	0%	5%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%
D	0%	0%	3%	83%	6%	0%	0%	0%	0%	0%	0%	0%	6%	0%	1%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%
E	0%	0%	1%	4%	88%	0%	0%	0%	0%	0%	0%	0%	5%	0%	1%	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%
F	0%	0%	0%	0%	0%	96%	0%	0%	3%	0%	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%
G	0%	0%	0%	0%	0%	0%	92%	1%	0%	1%	0%	0%	0%	0%	0%	0%	2%	0%	0%	0%	0%	0%	0%	2%	0%
H	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
I	0%	0%	0%	0%	0%	0%	1%	0%	94%	1%	0%	1%	1%	0%	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	0%
J	0%	1%	0%	0%	0%	0%	0%	0%	0%	85%	0%	13%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%
K	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
L	0%	2%	0%	0%	0%	0%	0%	0%	1%	14%	0%	80%	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	1%
M	0%	1%	8%	5%	3%	0%	0%	0%	2%	0%	0%	0%	71%	0%	7%	1%	0%	0%	0%	0%	0%	0%	0%	0%	1%
N	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
O	0%	0%	0%	1%	1%	0%	0%	0%	0%	0%	0%	0%	6%	0%	82%	0%	3%	0%	0%	0%	0%	2%	4%	0%	0%
P	0%	0%	0%	0%	0%	0%	2%	0%	0%	0%	0%	0%	0%	0%	0%	92%	3%	0%	0%	0%	0%	0%	0%	2%	0%
Q	0%	0%	0%	0%	0%	0%	3%	0%	2%	0%	0%	0%	1%	0%	0%	2%	86%	0%	0%	1%	3%	0%	0%	1%	0%
R	1%	2%	0%	2%	0%	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	93%	0%	0%	0%	0%	0%	0%	0%
S	0%	2%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	98%	0%	0%	0%	0%	0%	0%
T	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	94%	2%	0%	2%	0%	0%
U	0%	0%	1%	0%	0%	1%	0%	0%	2%	2%	0%	1%	0%	0%	0%	0%	4%	0%	0%	0%	89%	0%	0%	0%	0%
V	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	1%	0%	0%	0%	0%	0%	0%	96%	0%	0%	0%
W	0%	0%	0%	0%	0%	0%	4%	0%	1%	0%	0%	0%	2%	0%	1%	2%	3%	0%	0%	0%	0%	1%	84%	0%	1%
X	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%	1%	1%	0%	0%	0%	1%	0%	3%	89%	4%
Y	0%	0%	1%	0%	1%	3%	0%	0%	0%	1%	0%	1%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%	90%

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	99%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%
B	0%	95%	0%	0%	0%	0%	0%	0%	0%	3%	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%
C	0%	0%	93%	0%	1%	0%	0%	0%	0%	0%	0%	0%	5%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%
D	0%	0%	3%	84%	0%	2%	0%	0%	0%	0%	0%	0%	7%	0%	0%	0%	0%	2%	0%	0%	0%	0%	0%	0%	0%	1%
E	0%	0%	2%	5%	88%	0%	0%	0%	0%	0%	0%	0%	4%	0%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%
F	0%	0%	0%	0%	0%	99%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
G	0%	0%	0%	0%	0%	0%	93%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	0%	1%	0%	5%
H	0%	0%	0%	0%	0%	0%	0%	99%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%
I	0%	0%	0%	0%	0%	0%	0%	0%	96%	1%	0%	1%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%	1%
J	0%	0%	0%	0%	0%	0%	0%	0%	0%	87%	0%	9%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	3%
K	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
L	0%	1%	0%	0%	0%	0%	0%	0%	1%	13%	0%	84%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%
M	0%	0%	6%	4%	2%	0%	0%	0%	1%	0%	0%	0%	71%	0%	6%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%	8%
N	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
O	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	4%	0%	88%	0%	1%	0%	0%	0%	0%	3%	3%	0%	0%	0%
P	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	93%	4%	0%	0%	0%	0%	0%	0%	2%	0%	0%
Q	0%	0%	0%	0%	0%	1%	3%	0%	1%	0%	0%	0%	1%	0%	0%	0%	88%	0%	0%	1%	2%	0%	0%	0%	0%	2%
R	1%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	97%	0%	0%	0%	0%	0%	0%	0%	1%
S	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	98%	0%	0%	0%	0%	0%	0%	0%
T	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	96%	0%	0%	2%	0%	0%	0%
U	0%	0%	1%	0%	0%	0%	0%	0%	2%	2%	0%	2%	0%	0%	0%	0%	3%	0%	0%	0%	90%	0%	0%	0%	0%	0%
V	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	1%	0%	0%	0%	0%	0%	0%	97%	0%	0%	0%	0%
W	0%	0%	0%	1%	0%	0%	2%	0%	0%	0%	0%	0%	1%	0%	0%	0%	1%	0%	0%	0%	0%	0%	88%	0%	1%	3%
X	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	0%	2%	85%	2%	9%
Y	0%	0%	0%	0%	0%	4%	0%	0%	0%	1%	0%	1%	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%	93%	0%
Z	0%	1%	0%	0%	0%	0%	0%	2%	0%	0%	0%	0%	2%	0%	8%	0%	0%	0%	0%	0%	1%	0%	0%	0%	0%	86%

Figure 6-5. Left: across-user performance confusion matrix. Mean accuracy is 90.7% across 25 activities and 12 users. Right: confusion matrix for a cross-users model with unknown class detection. Using confidence thresholds, global accuracy is 92.2%, while unknown rejection is 86.3%.

impressive results. This experimental effect can be allayed by explicitly including a post-removal collection round, which not only offers for a more realistic estimate of accuracy, but also lets one assess the accuracy drop-off.

Using the same model as before (*i.e.,* trained on laps one and two), I evaluated accuracy using data collected from lap 4 (*i.e.,* post removal). Overall, the system achieved an average accuracy of 88.3% (SD=16.5, max=98.9%, chance=4%). The confusion matrix is offered in Figure 6-4 (right). The 6.9% drop in accuracy from pre-to-post watch removal was much less than I expected and suggests that the signals and approach are fairly robust to placement variation. I strongly suspect that if additional laps of data were collected following a similar watch removal/replacement procedure, accuracy would rebound.

All-Users Accuracy

To answer the central question of whether a commodity smartwatch accelerometer provides sufficient information power to distinguish between a variety of hand activities, I ran a lap-fold, cross validation study. For example, I trained a model on all user data from on laps 1, 2 and 3, and then tested on lap 4 (*i.e.,* 90,000 train, 30,000 test instances). I repeated this process for all lap combinations and averaged the results.

Across all participants, this “all users” model achieves a mean accuracy of 90.7% (SD=2.2, chance=4%). Figure 6-5 (left) shows the confusion matrix.

Leave-One-User-Out Accuracy (Across User)

Finally, I ran a leave-one-user-out analysis to investigate performance across users. Here, data from one participant (laps 1-4) served as a hold-out set, while data from all remaining participants are used for training. I repeat this process for all users and average the results. Across all participants, mean leave-one-out accuracy was 79.2% (SD=6.4, max=84.8%). This is a 10% drop compared to the previous result (90.7%), which simulated a general model seeded with some per-user calibration data (*i.e.,* 1/12th of corpus).

False Positive Rejection

In a worn input system – especially one that is hand-centric – it is vital to consider mechanisms for rejecting false positive events. For this, I take advantage of the per-class confidence scores output from our classifier’s softmax layer. When participants performed a (known) hand activity, the top ranked class had an average confidence of 98.0%, while the second highest ranked class had a mean confidence of 2%. This significant drop-off suggested that confidence could be a good predictor of “unknown-ness”. For example, the software could label events as unknown if the most confident class was below 50%. To identify a reasonable confidence threshold, I ran a simulation using my study data varying this threshold from 0 to 100%. The results, plotted in Figure 6-6 (left), suggest rejecting events when the top-ranked class is under 90% confident, offering a balance between false positive rejection and missed detections.

In addition to the simulation above, I ran another experiment where I trained a model with “negative” example data extracted from the experience sampling study. More specifically, I randomly selected 30K data instances that participants had labeled with hand activities not included in the test set of 25 (which included *e.g.,* driving, smoking and doing makeup). I labeled these diverse instances as an “unknown” hand activity class. Next, I performed a random 80/20 train-test split on the unknown class dataset, and then added this to the all-users model’s train and test datasets. After retraining, the

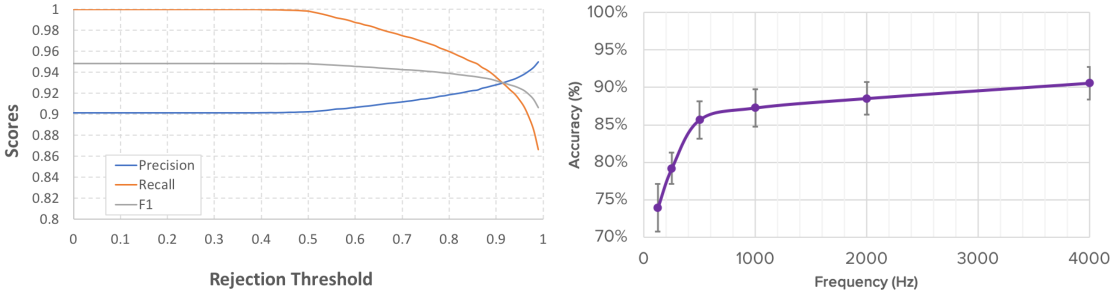


Figure 6-6. Left: precision vs. recall characterizations of the model. I prevent false positive occurrences by setting a confidence value cutoff, but at the expense of missing events. Right: post-hoc analysis of simulated sampling frequency vs. hand activity classification accuracy.

model correctly predicted (*i.e.,* rejected) 76.0% of the unknown activities, while the overall accuracy was 87.9%. If I include the confidence threshold identified in the previous simulation (confidence > 90%), the overall accuracy is 92.2% (with unknown detection of 86.3%). Figure 6-5 (right) provides this confusion matrix; note the confusion along the Z column, which we use for the unknown class.

Finally, I also ran a clustering experiment (t-SNE; based on the top-3 PCA components of our input data; Figure 6-7) to visualize the discriminability of the signals, as it may be possible to employ clustering techniques to mitigate false positives, where events that are “distant” from known hand activity clusters are rejected. This distance-based method could also be used to capture negative example data, or prompt wearers for labels for future recognition.

Sampling Frequency vs. Accuracy

I ran a final post hoc experiment to investigate the effect of accelerometer sampling frequency on classification accuracy. For this, I created downsampled versions of the original 4kHz data to simulate lower sampling rates: 2kHz, 1kHz, 500Hz, 250Hz, and 125Hz. As with the 4kHz data, this was featurized into three-axis, 0-128 Hz, three-second spectrograms. I performed cross-lap validation for each sample rate (train on one round, test on remaining rounds, four rounds total), and combined the results, shown in Figure 6-6 (right). There is a clear, monotonic decrease in accuracy as sample rate decreases, with a marked cliff around 500Hz.

LIMITATIONS

The most immediate limitation of my technique is the need for smartwatches to be worn on the active arm. Most often, this will be a wearer's dominant arm, whereas it is more common for watches to be worn on the passive arm. However, detection still works for two-handed activities such as clapping, washing hands, or typing. Detecting events on the passive arm is an area I plan to explore in future work. I also note that I did not explore simultaneous hand activities, though these appear rare for a single hand.

I also acknowledge that the 25 hand activities I evaluated, though large for a recognition study, are a small fraction of the ways we engage our arms and hands in the real world. As reported earlier, there is an exceptionally long tail of hand actions and activities that will certainly prove challenging to distinguish. Thus, future work should focus on specific activities that can especially benefit from computational support (*e.g.,* contextual aware assistance, smoking secession, elder care, hand-arm vibration syndrome (HAVS), typing RSI). Fortunately, as classifiers become more robust (perhaps through mass adoption of consumer smartwatches), over-the-air updates could unlock recognition of new classes incrementally.

Finally, as discussed earlier, the FFTs use long windows, both to mitigate noise and capture high-resolution spectral data for lower frequencies. As a consequence, this incurs a latency penalty of a few seconds in recognizing events. For extended activities, like eating a meal, where classification might trigger devices to enter a "do not disturb"

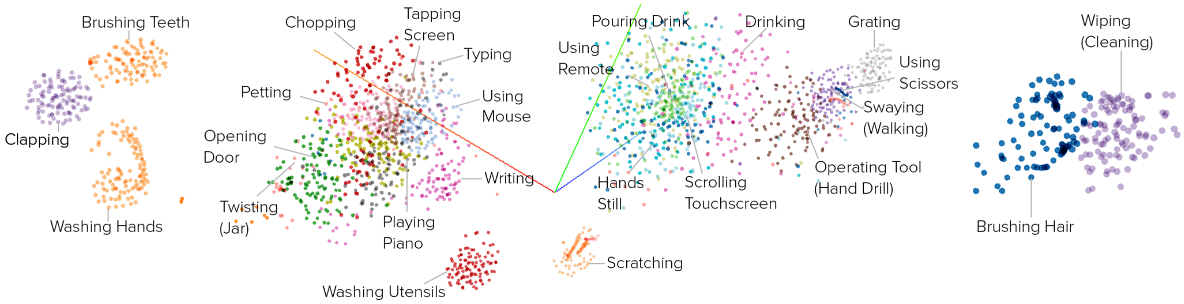


Figure 6-7. Clustering results from a t-SNE dimensionality reduction (perplexity=40, 5000 iterations) on a random subset of our 25 hand activities. Note how less intense activities (e.g., pouring drink, scrolling touch screen) cluster together, while more vigorous hand activities (e.g., scratching and wiping) emerge as distinct groups.

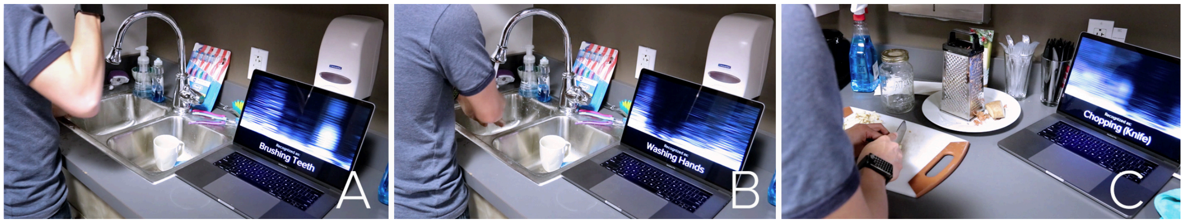


Figure 6-8. Real-time detection of hand activities can unlock many applications, from personal informatics to skills assessment, and richer context-aware applications.

mode, a few seconds of latency is acceptable. However, for short-duration activities, like operating a TV remote control, users will want relevant information to be pulled up quickly (*e.g.*, to make a decision regarding entertainment). Future systems will likely want to use smaller windows with variable class confidence thresholds in order to support hand activities of longer and shorter durations.

EXAMPLE USE DOMAINS

Using commodity smartwatches for hand activity recognition is applicable to a wide range of application scenarios that have been well motivated in prior research. I believe this work points towards a more practical means to feasibility.

One obvious use for fine-grained hand activity sensing is personal informatics. Hand activities are a fine-grained, contextual channel that naturally complements ambulatory state [33]. Fine-grained activity tracing (*e.g.,* washing hands, brushing teeth Figure 6-8AB) has been shown to nudge users towards more healthy lifestyles [16], and spark personal reflection and social facilitation [75]. A system that knows what hands are doing could also have many health-related applications. For example, a smartwatch could track a user's typing behavior to prevent repetitive strain injury (RSI) [25]. Likewise, a smart-watch might be able to track smoking as part of a cessation regime [38], or monitor a construction workers tool use to prevent hand-arm vibration syndrome (HAVS) [258]. Eldercare monitoring systems [54, 211, 258] could also make use of this new and nuanced information source.

There has also been interesting research into automatic skill assessment [124, 252]. Prior work has looked at musical skill acquisition [293], sports performance [184], and rehabilitation [10]. Automatic assessment could create opportunities for in situ feedback and skill-level evaluation [252]. It may even be possible to detect skill degradation overtime, and the onset of motor impairments such as Parkinson's.

Most generally, hand activity recognition could unlock richer, context-sensitive applications [42, 148, 147, 157, 209]. Sequences of fine-grained hand activities could also be used to infer higher-level human activities. For example, filling a kettle, turning on the stove, and then later pouring the kettle, can be indicative of the user making a cup of tea. However, if hand activities occur out of order (e.g., pouring water before boiling it), it could suggest *e.g.,* the onset of dementia. Hand activities could also be valuable in augmenting methods that gauge human interruptibility [80, 112], for example delaying a notification while a user is chopping with a knife (Figure 6-8C).

CONCLUSION

There is great value in knowing what activities the hands are engaged in to support assistive computational experiences. In this paper, I investigated the feasibility of such sensing using commodity smartwatches, which are an immediately practical means for achieving this vision. My explorations started with an in-the-wild deployment and culminated with a controlled lab study. My classification pipeline demonstrates 95.2% accuracy across 25 hand activities, and can reject unknown hand activities at 86.3% accuracy. At a high level, I believe these results bring the promise of contextually responsive applications much closer to reality, especially as this approach requires no external infrastructure or instrumentation of objects.

PART II:

GENERAL-PURPOSE ENVIRONMENT SENSING

7. ENVIRONMENT SENSING

A COMPLEMENTARY APPROACH

Part one of this thesis examined the promise of wearables for making ubiquitous contextual sensing more practical. However, in some cases, user instrumentation (via wrist-worn devices) can create steep requirements that dampen adoption. For example, environments such as classrooms and cafés could benefit from context-aware sensing, but it is unrealistic to expect every user to wear a wrist-worn device. A complementary approach to my previous research thrust is to embed sensors at key probe points within an environment. Substantial prior work exists in this area, which I have represented diagrammatically in Figure 2-3. Along the y-axis is the number of distinct sensed facets (*e.g.*, states and events), while the x-axis is the number of sensors needed to achieve this output (*i.e.,* practicality). A single room can have dozens of complex environmental facets worth sensing (Figure 2-3, y-axis), ranging from "*Is the coffee brewed?*" to "*Is the dishwasher done*?" A single home might have hundreds of such facets, and an office building could have thousands. The cost of hundreds of physical sensors is significant (Figure 2-3, top-right), not including the even greater cost of deployment and maintenance. Moreover, extensively instrumenting an environment in this fashion will almost certainly carry an aesthetic and social cost.

RELATED SYSTEMS

The projects I tackle in this domain intersect with a range of HCI and sensing topics, ranging from special-purpose sensors, distributed sensing, and infrastructure-mediated sensing. Here, I discuss related systems that are relevant to my thesis.

Direct vs. Indirect Sensing

Many of the aforementioned systems I mentioned in Chapter 2 (Background) utilize *direct sensing*, that is, a sensor that physically couples to an object or infrastructure of interest. For example, most window sensors need to be physically attached to a window. This approach is popular as it generally yields excellent signal quality. However, powering such sensors can be problematic, as most objects do not have power outlets. Instead, such systems rely on batteries, which must be periodically recharged [45, 85, 290]. Other systems avoid this by requiring access to a power outlet [92, 93], though this limits possible sensor locations or requires cords be run across the environment—neither of which is desirable.

Fortunately, it is also possible to sense state and events *indirectly*, without having to physically couple to objects. For example, work by Kim and colleagues [127] explored sensing of appliance usage with a sensor installed *nearby*. When an appliance is in different modes of operation (*e.g.*, refrigerator compressor running, interior lights on/off), it emits characteristic electromagnetic noise that can be captured and recognized. Similarly, Ward and colleagues [283] were able to recognize tool use in a workshop through acoustic sensing. Indeed, many sensors are designed for indirect sensing, including non-contact thermometers, rangefinders, and motion sensors.

Overall, *indirect sensing* allows for greater flexibility in placement, often allowing sensors to be better integrated into the environment or even hidden, and thus less aesthetically and socially obtrusive. Ideally, it is possible to relocate to a nearby wall power outlet, eliminating the need for batteries. However, this typically comes at the cost of some sensing fidelity – the further you move away from an object or area of interest, the harder it becomes to sense and segment events. Moreover, some sensors require line-of-sight, which can make some sensor placements untenable.

General-Purpose Sensing

Increasingly, sensor "boards" are being populated with a wide variety of underlying sensors that affords flexible use (Table 2-1). Such boards might be considered general purpose, in that they can be attached to a variety of objects, and without modification, sense many facets. However, this is still ultimately a one-sensor to one-object mapping (*e.g.*, Sen.se's Mother [241]), and thus is more inline with the tenets of a distributed sensing system (*many-to-many*).

Therefore, the *ideal* sensing approach occupies the top-left of my taxonomy (Figure 2-1), wherein *one* sensor can enable *many* sensed facets, and more specifically, beyond any one single instrumented object. This *one-to-many* property is challenging, as it must be inherently *indirect* to achieve this breadth. The ultimate embodiment of this approach would be a single, omniscient sensor capable of digitizing an entire building.

PART II: ROADMAP

In the subsequent chapters, I describe Zensors and Synthetic Sensors in greater detail, and I further discuss how sparse general-purpose can form into *constellations*, enabling practical wide-area contextual sensing. These projects offer a complementary approach to context-driven implicit sensing and interaction, uncovering contributions that further advance research in this space and bringing it ever closer to feasibility.

Chapter 8: Zensors

Computer vision has come closest to achieving the goal of general-purpose sensing because cameras offer rich, indirect data that can be processed through machine learning to yield sensor-like feeds. Despite progress in this area, achieving human-level abstractions and accuracy is a persistent challenge. In Zensors (Chapter 8) [146], I repurpose disused mobile devices and their cameras into "universal" sensors.

More specifically, I fused machine learning with crowdsourcing to provide instant, human-intelligent environmental sensors that end users can set up without specialized training. Users affix a camera to a wall or window (Figure 8-1), launch the Zensors app, use the live camera view to select a region of interest, and supply a natural language question. After this elementary setup, the "sensor" is active and time-series data is immediately accessible.

With Zensors, users can go from question to live sensor feed in under 60 seconds, something that traditional sensing approaches cannot achieve without extensive deployment, calibration, and/or training data. Sensing questions can be diverse, such as "*Is the handicapped parking spot occupied?*", "*How long is the line?*", "Are my plants wilting?", "Are customers wearing their jackets inside?", and "Is the dog bowl empty?" By exposing this functionality as APIs, Zensors enables a wide variety of end-user applications for context-aware environments.

Chapter 9: Synthetic Sensors

While Zensors (and CV-based sensing approaches in general) are powerful, cameras have been widely studied and recognized for their high level of privacy invasion and social intrusiveness, and thus carry a deployment stigma. In Synthetic Sensors (Chapter 9) [149], I achieved much of the same sensing versatility and accuracy as Zensors, but using only low-level sensors.

The versatility of synthetic sensors is driven by a novel "sensor tag" (Figure 9-2) I created, equipped with a suite of denatured low-level sensors that indirectly detect events manifested in an environment. For example, when a faucet is turned on, a nearby sensor tag can pick up the vibrations induced by service pipes installed behind the wall, as well as the characteristic acoustic features of running water (Figure 9-3). Through machine learning, synthetic sensors virtualize sensor data into actionable feeds, powering end-user applications while simultaneously mitigating privacy issues. In my user study, I show that across 38 proof-of-concept synthetic sensors in five locations, spanning seven days, our system achieved an average sensing accuracy of 96.0%

At its most basic abstraction, synthetic sensors operate in a binary fashion (*e.g.,* "Is the faucet running?" Possible outputs: yes or no). I call these *first-order synthetic sensors*. Yet it is possible to build more complex, non-binary, *second-order synthetic sensors* by leveraging first order outputs as new machine learning features, enabling state, count, and duration sensors. Importantly, there is no reason to stop at second-order synthetic sensors. Indeed, first-order and second-order synthetic sensors could feed into third-order synthetic sensors (and beyond), with each subsequent level encapsulating richer and richer semantics.

Chapter 10: Sparse Wide-Area Sensing

Finally, instead of a single general-purpose sensor installed in a single-room, what happens if these sensors work in tandem, perhaps ten sensors in a building? Physical activities that manifest in environments as sound, vibration, illumination, or temperature, are rarely confined to one location. They instead propagate out into adjacent spaces, through openings and doors, across walls, and along structural members. Therefore, it is possible for a general-purpose sensor placed in one location to indirectly detect events nearby. This enables sensor constellations with a sparsity of less than one per room. Furthermore, nearby sensors could offer confirmatory signals, improving accuracy.

In response, I performed a deeper exploration in this space (Chapter 10), and my experiments reveal that it is possible to blanket *e.g.,* an entire home with one sensor per-room, with accuracies reaching up to 96%. Further experiments reveal how different “sensor combinations” can enable different types of accuracy tradeoffs. For instance, I have found that placing a sensor in an adjacent room can boost the detection and recognition of events by ~2%. Moreover, events that are “loud” and “vibrational” can be sensed by the “nearest room only” sensor, but only up to a certain extent.

8. ZENSORS: ADAPTIVE, RAPIDLY DEPLOYABLE, HUMAN-INTELLIGENT SENSOR FEEDS

INTRODUCTION

For decades, "intelligent" environments have promised to improve our lives by inferring context, activity and events in diverse environments, ranging from public spaces, offices, and labs, to homes and healthcare facilities. To achieve this vision, smart environments require sensors, and lots of them. However, installation continues to be expensive, special purpose, and often invasive (*e.g.,* wiring for power).

An even more challenging problem is that sensor output rarely matches the types of questions humans wish to ask. For example, a door opened/closed sensor may not answer the user's true question: *"are my children home from school?"* A restaurateur may want to know how many patrons need their beverages refilled, and graduate students want to know, *"is there free food in the kitchenette?"* Unfortunately, these sophisticated, multidimensional and often contextual questions are not easily answered by the simple sensors we deploy today. Although advances in sensing, computer vision (CV) and machine learning (ML) have brought us closer, systems that generalize across these broad and dynamic contexts do not yet exist.

In this work, I introduce *Zensors*, a new sensor approach that requires minimal and non-permanent sensor installation and provides human-centered and actionable sensor output. To achieve this, I fuse answers from crowd workers and automatic approaches to provide instant, human-intelligent sensors, which end users can set up in under one minute. To illustrate the utility of Zensors, let us return to the restaurant example. John, the proprietor, finds a disused smartphone or tablet and affixes it to the wall of his restaurant. He installs and launches the Zensors app, which uses the front facing camera to provide a live overview of the restaurant. John presses the "new sensor" button and circles the bar countertop, thus specifying a region of interest. John can then enter a plain text question, for example: *"how many drinks are almost empty?"*

By pressing "go", the sensor is activated and starts providing real-time data, in this case numerical. John can now use *e.g.,* a companion app to see a real time visualization of how many drinks need to be refilled, or use an end user programming tool to have the system automatically message a co-worker requesting help if the number exceeds ten. Within a few minutes, John could similarly set up sensors for: *"does table four need to be cleaned?"*, *"are customers wearing their coats inside?"*, *"is a check sitting on the table?"* and other questions relevant to the dining experience.

Unbeknownst to John, his sensors are initially powered by crowd workers interpreting his plain text question, providing immediate human-level accuracy, as well as rich, human-centered abstractions. However, using crowd workers can be costly and difficult to scale, and so ideally it is only used temporarily – answers from the crowd are recorded and used as labels to bootstrap automatic processes. More specifically, the system begins training and testing image-based machine learning classifiers, testing against the ground truth provided by the crowd labels. If the classifiers begin to achieve human-like accuracies, they begin to vote alongside crowd responses. Eventually, if sufficiently robust, the classifiers can take full control.

This human-computer handoff is seamless and invisible to end-users; as far as users like John are concerned, they have a sensor with human-level accuracy from minute one and onward. Even when the classifiers do not achieve the needed level of accuracy, I have designed savings measures into the crowd-based method to conserve costs. Through an API, Zensors enables a variety of applications that help realize the potential of smart environments. This abstracts the complexity of crowdsourcing, computer vision and machine learning, enabling developers to treat "zensors" just as they would traditional electro-mechanical sensors.

I designed Zensors to achieve five key criteria: 1) answer diverse natural language "sensor" questions, 2) with reasonably high-accuracy, 3) while being easy enough to be used by non-experts, 4) requiring zero training, and 5) receive live data within seconds. I am unaware of a single prior computer vision system that achieves these five properties. With Zensors, I make the following contributions:

- A new approach and architecture for hybrid crowd-ML powered sensors, with an API to enable access to sensor data streams.
- A proof-of-concept mobile application for easy, end-user authoring of on-demand intelligent sensors.
- A tool for end-user programming of case-based events that turn sensor output into meaningful actions.
- A study that demonstrates the accuracy and reliability of Zensors in a variety of settings.
- Evidence that our human-powered sensors can be used to train computer vision approaches *in situ*, leading to an automatic handoff in most cases.

RELATED SYSTEMS

Zensors touches on several areas including crowd-driven annotation, computer vision, and more generally, smart environments and activity sensing.

Crowd-Driven Annotation

Crowdsourcing allows systems to access human intelligence through online marketplaces such as Mechanical Turk. For example, the ESP Game asked workers to label images with keywords in order to make the web more accessible [272]. VizWiz [26] asks crowd workers to answer visual questions for blind users with a latency of under a minute. However, both VizWiz and the ESP Game have difficulty scaling because they elicit untyped, natural language responses to non-repeated images, making it difficult for machine learning algorithms to learn patterns with high accuracy.

Marcus *et al.* [169] explored how to perform SQL query estimations that computers cannot do alone, *e.g.,* answer how many people in a database of images were of a certain gender. In their system, users could pre-define functions that specified crowd tasks, which were used to find an answer estimation for query filtering. CrowdDB [84] explored a similar crowdsourcing concept, using the crowd to fill in missing data and overcome the closed-world assumption held in traditional DBs. In both of these examples, the language the user must use to pose queries is a slight variant of SQL, and the crowd's input is used to estimate relationships and missing data over large, pre-existing datasets. In contrast, Zensors turn a live stream of images into an easily accessible structured data stream.

Tohme [98] used the crowd to improve accessibility information in maps by asking workers to label curbs in Google StreetView images. Unlike Zensors, Tohme addressed a single, specific question, using a specialized interface to facilitate labeling. Tohme also included a computer vision component that enabled it to learn how to identify curbs in specific areas over time. They demonstrated that this automated approach could be successful. However it, too, relied on the detailed object segmentation and labeling information obtained from workers through the custom interface (and in combination with GPS and other metadata from Google Maps). Zensors, by contrast, targets general-purpose domains in which no additional information is available, and answers a broad range of user-defined questions.

VATIC [104] uses the crowd to annotate video with labels and object bounding boxes. Glance [150] annotates spans of time with user-requested event labels within minutes by leveraging the ability of large sets of crowd workers to concurrently complete an otherwise time-consuming task. Both VATIC and Glance are approaches for analyzing previously recorded, fixed-length video. In contrast, Zensors is focused on real-time analysis of still images, and offers the potential for handing sensor labeling off from the crowd to machine learning algorithms.

Legion:AR [151] recruits the crowd to provide activity labels using data from live video and RFID tags. It uses an active learning approach to request crowd labels for partial streams of live video. While this solution is related to Zensors, the crowd architecture is different. Legion:AR does not use computer vision, only attempts automatic activity recognition via RFID tags, and collects open-ended questions and answers, not *typed* values (*i.e.,* data type). More specifically, Legion:AR collects open-ended plain text answers, in contrast to the structured labels of Zensors. Further, Legion:AR gathers multiple workers into a single session for a long duration, having them synchronously generate multiple incomplete sets of labels that are merged into a final stream—a configuration that costs tens of dollars for a few minutes. Legion:AR monitors an *instrumented* space, and was not designed to be a general sensing platform.

Image and Video-Based Sensors

The field of computer vision has largely developed in pursuit of extracting meaning from images without the need for human assistance. Today, there exist robust and automatic approaches for applications as diverse as face detection [304] and traffic monitoring [123]. Most related to our present work are video-based, end-user sensing systems, such as the techniques proposed in Crayons [77] and the Eyepatch system [171]. Light Widgets [78] allowed users to create virtual interactive elements on everyday surfaces through video-based sensing. Other systems, *e.g.,* Slit-Tear Visualizations [255], aim to help users better recognize environmental events. These systems rely on users annotating images to provide training data or demarcate regions of interest. Researchers have also explored crowd-based approaches for improving CV techniques [58, 274], which could be used to enhance systems such as Zensors.

COMPARISON TO COMPUTER VISION

I wanted to understand the work requirements and economic implications of building smart environment sensing using existing software engineering approaches. Intuitively, one or more experienced programmers would be able to implement most of my proposed intelligent sensors. But, how long would it take, and at what cost?

I conducted informal interviews with nine freelance developers from oDesk.com, a popular online workplace. To recruit capable candidates, we created a targeted job posting (titled "Image-Based Sensing System"), asking experienced software developers to build a CV-based automatic "bus detection" system. The job posting included illustrative details and clear task requirements (*e.g.,* system should perform >90% accuracy), along with a labeled image set (*e.g.,* a view of a bus stop).

Interested candidates were asked a series of follow-up questions, most notably: "how long … the project will take?", "… hourly rate?" and "… hours will you expect to work per week?" Candidates were encouraged to ask follow-up questions.

Across all users, the average estimated project cost was $3,044 (SD=$2,113), with an average completion time of 4.5 weeks (SD=2.2 weeks), excluding time spent for data collection. While these results are anecdotal—given that none of the developers went

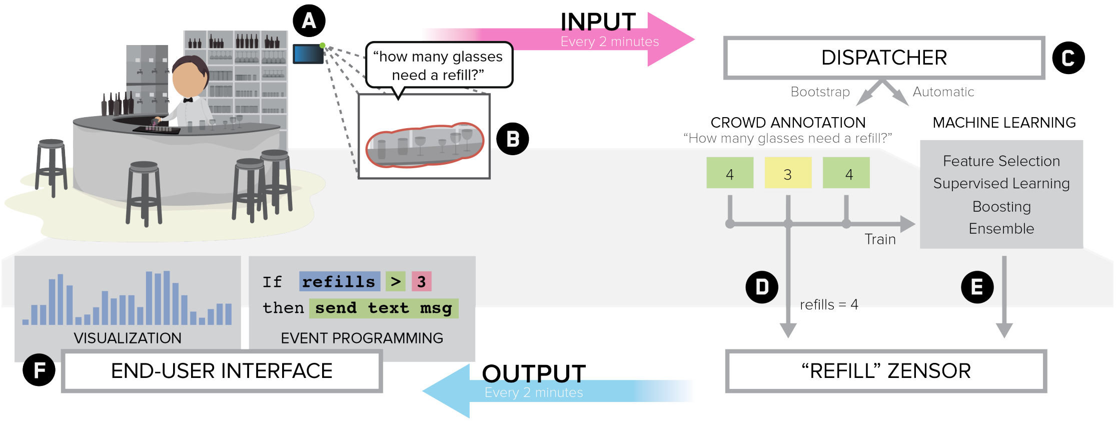


Figure 8-1. Zensors architecture. A bartender repurposes a tablet as a sensor host, affixing it to the wall behind the bar (A). Using the live view from the front facing camera, he selects a region of the scene and asks, "how many glasses need a refill?" (B). Periodically, the device takes snapshots, and forwards this data to a dispatcher (C). Initially, the dispatcher uses crowd workers to power the sensor, providing immediate human-level accuracy (D). In the background, answers from the crowd train a computer-vision-based, machine learning classifier (E). As it approaches crowd-level accuracy, the system employs a hybrid crowd-ML scheme to power the sensor stream. Sensor output can drive end-user applications, such as a real time visualizer (F, left) or event-based end-user programmable system (F, right).

ahead and actually built the system—I do believe it illustrates the significant complexity, cost and development time of special purpose CV-driven sensing systems.

ZENSORS SYSTEM

I now describe how Zensors provides end-users with intelligent sensing capabilities, leveraging both crowd-sourcing and machine learning.

System Architecture

First, a mobile application serves as the primary end-point for creating and modifying sensors. Users highlight a region of the camera image and an associated question, *e.g., "how many parking spots are available?"* As images stream from the device's front-facing camera, the system intelligently decides when to push requests to the crowd.

Next, crowd workers process these requests through a web-based interface. To reduce noise and reject malicious workers, several answers are collected per image, then fused together using quality-control algorithms (*e.g.,* voting) to determine the best response for a given instance. Finally, the responses gathered from the crowd are stored into a back-end database. These responses provide immediate, human-intelligent answers to the questions asked by users, and also serve as a corpus for training computer-vision based, machine learning classifiers.

Sensors from Images

Leveraging cameras as multi-purpose sensors. From mobile phones, security cameras, and Kinects in people's living rooms, cameras are everywhere. They continue to become more powerful, while remaining small. More importantly, time-series data from cameras offers rich contextual information about an activity or environment far more than what basic sensors (*e.g.,* proximity) can provide. One can ask several multi-dimensional questions from camera images across a time period, such as "how many people are smiling?", or "is the table messy?", all of which provide useful information in learning about the context or activity. Thus, the cost, availability, and information bandwidth that cameras offer make them an ideal "multi-purpose" commodity sensor.

Repurposing old mobile devices as sensor hosts. Users upgrade their devices on average once every two years [275]. It is not uncommon for people to have a slew of older smart devices stashed in a drawer or closet. Although older, these devices are capable computers, typically featuring one or more cameras, a touchscreen, and WiFi connectivity. This is the ideal platform for rapidly deployable, image-based sensing. Users simply download the Zensors app onto the devices, which allows them to create or modify sensors. Users then "stick" the device in a context of their choosing.

WiFi Cameras. Zensors can also utilize stand-alone Wi-Fi cameras, costing as little as $30 today. In this case, a web interface can be used to define sensors (Figure 8-3).

Privacy Preservation

Image Subregions. Contextual information from cameras creates an inherent tradeoff between information and privacy [29, 28, 113]. A naïve approach would utilize the entire raw image. However, this can easily violate privacy, especially when personally identifying information is present, or when images depict people in sensitive situations. To partially mitigate this issue, Zensors asks users to select an arbitrarily shaped subregion relating to the question they wish to ask; image data outside the subregion is masked away. This approach helps users to strike a balance between privacy and information content, and likewise, it reduces file size, removes unnecessary image elements, and simplifies the sensing effort for both human raters and CV techniques.

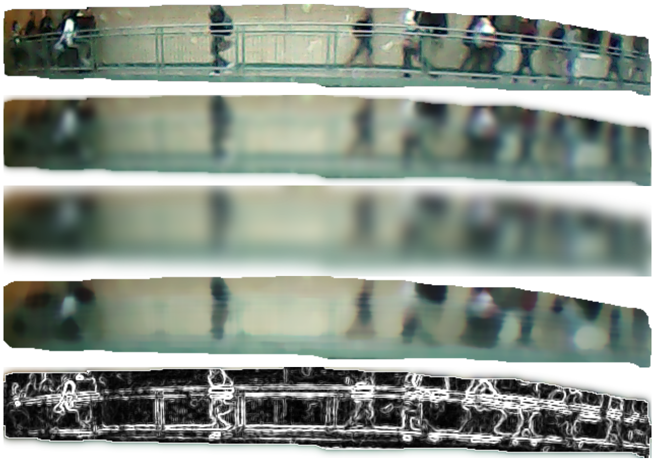


Figure 8-2. In addition to region masking, users select image obfuscation methods if desired. From top to bottom: raw image, light blurring, heavy blurring, median filter, and edge masking.

Image Obfuscation. For users wishing to add an additional level of privacy, Zensors offers several image obfuscation techniques. These are applied to images on the device before they are sent to the cloud. Image obfuscation and privacy has been previously researched, and therefore I integrated the guidelines suggested by Hudson [113] and Boyle [29, 28]. Users can choose to leave the subregion unaltered, or select from four obfuscation methods: light blur, heavy blur, median filter, and edge filter (Figure 8-2).

Creating New Sensors

Sensor Questions. Every sensor starts with a question. Users create a new sensor by selecting a region of the image, and entering a plain text question. For example, in Figure 8-1, the bartender highlights the bar area, and asks, “how many glasses need a refill?” Questions define the “capabilities” of a sensor, and thus, the quality of answers depends on several factors, such as the question’s context and relevance.

Data Types. To further add context and relevance, the system requires users to define a data type when creating new sensors. Data types curb variance and constrain the range of possible values returned by a sensor (*i.e.*, the answer to the sensor's question), and facilitate simple automated processing of the data stream. To this end, I categorize questions into four example data types:

`YesNo` – This type of question can be answered by either *yes* or *no*. It is analogous to an ON/OFF sensor mechanism. Examples include: "is the door open or closed?", "is there food in the kitchen?", or "can you see a bus in this image?"

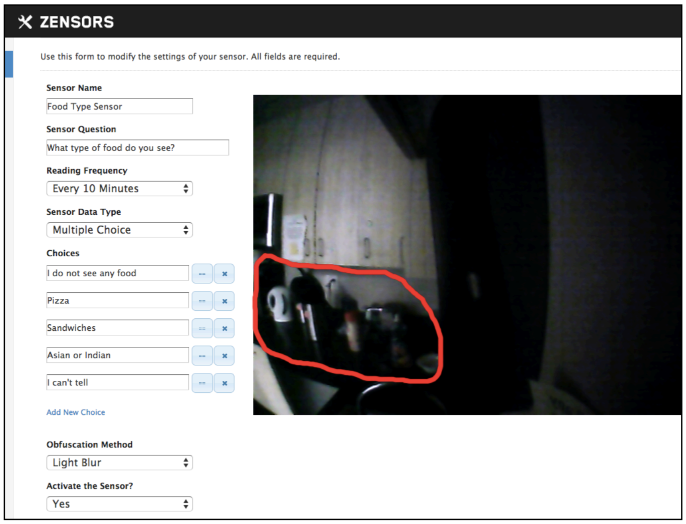
`Number` – Number data types are intended for questions that require counting. Numbers are continuous and are bound between a minimum and maximum range. Examples include: "how many cars do you see in the parking lot? (*min*=0, *max*=30)", and "what percentage of the water tank is full? (*min*=0, *max*=100)"

`Scale` – Scale data types are analogous to Likert-scale questions. Answers belong to discrete values specified within an increasing/decreasing scale. For this data type, users are required to supply scale-value pairs. Examples include: "how messy is this table? (1= Clean, 2=Average, 3= Messy)", or "how happy does the person look? (1=Very Sad, 2=Sad, 3=Neutral, 4=Happy, 5=Very Happy)"

`MultipleChoice` – When creating multiple-choice questions, users are required to specify the list of choices. Unlike scale data types, choice order is irrelevant. Examples include: "what type of food do you see? (None, Salad, Bagels, Other)" and "what are people doing? (reading, using computers, eating, other)".

Frequency. When creating sensors, users need to specify the frequency at which sensor readings are taken. Depending on the question, frequency readings can range from near real-time (*e.g.,* every one or two seconds for questions like "is the refrigerator door open? [`YesNo`]"), to extended periods (*e.g.,* once per day for: "what product is advertised on the billboard? [`MultipleChoice`])."

Web Interface. Along with the mobile application, I built a companion web interface for sensor management (Figure 8-3). Users link one or more sensors to a web account, where they can create, modify, and synchronize sensors across all of their devices. The web UI also makes it possible to create new sensors remotely. For example, users can "stick" a sensor device at an elevated vantage point (*e.g.,* for viewing an entire parking lot), and then manage and create sensors without having to physically touch the device.



Use this form to modify the settings of your sensor. All fields are required.

Sensor Name: Food Type Sensor

Sensor Question: What type of food do you see?

Reading Frequency: Every 10 Minutes

Sensor Data Type: Multiple Choice

Choices:
- I do not see any food
- Pizza
- Sandwiches
- Asian or Indian
- I can't tell

Obfuscation Method: Light Blur

Activate the Sensor?: Yes

Figure 8-3. Users create and synchronize Zensors across all of their devices using a web API, allowing them to manage sensors without physical device interaction.

Similar Image Detection and Rejection

Sensor image streams often have periods of little variation (*e.g.,* buildings after closing hours, outdoor scenes at night). Thus, to avoid soliciting redundant (and thus costly) responses from the crowd on highly similar images, I collapse runs of similar images. I calculate image similarity using a simple technique. First, I count the number of pixels that have changed from the previous frame using their RGB values and a predetermined *pixel difference threshold*. If the number of changed pixels in an image exceeds a certain *image area percentage threshold*, I consider the image to be different. Although this algorithm worked well for my purposes, note that more sophisticated approaches (see *e.g.,* [125]) could be used.

To determine optimal parameters, I performed a brute force optimization experiment. I compiled a corpus of roughly 6000 time-stamped images taken from multiple pilot sensor streams. I then manually labeled whether or not an image was the same as the previous image, providing a ground truth set. I then ran my image similarity algorithm, seeded with all combinations of the following thresholds: 2% to 40% *pixel difference threshold*, in 2% increments, and 0.1% to 5.0% *image area percentage threshold*, in 0.1% increments. This produced 130 result sets, which I compare to the ground truth using Jaccard's distance metric. By using a *pixel difference* and *image area* threshold of 10% and 1.0% respectively, a Jaccard distance of .64 is achieved. On average, this process removes roughly 40% of images – a significant saving.

Sensing with the Crowd

Images streamed from sensor devices are stored in a database. Crowd workers process un-answered instances through a web-based interface seen in Figure 8-4. The interface varies slightly based on the question/response type. Each sensor instance is answered (*i.e.,* labeled) by several different crowd workers; I use voting to determine the best response (at present, I use three workers, but other numbers are possible). The resulting value is then saved in the database and the instance is considered answered and ready for sharing with end users or powering applications.

My goal is to ensure workers are presented with a simple task that they can answer quickly. As such, I present one image to each worker and collect a single response. If workers are unable to answer an image-based question, they can mark it as an exception ("I can't tell" button, Figure 8-4), which informs the system that there is something amiss with the sensor itself (*e.g.,* occlusion, poorly defined question). In addition, workers are prompted to provide textual descriptions when exceptions occur. This approach provides actionable user feedback to help remedy the problem.

To recruit users fast enough to receive answers in real time, I use LegionTools [150], a toolkit for quickly recruiting workers from Mechanical Turk using a web-based interface. It leverages a retainer model [24], which pre-recruits workers so they are ready to respond to a task within as little as two seconds. When sensors are crowd-powered, this sets the lower-bound on our system latency.

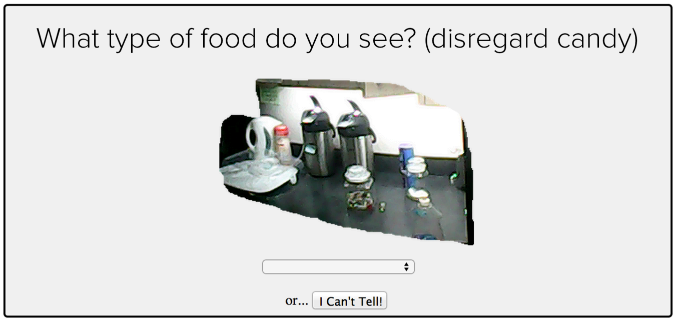

What type of food do you see? (disregard candy)

Figure 8-4. Our Mechanical Turk interface, which lets workers answer the question or raise an exception.

Training Automated Sensors

Solutions using crowd-power alone can be costly and difficult to scale (*i.e.,* more sensors requires more people). Zensors reduces its reliance on crowd workers over time by using crowd-provided answers to train machine learning classifiers, which are fast and inexpensive. However, even after machine learning has taken over processing sensors feeds, crowd workers are still needed to provide a periodic human-accuracy baseline to ensure high accuracy.

The classifier is trained on all the input data to date, except for the most recent full day of data, which is set aside for evaluation. For sensors that produce data infrequently, the test set can be extended to one week or more to ensure there are sufficient test instances. Histogram equalization is applied to each image to reduce the effect of lighting variances. Then, each input image is processed into a large number of global features. Additionally, each sensor image (which is itself a subregion of the larger original image) is broken into a grid of sub-images. In addition to a 1x1 grid (simply the image unchanged), I also use 4x4, 7x7 and 10x10.

Each of these variously-sized sub-images is then converted to a luminance image, and the mean, standard deviation and mean-squared error across the window are used as numerical features. This produces a total of 332 image features. Feature selection is used as a post-process to extract exemplar features for a given sensor feed. Of course, much more advanced computer vision and scene understanding approaches exist that are beyond the scope of this work.

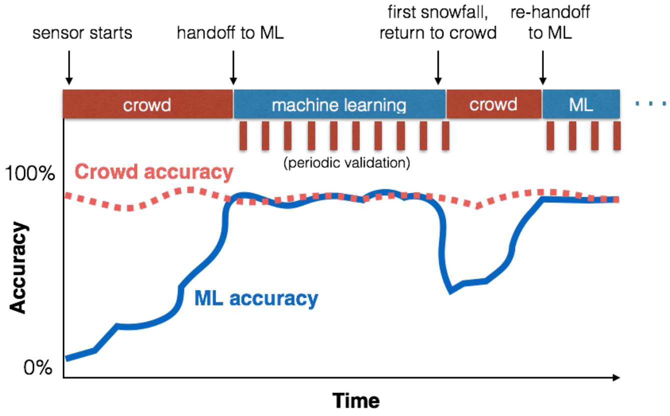


Figure 8-5. Sensors can toggle between the crowd and machine learning to adapt to environment changes. Note that end users and applications only ever see the max of the crowd and ML accuracies.

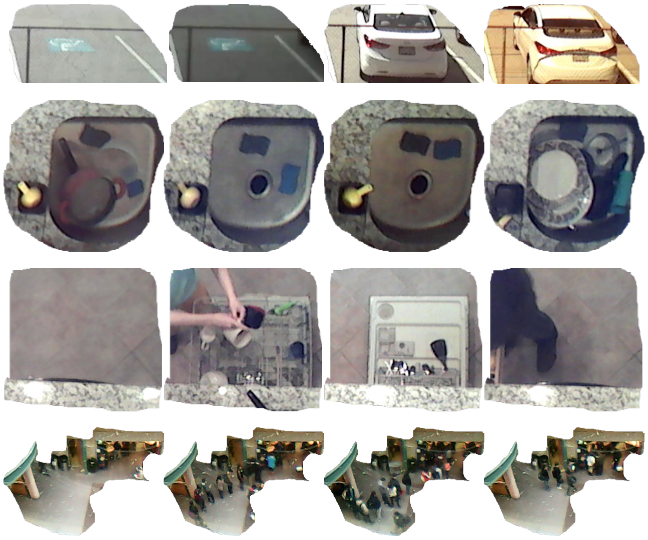


Figure 8-6. Sensor image time series. Associated questions top to bottom “do you see a parked car?”, “how many dishes in the sink?”, “do you see a dishwasher door?”, and “how orderly is the line?”

We use correlation-based feature selection [96] to select features from the training set, coupled with a backtracking best-first attribute searcher. Both algorithms are implemented in the Weka [95] machine learning toolkit. This selection process typically chooses between 10 and 30 features. Of note, the feature sets for different sensors rarely overlap.

I then train classifiers depending on the type of sensor. A “pre-classifier” is first trained to distinguish between exceptions and non-exceptions, to ensure that the main classifier is not polluted by incorrect data. For continuous (numeric or scale) sensors, I train a support vector machine regression classifier using the SMOReg algorithm. For discrete sensors (yes/no, multiple choice), I use a one-versus-one multiclass quadratic-kernel SVM trained with the SMO algorithm, and for simple binary sensors I train a single SVM. The SVM was chosen as the basic classifier because of its ease of training and predictable behavior, though other classification approaches are certainly possible and valid. A more robust version of this system would maintain a library of feature extractors and classification algorithms, selecting those exhibiting the best performance for a given sensor.

Machine Learning Handoff

As the training corpus grows from crowd labeled instances, the accuracy of the machine learning classifiers typically improves. Once the accuracy of the machine learning exceeds a predefined threshold (*e.g.,* 95%) for several days, the sensor hands off classification tasks to the machine learning algorithm. It is also possible to do a soft handoff, where the relative weighting between crowd and machine learning labels shifts over time.

Periodic Ground Truth Validation

To ensure continued sensor accuracy after the handoff to machine learning, the system periodically submits small batches of fresh sensor data to the crowd for labeling. This is used to benchmark the classifier accuracy. If accuracy is sufficiently robust, the machine learning can continue to power the sensor. However, if the accuracy has fallen below a threshold, the system can revert to crowd-power. This serves two immediate purposes: 1) the sensor immediately regains human-intelligence level accuracy, and 2) the system can incorporate new labels in its training for a hopeful future handoff.

In this way, Zensors can automatically handle infrequent changes (such as the first snow fall in a parking lot; Figure 8-5) that would prove challenging for most computer-vision-driven systems (which are first trained and then deployed). This ability to seamlessly toggle between crowd and automatic approaches, without sensor interruption, makes Zensors highly adaptive and robust.

End-User Programming

I built a basic end-user programming tool that lets users design event-based notifications using data from one or more sensors (*e.g.,* "send an email when the stove is ON *and* ZERO people are in the house"). These directives can be chained together as sets of conjunctions (“and” clauses), and “or” clauses for alternative responses. Multiple chains can be defined to represent different configurations. These disjunctions of conjunctions comprise a fully expressive set of logical terms.

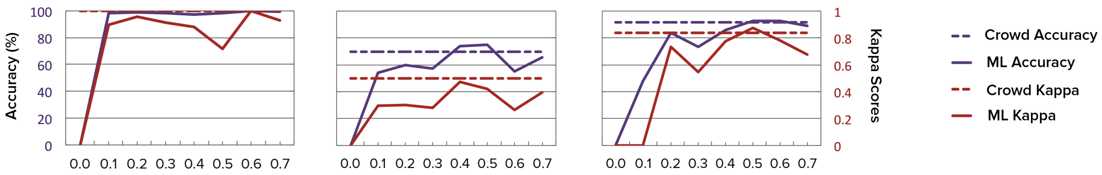


Figure 8-7. “Live” accuracy evaluation of three sensors. This is the accuracy of a sensor over the course of its deployment, as viewed “live” by its operator. X-axes represent training set cutoff time *t*. Left: “do you see a dishwasher door.” Middle: “how messy is the counter.” Right: “which parking spots are occupied.”

This interface works similar to the popular *If This Than That* tool (ifttt.com) – users can select a sensor, select a value and comparison operator, and then select an action type from the set of supported APIs. Our current implementation allows users to select from an email, text message, audio, or chat alert. For each alert type, users are able to define a custom message, which can also display the current value of the sensor by using a specially marked variable.

PROTOTYPE DEPLOYMENT

The goal of our prototype deployment was to illustrate that even with a basic approach, the Zensors architecture can achieve high accuracy, at low cost, quickly, and authorable by end users using plain text queries.

I deployed 16 sensors across four diverse environments: a home kitchen, office kitchenette, building food court, and parking lot (see Figure 8-6 for four examples). Sensor questions ranged from “is this café door open?” to “what type of food is on the counter?” A range of reporting frequencies (from once per minute to twice per hour) and deployment durations (10 days to 3 weeks) were represented by the sample sensor set. I also manually labeled images from seven sensors to create a ground-truth dataset for later experiments. These “expert” labels have the advantage of superior context understanding as well as being able to view the entire dataset, not just a small snapshot.

Accuracy of the Crowd

To analyze how well the sensors can quickly provide accurate sensing data, I measured the precision, recall, and latency of the aggregated crowd responses. Figure 8-7 shows the accuracy of crowd workers' ratings, using expert labels as the ground truth. Cohen's kappa [44] is calculated to mitigate the effects of skewed class distribution (*e.g.,* the leftover food sensor returned "no" over 75% of the time). Crowd accuracy reaches as high as 96.8% (kappa score 0.859), with mean accuracy 77.4% (median 76.0%). The crowd performed very well on three sensors (accessible parking spots occupied, number of cars in parking lot, and dishwasher door), moderately well on one sensor (leftover food), and poorly on three sensors (food type, line length, countertop messy).

The food type sensor required users to distinguish between seven types of cuisine ("I do not see any food", "Pizza", "Sandwiches", "Cake or pastries", "Asian or Indian", "Salad", "Bagels or doughnuts", "Other cuisine or I can't tell") based on a very low-resolution image, while the line length sensor and countertop sensors both involved subjective judgments (*e.g.,* "is the line orderly", "how messy is the countertop"). By contrast, quantitative questions ("is there food here", "is the door closed", "how many cars are there") generally had superior performance.

In designing questions to be posed to the crowd, operators may make assumptions that are not obvious to crowd workers, leading to incorrect results. In one example, workers were asked to identify the presence of food on a kitchen countertop. The countertop has a permanent candy jar, which the experimenters assumed would not be classified as food, yet several crowd workers marked the otherwise-empty countertop as having food. Based on the observed results, the question was amended to explicitly exclude candy, after which the expected results were obtained.

Estimating Live Accuracy

This experiment sought to estimate the accuracy of a sensor, over the course of its deployment, as viewed "live" by its operator. For each sensor, I defined ten time periods each covering one-tenth of the data, numbered t=0.1 through t=1.0. To estimate live accuracy at time t, I trained on all data up to time t, and then tested on all data from time t to time $t+0.3$ (*i.e.,* I tested on a sliding window of 30% of the data). The results for three representative sensors are shown in Figure 8-7, compared against crowd accuracies. In many cases, a relatively small portion of the data is needed to reach crowd-level accuracies.

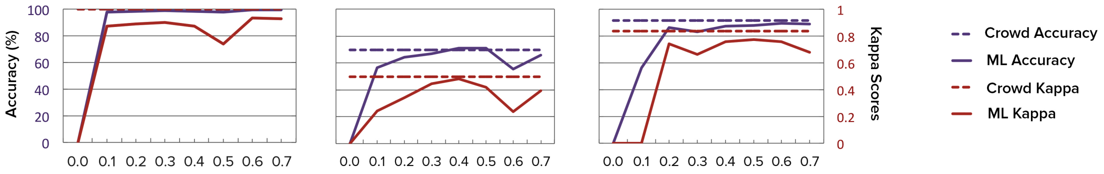


Figure 8-8. “Future” accuracy evaluation of three sensors. This is the accuracy of the sensor assuming ML handoff at time *t*. X-axes represent ML handoff time *t*. Left: “do you see a dishwasher door.” Middle: “how messy is the counter.” Right: “which parking spots are occupied.”

Assessing Future Accuracy Post-Handoff

Alternatively, it is equally important to assess what the accuracy of a sensor would be going forward, assuming a ML handoff occurs at time *t*. To assess this, I simulate a complete ML handoff at each time increment. All data up to that point is used for training, while all future data is used for testing. I stop this analysis when less than 30% of the data is available for testing, to avoid noisy results from insufficient test data. These results are summarized in Figure 8-8, compared against the overall crowd accuracies. The accuracies follow similar curves to the curves of Figure 8-7, suggesting that live classification accuracy may be able to predict future post-handoff classification accuracy.

Similar Image Rejection in Practice

As described previously, the system sends labeling requests to the crowd only when there is a sufficient visual change between consecutive images, otherwise the last sensor value is simply copied forward. In my deployment, I found that the image similarly mechanism rejected an average of 61.2% of images (SD=17.2%, minimum=40.5%, maximum=93.7%).

Sensors that Fail

It is important to acknowledge that some of the sensors failed to produce reliable output. I initially hypothesized that failure would primarily be due to shortcomings in the computer vision implementation. However, I found that the classifiers work well in practice, with six of the seven sensors (for which I had expert labels, and thus a ground truth) getting to within 90% of crowd accuracy when trained on half of the crowd-labeled data (mean 98.1%, SD=14.4%; tested on the second half). Instead, I found that the classification bottleneck for several of the sensors was caused by the poor accuracy of the crowd answers (as compared against our ground truth). For these underperforming sensors, I found a common theme: the sensor questions were subjective or required additional context.

For example, one sensor asked, "how orderly is the line?" (Figure 8-6) with three possible answers: "no people visible in image", "people present, but no obvious organization", and "people standing in a line". Because this is subjective (*e.g.,* relative to local cultural norms) I found that crowd workers provided widely varying answers. Another sensor was tasked with sensing whether a dishwasher was opened or closed (see Figure 8-6 for some example images). In piloting, the question was defined as "is the dishwasher door open?" However, this appeared to confuse crowd workers, reducing sensor accuracy. I hypothesize that this problem was caused by the fact that most of the time, no dishwasher was visible in the closed state. When presented with the question of "is the dishwasher door open?", the crowd presumably wondered "what dishwasher?". I found that rephrasing the question to be more context-free – "Do you see a dishwasher door?" – significantly boosted accuracy.

There are a number of ways to alleviate "bad" sensor questions. One approach is to suggest example questions or provide structured question templates (*e.g.*, "do you see a __ [in/on] the __?"), helping end-users formulate questions with less ambiguity. Additionally, the "I can't tell" button in the crowd interface (see "sensing with the crowd" section) could allow the system to flag sensors causing confusion and suggest the question or image sub-region be modified. Another approach is for the crowd labeling interface to provide exemplar images, one for each possible answer (*e.g.,* show exemplars of both dishwasher door states).

Zensors Economics

The Human Intelligence Tasks (HIT) I used for Zensors paid 2 cents each, a pay rate chosen to be above the U.S. minimum wage even for slower workers. This means, *e.g.,* that a sensor that takes images every 10 minutes would cost roughly $100 per month (assuming an average similar image rejection rate) to be fully human-powered (30 days * 24 hours * 6 images per hour * 40% different images * $0.02 * 3 workers). To offer a concrete example, the median sensor in terms of images captured was the dishwasher door sensor, which triggered every minute. It captured 528 non-similar images over a 7-day deployment, which translates to $135/month in costs assuming only human-power.

Sensor Name / Question	**Freq.**	**CP Cost per Mo.**	**Exp. Cost per Mo.**
Do you see a dishwasher door?	1 min	$135	$35
How messy is the counter?	10 Min	$82	$22
Do you see a parked car?	30 Min	$30	$8
How many dishes in the sink?	10 Min	$28	$7
How many cars do you see?	30 Min	$43	$12
What type of food do you see?	10 Min	$87	$23

Table 8-1. Estimated monthly costs if sensors were fully crowd-powered (CP cost), as well as expected cost (Exp. Cost) assuming ML handoff after week one and continued periodic validation.

For many of the sensors, I found that the automatic classification pipeline could reasonably approximate the crowd's answers using the first week as training data. Once there is sufficient agreement between crowd and machine learning, I can decrease the number of human workers from three to two, and eventually to one (and recruit more workers when there is significant disagreement). This means I can reduce the price by 67% even before the machine learning is fully trained, without reducing accuracy. To get to a point where machine learning can shoulder the entire load, I found that the test sensors took between 90 and 687 data points (depending on polling rate and setting). This means that I can train an automated sensor for as little as $5.40 (and the worst sensor for $41).

One of the strengths of Zensors is its ability to use human-intelligence to handle previously unseen scenarios. As such, even if a handoff to ML is possible, there is a continued cost to validate that the automated system is still working By periodically having the crowd spot-check the output of sensors, the system can detect e.g., errors and scene changes, switching back to human-power if needed. This periodic validation can run at different intensities. Validating e.g., 1 in every 50 instances would cost roughly 1/50th the typical human-powered cost (about a few dollars per month).

When ML Handoff is not Possible

Finally, there may be cases where the system cannot attain a full ML handoff (*e.g.,* poor image resolution, noisy training data, or simply a hard question incompatible with CV approaches). As a result, the system will need to rely on a fully crowd-powered approach for an indefinite period, which can be relatively expensive; Table 8-1 offers some example costs from our deployment. However, even if ML handoff never occurs, end users and applications only ever see human-accuracy level answers.

CONCLUSION

In this chapter, I described and studied Zensors, a system that enables easy end-user creation of arbitrary sensors for any visually observable property. Zensors uses crowd-powered answers to produce near-instant sensor readings with high accuracy while requiring no explicit training. Once enough data labels have been collected, Zensors can seamlessly hand-off image classification to machine learning utilizing computer-vision-derived features. I discussed results from our deployment, which suggest our approach is feasible, accurate and inexpensive.

~

In the next chapter, I describe another project that attempts to answer a complementary question: can we build a system with the general-purpose properties of Zensors, but without a camera? I call this system *Synthetic Sensors*, which uses a portfolio of low-level sensors and machine learning for contextual sensing, all from a single device.

9. SYNTHETIC SENSORS: TOWARDS GENERAL-PURPOSE SENSING

INTRODUCTION

As mentioned in the previous Chapter, smart, sensing environments have long been studied and sought after. Today, such efforts might fall under catchphrases like the "smart home" or the "internet of things", but the goals have remained the same over decades—to apply sensing and computation to enhance the human experience, especially as it pertains to physical contexts (*e.g.*, home, office, workshop) and the amenities contained within. Numerous approaches have been attempted and articulated, though none have reached widespread use to date.

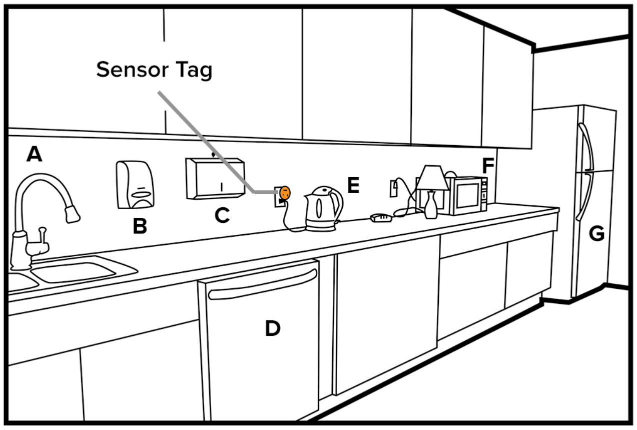


Figure 9-1. An example of general-purpose sensing, where one sensor (orange) enables the detection of many facets, including rich operational states of a faucet (A), soap dispenser (B), paper towel dispenser (C), dishwasher (D), kettle (E), microwave (F) and fridge (G).

One option is for users to upgrade their environments with newly released "smart" devices (*e.g.*, light switches, kitchen appliances), many of which contain sensing functionality. However, this sensing is generally limited to the appliance itself (*e.g.*, a smart light switch knows if it is on or off) or when it serves its core function (*e.g.*, a thermostat sensing occupancy). Likewise, few smart devices are interoperable, thus forming silos of sensed data that thwarts a holistic experience. Instead of achieving a smart home, the best one can hope for—at least in the foreseeable future—are small islands of smartness. This approach also carries a significant upgrade cost, which so far has proven unpopular with consumers, who generally upgrade appliances piecemeal.

To sidestep this issue, we are now seeing aftermarket products (*e.g.*, [70, 73, 130, 170, 190, 241, 218]) and research systems (*e.g.*, [154, 212,61]) that allow users to distribute sensors around their environments to capture a variety of events and states. For example, Sen.se's Mother product [241] allows users to attach "universal" sensor tags to objects, from which basic states can be discerned and tracked over time (*e.g.*, a tag on a coffee machine tracks how often coffee is made). This approach offers flexibility, but at the cost of having to instrument every object of interest in an environment.

As I will discuss, a single room can have dozens of complex environmental facets worth sensing, ranging from "is the coffee brewed" to "is the tap dripping." A single home might have hundreds of such facets, and an office building could have thousands. The cost of hundreds of physical sensors is significant, not including the even greater cost of deployment and maintenance. Moreover, extensively instrumenting an environment in this fashion will almost certainly carry an aesthetic and social cost [19].

A lightweight, general-purpose sensing approach could overcome many of these issues. Ideally, a handful of "*super*" sensors could blanket an entire environment – *one per room or less*. To be *minimally obtrusive*, these sensors should be capable of sensing environmental facets *indirectly* (*i.e.*, from afar) and be *plug and play* – forgoing batteries by using wall power, while still offering omniscience despite potential sub-optimal placement. Further, such a system should be able to answer *questions* of interest to users, abstracting raw sensor data (*e.g.*, z-axis acceleration) into *actionable* feeds, encapsulating human semantics (*e.g.*, a knock on the door), all while *preserving occupant privacy*.

In this Chapter, I describe the structured exploration process I employed to identify opportunities in this problem domain, ultimately leading to the creation of a novel sensing system and architecture that achieves most of the properties described above. First, I provide a comprehensive review of sensors, both academic and commercial, that claim some level of generality in their sensing. Similarly, I conducted a probe into what environmental facets users care to know, and at what level of fidelity has acceptable privacy trade-offs. I then merged what I learned from this two-pronged effort to create a novel sensor tag (Figure 9-2).

I deployed my sensor tags across many months and environments to collect data and investigate ways to achieve our desired versatility, accuracy and privacy preservation. This directly informed the development of a novel, general-purpose, sensing architecture that denatures and virtualizes raw sensor data, and through a machine-learning pipeline, yields what I call *Synthetic Sensors*. Like conventional sensors, these can be used to power interactive applications and responsive environments. I conclude this Chapter with a formal evaluation, deploying the entire sensing pipeline, followed by a digest of significant findings and implications.

RELATED SYSTEMS

Increasingly, sensor "boards" are being populated with a wide variety of underlying sensors that affords flexible use (Table 2-1). Such boards might be considered general purpose, in that they can be attached to a variety of objects, and without modification, sense many facets. However, this is still ultimately a one-sensor to one-object mapping (*e.g.*, Sen.se's Mother [241]), and thus is more inline with the tenets of a distributed sensing system (*many-to-many*).

The ideal sensing approach occupies the top-left of the sensor-utility taxonomy (Figure 2-1), wherein *one* sensor can enable *many* sensed facets, and more specifically, beyond any one single instrumented object. This *one-to-many* property is challenging, as it must be inherently *indirect* to achieve this breadth. The ultimate embodiment of this approach would be a single, omniscient sensor capable of digitizing an entire building.

Computer vision (CV) has come closest to achieving this goal. Cameras offer rich, indirect data, which can be processed through *e.g.*, machine learning to yield sensor-like feeds. There is a large body of work in video-based sensing (see *e.g.*, [77, 171, 255]). Achieving human-level abstractions and accuracy is a persistent challenge, leading to the creation of mixed CV- and crowd-powered systems (*e.g.*, [26, 91, 146]).

Most closely related to this work is Zensors (Chapter 8), which explicitly used a sensor metaphor (as opposed to a Q/A metaphor [26]). Using a commodity camera, a wide variety of environmental facets could be digitized, such as “how many dishes are in the sink?” To achieve this level of general-purposeness, Zensors initially uses crowd answers, which are simultaneously used as labels to bootstrap machine learning classifiers, allowing for a future handoff.

While these CV-based sensing approaches are powerful, cameras have been widely studied and recognized for their high level of privacy invasion and social intrusiveness [19, 29, 28, 113], and thus carry a heavy deployment stigma. To date, this has hindered their use in many environments ripe for sensing, such as homes, schools, care facilities, industrial settings and work environments. In this work, I show that I can achieve much of the same sensing versatility and accuracy without the use of a camera.

EXPLORATORY STUDIES

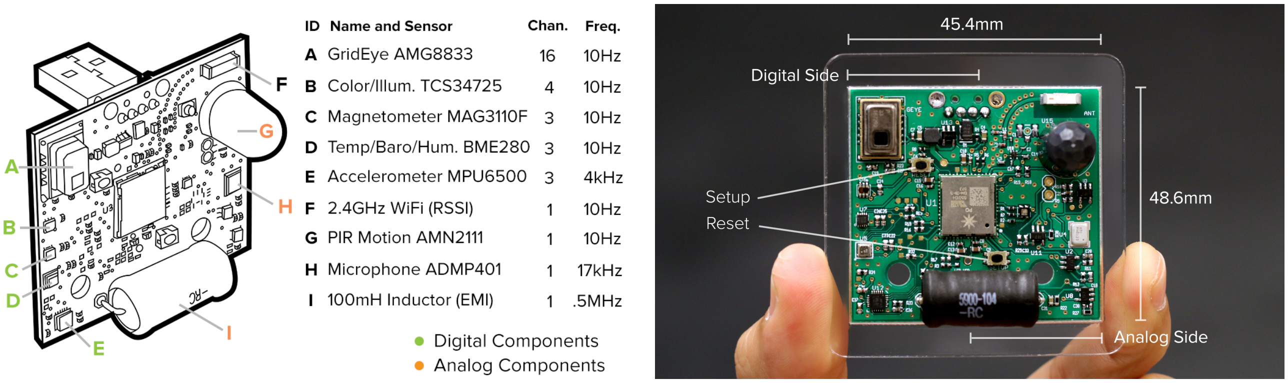
As a first step in my exploration of general-purpose sensing, I conducted two focused probes. This grounded basic assumptions and informed the design of my system.

Survey of Sensor Boards

There is an emerging class of small, screen-less devices equipped with an array of sensors and wireless connectivity, often described as “sensor boards” or “tags”. For example, the Texas Instruments SimpleLink SensorTag packs five sensors into a matchbook-sized, coin-battery-powered package [56]. These devices are intended to facilitate “quick and easy prototyping of IoT experiences.” I performed an extensive survey of these boards, drawn from commercial and academic systems, allowing me to build an inventory of sensing capabilities. The high-level results of this search are offered in Table 9-1.

Facet & Privacy Elicitation Study

In my second probe, I sought to better understand the perceived utility of a “perfect” and omniscient, general-purpose sensor. For this, I conducted an elicitation study (10 interaction design Masters students, 4 female, mean age 24.4, two hours, paid $20) that



ID	Name and Sensor	Chan.	Freq.
A	GridEye AMG8833	16	10Hz
B	Color/Illum. TCS34725	4	10Hz
C	Magnetometer MAG3110F	3	10Hz
D	Temp/Baro/Hum. BME280	3	10Hz
E	Accelerometer MPU6500	3	4kHz
F	2.4GHz WiFi (RSSI)	1	10Hz
G	PIR Motion AMN2111	1	10Hz
H	Microphone ADMP401	1	17kHz
I	100mH Inductor (EMI)	1	.5MHz

● Digital Components
● Analog Components

Figure 9-2. Left: our sensor tag features nine discrete sensors, able to capture twelve unique sensor dimensions. Photo of our general-purpose sensor tag

allowed me to gather facets of interest about six local environments (a common area, kitchen, workshop, classroom, office and bathroom). In total, following a group affinity diagraming exercise, 107 unique facets were identified. I also used this opportunity to informally inquire about the perceived privacy implications of such sensed facets. During discussion, participants unanimously desired that "sensor data be stored in a way that cannot identify individuals." I asked participants to rank privacy facets on a scale of 0 (no privacy danger) to 5 (high privacy danger). Unsurprisingly, facets along "who" dimensions (mean 3.76, SD 1.34) ranked significantly higher ($p<0.01$) in their invasiveness than "what" dimensions (mean 0.92, SD 1.02). Reinforcing my initial notion (and prior research [3,7]), participants uniformly rejected the use of cameras.

CUSTOM SENSOR TAG

I set out to design a novel sensor tag (Figure 9-2), which integrates the *union* of the sensing capabilities across all of the devices in Table 1, minus a camera. Not only does this serve as an interesting vehicle for investigation (*e.g.*, what sensors are most accurate and useful?), but also an *extreme* embodiment of board design using many low-level sensors – one that we hoped could approach the versatility of camera-based approaches, but without the stigma and privacy implications. We incorporated nine physical sensors capturing twelve distinct sensor dimensions (Figure 9-2, left).

The heart of the sensor tag design is a Particle Photon STM32F205 microcontroller with a 120MHz CPU. We strategically placed sensors on the PCB to ensure optimal

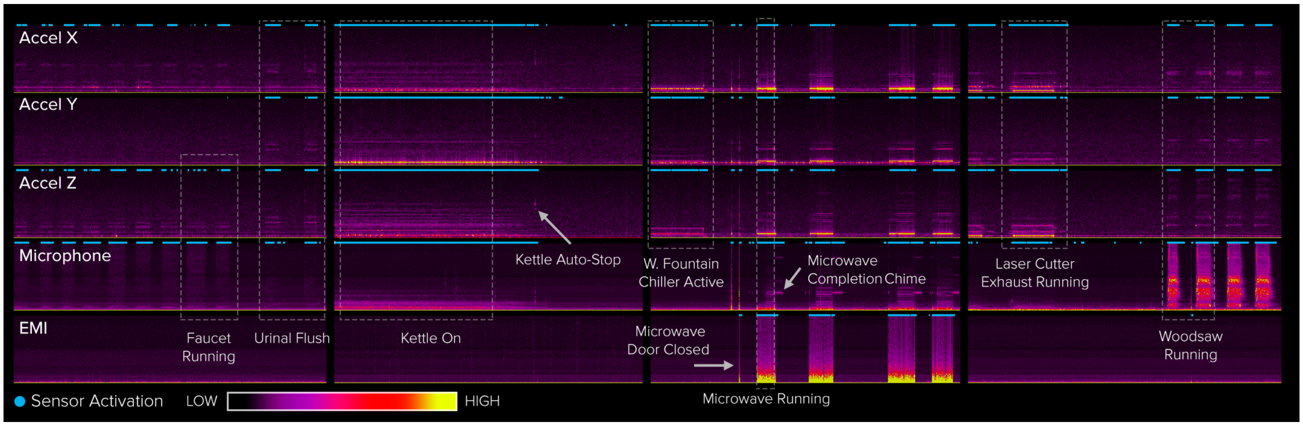


Figure 9-3. Stacked spectrograms of accelerometer (at 4 kHz sampling rate), microphone (17 kHz) and EMI (500 kHz) sensors. A variety of events are illustrated here, with many signals easily discerned by the naked eye.

performance (*e.g.*, ambient light sensor faces outwards), and we spatially separated analog and digital components to isolate unintended electrical noise from affecting the performance of neighboring components. For connectivity, we considered industry standards such as Ethernet, ZigBee, and Bluetooth, but ultimately chose WiFi for its ubiquity, ease-of-setup, range and high bandwidth.

Finally, the board uses a Type A USB 2.0 connector, which can be used for power (*e.g.*, with a DC wall wart) or to deploy software. We intentionally designed the board so that it can be easily plugged-in to a power outlet (in line with our goals of being maintenance free). From this placement, we hoped to be "omniscient" through clever signal processing and machine learning. For this reason, power consumption was not a design objective (for reference: approximately 120mA at 5V when fully streaming).

PILOT DEPLOYMENT AND FINDINGS

At this stage of development, the sensor tags provided a raw stream of high fidelity data (*e.g.*, an audio-quality microphone stream), which was logged to a secured server. Two questions were paramount: 1) was the captured sensor data sufficiently descriptive to enable general-purpose sensing? And 2) was the sensed data adequately preserving occupant privacy, especially identity?

To explore possible tradeoffs, I deployed five sensor tags across thirteen diverse environments I controlled for a collective duration of 6.5 months. During this period, I iteratively refined the sensor's software, affording the ability to test parameters and system features *live* and in *real world* settings. This led to several critical insights:

Immediate Featurization. I found there was little advantage in transmitting raw data from the sensor boards, and instead, all data could be featurized on-sensor. Not only does this reduce network overhead, but it also *denatures* the data, better protecting privacy while still preserving the essence of the signal (with appropriate tuning). In particular, I selected features (discussed later) that do not permit reconstruction of the original signal.

Sensing Fidelity. My pilot deployments showed that the sensor tags were capable of capturing rich, nuanced data. For example in Figure 9-3, I can see not only the coarse event of "microwave in use", but also its door being opened and closed, as well as the completion chime, revealing *state*. In general, I found that sensed signals could be broadly categorized into three temporal classes: *sub-second*, *seconds-to-minutes*, and *minutes-to-hours* scale. For example, a knock on a door lasts a fraction of a second, and so for a sensor to capture this (*e.g.*, acoustically), it must operate at a sampling interval capable of digitizing sub-second events. On the other hand, other facets change slowly, such as room temperature and humidity.

In response, I tuned the raw sensor sampling rates over the course of deployment, collecting data at the speed needed to capture environmental events, but with no unnecessary fidelity. Specifically, I sample temperature, humidity, barometric pressure, light color, light intensity, magnetometer, Wifi RSSI, GridEye (8x8 directional heat sensor), and passive infrared (PIR) motion sensors at 10Hz. All three axes of the accelerometer are sampled at 4 kHz, the microphone at 17 kHz, and the and EMI sensor at 500 kHz. Note that when accelerometers are sampled at high speed, they can detect minute oscillatory vibrations propagating (see Chapter 5) through structural elements in an environment (*e.g.,* drywall, studs, joists), very much like a geophone.

Sensor Activation Groups. My pilot deployments revealed that events tend to activate particular subsets of sensor channels. For instance, a "lights on" event will activate the light sensor, but not the temperature or vibration sensors. Similarly, a door knock might activate the microphone and x-axis of our accelerometer, but not our EMI sensor. I use "activate" to mean any statistical deviation from the environmental norm. As I will discuss, I can leverage these sensor activation groups to improve the system's robustness to noise by only dispatching events to the classification engine if the appropriate set of sensors are activated.

SYNTHETIC SENSORS

My exploratory studies revealed that while low-level sensor data can be high-fidelity, it often does not answer users' true intent. For example, the average user does not care about a *spectrogram of EMI* emissions from their coffee maker – they want to know *when their coffee is brewed*. Therefore, a key to unlocking general-purpose sensing is to support the "virtualization" of low-level data into semantically relevant representations. I introduce a sensing abstraction pattern that enables versatile, user-centered, general-purpose sensing, called *Synthetic Sensors*. In this framework, sensor data exposed to end-users is "virtualized" into higher-level constructs, ones that more faithfully translate to users' mental models of their contexts and environments. This "top-down" approach shifts the burden away from users (*e.g.*, "what can I do with accelerometer data?") and unto the sensing system itself (*e.g.,* user demonstrates a running faucet while the system learns its vibrational signature). Such output better matches human semantics (*e.g.*, "is the laser cutter exhaust running?") and end-user applications can use this knowledge to power rich, context-sensitive applications (*e.g.*, "if exhaust is turned off, send warning about fumes").

Overall Architecture

First, as already discussed, I detect events that manifest in an environment through low-level sensor data. For example, when a faucet is running, a nearby sensor tag can pick up vibrations induced by service pipes behind the wall, as well as characteristic acoustic features of running water. Next, a featurization layer converts raw sensor data into an abstract and compact representation. This happens in our embedded software, which means raw data never leaves the sensor tag. Finally, the "triggered" sensors form an *activation group* (*e.g.,* X- and Z-axis of accelerometer, plus microphone), which becomes input to the machine learning layer.

The system supports two machine learning modalities: manual training (*e.g.,* via user demonstration and annotation using a custom interface we build) or automatic learning (*e.g.*, through unsupervised clustering methods). The output of the machine learning layer is a "synthetic sensor" that abstracts low-level data (*e.g.,* vibration, light color, EMI sensors) into user-centered representations (*e.g.,* coffee ready sensor). Finally, data from one or more synthetic sensors can be used to power end-user applications (*e.g.,* estimating kitchen water usage, sending a text when the laundry dryer is done).

On-Board Featurization

Data from our high-sample-rate sensors are transformed into a spectral representation via a 256-sample sliding window FFT (10% overlapping), ten times per second. Note that phase information is discarded. The raw 8x8 GridEye matrix is flattened into row and column means (16 features). For other low-sample-rate sensors, I compute seven statistical features (min, max, range, mean, sum, standard deviation and centroid) on a rolling one-second buffer (at 10Hz). The featurized data for every sensor is concatenated and sent to a server as a single data frame, encrypted with 128-bit AES.

Event Trigger Detection

Once data is sent, I perform automatic event segmentation on the server side. To reduce the effects of environmental noise, the server uses an adaptive background model for each sensor channel (rolling mean and standard deviation). All incoming streams are compared against the background profile using a normalized Euclidean distance metric (similar to [161]). Sensor channels that exceed their individual thresholds are tagged as "activated". I also apply hysteresis to avoid detection jitter. Thresholds were empirically obtained by running sensors for several days while tracking their longitudinal variances. All triggered sensors form an activation group.

Of note, classification of simultaneous events is possible, especially if the activation groups are mutually exclusive. For overlapping activation groups, cross talk between events is inevitable. Nonetheless, my evaluations suggest that many events contain discriminative signals even when using shared channels.

Server-Side Feature Computation

The set of activated sensors serves as useful metadata to describe an event. Likewise, I use activation groups to assemble an amalgamated feature vector (*e.g.*, a boiling kettle event combines features extracted from the GridEye, accelerometer and microphone). Then, if any high-sample-rate sensors were included, I compute additional features on the server. Specifically, for vibrations, acoustics and EMI, I compute band ratios of 16-bin downsampled spectral representations (120 additional features), along with additional statistical features derived from the FFT (min, max, range, mean, sum, standard deviation, and centroid). For acoustic data, I also compute MFCCs [306]. Data from all other sensors are simply normalized. Finally, these features are fed to a machine learning model for classification.

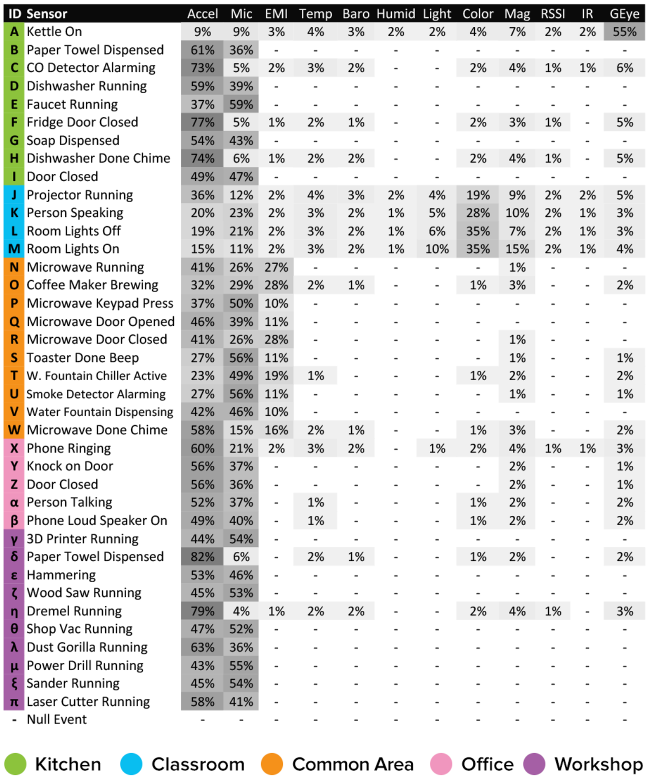
Learning Modalities

In manual mode, users train the system by demonstrating an event of interest, à la "programming by demonstration" [61, 103, 104], supplying supervised labeled data. The feature sets, along with their associated labels are fed into a plurality-based ensemble classification model (implemented using the Weka Toolkit [95]), similar to the approach used by Ravi et al. [212]. I use base-level SVMs trained for each synthetic sensor, along with a global (multi-class) SVM trained on all sensors. This ensemble model promotes robustness against false positives, while supporting the ability to detect simultaneous events.

In automatic learning mode, the system attempts to extract environmental facets via unsupervised learning techniques. I use a two-stage clustering process. First, I reduce the dimensionality of the data set using a multi-layer perceptron configured as an AutoEncoder [108], with five non-overlapping sigmoid functions in the hidden layer. Because the output of the AutoEncoder is the same as the input values, the hidden layer will learn the best reduced representation of the feature set. Finally, this reduced feature set is used as input to an expectation maximization (EM) clustering algorithm. These were implemented via python scikit-learn [199], Weka [95] and Theano [262].

EVALUATION

I explored several key questions to validate the feasibility of synthetic sensors. Foremost, how versatile and generic is our approach across diverse environments and



ID	Sensor	Accel	Mic	EMI	Temp	Baro	Humid	Light	Color	Mag	RSSI	IR	GEye
A	Kettle On	9%	9%	3%	4%	3%	2%	2%	4%	7%	2%	2%	55%
B	Paper Towel Dispensed	61%	36%	-	-	-	-	-	-	-	-	-	-
C	CO Detector Alarming	73%	5%	2%	3%	2%	-	-	2%	4%	1%	1%	6%
D	Dishwasher Running	59%	39%	-	-	-	-	-	-	-	-	-	-
E	Faucet Running	37%	59%	-	-	-	-	-	-	-	-	-	-
F	Fridge Door Closed	77%	5%	1%	2%	1%	-	-	2%	3%	1%	-	5%
G	Soap Dispensed	54%	43%	-	-	-	-	-	-	-	-	-	-
H	Dishwasher Done Chime	74%	6%	1%	2%	2%	-	-	2%	4%	1%	-	5%
I	Door Closed	49%	47%	-	-	-	-	-	-	-	-	-	-
J	Projector Running	36%	12%	2%	4%	3%	2%	4%	19%	9%	2%	2%	5%
K	Person Speaking	20%	23%	2%	3%	2%	1%	5%	28%	10%	2%	1%	3%
L	Room Lights Off	19%	21%	2%	3%	2%	1%	6%	35%	7%	2%	1%	3%
M	Room Lights On	15%	11%	2%	3%	2%	1%	10%	35%	15%	2%	1%	4%
N	Microwave Running	41%	26%	27%	-	-	-	-	-	1%	-	-	-
O	Coffee Maker Brewing	32%	29%	28%	2%	1%	-	-	1%	3%	-	-	2%
P	Microwave Keypad Press	37%	50%	10%	-	-	-	-	-	-	-	-	-
Q	Microwave Door Opened	46%	39%	11%	-	-	-	-	-	-	-	-	-
R	Microwave Door Closed	41%	26%	28%	-	-	-	-	-	1%	-	-	-
S	Toaster Done Beep	27%	56%	11%	-	-	-	-	-	1%	-	-	1%
T	W. Fountain Chiller Active	23%	49%	19%	1%	-	-	-	1%	2%	-	-	2%
U	Smoke Detector Alarming	27%	56%	11%	-	-	-	-	-	1%	-	-	1%
V	Water Fountain Dispensing	42%	46%	10%	-	-	-	-	-	-	-	-	-
W	Microwave Done Chime	58%	15%	16%	2%	1%	-	-	1%	3%	-	-	2%
X	Phone Ringing	60%	21%	2%	3%	2%	-	1%	2%	4%	1%	1%	3%
Y	Knock on Door	56%	37%	-	-	-	-	-	-	2%	-	-	1%
Z	Door Closed	56%	36%	-	-	-	-	-	-	2%	-	-	1%
α	Person Talking	52%	37%	-	1%	-	-	-	1%	2%	-	-	2%
β	Phone Loud Speaker On	49%	40%	-	1%	-	-	-	1%	2%	-	-	2%
γ	3D Printer Running	44%	54%	-	-	-	-	-	-	-	-	-	-
δ	Paper Towel Dispensed	82%	6%	-	2%	1%	-	-	1%	2%	-	-	2%
ε	Hammering	53%	46%	-	-	-	-	-	-	-	-	-	-
ζ	Wood Saw Running	45%	53%	-	-	-	-	-	-	-	-	-	-
η	Dremel Running	79%	4%	1%	2%	2%	-	-	2%	4%	1%	-	3%
θ	Shop Vac Running	47%	52%	-	-	-	-	-	-	-	-	-	-
λ	Dust Gorilla Running	63%	36%	-	-	-	-	-	-	-	-	-	-
μ	Power Drill Running	43%	55%	-	-	-	-	-	-	-	-	-	-
ξ	Sander Running	45%	54%	-	-	-	-	-	-	-	-	-	-
π	Laser Cutter Running	58%	41%	-	-	-	-	-	-	-	-	-	-
-	Null Event	-	-	-	-	-	-	-	-	-	-	-	-

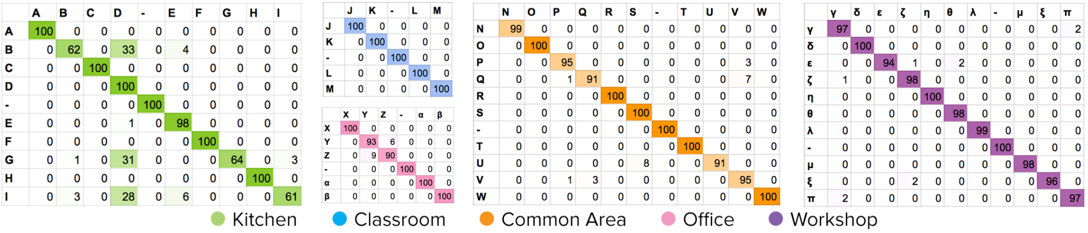
● Kitchen ● Classroom ● Common Area ● Office ● Workshop

Table 9-1. List of synthetic sensors studied across our two-week deployment. Percentages are based on a sensor's feature merit (*i.e.*, SVM weights).

events? Are the signals captured from the environment stable and consistent over time? How robust is the system to environmental noise? And finally, once deployed, how accurate can synthetic sensors be?

Deployment

To answer these questions, I conducted a two-week, *in situ* deployment across a range of environmental contexts. Specifically, I returned to five of the six locations I explored in the Facet & Privacy Elicitation Study: a kitchen (~140 sq. ft.), an office (~71 sq. ft.), a workshop (~446 sq. ft.), a common area (~156 sq. ft.) and a classroom (~1000 sq. ft.), spanning an entire building at my institution. In each room, a single sensor tag was plugged into a centrally located, available, electrical wall socket. Building occupants went about their daily routines uninterrupted. Each tag ran continuously for roughly



	A	B	C	D	-	E	F	G	H	I
A	100	0	0	0	0	0	0	0	0	0
B	0	62	0	33	0	4	0	0	0	0
C	0	0	100	0	0	0	0	0	0	0
D	0	0	0	100	0	0	0	0	0	0
-	0	0	0	0	100	0	0	0	0	0
E	0	0	0	1	0	98	0	0	0	0
F	0	0	0	0	0	0	100	0	0	0
G	0	1	0	31	0	0	0	64	0	3
H	0	0	0	0	0	0	0	0	100	0
I	0	3	0	28	0	6	0	0	0	61

	J	K	-	L	M
J	100	0	0	0	0
K	0	100	0	0	0
-	0	0	100	0	0
L	0	0	0	100	0
M	0	0	0	0	100

	X	Y	Z	-	α	β
X	100	0	0	0	0	0
Y	0	93	6	0	0	0
Z	0	9	90	0	0	0
-	0	0	0	100	0	0
α	0	0	0	0	100	0
β	0	0	0	0	0	100

	N	O	P	Q	R	S	-	T	U	V	W
N	99	0	0	0	0	0	0	0	0	0	0
O	0	100	0	0	0	0	0	0	0	0	0
P	0	0	95	0	0	0	0	0	0	3	0
Q	0	0	1	91	0	0	0	0	0	7	0
R	0	0	0	0	100	0	0	0	0	0	0
S	0	0	0	0	0	100	0	0	0	0	0
-	0	0	0	0	0	0	100	0	0	0	0
T	0	0	0	0	0	0	0	100	0	0	0
U	0	0	0	0	0	8	0	0	91	0	0
V	0	0	1	3	0	0	0	0	0	95	0
W	0	0	0	0	0	0	0	0	0	0	100

	γ	δ	ε	ζ	η	θ	λ	-	μ	ξ	π
γ	97	0	0	0	0	0	0	0	0	0	2
δ	0	100	0	0	0	0	0	0	0	0	0
ε	0	0	94	1	0	2	0	0	0	0	0
ζ	1	0	0	98	0	0	0	0	0	0	0
η	0	0	0	0	100	0	0	0	0	0	0
θ	0	0	0	0	0	98	0	0	0	0	0
λ	0	0	0	0	0	0	99	0	0	0	0
-	0	0	0	0	0	0	0	100	0	0	0
μ	0	0	0	0	0	0	0	0	98	0	0
ξ	0	0	0	2	0	0	0	0	0	96	0
π	2	0	0	0	0	0	0	0	0	0	97

Figure 9-4. Confusion matrices for the 38 synthetic sensors I deployed across the five test locations. Results shown here are from training on data from days 1 through 7, and testing on data from day 14. Use Table 9-1 as key for names.

336 hours (for a cumulative period of roughly 1,700 hours). Featurized data was streamed and stored to a secure local server for processing and analysis.

Versatility of General Purpose Sensing

I examined the list of environmental facet questions (*i.e.*, synthetic sensors) that the earlier study participants elicited, which I pruned to facets that could be practically sensed. Specifically, I removed facets that required a camera (*e.g.*, “what is written on the whiteboard?”), and likewise eliminated facets that did not manifest physical output that could be sensed (“where did I leave my keys?”). This pruned the original set from 107 facets to 59. From the remaining facets, I selected 38 to be the test synthetic sensors (Table 9-1).

Signal Fidelity and Sensing Accuracy

To understand the fidelity and discriminative power of the signals produced from the sensor tags, I conducted an accuracy evaluation, spanning multiple days and locations. In each test location, I demonstrated instances of each facet of interest (mean repeats = 6.0, max 8). For instance, in the workshop, I collected data for the “Laser Cutter Exhaust” synthetic sensor by turning on and off the exhaust several times. I collected data every day for a week, which was labeled offline using a custom tool. This yielded a total of ~150K labeled data instances spanning the 38 synthetic sensors located in five locations. The labeling process took approximately one hour per sensor for a day’s worth of data. This task was tedious, but critical, as it established a ground truth from which to assess accuracy. Note that I also captured “null instances” (*i.e.,* no event) that were derived from captured background instances.

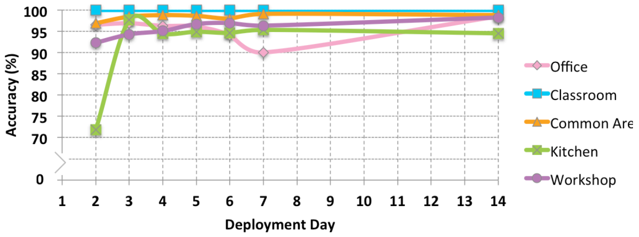


Figure 9-5. Learning curves for our sensor deployments, combined per test location.

To evaluate accuracy, I started by training the classifier using data from day 1, and then testing the classifier using data collected on day 2. This effectively simulates, *post hoc*, what the accuracy would have been on day 2. I then repeat this process, using data from days 1 and 2 for training, and testing on day 3's data. I continue this process up to day 7, which is trained on data from days 1 through 6. In this way, I can construct a learning curve, which reveals accuracy improvements (Figure 9-5).

Across our 38 synthetic sensors in five locations, spanning all seven days, the system achieved an average sensing accuracy of 96.0% (SD=5.2%). Note that the accuracy on day 2 is already relatively high (91.5%, SD=11.3%). I also reiterate that a "day" in this context does not imply a "day's worth" of data, but simply the demonstrated instances for a day (*i.e.*, a few minutes of demonstration data per sensor).

Sensing Stability Over Time

It is possible for environmental facets to change their physical manifestation over time (due to *e.g.*, ambient air temperature, or shifts in physical position). Therefore, it is important to explore whether signals are sufficiently reliable over time to enable robust synthetic sensors. Thus, in addition to collecting data on days 1 through 7, I also collected and labeled one additional round of data on day 14. This weeklong separation (without intervening data collection days) is a useful, if basic test of signal stability.

More specifically, I trained the system on data from days 1 though 7, and tested on day 14's data. Overall, across the 38 synthetic sensors in five locations, the system was 98.0% accurate (SD=2.1%), similar to day 7's results, showing no degradation in accuracy. Day 14's results are also plotted in Figure 9-5, and I further provide the full confusion matrices for each room's sensors in Figure 9-4. Note that a vast majority of the synthetic sensors perform well – near or at 100% accuracy – with just three sensors performing poorly (in the 60% accuracy range). Overall, I believe these results are encouraging and suggest that synthetic sensors can achieve their general-purpose aim.

Noise Robustness

Human environments are noisy, not just in the acoustic channel, but all sensor channels. A robust system must differentiate between true events and a much larger class of false triggers. In response, I conducted a brief noise robustness study that examined the behavior of synthetic sensors when exposed to deliberately noisy conditions.

I selected a high-traffic location (common area) and manually monitored the performance of the classifier (trained on data from days 1-7). An experimenter logged location activity *in vivo,* while simultaneously monitoring classification output. The range of activities observed was diverse, from "sneezing" and "clipping nails", to "people chatting" and a "FedEx delivery." The experimenters also injected their own events, including jumping jacks, whistling, clapping, and feet stomping.

The observation lasted for two hours, and within this period, the experimenter recorded 13 false positive triggers. Admittedly, a longer duration would have been preferable, but the labor involved to annotate a longer period was problematic at the time of the study. Regardless, I believe this result is useful and promising, though the false positive rate is higher than I hoped. It suggests future work is needed on mitigating false positives, perhaps by supplying more negative examples or employing more sophisticated ML techniques, like dropout training.

Automatic Event Learning

I performed a preliminary evaluation of our system's ability to automatically extract and identify events of interest *without* user input (*i.e.*, segmentation or labeling). As briefly discussed in the implementation section, I used a two-step process: multi-layer perceptron followed by an EM clustering algorithm. To evaluate the effectiveness of auto-generated clusters, I used labeled data from days 1 though 7 and performed a cluster membership evaluation. I found mixed results, ranging from a high of 88.1% mean accuracy in the classroom location (five synthetic sensors, thus chance=20%) to a low of 30.0% in the workshop setting (chance=11%). In most locations, clusters were missing for some user-labeled facets, and often featured scores of unknown clusters for things the system had learned by itself. Much future work could be done in this area. More sophisticated clustering and information retrieval techniques could help [48,60], as could correlating sensor data with known events and activities (*e.g.*, room calendar) to power a knowledge-driven inference approach [202, 294].

SENSOR TYPE AND SAMPLE RATE IMPLICATIONS

Most of the synthetic sensors I have described so far have been *sub-second scale* events, and thus most heavily rely on the board's high-sample-rate sensors. This bias can be readily seen in Table 2, which provides a weighted breakdown of merit as calculated by SVM weights when all features are supplied to the classifier. Three sensors in particular stand out as most useful: microphone (17 kHz sample rate), accelerometer (4 kHz) and EMI (500 kHz) – the three highest sample rate sensors on our board.

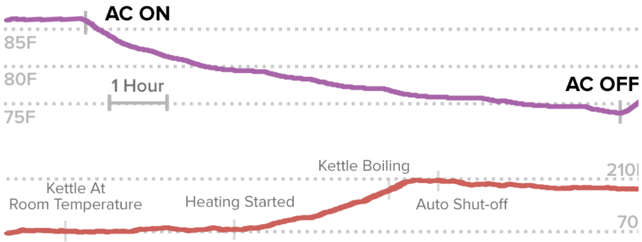


Figure 9-6: Room temp variation caused by an AC unit (top). Heat radiating from the kettle can be captured by the GridEye.

Foremost, I stress that this result should not be over generalized to suggest that using these three sensors alone are sufficient for general purpose sensing. In many cases, the other sensors provide useful data for *e.g.*, edge cases, and can make the difference between 85% and 95% accuracy. Second, as already noted, the sensor questions elicited from the participates were heavily skewed towards instantaneous events, chiefly because I spent roughly ten minutes in each location during the Facet & Privacy Elicitation study, which likely inhibited participants from fully considering environmental facets that might change over longer durations (like a draft in a poorly sealed window).

Finally, every environment is different and there is no doubt there exists a long tail of questions that could be asked by end users. Although microphone, accelerometer and EMI might enable 90% of possible sensor questions, to be truly general purpose, a sensor board will need to approach 100% coverage. It can be seen from Table 9-1 that other, infrequently-used sensors are occasionally critical in classifying some questions, for example, the *GridEye* sensor for detecting kettle use (Table 9-1A), and the *light color sensor* for detecting when the classroom lights are on/off (Table 9-1, M/L). If these two sensors were dropped from our board, these two questions would likely have been unanswerable (at acceptable accuracies).

As an initial exploration of how other sensors can come into play – especially for sensing longer-duration environmental facets – I ran a series of small, targeted deployments in mostly new locations and at different times of the year. I highlight events of interest to underscore the potential utility of other sensor channels.

Room Temperature Fluctuation. I used our sensor tag to capture temperature variation in a room with a window-mounted air conditioner on a warm summer's night (Figure 9-6). Note the accelerated slope of the temperature when the AC is turned on and off. Another example of HVAC cycling can be seen in Figure 9-8 (note also the change in behavior when the thermostat target temperature is moved from 70 to 72°).

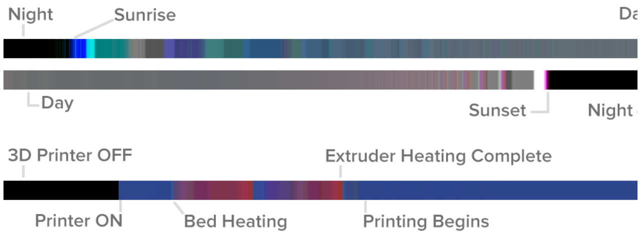


Figure 9-7: Daylight variations (top). Various states of a MakerBot Replicator 2X over a ~30 minute period, captured by a light color sensor (bottom).

Non-Contact Temperature Sensing. The GridEye sensor acts like a very low-resolution thermal camera (8×8 pixels), which is well suited for detecting localized thermal events. For example, in the kitchen location, the kettle occupies part of the sensor's field of view, and as such, the radiant heat of the kettle can be tracked, from which its operational state can be inferred (Figure 9-6).

Light Color Sensing. I also found ambient light color to be a versatile sensing channel. For example, colors cast by artificial lighting or sunrise/sunset (Figures 9-7), can provide clues about the state of the environment (*e.g.*, bedroom window open, TV on). Additionally, many devices communicate their operational states through color LEDs (Figure 9-7, bottom), which can be captured and aid classification of different states.

Multiple Sensors. Figures 9-8 and 9-9 offer more complex examples of how multiple sensors can work together to characterize environments. For example, in Figure 9-8, one can see that opening a garage door causes a temperature, color and illumination change in the garage. Having multiple confirmatory signals generally yields superior classification accuracies. Figure 9-9 shows how high- and low-sample-rate sensors can work together to capture the state of a car and an apartment.

SECOND-ORDER SYNTHETIC SENSORS

Up to this point, the synthetic sensors that I have discussed all operate in a binary fashion (*e.g.,* is the "faucet running?" Possible outputs: yes or no). These are what I call *first-order synthetic sensors*. I can build more complex, non-binary, *second-order synthetic sensors* that leverage first order outputs as new ML features. I explored three 2nd-order classes: state, count and duration. The first-order sensors used in this section were trained using data from days 1 though 7 in the deployment study.

State

Two or more first-order synthetic sensors can be used as features to produce a second-order synthetic sensor for tracking the multi-class *state* of an object or environment. For example, in the two-week deployment study, I had five first-order synthetic sensors about a single microwave (running, keypad presses, door opened, door closed, completion chime; see Table 9-). From these five, individual, binary-output, first-order synthetic sensors, I created a microwave state, second-order sensor (five-classes: *available*, *door ajar*, *in-use*, *interrupted*, or *finished*; Figure 9-10). For example, when the completion chime is detected, the state will change from *in-use* to *finished*, and will

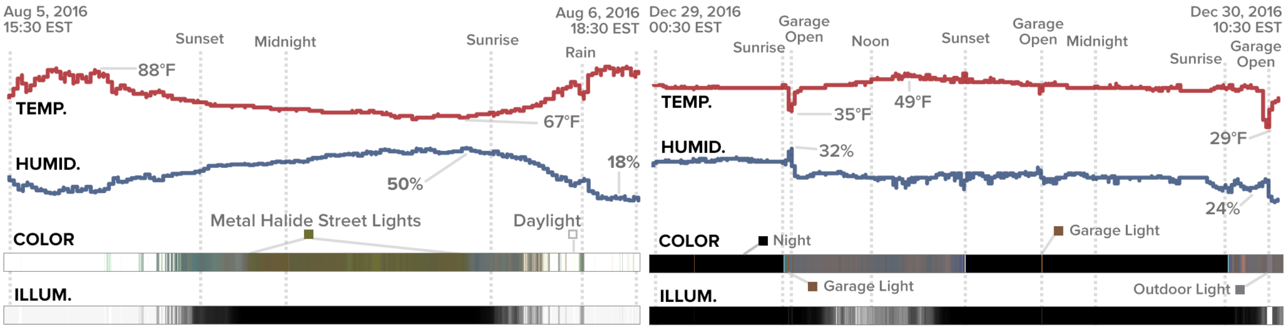


Figure 9-8: Left: data captured over a ~24 hour period in an outdoor parking (summer). Right: data captured in a two-car garage over a ~36 hour period (winter).

stay *finished* until a door close event is detected, after which the items inside are presumed to have been removed, and the state is set to *available*.

To test the microwave-state sensor's accuracy, I manually cycled the microwave through its five possible states (ten repetitions per state), and recorded if it matched the classifier's output. Overall, the sensor was ~94% accurate.

Count

In addition to states, it is also possible to build second-order synthetic sensors that can *count* the occurrence of first-order events. For example, it is possible to use a door opened first-order sensor to track how many times a restroom is accessed. This, in turn, could be used to trigger a message to staff to inspect the restroom every 100 visits.

As a real-world demonstration, I built a second-order count sensor that tracked the number of towels dispensed by the dispenser in the kitchen location. To test this counter's accuracy, I manually dispensed 100 towels. At the end, our sensor reported 92 towels dispensed. As shown here, errors can accumulate, but nonetheless offers a

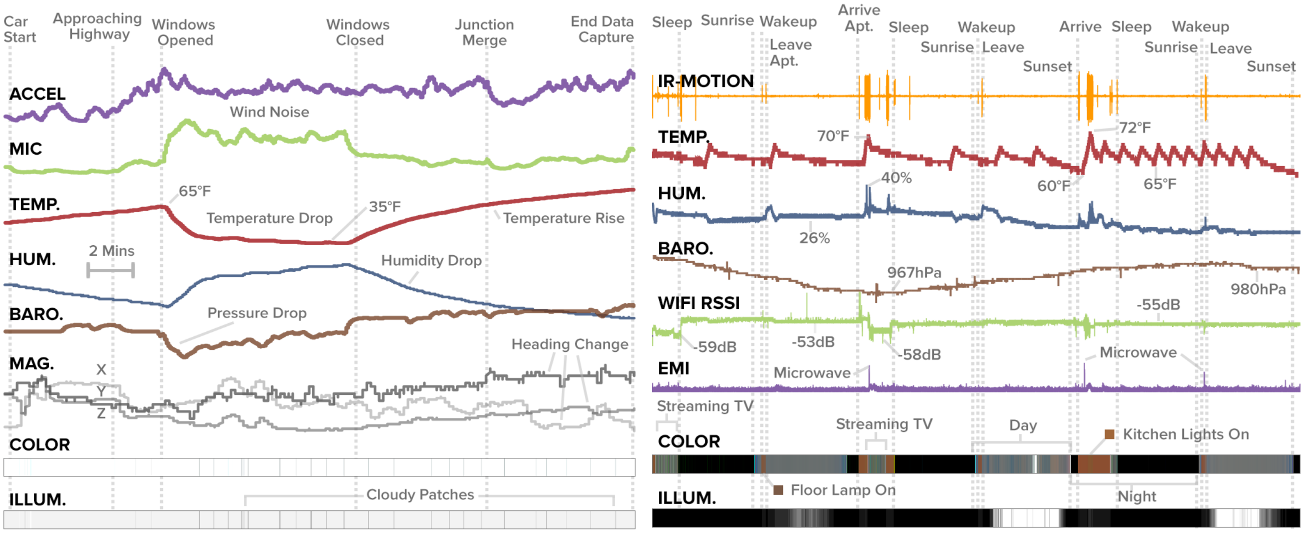


Figure 9-9: Left: sensor in moving car. Here, high- and low-sample-rate sensors offer complimentary readings useful in detecting complex events, *e.g.*, when a window is opened. Right: Data captured in an apartment over ~72 hours.

reasonable approximation. Similar to the previous example, when the dispenser runs low, an order for more supplies could be automatically placed.

Duration

Similar to count, it is also possible to create second-order synthetic sensors that track the cumulative duration of an event, for example energy consumption or water usage. I performed two simple evaluations. First, using the *microwave running* first-order sensor (Table 9-1N), I built a second-order "microwave usage" duration sensor. To test it, I ran the microwave 15 times with random durations (between 2 and 60 seconds). At the end of each run, I compared our sensor's estimated duration to the real value. Across 15 trials, the microwave usage sensor achieved a mean error of 0.5 secs. (SD=0.4 sec).

As a second example, I used the *faucet running* first order sensor (Table 9-1E) to estimate water usage. To convert time into a volume of water, I used a calibration process similar to UpStream [32]. To test this sensor, I filled a large measuring cup with a random target volume of water (between 100-1000mL), and compared the true

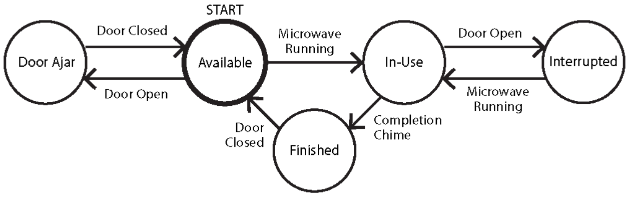


Figure 9-10. An example state machine for a second-order “microwave state” synthetic sensor.

value to the classifier’s output. I repeated this procedure ten times, revealing a mean error of 75mL (SD=80mL).

NTH-ORDER SYNTHETIC SENSORS

Importantly, there is no reason to stop at second-order synthetic sensors. Indeed, first-order and second-order synthetic sensors could feed into third-order synthetic sensors (and beyond), with each subsequent level encapsulating richer and richer semantics. For example, appliance-level second-order sensors could feed into a kitchen-level third-order sensor, which could feed into a house-level sensor, and so on. A house-level synthetic sensor, ultimately drawing on scores of low-level sensors across many rooms, may even be able to classify complex facets like human activity. I hope to extend the system in the near future to study and explore such possibilities.

CONCLUSION

In this Chapter, I introduced *Synthetic Sensors*, a sensing abstraction that unlocks the potential for versatile and user-centered, general-purpose sensing. This allows everyday locations to become "smart environments" without invasive instrumentation. Guided by formative studies, I designed and built novel sensor tags, which served as a vehicle to explore this problem domain. Real-world deployments show that general-purpose sensing can be flexible and robust, able to power a wide range of applications.

10. SPARSE GENERAL-PURPOSE SENSORS FOR WIDE-AREA SENSING

INTRODUCTION

In the previous Chapters, I put forward the notion of general-purpose sensors, with a stated goal of "one sensor per room." Using either a camera (Chapter 8) or a multitude of low-level sensors on a single board (Chapter 9), these aim to digitize a wide range of environmental facets in a room with no direct instrumentation of objects. Today, devices such as Google Home, Amazon Alexa or Apple HomePod [11] are commercial examples that operate at single room scale, and have the requisite sound sensing capability for activity recognition. These approaches can offer all of the benefits of full sensor saturation, but without the aesthetic, maintenance, and cost drawbacks.

However, the previous Chapters have only investigated single room sensing, and did not consider sensing events in neighboring spaces, nor leverage potentially complementary signals if more than one sensor was deployed in *e.g.*, a home (Figure 10-1). Moreover, the previous projects have generally been confined to a lab setting and not authentic use environments, and the range of sensed classes has been small. In this Chapter, I directly address these open questions, move beyond lab studies, and seek to quantify four key questions:

> Q1: What is the accuracy difference when sensing activities from *inside* a room vs. from *outside* of it?
>
> Q2: Is there a benefit in using data from *another sensor*, for example, a sensor in a *nearby* room?

Q3: What is the effect of using *all available sensor data* at a location (*e.g.*, *every sensor* in a house)?

Q4: What events, if any, are detectable when *only sensor data from outside* of a room is available.

To answer these questions, I crafted a study procedure that allowed me to evaluate different spatial configurations of sensors: (1) in room sensing, (2) in room + nearest room sensing, (3) all room sensing, (4) only-nearest-room sensing, and (5) all-but-in-room sensing. These conditions, in turn, serve as experimental proxies for the questions above. To quantify performance, I ran a deployment study capturing real-world data using thirty general-purpose sensors distributed at three distinct use locations (small business, residential and institutional). At these locations, I studied 100 sensed events (67 unique classes). I also investigate how and what types of signals propagate in typical environments, as well as robustness across different rooms and locations.

At a high level, I found that in-room sensing (which is the most pervasive contemporary practice) can benefit from sensor data from adjacent rooms. While not a surprising result, this work is the first to quantify it in a comprehensive study across 30 rooms and 100 sensed events. I show that in-room accuracy (94.6%) can be improved by 1.8% when leveraging sensor data from a neighboring room (96.4%). Although this accuracy gain appears modest, it is a meaningful (and statistically significant) reduction in classification error. Often one of the most significant challenges in moving technology from the lab and into the real world is closing the gap between 95 and 99% accuracy. Other results are less intuitive, for example, utilizing all sensors in a house can actually harm accuracy due to feature explosion and added noise. I also show that sensing without any sensor in a room (but perhaps surrounded by sensors) remains a stubborn research challenge. As before, I quantify many of these widely held assumptions in order to underscore where research challenges remain.

PROPAGATION OF EVENTS

Physical events manifest as sound, vibration, illumination, temperature and other forms of energy, which are rarely confined to one location. Instead, they propagate outwards, into interior spaces through openings and doors, across walls, and along structural members [194]. Therefore, it is possible for a general-purpose sensor placed in one

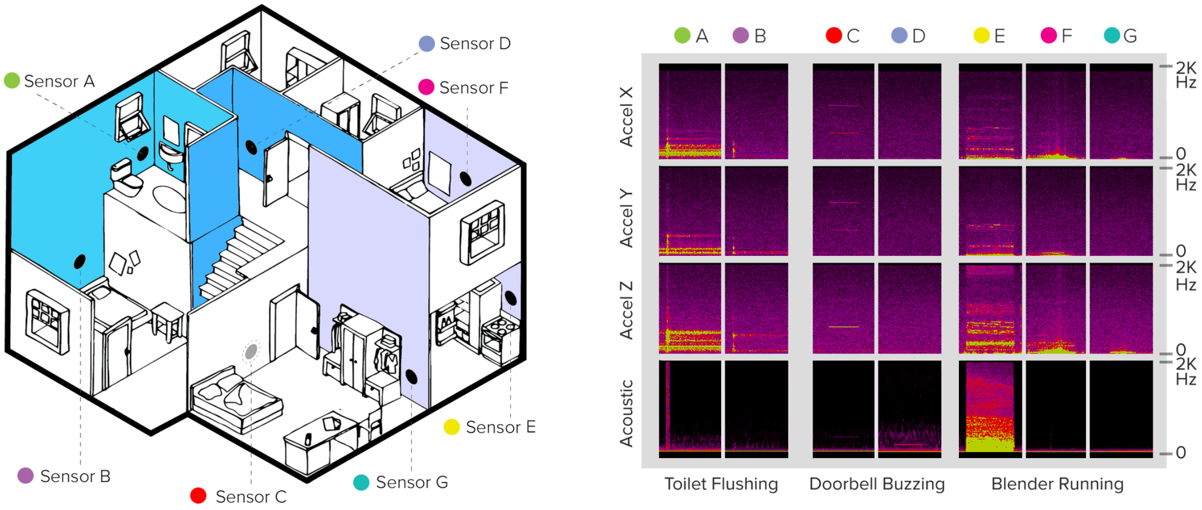


Figure 10-1. An example of a sparse constellation setup, wherein a small group of sensors, one per room or less, can detect a wide range of events happening across *e.g.,* an entire home. Sensors that share *e.g.,* a common wall (Sensor A & B, Sensor E & F) or a contiguous space (Sensor C & D) can be used to correlate and confirm events, using signals such as changes in vibration, sound, motion or lighting.

location to indirectly detect events in another. In theory, this should enable sensor constellations with a sparsity of less than one per room. Furthermore, other sensors can offer confirmatory signals, improving accuracy and robustness.

Figure 10-1 offers an example of this effect with a sparse sensor constellation deployed in a home. Here, Sensor A (bathroom) and B (lower bedroom) share a common wall, Sensor C (first floor hallway) and D (second floor hallway) share an open space, while Sensor E (kitchen), F (upper bedroom) and G (master bedroom) are simply proximate to one another. When an event occurs, such as a toilet being flushed, it produces characteristic sound and vibrations (Figure 10-1, Right; Sensors A & B). These physical manifestations are readily captured by the accelerometer and microphone of Sensor A (Figure 10-1, Sensor A), located in the room, and also by the accelerometer of Sensor B (Figure 10-1, Sensor B) in the room below and adjacent to the plumbing stack. A similar effect can be seen when the doorbell buzzes (Figure 10-1, Sensors C & D), or when a blender is running in the kitchen (Figure 10-1, Sensors E, F & G). Depending on the interior layout, changes in lighting, temperature, motion and other non-vibroacoustic channels can be useful in supporting robust recognition of event.

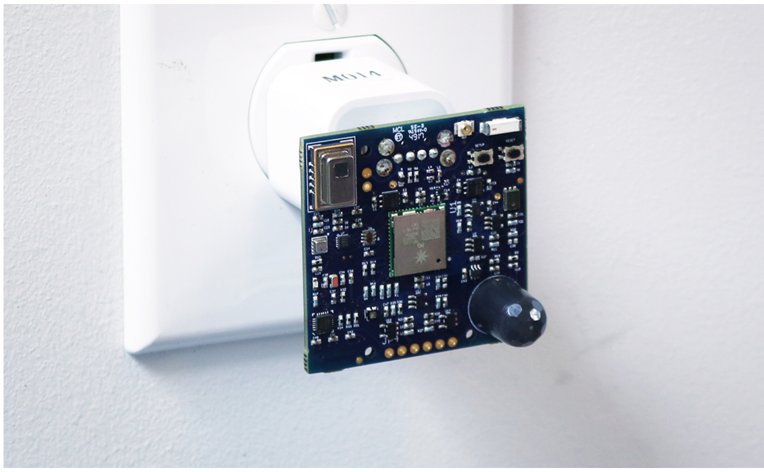


Figure 10-2. The general-purpose sensing board [20] for my constellation deployments.

DEPLOYMENT HARDWARE AND SOFTWARE

For deployment, I take advantage of the general-purpose sensing hardware developed in Synthetic Sensors (Chapter 9), shown in Figure 10-2. This WiFi-connected board offers an array of low-level sensors (18 channels in total: 3-axis accelerometer, microphone, electromagnetic interference, temperature, humidity, barometer, illumination, RGB light color, infrared motion, 3-axis magnetometer, non-contact thermal, and WiFi RSSI), sampled at high rates and buffered locally.

Ten times a second, the board computes and transmits a feature vector derived from its buffered on-board sensors. Specifically, it computes seven statistical features (min, max, sum, mean, standard deviation, range, and centroid) for all 18 channels. Additionally, for its high-sample-rate channels (microphone, electromagnetic interference, and three accelerometer axes), the board also sends 128-length, real-valued FFTs. In total, the board computes 1207 features (see Chapter 9 for full details). This featurized data not only reduces bandwidth, but also denatures data to help preserve privacy. Per sensor data throughput is ~20KBps. One improvement to the sensor firmware was the addition of Network Time Protocol (NTP), which the boards use to accurately timestamp all data transmissions, facilitating multi-sensor data fusion.

To receive data, I built a backend to handle incoming sensor streams (which connect to the Internet over WiFi). Transmitted data is decoded on the server and can be serialized

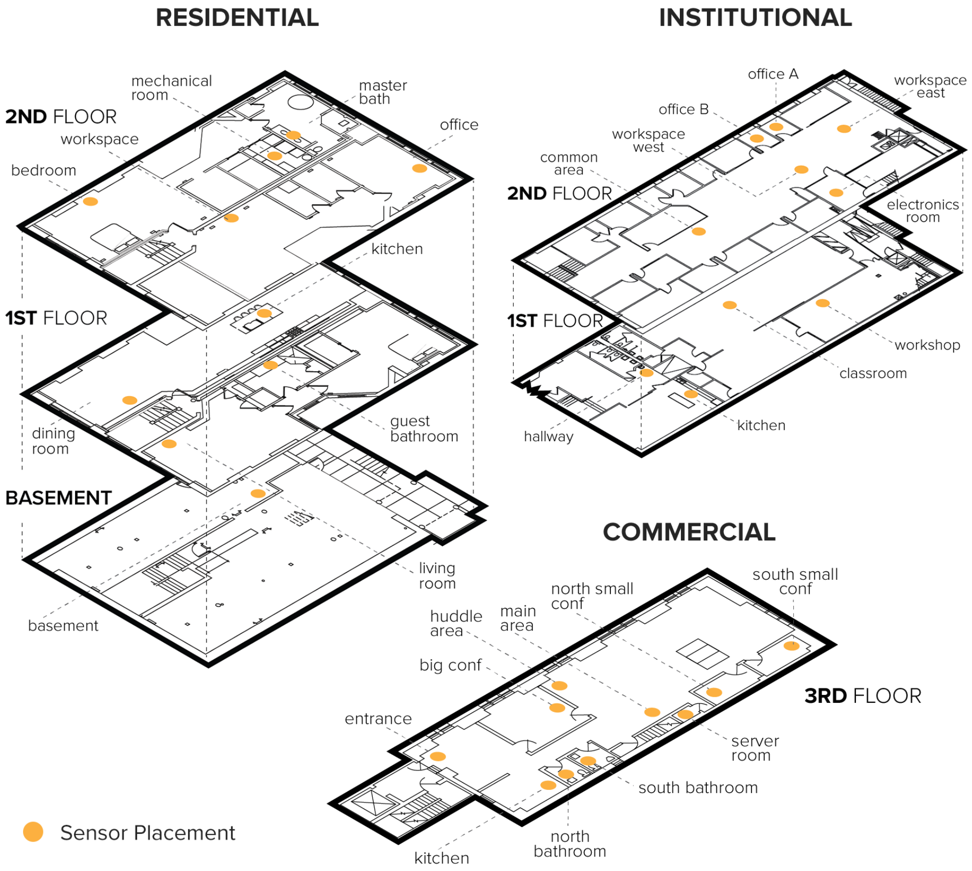


Figure 10-3. Floor plan and sensor placements at our three test contexts: Home, Institution, and Business.

to disk for offline processing or forwarded to a delegate application for real-time visualization or analysis. The backend takes care of all data synchronization using sensor packet timestamps. For each incoming sensor stream, the backend performs a mean windowing (w=5, analogous to average pooling [16]) to reduce noise. Finally, the backend assembles all window-averaged features from each sensor in a constellation into a superset of synchronized features (*i.e.*, 10 sensors × 1207 features), which serves as input to one or more endpoints for machine learning.

DEPLOYMENT STUDY

The system backend and fleet of synchronized sensors provided a unique technical vehicle to investigate the efficacy of sparse, general-purpose sensor constellations in

Location	Room	Sensed Event	L	V	S	In-Room Only	In Room+ Nearest	All Rooms	Nearest Only	All But In-Room
Home Location	Office	Door Closed	✓	✓		99%	100%	96%	82%	33%
		Printer On	✓			99%	100%	100%	32%	26%
		Space Heater Running			✓	100%	100%	100%	7%	5%
	Basement	Back Door Closed	✓	✓		100%	100%	100%	24%	46%
	Bedroom	Door Closed	✓	✓		86%	98%	96%	96%	89%
	Workspace	Chair Rolling		✓		98%	98%	95%	55%	38%
		Computer On	✓			94%	97%	86%	0%	0%
		Music Playing	✓			99%	99%	97%	53%	34%
	Dining Room	Door Closed	✓	✓		100%	100%	100%	81%	79%
	Guest Bathroom	Door Closed	✓	✓		96%	95%	96%	2%	18%
		Left Faucet Running	✓	✓		90%	92%	84%	15%	3%
		Right Faucet Running	✓	✓		79%	88%	80%	9%	1%
		Shower On	✓	✓		98%	99%	97%	14%	42%
		Toilet Flushing	✓	✓		98%	99%	100%	28%	12%
	Kitchen	Blender On	✓	✓		100%	100%	99%	100%	73%
		Coffee Grinder On	✓			100%	100%	100%	100%	66%
		Dishwasher On	✓	✓		93%	95%	98%	9%	3%
		Faucet On	✓	✓		92%	91%	91%	55%	15%
		Garbage Disposal On	✓	✓		100%	100%	100%	46%	58%
		Left Burner On			✓	98%	98%	98%	19%	22%
		Right Burner On			✓	98%	98%	98%	4%	4%
		Toaster Finished	✓			98%	98%	99%	87%	11%
	Living Room	Fireplace On			✓	97%	97%	97%	56%	38%
		TV Playing	✓			97%	97%	98%	49%	44%
	Master Bath	Electric Toothbrush On	✓			98%	98%	98%	55%	23%
		Hair Dryer On	✓			100%	100%	100%	73%	49%
		Left Faucet Running	✓	✓		93%	100%	98%	100%	100%
		Right Faucet Running	✓	✓		88%	98%	94%	86%	7%
		Shower Running	✓	✓		90%	95%	96%	27%	25%
		Toilet Flushing	✓			83%	74%	69%	6%	0%
	Mechanical Room	Furnace Running	✓	✓		100%	100%	100%	100%	96%
		Water Heater Running	✓	✓		100%	100%	100%	97%	65%
		Laundry Dryer Running	✓	✓		100%	100%	100%	100%	88%
		Utility Sink Running	✓	✓		96%	97%	98%	85%	46%
Institution Location	Office A	Door Closed	✓	✓		96%	96%	96%	95%	90%
		Door Knock	✓	✓		99%	99%	99%	40%	25%
		Phone Ringing	✓			100%	100%	99%	48%	48%
	Classroom	Lecturer Talking	✓			99%	99%	100%	11%	32%
		Partition Drag	✓	✓		100%	100%	100%	100%	99%
		Projector Running		✓		100%	100%	100%	54%	41%
	Common Area	Coffee Brewing	✓	✓		99%	100%	100%	31%	35%
		Microwave Running	✓	✓		100%	100%	100%	71%	26%
		Toaster Finished	✓			100%	99%	100%	4%	2%
		Water Fountain Running		✓		97%	99%	98%	6%	18%
	Electronics Room	Bench Grinder Running	✓	✓		85%	86%	86%	58%	57%
		Drill Press On	✓	✓		86%	86%	86%	44%	33%
		Extractor Fan On	✓	✓		83%	86%	85%	10%	14%
		Staple Gun Fired	✓			89%	95%	97%	15%	57%
		West Door Closed	✓	✓		97%	100%	94%	98%	79%
	Hallway	Chiller Running	✓	✓		100%	100%	100%	79%	56%
		Door Closed	✓	✓		100%	100%	100%	17%	27%
		Water Fountain Running	✓	✓		100%	100%	100%	35%	19%
Institution Location	Office B	Door Closed	✓	✓		100%	100%	98%	100%	93%
		Door Knock	✓	✓		91%	92%	96%	91%	58%
		Fridge Closed		✓		84%	87%	92%	84%	56%
	Kitchen	Coffee Grinding	✓	✓		100%	100%	100%	92%	77%
		Drawer Actuated		✓		99%	97%	99%	22%	7%
		Faucet Running	✓	✓		100%	100%	100%	35%	15%
		Microwave Running		✓		99%	100%	100%	84%	89%
	Lab East	Mouse Moving On Table		✓		84%	95%	92%	30%	9%
		Objects Placed On Table		✓		90%	92%	88%	27%	7%
		Typing On Table		✓		94%	97%	96%	25%	25%
		Writing On Table		✓		98%	98%	99%	18%	39%
	Lab West	Card Swiped	✓			100%	99%	99%	65%	53%
		Closet Door Closed	✓	✓		85%	95%	88%	91%	73%
		Door Closed	✓	✓		91%	91%	89%	76%	76%
	Workshop	Air Compressor Running	✓	✓		100%	100%	100%	96%	74%
		Exhaust Fan On	✓	✓		100%	100%	100%	99%	93%
		Laser Exhaust Running	✓	✓		100%	100%	100%	99%	77%
		Mitre Saw Running	✓			100%	100%	100%	86%	57%
Business Location	Big Conf	North Door Closed	✓	✓		98%	97%	86%	94%	85%
		South Door Closed	✓	✓		99%	99%	99%	100%	83%
	Entrance	Front Door Closed	✓	✓		92%	94%	95%	81%	81%
	Huddle Area	Front Door Buzzer	✓	✓		71%	75%	75%	86%	86%
		Whiteboard In Use		✓		100%	100%	100%	73%	100%
	Kitchen	Dishwasher Running	✓	✓		98%	100%	100%	100%	100%
		Espresso Running	✓	✓		70%	95%	86%	96%	75%
		Faucet Running	✓	✓		65%	95%	94%	86%	85%
		Garbage Disposal On	✓	✓		94%	100%	100%	100%	100%
		Kettle Running	✓	✓		84%	98%	91%	62%	49%
		Keurig Running	✓	✓		95%	93%	95%	68%	46%
		Microwave Running	✓	✓		100%	100%	100%	86%	95%
		Toaster Finished	✓			85%	90%	91%	93%	81%
	Main Area	Paper Shredder On	✓	✓		100%	100%	100%	99%	97%
		Printer Drawer Closed	✓	✓		99%	97%	97%	86%	91%
		Printer Printing	✓	✓		100%	100%	100%	88%	63%
		Scale Key Pressed	✓			91%	95%	94%	54%	9%
	Bathroom A	Door Closed	✓	✓		98%	97%	98%	98%	97%
		Faucet Running	✓	✓		96%	100%	100%	55%	46%
		Toilet Flushing	✓	✓		93%	100%	99%	89%	75%
	Server Room	Door Closed	✓	✓		100%	98%	100%	97%	97%
	North Conf	Door Closed	✓	✓		89%	93%	93%	88%	28%
		Whiteboard In Use		✓		87%	92%	93%	45%	53%
	South Conf	Door Closed	✓	✓		83%	83%	84%	61%	57%
		Whiteboard In Use		✓		99%	99%	99%	55%	41%
	Bathroom B	Door Closed	✓	✓		84%	80%	81%	75%	70%
		Faucet Running	✓	✓		88%	92%	84%	80%	78%
		Paper Towel Dispensed		✓		77%	87%	73%	56%	35%
		Toilet Flushing	✓	✓		82%	98%	92%	96%	91%
		Urinal Flushing	✓	✓		90%	100%	99%	100%	100%
	All Locations	Background Class				96%	97%	97%	67%	60%
		Average Accuracy				**95%**	**96%**	**96%**	**63%**	**54%**
		Standard Deviation				**7.1%**	**5.3%**	**6.2%**	**31.4%**	**30.1%**

Legend: L = Loud, V = Vibrational, S = Line-of-Sight

Table 10-1. Deployment results broken out by evaluation condition (colorized columns) and sensed event (rows). L column denotes acoustically loud events; V for vibrational events, and S for events requiring line-of-sight.

different spatial configurations. I now describe where I deployed these sensors, what they were tasked with sensing, and the data collection procedure.

Study Locations

As a test bed, I selected three distinct locations: a *home* (Figure 10-3 top left), a non-profit *institution* (Figure 10-3 top right) and a small *business* (Figure 10-3 bottom). These locations captured diversity in size, layout, construction, function, and appliances available to sense. According to municipal records, the home, institution and business were last renovated, inspected and brought up to code in 2015, 2014 and 2010 respectively. The home and institution both followed the 2012 International

Building Code (IBC) [115], while the business followed the 2009 IBC (both of which provide, *e.g.,* standards for electrical outlet spacing). All three structures are typical northeastern US construction, with loadbearing masonry exteriors. The home and business used wood floor joists, while the intuitional building used steel beams. For walls, the home used wood studs, while the business and institution locations used metal studs. All three used drywall for wall surfaces. Occupants in each location were briefed about the deployment and signed participation waivers, but otherwise went about their usual activities.

Sensed Events

In order for the results to be representative across a wide range of uses, I sought to develop a large and diverse set of sensed events. I started by asking building occupants about environmental facets a "smart building" might wish to know about. From this list, a research team curated a final set, selecting those plausible to sense with our hardware. In total, I included 100 sensed events, encompassing 67 unique classes, across our three test locations, summarized in Table 10-1.

Sensor Placement

I deployed ten sensor tags in each study location, at a density of one per room. I define a room as a contiguous area separated physically or functionally by walls, flooring, elevation change, walkways, furniture or other demarcations. Sensor placement followed a strict procedure, to avoid experimenter bias:

1. Using a pre-determined list of desired sensed events, find the spatial centroid of the events in the room.
2. Plug the sensor into the closest (*i.e.,* Euclidian distance) wall electrical outlet to the centroid.

This placement procedure is both logical and realistic, and I further confirmed with occupants that this is where they would have placed the sensor as well. In some cases, this led to suboptimal placement (*e.g.,* sensor placed farther away from more subtle signals). However, I do not view this as an experimental confound, but rather as ecological validity. I did not know *a priori* which event was harder to sense than others, and more importantly, it is unrealistic to expect end users to move sensors around to improve accuracy. As a result, the study results offer a realistic assessment of accuracy should such sensors be deployed in a logical, but perhaps not optimal way. Moreover, I purposefully included a large number of sensed events (100) as a way to assess population level accuracy, which is more robust than drawing conclusions on individual sensor placements or specific events.

Procedure

I initially considered collecting data serendipitously, as occupants triggered events in the course of their days, which I would label post hoc by *e.g.*, listening to audio. However, there were three significant drawbacks to this procedure. Foremost was privacy – capturing data that facilitated accurate labeling required a high degree of privacy invasion (either a camera or audio feed – neither of which the pilot participants would accept for more than a brief deployment period). Second, even with video and/or audio, I often found it challenging to know exactly what event was happening (faucet running vs. toilet bowl refilling; coffee maker vs. microwave running) due to *e.g.*, visual occlusion and ambiguous sounds. This made building a reliable ground truth for evaluation difficult and laborious. Finally, I found that many interesting events happened infrequently (*e.g.*, paper shredder, bench grinder, toaster), which precluded building a sufficient data set for either training or analysis.

As a necessary experimental compromise, I decided to visit each of the 30 rooms (three locations, ten rooms each) once per day for a week and manually trigger events (*e.g.*, ran the microwave, flushed the toilet). For transient events, such as a door closing, I repeated the action several times over a ten second period. The order I visited locations on any given day was randomized, and then within location, room order was randomized, and then finally within rooms, event performance order was randomized.

All data was captured during regular hours, with occupants going about their normal routines. Throughout data collection, I did not control events happening in other rooms (*e.g.*, I collected microwave events in the kitchen, while someone could be washing a cup, flushing a toilet, or printing a document). It was common for captured data to have background noise, including *e.g.*, human chatter, road noise from outside, HVAC and occupants walking around. This noise is purposely part of our data, and one of the

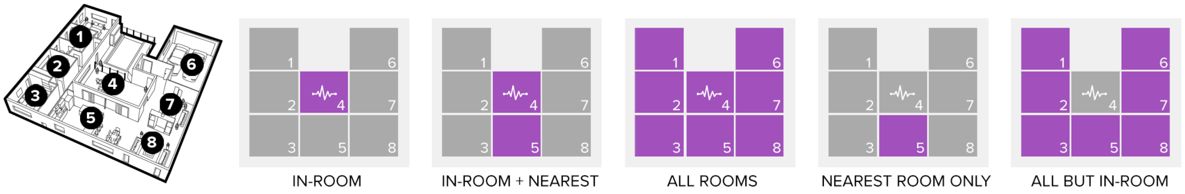


Figure 10-4. Far left, hypothetical layout with eight rooms, with signal of interest in Room 4. To the right, pictorial representations of the five spatial conditions (purple rooms = sensor contributing to recognition).

reasons why I ran a deployment as opposed to an entirely controlled lab study (like that in Chapter 9, which offers a different and complimentary experimental result). I also collected a "background" class (*i.e.,* ambient noise, no event happening) for each room. To facilitate the tedious task of labeling events, I built an annotation tool that supported playback and the ability to overlay synchronized sensor data. Data labeling was completed immediately after performing (known-order) events in a room. In total, this yielded ~640K labeled instances, representing ~17.8 hours of data.

RESULTS

To investigate the four research questions (Q1-Q4), I developed five spatial configuration conditions that serve as probes to quantify system performance: (1) in room sensing, (2) in room + nearest room sensing, (3) all room sensing, (4) only-nearest-room sensing, and (5) all-but-in-room sensing (Figure 10-4). I then built machine learning models for each room (30) × each spatial condition (5).

Analysis Methodology

To evaluate classification performance, I ran day-fold, cross-validation analyses for each model (see next section for details on machine learning). In this day-fold validation scheme, I procedurally choose one day out of the week-long deployment as a test dataset (*e.g.*, Day 3), and use the remaining days (*e.g.*, Days 1-2 and 4-7) for training data. Separating the data in this fashion prevents temporal overfitting. I repeat this process seven times, for all days in the week, and then average the results. This method enables quantifying the effect of several constellation configurations across variations in location and time. Finally, note that although I do not specifically discuss “false positives”, the per-room models incorporate a “background” event that serves as a “negative” class for mitigating false event triggers.

In-Room Sensing Accuracy

In this condition, I evaluated classification accuracy using only data collected from the sensor operating *in the room* where an activity occurred (Figure 10-4). Due to physical proximity, in-room sensors are the most likely to provide robust classification, and thus serve as a baseline (Q1). Across all 100 sensed events (in 30 rooms in three locations), the event classifiers achieved a mean accuracy of 94.6% (SD 7.1%). In-room accuracy was 95.9% (SD 5.4%) at the Home location, 95.8% (SD 5.7%) at the Institution location, and 91.9% (SD 9.2%) at the Business location. 80 out of the 100 sensed events achieved accuracies greater than 90%, while 33 were 100% accurate. Table 10-1, “In-Room Only” column, lists the individual accuracies for all 100 sensed events. These baseline accuracies are consistent with results depicted in Chapter 9, indicating that indeed, physical events in a room can be reasonably sensed from the vantage point of a single general-purpose sensor.

In-Room + Nearest Room Sensing

Next, I wished to investigate whether other sensors can offer complementary and confirmatory data to improve classification accuracies beyond that of what in-room sensors can provide alone (Q2). To do this, I trained, and then tested, the models using data from sensors *in the room* plus the *nearest room* (Figure 10-4). For example, to detect events in a bedroom, I utilize the bedroom sensor as well as the sensor in an adjoining bathroom. Across all events and rooms, I found an average classification accuracy of 96.4% (SD 5.3%). This is a significant gain of 1.8% over using only in-room sensor data. This yields a statistically significant result ($p<0.001$) using a paired t-test ($n=100$) against the baseline condition (*i.e.*, the in-room configuration). Broken down by location, the Home achieved 96.8% accuracy (a gain of 0.9%), Institution had 96.9% accuracy (+1.1%), and Business had 95.5% (+3.6%). Table 10-1, “In-Room + Nearest” column, provides full results.

Although these improvements in accuracy may seem small – single percent increases – it is important to note they constitute a significant decrease in error (roughly a third).

Closing the gap between 95% and 99% accuracy is often a substantial challenge. However, it also can make the difference between a technology that sits in the literature vs. meaningfully deployed in the real world.

All Room Sensing

Next, I took the second research question (Q2) to its full conclusion to observe the effect on sensing accuracy if *all* sensors in a building are utilized to detect events (Q3; Figure 10-4). For example, to detect events in a kitchen, I use sensor data from every room, including the kitchen. In this configuration, the average accuracy across all 100 events was 95.7% (SD 6.2%), a 1.1% increase from in-room sensing. This too is a significant improvement over in room sensing (paired t-test; p<.01). The improvement was consistent across the test locations: Home was 95.8% (+0.1%), Institution was 96.9% (+1.1%), and Business was 94.1% (+2.2%).

Though better than in-room accuracy, it was significantly worse than in room + nearest room sensing (paired t-test; p<.05). I suspect this dampening of performance is chiefly due to the substantial increase in machine learning features (from ten sensors, vs. just one or two), most of which likely contain no information power, introduce unwanted noise, and may lead to overfitting of some classifiers. Please also see Table 10-1, "All Rooms" column, for individual event results.

Nearest Room Only Sensing

The results discussed thus far suggest improved recognition accuracy when in-room sensors are complemented with other sensors, either nearby or throughout a building. This a positive result for the notion of deploying a constellation of sensors at a sparsity of roughly one per room. However, I also sought to investigate if it was possible to support recognition of events at densities of *less than one per room* (Q4)*,* potentially making deployments lower cost and even more practical. To start, I examined a *nearest room only* spatial configuration (Figure 10-4), where I include sensor data from the nearest room, but *not* the sensor in the room where the event actually occurred (Q1).

Across all rooms and events, *nearest-room-only* sensing accuracy is 63.4% (STD 31.4%), a 31.2% decrease from the baseline, *in-room* sensing condition, which is significantly worse (paired t-test; p<.001). Home location drops to 55.2% accuracy (-40.7%), Institution drops to 57.6% (-38.2%), and Business drops to 79.1% (-12.8%). For the latter location, I suspect a more compact physical footprint helped buffer losses, as the nearest sensor was often still fairly close, offering a greater chance to capture signals of interest. More interestingly, the drop in accuracy is not uniform across all sensed events. Indeed, 21 of the 100 events had unchanged accuracies or even improved in accuracy. If I consider an accuracy drop of less than 10% acceptable, 36 of the sensed events qualify, despite not having a sensor in the room where the event occurred, which is a promising result. Specifically, it suggests that for most events, in room sensing is required to achieve high accuracy, however there are some classes of events that produce physical manifestations that are detectable beyond the immediate room, allowing for remote detection and thus sparser sensor distributions, a topic I revisit in later discussion.

All But-In-Room

Building on the previous section, I also investigated the feasibility of supporting events when no sensor is present in a room (Q1), but leveraging *all other* sensors available at a location (Q3/Q4). In this *all but in-room* condition (Figure 10-4), average sensing accuracy degrades to 53.8% (STD 30.1%), a 40.8% drop-off from in room sensing. For the Home location, overall accuracy is 42.5% (-53.4%), 47.8% at the Institution location (-48.0%), and 73.3% for the Business location (-18.6%). This result is worse than just relying on the single nearest sensor (marginal significance, p=0.11). Similar to the *all rooms* condition, I suspect this accuracy degradation versus our *nearest-room-only* condition is from overfitting of the models due to excessive features, most of which are from sensors unable to capture any useful signal.

Learning Curves

I conducted a supplemental analysis to understand how different sensor configurations might impact the training of classifiers. For this, I computed per-day cumulative learning curves, which illustrate accuracy gains as more training data becomes available. More specifically, I use data collected on Day 1, and test it against data from Days 2-7, simulating classification accuracy as though the model only had one day to learn before being deployed. For Day 2, I train on data from Days 1 and 2, and test on

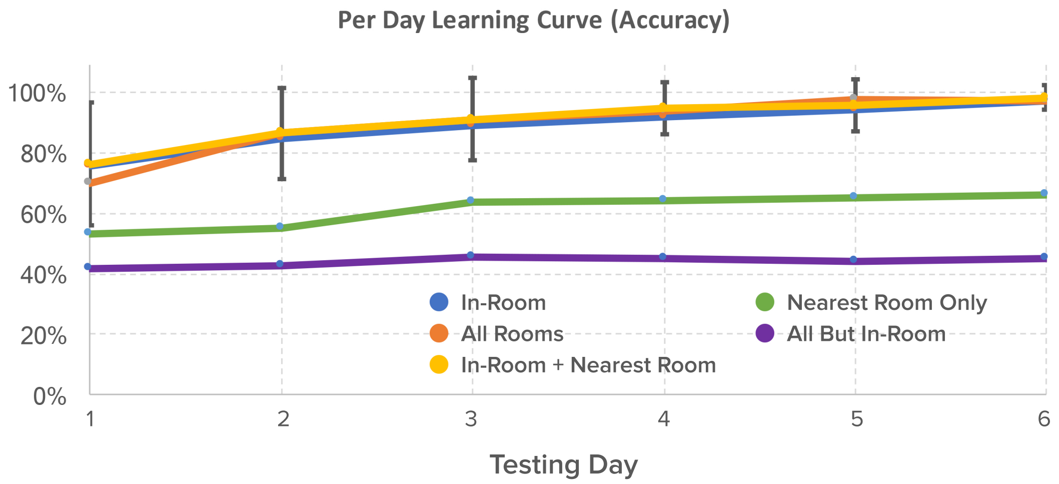


Figure 10-5. Accuracy with increasing days of training data for different sensor spatial configurations. Error bars illustrate how accuracy variance decreases over

the remaining data (Days 3-7). I repeat this process for all days in the training set, up to day 6, which is tested on day 7. The results (Figure 10-5) mirror previous results, with *in room*, *in room + nearest room* and *all room* spatial configurations performing the best. For these three conditions, accuracy appears to have a positive trajectory on day 7, suggesting deployments longer than a week might yield even stronger accuracies. Next best is training on *nearest-room-only* data, followed by *all-but-in-room* data, with both accuracies appearing to plateau by ~Day 3.

DISCUSSION AND IMPLICATIONS

Synthetic Sensors–the project most related to this research–demonstrated 96.0% accuracy across 38 sensed events at a single controlled location. The larger and considerably more ecologically valid study in this work yielded a similar 94.6% accuracy over 100 sensed events. Moreover, 54 sensed events achieved accuracies ≥99%, reinforcing the central premise of prior research that room-scale, general-purpose sensing can achieve useful accuracies *without* direct instrumentation.

Benefit of Constellations & Reinforcement

Moving beyond prior work, I quantified for the first time the benefit of leveraging other general-purpose sensors deployed at a location. In particular, I tested two ways to supplement data from in-room sensors: leveraging data from *one nearby sensor*, and also leveraging data from *all sensors* at a location. In general, I found this generally improves event recognition, by around 1.5% in my study. In some cases, the gain is significant, and perhaps key to unlocking usable accuracies. For example, the *faucet running* event in the Business Bathroom sees *in-room* accuracy of 65.2% jump to 94.6% when adding *nearest-room* data (in this case, due to plumbing inducing vibrations on an adjoining wall). Another example in the Business location is the *espresso machine running* event, which jumps from 69.8% to 95.0% (also due to vibrations induced in a neighboring room). These two examples, and 14 other events, improved by at least 5% in accuracy when allowing classifiers to utilize nearest-room sensor data, demonstrating the value of distributed sensor fusion. Moreover, this benefit is mutual: deploying in adjacent rooms *reinforces* recognition in both rooms.

More Sensor Data is Not Always Better

One of my initial hypotheses was that more sensor data would be better, and even though some sensors may be far away, they would still offer weak, but useful signals to support recognition. This did not appear to be the case in my deployment, as the results show that *in room + nearest room* sensing was significantly better than leveraging all available data (*i.e., all rooms* condition). As noted previously, I believe this is due to the substantial growth in machine learning features, which add little or no information power, impeding model training. I speculate this could be overcome with more training data, where perhaps weak and noisy signals could ultimately improve classification. However, end users will demand robust classification “out of the box”, and so it may be that deployments have to start with in-room sensing, slowly add nearest-room data, and finally transition to whole house data.

Benefit of Proximity

My results also serve to confirm that proximity to physical events is vital to enable robust classification when using a general-purpose sensor (the three best spatial conditions all took advantage of data from an in-room sensor). Importantly, this detriment cannot be overcome simply by having more sensors elsewhere, as illustrated by the *nearest-room-only* and *all-but-in-room* conditions. For example, despite offering classifiers nine times the sensor data, including from the nearest room, *all-but-*

in-room sensing is 40.8% worse on average than *in room* sensing (53.8% vs. 94.6%) – accuracies untenable for supporting end-user applications.

Overcoming Lack of Proximity

Interestingly, the accuracy drop when having to rely on out-of-room sensors is bi-modal, with some event classes essentially operating at chance (*i.e.*, no useful signal to operate on), while roughly a third of sensed events saw little or no degradation in accuracy. I suspected this behavior was correlated with how events manifested and propagated physically. To investigate this, I categorized the event classes into three properties of interest:

1) Is the event acoustically *loud*? Loud sounds will travel to adjacent spaces (*e.g.*, a door closing, a large appliance running), where they can be detected by other sensors.

2) Did the event cause *vibration*? Though often subtle, vibrations propagate through building structures (*e.g.*, a toilet flushing, laundry machine spinning) with much less degradation than sound through air. Additionally, this can travel between floors, and along walls/floor/ceilings to rooms not immediately connected. With sensors plugged into wall power sockets (which are typically mounted to a wall stud), they are in an ideal position to capture structural vibrations originating elsewhere.

3) Does the event produce signals that require *line of sight* (*i.e.*, a direct, unimpeded path between event and sensor). This is a property of some of our sensor's channels, such as non-contact temperature (*e.g.*, stove burner on) and infrared motion sensing (*e.g.*, occupancy).

I applied these property labels to the sensed events in Table 10-1; Figure 10-6 offers an accuracy comparison across spatial configurations broken out by these properties. Of the events that experience less than 10% drops in accuracy when no sensor is present in the room they occur, 90% are loud, vibrational or both. This suggests that these classes of physical event may be the use cases where in-room sensor placement can be

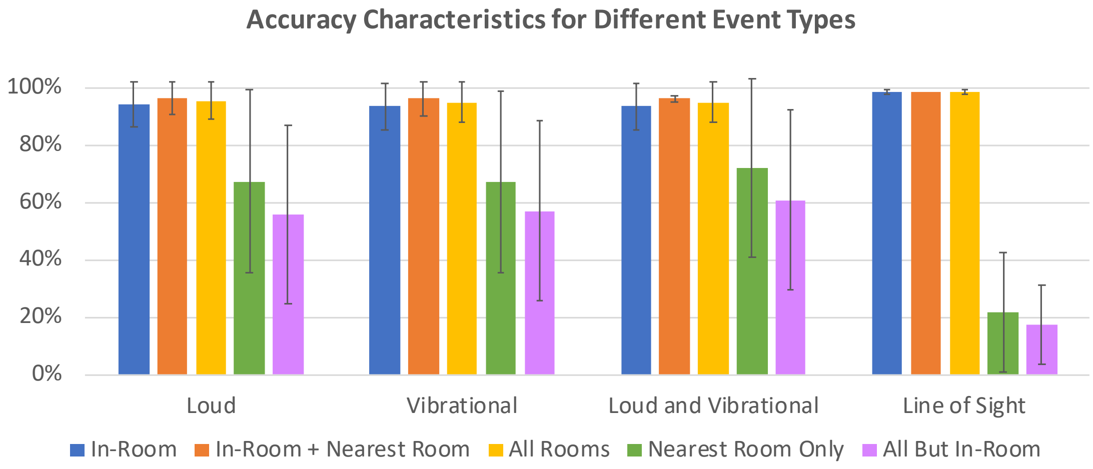


Figure 10-6. Accuracy across event properties and sensor spatial configurations.

skipped, and the sensor can be deployed in areas with more subtle signals. I caution this effect, however, only works well for noisy or strong vibratory signals (*e.g.*, doors closing, HVAC, motor-power appliances), and is not universal. For example, the toaster in the Home Kitchen dropped in accuracy from 97.6% *in room* accuracy to 86.6% *nearest-room-only* accuracy, despite emitting a completion chime. Although loud, and certainly audible in the adjacent room, there is nonetheless a substantial drop in accuracy, especially when there is any noise present in the adjacent room. Likewise, small vibrations, especially those local to a surface, such as typing, do not appear to have enough energy or coupling to structural members to be detected from outside of their room (or indeed even in the room, as seen in the case of *mouse moving on table*, with in-room accuracies of 84.0%).

Line of Sight

I identified five sensed events that *required* line of sight (through a feature selection analysis that showed a majority of information power was derived from sensor channels requiring line of sight). In all five cases, this was due to a reliance on non-contact thermal sensing to detect heat-related events (*e.g.*, fireplace on). When moving from in-room sensing *with line of sight*, to the nearest-room sensor with *no line of sight*, the average accuracy went from 97.6% to 17.3% (Figure 10-6), a much sharper drop than the 95 events not requiring line of sight (average *nearest-room-only* accuracy of

65.2%). This reinforces the notion that some events require not just in-room sensing, but line of sight sensing, and when this is impossible (*e.g.*, no available outlet, occlusion from furniture or occupants, or aesthetic reasons), it can render that event impossible to sense, even if leveraging all sensors in a constellation.

Redundancy

A final benefit of sparse sensor constellations is *redundancy.* In real world environments, sensors will lose connectivity, be occluded by people and furniture, get damaged, and outright fail. Ideally, we want ubiquitous sensing systems to gracefully continue when such failures occur, even if it means operating at reduced accuracies. My results suggest this is indeed possible, by transferring the responsibility to the next nearest sensor. One can see in the *nearest-room-only* spatial condition (which offers a proxy for this failure event) that 45 of the 100 events operate at ≥80% accuracy (22 events at ≥ 95%), which may be acceptable in such failure modes, as opposed to a loss of sensing entirely.

Placement Guide

Although I believe an ideal sensing system should require no special placement consideration (*i.e.,* essentially "plug and play"), there are nonetheless lightweight optimizations that can be considered to increase the chances of a successful deployment. The most immediate rule of thumb is for the sensor to be as close as possible to events of interest, as borne out by the main study results. Next, users should consider the relative signal strength of events, *e.g.,* subtle noises and vibrations are less likely to travel and must contend with background noise. If a user can hear or feel an event, that is a strong indicator of actionable SNR. Likewise, for events requiring line of sight, it may be that sensors have to move to a less desirable location in order to have visibility. This tradeoff is impossible to quantify, as spaces are so varied. With regards to signals that are able to be sensed through walls, such as plumbing and highly vibratory appliances, my results still suggest in-room placement is the strongest avenue. However, if out-of-room deployment is desired, adjacent walls are the next most reliable instrumentation point.

I acknowledge this guide offers few concrete recommendations, largely because spaces (their layout and construction) are so varied, which precludes all but the most generic recommendations. This complexity might suggest that companion apps are needed for rapid, end user evaluation of sensor placements.

Performance Across Rooms

In my study design, I included a large number of rooms – of varying size and function – so as to provide a more generalizable result that was not tightly coupled to a specific location or event type. Furthermore, as noted earlier, this provided an uncontrolled, but real-world distribution of electrical outlets for sensor deployment. Without doubt, some rooms were more favorable, while others were more challenging. With my study data and accuracy results, I can explore this distribution to better understand underlying effects. For simplicity, I use results from our *in room only* sensing condition.

Foremost, when averaging accuracies of all classifiers operating in a room, one can see that room accuracies vary from 65.2% to 100% (mean 94.6%, SD 7.1%). Of course, rooms vary in size, function, sensor placement and challenge of the sensed facets, and so a more reliable test of generalizability across environments is to look at sensed events that were common across rooms. For example, I ran *door closed* classifiers in 17 rooms, which varied in accuracy from 83.2% to 100.0% (mean of 94.3%, SD 6.1%) – an extremely tight clustering which suggests this signal is robust across locations and sensor placements.

Conversely, *faucet running*, the next most common classifier (included in 8 rooms) varied in accuracy from 65.2% to 100.0% (mean of 88.8%, SD 10.2%) – a wider distribution of accuracies. This is almost certainly due to a much weaker signal (vibro-acoustically) than a door closing and latching shut. In locations where the sensor happened to be placed far from the faucet in the room, and especially when decoupled from the wall containing the plumbing stack, accuracy dropped substantially (94.6% when proximate, 80.1% when far). Thus, sensor placement does appear to be important factor in accuracy for some signals, much more so than the room they operate in, and it suggests that users should be encouraged to place sensors nearer to areas of interest when outlets permit.

STUDY LIMITATIONS AND FUTURE WORK

Although I conducted a large deployment in 30 rooms across three buildings, this is still a small subset of the diverse and complex environments present in the real world. Likewise, the 67 unique sensed events do not represent all activities that users and smart buildings may wish to monitor. Although I believe the deployment study is a significant contribution to this long-standing problem domain, tackling key questions, many challenges remain before such sensing can move out into the real world.

For example, many physical events can be subtle or imperceptible to the types of sensors I deployed (*e.g.*, is the cat sleeping, is the cooler out of water). Indeed, most of the sensed events relied on sound and vibration, but none used change in magnetic field, temperature, or humidity (despite being captured by the sensor board I used). This was partly due to the fact that the events I studied were on the milliseconds-to-seconds timescale, whereas interesting variations in signals like temperature and humidity typically happen on the order of hours or days. Such longer-term sensing might be able to detect events such as the ingress of moisture or poor insulation, which *e.g.*, home owners might wish to be notified about.

Additionally, events in the real world are chaotic, noisy, and can happen simultaneously, whereas the events captured in this study were mostly isolated. Although there were uncontrolled background activities happening during the study, I did not explicitly perform multiple simultaneous events, which was combinatorically explosive and logistically challenging. Anecdotally, however, I can report that simultaneous recognition of events occurring in different rooms is straightforward when using any of the spatial configurations utilizing in-room sensors (where the in-room sensor dominates classification). However, recognizing multiple simultaneous events in the same room remains a significant challenge. I have started to examine signal processing techniques such as adaptive background subtraction [93] and source separation [185] to explore this capability in future work.

Finally, I note that several events in the deployment were common across rooms and locations, such as detecting doors closing and faucets running. Many of these exhibited similar sound and vibrational properties regardless of location. For instance, a door closing is typified by a loud sound and vibrational impulse, where as a faucet produces a steady white-noise-esque sound with low frequency vibrations. I believe these types of classes could be learned by a “universal” model that would require no on-site training, a challenge I address in the next Chapters.

CONCLUSION

Many previous distributed sensing systems have envisioned densely instrumented environments. However, through clever sensing, I believe many of the same benefits can be achieved in a more practical and less obtrusive manner through sparse arrays of general-purpose sensors. In this work, I explored the efficacy of this approach, wherein a constellation of sensors work synergistically to support sensing of physical events across multi-room environments. I found that one sensor per room can unlock high accuracies across a wide variety of sensed events, supporting and extending prior study results. More interestingly, by leveraging sensors in other rooms, accuracy can be further improved. I also found that for some classes of events, detection is possible even when no sensor is present in the room, instead relying entirely on other sensors in the building. Taken together, these results suggest that near-complete sensor saturation of everyday spaces is within reach using sparse sensor constellations.

PART III:

REDUCING USER BURDEN

II. TOWARDS SCALABILITY

Part III of this thesis builds upon the work discussed in the previous Chapters. In Chapter 9 and Chapter 10, I showed how Synthetic Sensors offers a practical vehicle for general-purpose sensing. However, its general sensing and machine learning approach fundamentally incurs a heavy data collection and training burden. To build robust models, large amounts of data is required, which is time consuming, error-prone, and difficult to scale (especially if end users are involved). In the next two Chapters, I aim to reduce data collection and training burden by leveraging 1) sensors that already exist, 2) leveraging large amounts of pre-existing, high-quality data, and 3) leveraging semi-supervised machine learning strategies to further reduce user effort.

PART III: ROADMAP

In the next two Chapters, I describe Ubicoustics and Listen Learner in detail. Specifically, I describe the different sensing opportunities and user-burden implications afforded by each approach, ultimately laying the ground work for more accurate and more scalable context-driven systems.

Chapter 12: Ubicoustics

In Chapter 9, I showed how acoustic information constitutes a significant portion of the classification merit for activity recognition. I leverage this insight as the next step towards making context-driven ubiquitous sensing more practical. It turns out that microphones are one of the most common sensors found in consumer devices but weakly utilized as a contextual sensing platform. Further, microphones are the exact

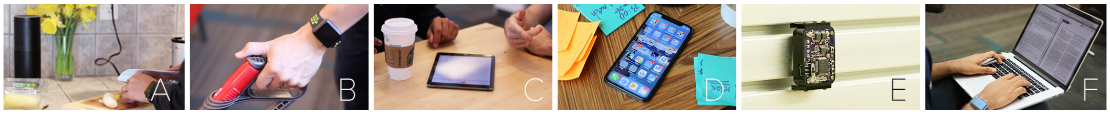


Figure 11-1. Microphones are one of the most common sensors found in most devices. Leveraging this signal source could enable plug-and-play activity recognition, requiring no *in situ* training, across diverse platforms.

sensors used to record *professional sound effect libraries* traditionally used in the entertainment industry. Unlike internet-mined datasets (which are noisy and weakly labelled), professional sound effects contain noise-free, well-segmented, well-labeled, and highly diverse everyday sounds. Because of its atomic properties, they can be easily transformed and projected into hundreds of realistic environments (*e.g.*, applying impulse functions and mixing with background tracks), synthetically growing a dataset with a high signal-to-noise ratio.

Armed with a corpora of context-specific, high-quality data, I fine-tune existing state-of-the-art deep learning models (*e.g.*, leveraging pre-trained weights), enabling sound activity recognition without *in-situ* training. My evaluations, conducted across 30 activities, 50 locations and multiple device platforms (*e.g.*, from watches, to smart speakers, and IoT sensors), show that models tuned with Ubicoustics can achieve superior accuracy than those trained on internet-mined data alone. More surprising is that system accuracy is comparable to human-level performance (baselined on 600 human coders). By leveraging sensors and high-quality data that already exist, Ubicoustics offers another promising path towards practical context sensing.

Chapter 13: One-Shot Activity Recognition

Finally, a key challenge for acoustic activity recognition is building classifiers that can recognize highly localized events with minimal user intervention or *in-situ* training. To

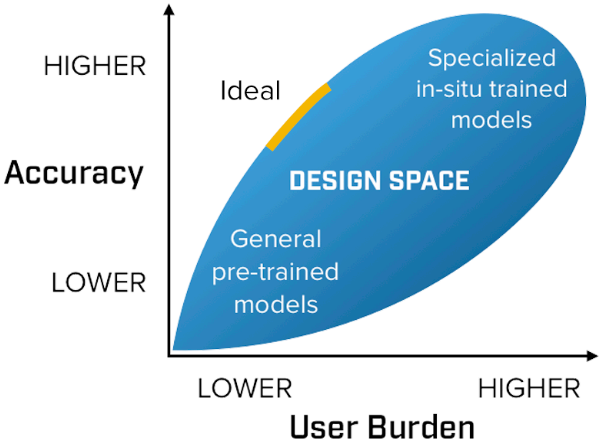

HIGHER
Accuracy
LOWER
Ideal
Specialized in-situ trained models
DESIGN SPACE
General pre-trained models
LOWER
HIGHER
User Burden

Figure 11-2. A taxonomy depicting the different approaches for activity recognition, along with their accuracy (Y-axis) and user-burden implications (X-axis). An ideal approach (top-left) supports higher accuracies with minimal user burden.

train these classifiers, two predominant approaches have been proposed, with characteristic accuracy and user burden implications (Figure 11-2). First is to train a system manually, after it is deployed, most often by demonstrating different activities and having the user provide class labels (Figure 11-2, top-right). Because data is collected *in-situ*, accuracy tends to be quite high. However, the burden to the user is also high. The other approach is to provide users with classifiers that are already trained, and work "out of the box" (Figure 11-2, bottom-left).

This is achieved by training a classifier on a large, general corpus of acoustic data. Because the classifier has no data for a user's specific environment, it tends to be less accurate, but the burden to the user is low. In Listen Learner, I propose and evaluate a balanced approach that seeks to provide high classification accuracy, while minimizing user burden. This approach requires no up-front data, and instead, learns acoustic events over time and requires no manual demonstration. This approach learns events *in-situ*— it is highly tuned to the environment and objects of interest, and thus can offer superior accuracy than general, pre-trained classifiers.

12. UBICOUSTICS: PLUG-AND-PLAY ACOUSTIC ACTIVITY RECOGNITION

INTRODUCTION

As mentioned in the previous Chapter, microphones are the most common sensor found in consumer electronics today, from smart speakers and phones to tablets and televisions. Despite sound being an incredibly rich information source, offering powerful insights about physical and social context, modern computing devices do not utilize their microphones to understand what is going on around them. For example, a smart speaker sitting on a kitchen countertop cannot figure out if it is in a kitchen, let alone know what a user is doing in a kitchen. Likewise, a watch worn on the wrist is oblivious to its user *e.g.,* cooking or cleaning. This inability for "smart" devices to recognize what is happening around them in the physical world is a major impediment to them truly augmenting human activities.

Real-time, sound-based classification of activities and context is not new. There have been many previous application-specific efforts that focus on a constrained set of recognized classes [135, 244, 298, 282]. For example, Ward *et al.* [282] developed a microphone-equipped necklace in conjunction with accelerometers mounted on the arms that could distinguish between nine workshop tools. In these types of constrained uses, the training data for machine learning is generally domain-specific and captured by the researchers themselves.

I sought to build a more general-purpose and flexible sound recognition pipeline – one that could be deployed to an existing device as a software update and work immediately, requiring no end-user or *in situ* data collection (*i.e.*, no training or calibration). Such a system should be "plug-and-play" – *e.g.*, plug in your smart

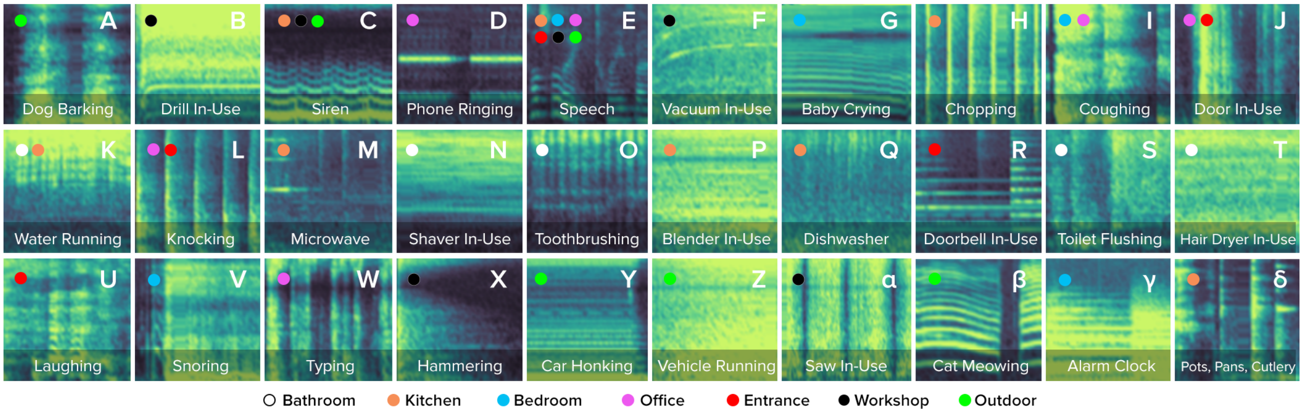


Figure 12-1. Example log Mel spectrograms (based on [106]) of 960ms audio for 30 test event classes. These 96×64 input vectors are used to tune a VGG-16 model pre-trained on YouTube-8M.

speaker, and it can immediately discern all of your kitchen appliances by sound. This is a challenging task, and very few sound-based recognition systems achieve usable end-user accuracies, despite offering pre-trained models that are meant to be integrated into applications (*e.g.*, Youtube-8M [5], SoundNet [15]).

I propose a novel approach that brings the vision of plug-and-play activity recognition closer to reality. This process starts by taking an existing, state-of-the-art sound labeling model and tuning it with high-quality data from professional sound effect libraries for specific contexts (*e.g.*, a kitchen and its appliances). I found professional sound effect libraries to be a particularly rich source of high-quality, well-segmented, and accurately-labeled data for everyday events. These large databases are employed in the entertainment industry for post-production sound design (and to a lesser extent in live broadcast and digital games).

Sound effects can also be easily transformed into hundreds of realistic variations (synthetically growing a dataset, as opposed to finding or recording more data) by adjusting key audio properties such as amplitude and persistence, as well mixing sounds with various background tracks. I show that models tuned on sound effects can achieve superior accuracy to those trained on internet-mined data alone. I also evaluate the robustness of this approach across different physical contexts and device categories. Results show that this system can achieve human-level performance, both in terms of recognition accuracy and false positive rejection.

Overall, the research discussed in this Chapter makes the following contributions in the area of activity recognition and ubiquitous sensing:

1. A real-time, activity recognition system that demonstrates accuracies and class diversity approaching end user feasibility, requiring no in-situ data collection, and using nothing but commodity microphones for input.

2. A comprehensive suite of experiments that quantify the performance of our system across 4 data augmentations, 7 device categories, 7 location contexts, and 30 recognition classes, providing insights into the feasibility of sound-based activity recognition that generalize beyond our implementation.

3. In addition to conventional testing with existing sound datasets, this exploration moves beyond prior work by capturing a new, real-world dataset with improved ecological validity. I also benchmark these results against human accuracy (600 participants) to contextualize performance.

WHY SOUND EFFECTS?

Properties of Sound Effects

First, sound effects are ***atomic*** – a clip labeled as "door knock" or "cat meow" is tightly segmented and contains only that one sound. Sound effects are also ***pure*** – clips are generally recorded in professional studios and are devoid of artifacts like background noise and echoes. Such purity is mandatory, as the sounds are meant to be transformed and composited into richer soundscapes. Third, sound effect libraries are ***diverse***; post-production sound editors search for the perfect sound based on the materials in the scene and mood of the shot. For this reason, libraries often contain hundreds of variations of the same sound effect.

Properties of Sound

Sound itself has three important properties. Foremost, sound data is ***scalable***, both in amplitude and duration. Second, sounds are ***transformable***, able to be projected into synthetic environments by altering their equalization ("EQ"), reverb and damping (*i.e.*, persistence). In this manner, an effect can be made to sound like it is in a furnished living room or small bathroom. Finally, audio is innately ***additive*** (though not subtractive), allowing two or more effects to be trivially blended. One can take a sound effect and trivially combine it with an ambient track of a bustling market, tranquil forest, or ambient hum of HVAC.

Taken together, this means one can take a *single* sound effect, and transmute it into *hundreds* of realistic variations. When applied to entire sound effect libraries, it is possible to achieve a scale of data ideal for training deep learning models, while retaining all of the benefits of a highly-curated corpus. I show in subsequent evaluations that such models outperform those trained on comparably-sized, internet-mined data.

Categories of Sound Effects

There are three main categories of sound effects: *hard*, *natural*, and *background* sounds [220]. Hard sounds are closely linked to on-screen action (*e.g.*, cough, door closing). Natural sounds are subtle effects that add realism to a scene and action (*e.g.*, leaves rustling, fabric chafing). Finally, background sounds (*e.g.*, HVAC hum, engine noise) are used to build immersive soundscapes, smooth breaks in dialogue, and anchor visual transitions. In Ubicoustics, I rely on *hard* sounds for training data and use *background* sounds for our mix augmentations, described later.

Note that many professional sound effects are produced through Foley – the recreation or simulation of a sound. In interviews with sound production and Foley artists (with on-screen credits) conducted as part of background research, I found that Foley is generally used when scenes require precise audio-visual timing (*e.g.*, person walking on snow) or for actions where natural examples are rare (*e.g.*, blood spatter). I confirmed that the commonplace environmental sounds that I focus on are rarely Foleyed, as these are easier and cheaper to record than simulate (*e.g.*, coughing, toilet flush, blender, vacuum).

RELATED WORK

There are many approaches for sensing human activity, from special-purpose sensors such as geophones [134], water pressure sensors [85], powerline sensing [93] and RF tags [157], to generic approaches such as computer vision (see Chapter 8). In this section, I focus on sound-driven methods that most closely relate to my efforts.

Sound Event Classification

There are a number of machine learning models that leverage publicly-available audio datasets to classify sound events. For instance, Salamon *et al.* [228] employed scattering transform for urban sound classification (classes include jackhammer and car horn), while Foggia *et al.* [82] employ a bag of words-based approach with a multi-class SVM for detection for surveillance uses (*e.g.*, screaming, glass breaking, gunshot). Sound has also been used for scene and context recognition, for example, Eronen *et al*. [76] used clustering and hidden Markov models for audio context recognition (including outdoors, vehicles and homes).

More recently, deep learning has been applied to sound event classification. For example, Lane *et al.* leverage several fully connected layers for audio sensing in DeepEar [135], which focused on classification of high-level categories. The closest to Ubicoustics is a four-class "voicing, music, water and traffic" set, which is quite different from the fine-grained activity classes I aim to support (toilet flushing, chopping, knocking, coughing, microwaving, typing, etc.). SoundNet [15] used video data and computer vision to identify objects in a scene, and then used the resulting labels to learn sounds.

Convolutional neural networks (CNNs) have also been employed for sound event classification. McLoughlin *et al.* [173], Ephrat *et al.* [74] and Phan *et al.* [204] present CNN-based deep learning architectures, while Parascandolo *et al.* [195] use Convolutional Recurrent Neural Networks (CRNNs). Hershey *et al.* [106] compare various CNN architectures for acoustic event detection and benchmark on the AudioSet dataset [5]. The Never-Ending Learner of Sounds (NELS) [71] crawls the web, continuously training a CNN using semi-structured online data, creating an index of sounds. I leveraged this prior work heavily in the design of Ubicoustics.

Sound data augmentation has also been previously explored to improve the robustness of acoustic classification models. For example, McFee *et al.* [172] used pitch shift, time stretch, dynamic range compression, and background noise addition for music. Salamon *et al.* [228] used the same augmentation techniques in conjunction with a

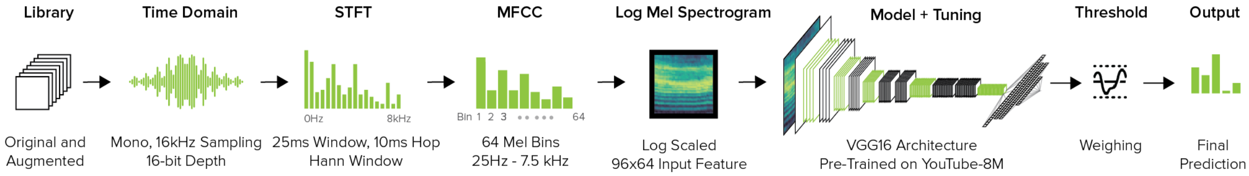

Library
Time Domain
STFT
MFCC
Log Mel Spectrogram
Model + Tuning
Threshold
Output
0Hz
8kHz
Bin 1 2 3
64
Original and Augmented
Mono, 16kHz Sampling 16-bit Depth
25ms Window, 10ms Hop Hann Window
64 Mel Bins 25Hz - 7.5 kHz
Log Scaled 96x64 Input Feature
VGG16 Architecture Pre-Trained on YouTube-8M
Weighing
Final Prediction

Figure 12-2. The end-to-end process of Ubicoustics, which leverages professional sound effect libraries to fine-tune a CNN-based classifier for activity recognition.

CNN, evaluating on the UrbanSound8K dataset. I drew inspiration from these prior efforts when developing my set of augmentations.

Real-Time, Sound-Driven Activity Recognition

Beyond "offline" event labeling, sound has long been used for inferring *real-time activities*. Likely the most pervasive system is ShotSpotter [244], which provides gunshot detection and localization for law enforcement. For a more general set of activities, Storkx *et al.* [253] used Mel-frequency cepstral coefficients (MFCC) with non-Markovian ensemble voting to discriminate among 22 human activities from bathroom and kitchen contexts (including blender mixing, pouring water, sorting dishes). Lu *et al.* [165] demonstrated speech, music and ambient sound detection using a phone microphone. Synthetic Sensors (Chapter 9) used a custom board equipped with acoustic sensing capabilities (among many sensor channels) to distinguish 38 environmental events. There have also been portable systems using body-worn microphones and accelerometers, including the Mobile Sensing Platform [40], BodyScope [298] and work by Ward *et al.* [282]. This prior work requires training within a user's environment and focuses on specific domains and devices.

UBICOUSTICS

I now describe my process for enabling ubiquitous acoustic activity sensing, which is illustrated in Figure 12-2.

Contexts and Classes

The first step is defining a context of use (*e.g.*, construction site, hospital, dentist office, café), which limits the set of classes to be recognized. These classes can then be mined from professional sound effect libraries, often just by using the name of the class itself as the search term, though I found more clips can be identified with some keyword variation (*e.g.*, not just "faucet", but also "tap running", "water flowing", etc.). The result is a large corpus of sound effects covering the classes of interest.

Sound Pre-Processing

Once the corpus is assembled, I standardize all sounds into a single format, as libraries come in different file formats (*e.g.*, WAVs, AIFFs), bit depths (8 to 32 bits), sample rates (16-48 kHz), and number of channels (mono to 5.1 surround sound). I selected 16 kHz mono (16-bit) as the standard format, as I found this to be the lowest common denominator. Once converted, I removed silences greater than one second anywhere in the clip. At this point, I have what I call an "*original*" sound set (*i.e.*, no augmentations).

Amplification Augmentation

To begin to add variation to a sound dataset, I first apply an amplitude augmentation. I produce two variations for each input sound, one quieter (25% of original volume) and one louder (raising peak amplitude to -0.1dB).

Persistence Augmentation

The next augmentation modifies the persistence of a sound effect (Figure 12-3), which includes reverberations and non-linear damping (see *e.g.*, [69, 223] for more background). By modifying these parameters, I can simulate sounds in a variety of physical spaces (*e.g.*, kitchen, hallway, bathroom).

I used two methods to generate realistic persistence transformations. First, I selected six professional "reverb" effects provided by Adobe Audition (Figure 12-3, grey dots). Second, I created four custom convolutional reverbs by capturing impulse

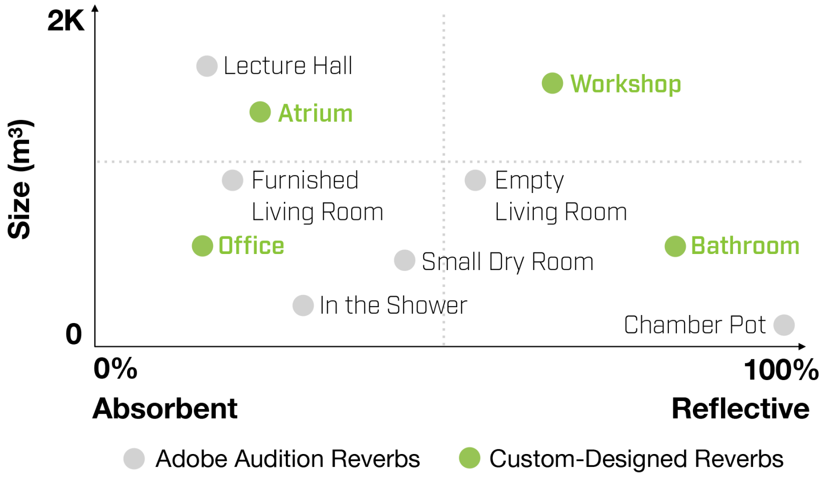


Figure 12-3. Persistence of sound transformations plotted by room size and sound absorption.

functions [12] in exemplary rooms: a bathroom, large atrium, workshop and small office (Figure 12-3, green dots). To create these custom effects, I placed a speaker/microphone setup in a target room. I then emitted a sinusoidal sweep and recorded the frequency response, which I de-convolved into an impulse function. I plot the ten persistence transformations in Figure 12-3 by the size of room and absorption level. In total, this augmentation process yields ten new sounds for every input sound.

Mixing Augmentation

The next augmentation blends sound effects with background sounds I sourced. This mixing process introduces foreign elements to original sounds, adding variability. Each input sound is mixed with a randomly selected background segment, which includes indoor (*e.g.*, HVAC), outdoor (*e.g.*, birds chirping), urban (*e.g.*, traffic), and social (*e.g.*, cafe) background noise. This way, I create six new sounds for every input sound.

Combining Augmentations

Finally, I can stack and combine augmentations, creating even greater variety. For example, I can take a “brushing teeth” sound effect and make it louder, apply a bathroom-like reverberation, and add background noise from an exhaust fan. Note the order of operations is important. For instance, background tracks generally already include reverb, as they are recorded on location, and thus re-projecting them into a second environment leads to less realistic output.

Featurization

Once the sound dataset is assembled, I compute its features. There are many existing featurization stacks for audio data; in our implementation, I chose the method described in Hershey *et al.* [106]. First, I segment files into 960 ms audio segments and compute short-time Fourier Transforms for each segment (using a 25 ms window and step size of 10 ms), which yields a 96-length spectrogram. I then convert the linear spectrogram into a 64-bin log-scaled Mel spectrogram and generate a 96×64 input frame for every 960 ms of audio, which is fed into our classification model.

Model Architecture

I build upon the YouTube-8M VGG-16 [106] model, which is a variant of the VGG16 architecture trained on 8 million YouTube videos. The architecture contains four convolutional layers (3×3 kernel, step size = 2, depth = 64, 128, 256, and 512, ReLU activation [186]), with intermediary max pool layers [88], and a 128-wide fully connected embedding layer. I modified this pre-trained model by removing the last fully connected layer and replacing it with a custom fully connected layer. Finally, I tune the entire architecture with professional sound effect datasets.

Devices

In a commercial implementation, I envision models running locally on devices, as opposed to streaming data to cloud infrastructure (common approach). Local classification has obvious latency and privacy benefits. As a proof of concept, I deployed the model to three exemplary devices spanning a range of computational abilities: Apple MacBook Pro 2017, iPhone 7 smartphone, and Raspberry Pi Zero W with a ReSpeaker dual mic shield [240]. The models run at 15.2, 8.3 and 0.7 frames per second on these devices respectively. This performance is already sufficiently granular for most of the activities I studied (which last on the order of seconds), and suggests that with additional engineering and optimization, interactivity is possible.

EVALUATION

I sought to answer several key questions: What is the performance of a classifier tuned with sound effects? Does sound augmentation improve performance? How well does the model perform when tested on live, real-world data? Does the model work across different devices? And how does the technique compare to human accuracy?

Contexts. For our evaluation, I selected seven location contexts in which everyday activities occur: 1) bathroom, 2) bedroom, 3) house entrance, 4) kitchen, 5) office, 6) outdoor and 7) workshop. These contexts offer realistic scenarios with constrained event classes. For instance, it is highly unlikely for a "blender" event to occur in a bathroom, or for "chopping" to happen in a bedroom. This allows one to tune models for particular contexts with more tractable class sets.

Classes. For each context, I selected commonplace events using the following selection criteria: a) does the event happen frequently in that context? b) does it produce enough acoustic energy to be heard by a microphone? and c) can knowledge of the event enable useful applications? In total, I selected 30 events across seven contexts (Figure 12-1).

Sound Sources. There are dozens of large sound effect libraries to draw upon for tuning data. As a representative cross-section, I selected five libraries that were available online or licensed by our institution, listed in Table 12-1.

Name	Total Sounds	Sounds Used	Hours Used
BBC Sound Library [18]	29K	740	1.9
Network Sound Library [187]	10K	492	1.3
SoundSnap [248]	250K	4072	10.4
FreeSound [83]	372K	8929	22.2
AudioSet [5]	2000K	7899	18.2

Table 12-1. The sound sources I used for our datasets.

Tune and Test Sets. From the sources listed in Table 12-1, I extracted sound data covering the 30 classes. I split this corpus into train (*i.e.*, tune) and test sets: *SFX-Orig* and *SFX-Test*. These models are also tuned on different augmentations of *SFX-Orig*: amplification (*SFX-Amp*), persistence of sound (*SFX-Persist*), and mixing (*SFX-Mix*).

I also created a corpus containing all augmentations (*SFX-All*), comprising ~500 hrs. of sounds clips for our 30 classes. These datasets are summarized in Table 12-2.

Set Name	Contains	Data Hours
SFX-Orig	Processed, but otherwise unaugmented sounds	54.6
SFX-Amp	SFX-Orig + Amplitude Augmentations	152.0
SFX-Persist	SFX-Orig + Persistence Augmen. (15% draw)	136.2
SFX-Mix	SFX-Orig + Mix Augmentations (75% draw)	300.6
SFX-All	SFX-Orig + SFX-Amp + SFX-Persist + SFX-Mix	479.5
SFX-Test	Unaugmented sounds; holdout test set.	8.8
In-the-Wild Test	Sounds recorded on seven exemplary devices (see Table 3); holdout test set.	12.3

Table 12-2. Summary of datasets I created for our evaluations.

Test Devices. In order to test the robustness of the models across different microphones, placements, and platforms, I developed software to capture and stream audio from a diverse set of hardware platforms (Table 12-3). These devices connected over various means to an independent laptop capable of recording synchronized streams and performing live classification.

Device Type	Implementation	Comm.
Smartphone *(two placements)*	iPhone 5C, Swift iOS app	Wi-Fi
Smartwatch	LG W100, Android app	Bluetooth
Smart speaker	Jabra Speak 410	USB
IoT Sensor	Custom hardware	Bluetooth
Laptop	MacBook Pro 2013, Python app	Wi-Fi
Tablet	iPad 3, Swift iOS app	Wi-Fi

Table 12-3. Devices I used to capture data for our *In-The-Wild Test* set. Two identical smartphones were used to record on-table and in-pocket data.

Collecting In-the-Wild Sounds

In addition to evaluating our models on *SFX-Test* (8.8 hours of mined sound effects), I also sought to test on more ecologically valid “in-the-wild” dataset, captured not in a studio, but with microphones found in real-world devices recording in real-world environments. In response, I recruited 12 participants (mean age 29.3) who performed or triggered events across 50 rooms, spanning dozens of homes and buildings. The experimenter used a laptop to synchronously capture audio data from the seven test devices (Table 12-3). The interface allowed the experimenter to demarcate the start and end of events, as well as enter a ground truth label.

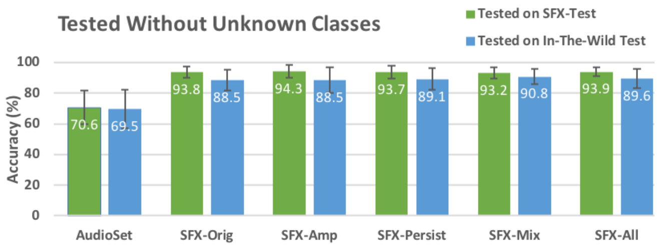


Figure 12-4. Recognition accuracies when evaluated without any unknown classes and checkpointed on best model.

When collecting data in a room (*e.g.*, kitchen 5), devices were placed in a realistic fashion. For example, the laptop, tablet and smart speaker were placed on a logical flat surface, while participants wore the smartwatch, and the IoT sensor was plugged-in to a nearby power outlet. For the smartphone category, I captured data for two placements (using two identical phones): a) phone in a participant's pocket, and b) phone on a surface. Devices were never more than 3 meters from an event source, and I avoided making changes to the physical layout (no special tables, no appliances moved, etc.).

In each location, I collected three rounds of data per event, in a random order. Sometimes this was activating an appliance (*e.g.,* running a microwave) while other times it was a physical task (*e.g.,* chopping vegetables). In all cases, the materials and equipment were participants' own. For events such as "coughing" and "laughing," I asked users to perform the action as naturally as possible. I excluded events that were challenging to induce (*e.g.*, baby crying, hazard alarm). All data was collected between 10am and 8pm; other occupants were free to go about their daily routines, which injected some natural noise. In total, I collected 12.3 hours of labeled/segmented data, which I call *In-The-Wild Test*.

RESULTS AND DISCUSSION

I now describe the results from a series of integrated experiments. A summary of main study results can be found in Figures 12-4 through 12-7. First, I evaluate the "plug-and-play" accuracy of Ubicoustics, including rejection of unknown sounds. I then compare performance to human annotators, which serve as a gold standard. Finally, I investigate the effects of augmentations, device categories, and location contexts.

Accuracy

For all accuracy metrics, I use clip-level prediction. More concretely, I record a model's output across an entire sound clip and return the top predicted result, based on cumulative confidence. As noted above, I use a dedicated model for each context (tuned only for classes that belong to that particular context). At the end of every tuning epoch, I checkpoint the model against both the *SFX-Test* and *In-the-Wild Test* datasets respectively and report the accuracy of the best performing epoch (a common, but artificial method we improve upon in the next section).

Overall, per-context models tuned on *SFX-All* and tested on *SFX-Test* achieved an average accuracy of 93.9% (SD=3.7%; Figure 12-4, *SFX-All*, green bar). When tested on *In-the-Wild Test,* average accuracy dropped to 89.6% (SD=6.3%; Figure 12-4, *SFX-All*, blue bar). When I tune the model using only AudioSet data leveraging the 30-class setup, the system achieves an average accuracy of 70.6% and 69.5% on *SFX-Test* and *In-the-Wild Test,* respectively (Figure 12-4, AudioSet). This result underscores the significant boost in accuracy when tuning on sound effect libraries. If I disregard context, and tune/test on all 30 classes, the *SFX-All* tuned model achieves an accuracy of 82.1% and 68.4% on *SFX-Test* and *In-The-Wild Test* respectively.

Better Estimating Real-World Accuracy

Although the aforementioned 89.6% accuracy follows a standard evaluation procedure, it does not offer a fair depiction of "plug-and-play" accuracy, as one would experience

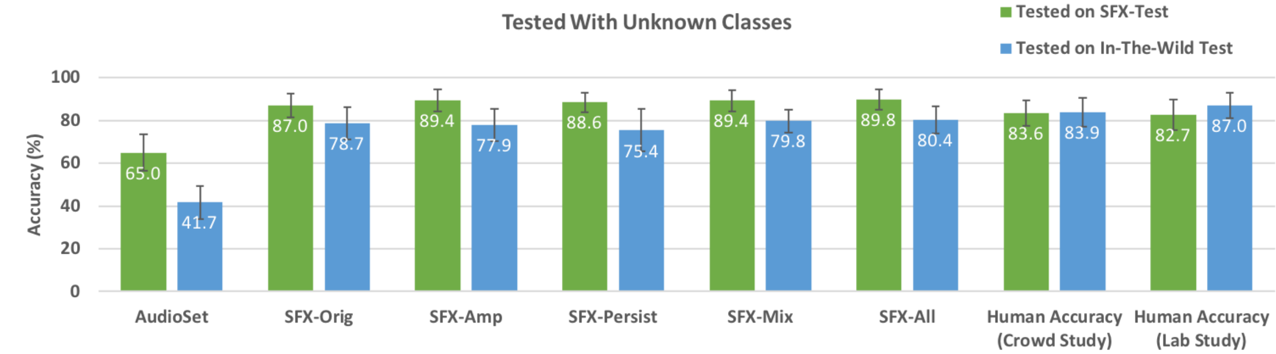
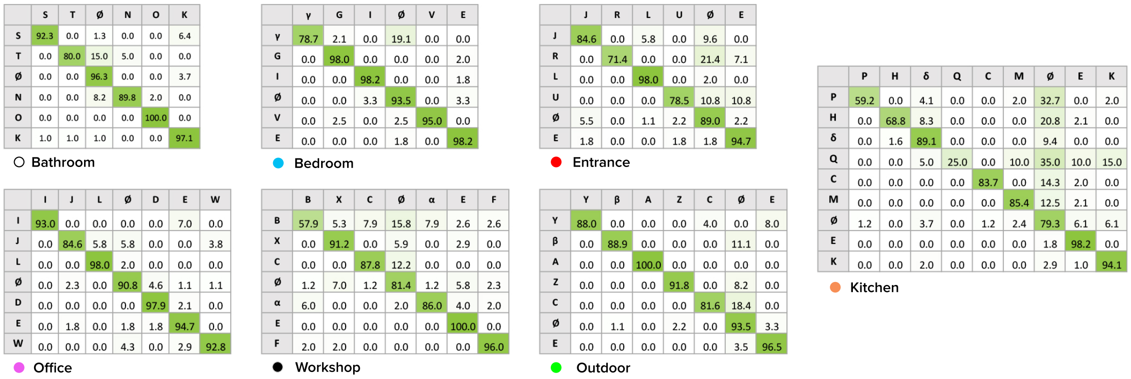


Figure 12-5. Recognition accuracies when evaluated with unknown classes comprising 20% of the test data. *In-the-Wild Test* is checkpointed on *SFX-Test* to remove checkpoint bias (*SFX-Test* is checkpointed on *SFX-Test*).



	S	T	Ø	N	O	K
S	92.3	0.0	1.3	0.0	0.0	6.4
T	0.0	80.0	15.0	5.0	0.0	0.0
Ø	0.0	0.0	96.3	0.0	0.0	3.7
N	0.0	0.0	8.2	89.8	2.0	0.0
O	0.0	0.0	0.0	0.0	100.0	0.0
K	1.0	1.0	1.0	0.0	0.0	97.1

○ Bathroom

	γ	G	I	Ø	V	E
γ	78.7	2.1	0.0	19.1	0.0	0.0
G	0.0	98.0	0.0	0.0	0.0	2.0
I	0.0	0.0	98.2	0.0	0.0	1.8
Ø	0.0	0.0	3.3	93.5	0.0	3.3
V	0.0	2.5	0.0	2.5	95.0	0.0
E	0.0	0.0	0.0	1.8	0.0	98.2

● Bedroom

	J	R	L	U	Ø	E
J	84.6	0.0	5.8	0.0	9.6	0.0
R	0.0	71.4	0.0	0.0	21.4	7.1
L	0.0	0.0	98.0	0.0	2.0	0.0
U	0.0	0.0	0.0	78.5	10.8	10.8
Ø	5.5	0.0	1.1	2.2	89.0	2.2
E	1.8	0.0	0.0	1.8	1.8	94.7

● Entrance

	P	H	δ	Q	C	M	Ø	E	K
P	59.2	0.0	4.1	0.0	0.0	2.0	32.7	0.0	2.0
H	0.0	68.8	8.3	0.0	0.0	0.0	20.8	2.1	0.0
δ	0.0	1.6	89.1	0.0	0.0	0.0	9.4	0.0	0.0
Q	0.0	0.0	5.0	25.0	0.0	10.0	35.0	10.0	15.0
C	0.0	0.0	0.0	0.0	83.7	0.0	14.3	2.0	0.0
M	0.0	0.0	0.0	0.0	0.0	85.4	12.5	2.1	0.0
Ø	1.2	0.0	3.7	0.0	1.2	2.4	79.3	6.1	6.1
E	0.0	0.0	0.0	0.0	0.0	0.0	1.8	98.2	0.0
K	0.0	0.0	2.0	0.0	0.0	0.0	2.9	1.0	94.1

● Kitchen

	I	J	L	Ø	D	E	W
I	93.0	0.0	0.0	0.0	0.0	7.0	0.0
J	0.0	84.6	5.8	5.8	0.0	0.0	3.8
L	0.0	0.0	98.0	2.0	0.0	0.0	0.0
Ø	0.0	2.3	0.0	90.8	4.6	1.1	1.1
D	0.0	0.0	0.0	0.0	97.9	2.1	0.0
E	0.0	1.8	0.0	1.8	1.8	94.7	0.0
W	0.0	0.0	0.0	4.3	0.0	2.9	92.8

● Office

	B	X	C	Ø	α	E	F
B	57.9	5.3	7.9	15.8	7.9	2.6	2.6
X	0.0	91.2	0.0	5.9	0.0	2.9	0.0
C	0.0	0.0	87.8	12.2	0.0	0.0	0.0
Ø	1.2	7.0	1.2	81.4	1.2	5.8	2.3
α	6.0	0.0	0.0	2.0	86.0	4.0	2.0
E	0.0	0.0	0.0	0.0	0.0	100.0	0.0
F	2.0	2.0	0.0	0.0	0.0	0.0	96.0

● Workshop

	Y	β	A	Z	C	Ø	E
Y	88.0	0.0	0.0	0.0	4.0	0.0	8.0
β	0.0	88.9	0.0	0.0	0.0	11.1	0.0
A	0.0	0.0	100.0	0.0	0.0	0.0	0.0
Z	0.0	0.0	0.0	91.8	0.0	8.2	0.0
C	0.0	0.0	0.0	0.0	81.6	18.4	0.0
Ø	0.0	1.1	0.0	2.2	0.0	93.5	3.3
E	0.0	0.0	0.0	0.0	0.0	3.5	96.5

● Outdoor

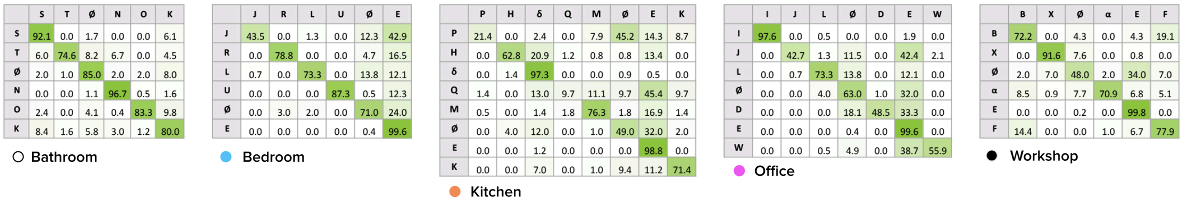
Figure 12-6. Confusion matrices for *SFX-Test* with 20% unknown classes. Class letter legend found in Figure 12-1 (Ø is unknown class).

in a real-world deployment. Foremost, when deploying to, *e.g.*, a user's home, there is no test data on which to checkpoint model training. Second, real-world deployments are subjected to "unknown" sounds, never before heard by the classifier. Evaluating models using only classes they know offers no insights into how interactive systems will handle false positive events. Thus, I ran two additional experiments to more conservatively estimate the system's performance. First, I tuned per-context models using *SFX-All*, checkpointed on *SFX-Test,* and evaluated on *In-the-Wild Test*. This procedure inherently removes checkpointing bias. Using this more stringent procedure, average accuracy across per-context classifiers was 84.8% (SD=6.6%).

As a second, even harder test, I devised an experiment that included 20% "unknown" classes (*i.e.*, sounds we drew from other contexts) in the test set that the model should ignore. This required some alterations to the pipeline. Instead of many per-context models, I tuned a single model using data from all 30 classes in *SFX-All*. A sound is classified as "unknown" (and ignored) if no in-context class exceeds a confidence threshold. Using this evaluation procedure on *In-the-Wild Test* (checkpointed on *SFX-Test*), I found an across-context accuracy of 80.4% (SD=6.4%; Figure 12-5, *SFX-All*), which I believe is a much closer estimation of plug-and-play performance. See Figures 12-6 and 12-7 for the confusion matrices for these two experiments.

Comparison to Human Performance

While 100% accuracy is the ultimate goal of any interactive system, some problem domains are particularly ambiguous or challenging and can benefit from additional



○ Bathroom

	S	T	Ø	N	O	K
S	92.1	0.0	1.7	0.0	0.0	6.1
T	6.0	74.6	8.2	6.7	0.0	4.5
Ø	2.0	1.0	85.0	2.0	2.0	8.0
N	0.0	0.0	1.1	96.7	0.5	1.6
O	2.4	0.0	4.1	0.4	83.3	9.8
K	8.4	1.6	5.8	3.0	1.2	80.0

● Bedroom

	J	R	L	U	Ø	E
J	43.5	0.0	1.3	0.0	12.3	42.9
R	0.0	78.8	0.0	0.0	4.7	16.5
L	0.7	0.0	73.3	0.0	13.8	12.1
U	0.0	0.0	0.0	87.3	0.5	12.3
Ø	0.0	3.0	2.0	0.0	71.0	24.0
E	0.0	0.0	0.0	0.0	0.4	99.6

● Kitchen

	P	H	δ	Q	M	Ø	E	K
P	21.4	0.0	2.4	0.0	7.9	45.2	14.3	8.7
H	0.0	62.8	20.9	1.2	0.8	0.8	13.4	0.0
δ	0.0	1.4	97.3	0.0	0.0	0.9	0.5	0.0
Q	1.4	0.0	13.0	9.7	11.1	9.7	45.4	9.7
M	0.5	0.0	1.4	1.8	76.3	1.8	16.9	1.4
Ø	0.0	4.0	12.0	0.0	1.0	49.0	32.0	2.0
E	0.0	0.0	1.2	0.0	0.0	0.0	98.8	0.0
K	0.0	0.0	7.0	0.0	1.0	9.4	11.2	71.4

● Office

	I	J	L	Ø	D	E	W
I	97.6	0.0	0.5	0.0	0.0	1.9	0.0
J	0.0	42.7	1.3	11.5	0.0	42.4	2.1
L	0.0	0.7	73.3	13.8	0.0	12.1	0.0
Ø	0.0	0.0	4.0	63.0	1.0	32.0	0.0
D	0.0	0.0	0.0	18.1	48.5	33.3	0.0
E	0.0	0.0	0.0	0.4	0.0	99.6	0.0
W	0.0	0.0	0.5	4.9	0.0	38.7	55.9

● Workshop

	B	X	Ø	α	E	F
B	72.2	0.0	4.3	0.0	4.3	19.1
X	0.0	91.6	7.6	0.0	0.8	0.0
Ø	2.0	7.0	48.0	2.0	34.0	7.0
α	8.5	0.9	7.7	70.9	6.8	5.1
E	0.0	0.0	0.2	0.0	99.8	0.0
F	14.4	0.0	0.0	1.0	6.7	77.9

Figure 12-7. Confusion matrices for *In-the-Wild Test* with 20% unknown classes. Class legend found in Figure 12-1 (Ø is unknown class).

baselines to contextualize performance. In the case of sound classification, humans offer an excellent gold standard, as they can draw upon a lifetime of real-world experiences and leverage contextual knowledge in sophisticated ways (*e.g.,* a small motorized appliance in a kitchen is more likely to be a blender than a miter saw). As such, I conducted two studies to establish human accuracy on the *SFX-Test* and *In-the-Wild Test* datasets.

First, I ran a crowd-sourced study on Amazon Mechanical Turk. The crowd interface noted the context (*e.g.*, office) and allowed users to play (and replay) a single sound. Given these two pieces of information, the task was to select the best label (*e.g.*, telephone ringing) from a dropdown list of classes found in that context. Participants could also choose "unknown" if they felt none of the options were correct. In one round of labeling, each of the classes appeared once, plus seven out-of-context ("unknown") sounds (one injected for each context). 250 crowdworkers completed three labeling rounds on *SFX-Test* (producing 27,750 labels), and another 250 crowdworkers labeled the *In-the-Wild Test* set (producing 21,750 labels; less because the real-world data omitted 8 classes, *e.g.*, baby crying).

A potential danger in online studies is reduced accuracy from malicious or apathetic crowdworkers. Thus, as an additional human benchmark, I ran a monitored, in-lab study. This used the same interface and followed the same procedure as the crowd study, but collected four rounds of data instead of three. In total, 50 participants labeled 7,400 sounds from *SFX-Test*, and another 50 participants labeled 5,800 sounds from *In-the-Wild Test*.

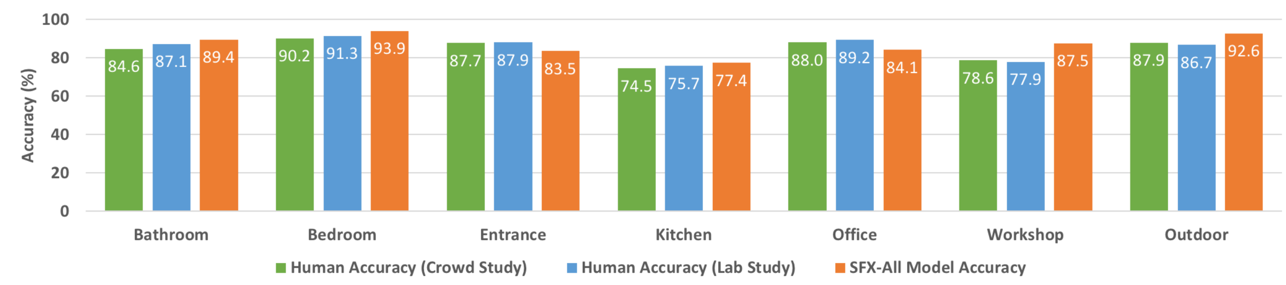


Figure 12-8. Per-context accuracy (trained on *SFX-All,* tested on *SFX-Test* & *In-The-Wild Test* combined, plus 20% unknown classes.

Across all contexts, the average accuracy on the *SFX-Test* data was 83.6% (SD=5.9%) for the crowd workers, and 82.7% (SD=7.0%) for the in-lab participants. For the *In-the-Wild Test* set, the accuracy was 83.9% (SD=6.6%) for the crowdworkers and 87.0% (SD=6.1%) for the in-lab labelers. For reference, under comparable test conditions, the system achieves 89.8% and 80.4% accuracy on *SFX-Test* and *In-the-Wild Test* respectively (Figure 12-5), which is very close to human performance (no significant difference). See also Figure 12-8 for a breakdown of human accuracy across contexts.

Location Context

I tested Ubicoustics' performance across seven location contexts (Figure 12-8), which ranged from 77.4% in the kitchen to 93.9% in the bedroom. Model performance roughly correlates with human performance (R=0.63).

Limiting classes to a context is only possible if a device knows its location. In the case of a smart speaker, a user could specify a location during setup, but a smartwatch is rarely stationary. Thus, I also evaluated the model's ability to automatically infer its physical context (*e.g.*, kitchen *vs*. office). Such a capability could enable devices to automatically load per-context classifiers without user intervention. For this, I used the predicted sound class itself as a proxy for the origin context. For example, if Ubicoustics predicts a *microwave* event, one can can infer that the device is in a kitchen.

To simulate this experimentally, I passed the 30-class *SFX-All* model ten random clips for each context in the test data (*SFX-Test* and *In-the-Wild Test*). The model classifies these clips individually, and the output is used to cast a vote for a context. A few classes have special voting logic: if *water running* is detected, votes for both *bathroom* and *kitchen* are cast, and similarly, *knock* casts votes for both *office* and *entrance*. There are also a set of context-free classes (*i.e.*, can happen anywhere) that do not cast any votes (dog bark, cat meow, vacuum, speech, phone ringing, laugh, cough, door, baby cry and hazard alarm). Once all ten sounds have been processed, the context with the highest vote count is chosen and validated against the ground truth context. We repeated this process 1000 times with random contexts and sounds within that context. Overall, automatic context recognition accuracy is 99.4% (SD=1.5%) for the *SFX-Test* dataset and 92.2% (SD=14.4%) for the *In-the-Wild* dataset. These preliminary results show that it may be possible for interactive systems to automatically deduce their context of use by listening for a short period after setup.

Efficacy of Augmentations

The results reported thus far are based on models tuned on *SFX-All*, which is the superset of all of the data augmentations. To investigate the effects of each augmentation type, I ran the same procedure as the main accuracy studies (20% unknown test sounds, checkpointed on self) but with different tuning sets: *SFX-Orig* (*i.e.*, no augmentations), *SFX-Amp* (amplification), *SFX-Persist* (persistence of sound), *SFX-Mix* (mixing), and *SFX-All* (all augmentations). In this experiment, *SFX-Orig* serves as a baseline. Combining results from *SFX-Test* and *In-The-Wild Test*, the average delta over the baseline for *SFX-Amp* is +1.6%, *SFX-Persist* is 0.0%, *SFX-Mix* is +1.8%, and *SFX-All* is +2.4%. All but *SFX-Persist* are a significant improvement over baseline (paired t-tests, p<.01). See Figure 12-5 for a breakdown.

Performance Across Platforms

If I break out the results by device (*In-the-Wild Test* plus 20% unknown test sounds), one can see that the laptop performed the best at 86.1% accuracy, followed by the watch at 84.1% (Figure 12-9). It appears that better quality microphones and being closer to

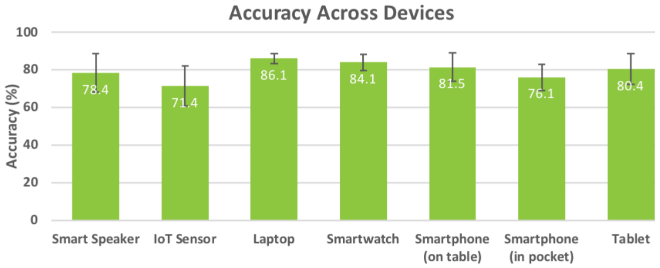


Figure 12-9. Per-device accuracies on the In-the-Wild Test.

events helps recognition. The IoT sensor performed the worst at 71.3%, likely because it was often the farthest sensor from the event source. I also saw a performance drop between phone-on-table at 81.5% *vs*. phone-in-pocket at 76.1% (which "muffled" the microphone and often added fabric chafing background noise).

Comparison to Prior Results

There have been a wide variety of metrics used to evaluate sound-based recognition systems that make apples-to-apples comparisons challenging. Here, I discuss baselines that are most relevant to this work. SoundNet [15] benchmarks its results on the DCASE challenge [254], where it achieves 88% on 10 classes. On the ESC-50 (50 classes) and ESC-10 (10 classes) [206] datasets, SoundNet reports an accuracy of 74.2% and 92.2% respectively. For reference, Ubicoustics achieve an accuracy of 82.1% on 30 classes (all contexts classifier) in the *SFX-Test* set.

Other systems employ metrics that are more relevant to audio indexing – given sound or video, a system produces prediction labels as metadata to facilitate search. Hershey *et al.* [106] report a best-case mAP (mean Average Precision) of 0.38 on the YouTube-8M dataset and 0.31 on the AudioSet dataset. Stated differently, for every sound clip broken into *n* smaller instances, the system makes the correct prediction roughly every third instance. While they benchmark their system as a retrieval system, I evaluate Ubicoustics as a recognition system.

Finally, in the activity recognition domain, BodyScope [298] used a wearable necklace to achieve a recognition accuracy of 71.5% (tested on in-the-wild data) across four activities (eating, drinking, speaking and laughing). Similarly, SoundSense [165] recognizes three classes (speech, music and ambient sound) with an accuracy of ~84%. Ubicoustics provides comparable accuracies, while offering a much richer set of activities without requiring any in situ training.

DISCUSSION

My evaluations show promising results that could make sound-based activity recognition more practical. That said, this work also has limitations, which I now discuss along with directions for future work.

Accuracy

Ubicoustics achieves an average per-context accuracy of 80.4% (on *In-the-Wild Test* data, checkpointed on independent data, with 20% unknown sounds injected), meaning that roughly one in five sounds is missed or misclassified. This performance is not sufficient to support end-user applications, though I note it is competitive with human accuracy. Better quality microphones and higher sampling rates could certainly increase accuracy. Likewise, it is also possible to leverage better and deeper model architectures, such as ResNets [105] and those that can model audio temporalities such as LSTMs and CRNNs [88].

Simultaneous Events

The real world is often noisy, rife with multiple sounds occurring simultaneously. However, the current system and experimentation chiefly focused on isolated sounds. No doubt in more chaotic environments, accuracy would suffer. Fortunately, as noted earlier, the additive nature of sound (and sound effect data) is perfect for generating compound sounds, while retaining the benefits of tight segmentation and good labels – an exploration I leave to future work.

Privacy

The richness of sound is a double-edged sword. On one hand, it enables fine grained activity sensing, while also capturing potentially sensitive audio, including spoken content. This is an inherent and unavoidable danger of using microphones as sensors. However, I note that always-listening devices – especially smartphones and smart speakers – are becoming more prevalent and accepted in homes and workplaces, and so the social stigma of such devices *may* wane in the coming years. In the meantime, to mitigate this technically (socially is more challenging), we convert all live audio data into low-resolution Mel spectrograms (64 bins), discarding phase data. With this signal, the model can readily detect speech, but the spoken content is challenging to recover. Moreover, I envision the model being run locally on devices (as shown possible with the laptop, smartphone, and IoT device), such that audio never has to be transmitted.

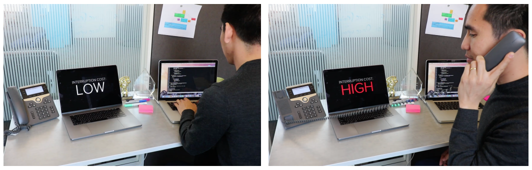


Figure 12-10. Devices could be made aware of user interruptibility, enabling more nuanced notification behaviors.

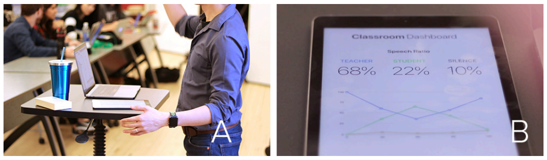


Figure 12-11. In a classroom setting (A), tablets could track and visualize instructional informatics, such as speaking ratio (B).

Bootstrapping Complementary Systems

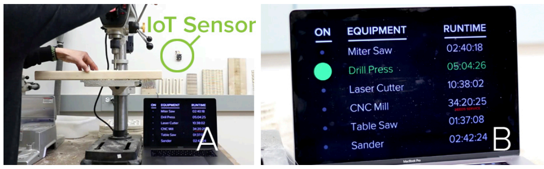
Dimensions beyond audio (*e.g.*, vibrations, motion) are useful for digitizing physical events in environments (see *e.g.*, [149]). However, data collection, segmentation and labeling are generally laborious. With Ubicoustics, it is possible to facilitate and bootstrap this process. For instance, in a wearable application, researchers can collect accelerometer data in tandem with audio. Ubicoustics can provide predictions for performed events (*e.g.*, typing, chopping, writing), as well as offering automatic segmentation of data.

EXAMPLE APPLICATIONS

Finally, I conclude with several example uses that demonstrate the potential of Ubicuostics (Figures 12-10 to 12-3), spanning a range of contexts and platforms.

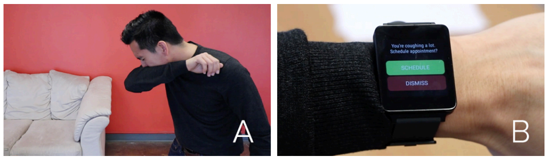
Context-Aware Assistants

Despite smart speakers like Amazon Alexa and Google Home being integrated into people's living spaces (*e.g.*, kitchen, living room, bathroom), these systems have no understanding of events happening around them. Ubicoustics enable new interactive opportunities that leverage real-time context-awareness. There are two main categories. First are *implicit* interactions, where systems can proactively provide users with assistive information. For example, a system could alert users when someone knocks on their front door or automatically move to the next step in a recipe after detecting,



ON	EQUIPMENT	RUNTIME
●	Miter Saw	02:40:18
●	Drill Press	05:04:26
●	Laser Cutter	10:38:02
●	CNC Mill	34:20:25 NEEDS SERVICE
●	Table Saw	01:37:08
●	Sander	02:42:24

Figure 12-12. IoT sensors could track equipment use (A) for safety and maintenance (B).



You're coughing a lot. Schedule appointment?

Figure 12-13. Wearables could track health information, such as coughing (A), and recommend actions (B).

e.g., chopping or a blender running for a defined period. Interactions can also be *explicit*, where users ask their smart assistants about physical events, for example, "is my *microwave done defrosting*?" Notifications about physical events are also possible, such as, *"send an alert when my laundry is done."* Additionally, knowledge of active tasks can suggest a user's interruptibility [80] and better manage interruptions (Fig. 12 -10).

Informatics

Simply logging the occurrence of events could also be valuable. For example, Ubicoustics could track a user's typing over the course of a day, prompting breaks. It could also track the ratio between typing and talking, encouraging more face-to-face interactions. In a classroom setting, a tablet or laptop could track the ratio between teacher and student speech (Figure 12-11), suggesting better instructional behaviors [30]. Finally, IoT sensors in an industrial setting could track equipment use, helping to schedule maintenance (Figure 12-13).

Mobile and Wearable Sensing

Smartwatches are unique in that they reside on the body, equipping users with a "sensor" that they carry everywhere they go. As found in the evaluations, watches are one of the stronger performing device categories, given their proximity to hand-triggered events. Interactions with objects could be logged for quantified self, safety, and assistive applications (Figure 12-13). Being proximate to users is also powerful for health sensing. For example, Ubicoustics can detect when a user coughs or sneezes more frequently. This could enable smartwatches to track the onset of symptoms and potentially nudge users towards healthy behaviors, such as washing hands or scheduling a doctor's appointment (Figure 12-13).

CONCLUSION

In this Chapter, I have shown that Ubicoustics can unlock real-time activity recognition by leveraging one of the most common sensors found in consumer electronics today – microphones – bringing the promise of smart devices and environments closer to reality. By leveraging existing state-of-the-art sound classification models and tuning them with sound effects, Ubicoustics enabled a general-purpose and flexible sound recognition pipeline that requires no *in situ* data collection, yielding a "plug and play" end user experience. I evaluated the robustness of this approach across different physical contexts and hardware platforms, and show that Ubicoustics can achieve high accuracies, both in terms of recognition accuracy and false positive rejection.

13. LISTEN LEARNER: ONE-SHOT ACOUSTIC ACTIVITY RECOGNITION

INTRODUCTION

In the previous Chapters, I described two predominant approaches for building activity recognition classifiers, each with characteristic accuracy and user burden implications (Figure 11-2). First is to train a system manually, after it is deployed, most often by demonstrating different activities and having the user provide class labels (Chapter 9, Synthetic Sensors). Because data is collected *in-situ*, accuracy tends to be quite high. However, the burden to the user is also high. The other approach is to provide users with classifiers that are already trained, and work “out of the box” (Chapter 12, Ubicoustics). This is achieved by training a classifier on a large, general corpus of acoustic data. Because the classifier has no data for a user’s specific environment, it tends to be less accurate, but the burden to the user is low.

In this Chapter, I cap-off my thesis by proposing and evaluating a balanced approach that seeks to provide high classification accuracy, while minimizing user burden. This approach, which I call Listen Learner, requires no up-front data, and instead, learns acoustic events over time and requires no manual demonstration (Figure 13-1). This approach learns events *in-situ*— it is highly tuned to the environment and objects of interest, and thus can offer superior accuracy than general, pre-trained classifiers.

LISTEN LEARNER

Listen Learner supports personalized acoustic activity recognition using *one-shot* labelling of a self-supervised model. It constructs a model using audio samples

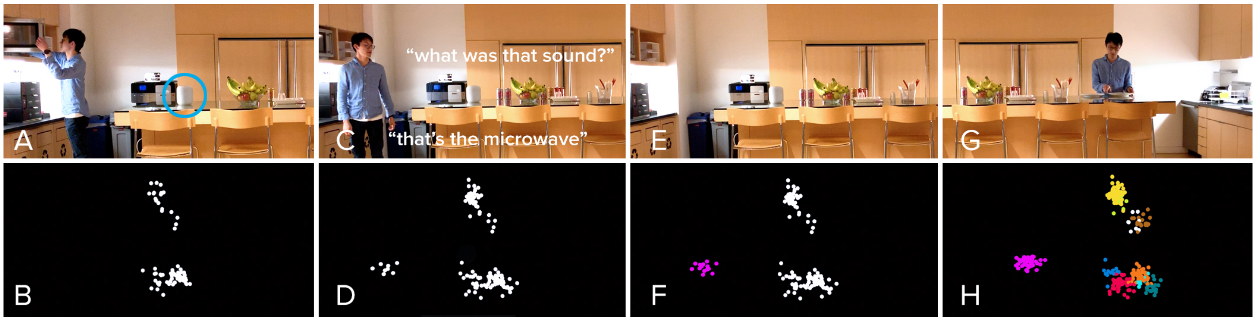


Figure 13-1. Listen Learner system overview. A smart device is deployed (A) in the user's environment (*e.g.,* a kitchen counter top), and it tries learn about acoustic events in the environment (B). When the system becomes confident about a set of identical events it has heard over time (*i.e.*, clusters forming, D), it prompts the users for a label (C). That cluster then becomes part of a set of known events (F, G). Over time, the system continues to learn more events (E) with minimal user intervention.

collected from its surroundings, allowing it to learn the specific acoustic properties of its environment and any user-specific activity classes that may be present. At the same time, it minimizes user effort by only requiring a single label per class.

Example Interaction

To further illustrate the utility of Listen Learner, I describe the following vignette:

Setup. Lisa deploys a smart speaker, equipped with Listen Learner, on her kitchen countertop. The system starts with no data or knowledge about its environment. As sounds in Lisa's kitchen occur, the device clusters live audio data using deep learning embeddings and acoustic direction as features. No raw audio is saved to the device or to the cloud, helping to preserve privacy.

One-Shot Labeling. Eventually, the system becomes confident that a cluster of data is a unique sound, at which point, it can prompt Lisa for a label the next time the sound occurs. The system asks: *"what sound was that?",* and Lisa responds with: *"that was my faucet."* This one answer can then be used to label a cluster. This provides enough training data to initialize a classifier that is inherently tuned to that event's sound and local environment. As time goes on, the system can continue to add data to clusters, training better and better classifiers. It can also continue to prompt Lisa for labels as new clusters emerge, thus slowly building up a library of recognized sounds.

Verification and Refinement. The system can also employ different conversational strategies to elicit labels from Lisa. For example, instead of asking an open-ended question, Listen Learner can make an initial guess using a general, pretrained model. The system asks: *"was that a blender?"*, in which Lisa responds: *"no, that was the coffee machine."* Finally, in cases where cluster boundaries are obscure, Listen Learner can ask refinement questions to aid in separation. The system asks: *"was that a faucet or a microwave?"*, in which Lisa responds: *"it's a microwave."* The library of sounds that Listen Learner builds over time can then be used to power assistive and context-driven applications.

Unlike traditional supervised learning methods that require numerous user-labelled examples at the training phase, Listen Learner *inverts* the annotation and training process. In this work, I describe the *cluster-classify* algorithm, which learns an ensemble of classifiers without any user intervention. The user is then queried *in-situ* to provide a label for the model the next time it is triggered. In this paper, I further describe a proof-of-concept implementation and evaluate the system on offline datasets and a real-world dataset collected using a custom sensing hardware. The rest of the paper describes key results, implications for user interaction, system limitations, and directions for future work.

RELATED WORK

I situate Listen Learner in the literature of contextual sensing for activity recognition and machine learning methods for real-world activity recognition.

Audio Event Classification

Audio-based sensing has emerged as an effective method for activity recognition, as it can capture rich information using small, widely available hardware while mitigating occlusion. Acoustic sensing has been deployed in both localized [165, 166, 249, 298] and wide-area [142, 228, 259] applications. Traditional approaches to audio classification involve computing statistical features on time-domain [263], frequency [176, 263] or wavelet [263, 266] representations. More recently, deep-learning architectures have been used to model the inherent non-linearities in acoustic data. Approaches include treating audio signals as one-dimensional signals [15, 268], or creating two-dimensional log-mel spectrograms [106] that serve as input to convolutional neural networks (CNNs), previously used for image classification [106, 142]. Finally, there has also been research in unsupervised learning of audio representations from large unstructured and semi-structured audio sources [119, 120, 152]. Listen Learner uses the "bottleneck" embedding representation of a CNN (similar to those previously used for audio event discovery and activity recognition [119, 158]) but fine-tuned on a library of professional sound effects [142].

Methods for Activity Recognition

In addition to finding appropriate sensing methods, a major challenge of human activity recognition (HAR) is training highly robust machine learning models (*i.e.,* accurate classification, minimal false positives). One approach is to make minimal assumptions about the environment by employing semi-supervised learning techniques such as Positive Unlabeled Learning (PUL) [72] to learn from a small number of positively labelled samples. In the context of HAR, Nguyen *et al.* propose using a specific form of PUL, called mPUL to decrease the amount of training data and reduce false-positives in real-world HAR applications by assuming "open-world conditions" [188]. Others have focused on active learning to minimize the size and impact of the training set on the model [111, 163, 242]. Finally, another strategy is to model activities using a set of semantic attributes—allowing activities to be more generally defined [267, 189, 276].

Relevant Machine Learning Approaches

A goal of Listen Learner is to require only limited training labels from users to reduce the burden of adapting it to new activity classes and contexts. Training a model using a (usually) large amount of unlabeled training instances along with a smaller number of labeled training instance is called semi-supervised learning [37], to differentiate it from supervised learning (all data is labeled) and unsupervised learning (no data is labeled) techniques. Co-training is a semi-supervised learning method for leveraging a small number of examples and a large unlabeled set to create a model with better classification performance [27]. In self-supervised approaches, the learning process itself is able to construct labels necessary to train a model using supervised learning.

One-shot and zero-shot learning allow models to recognize previously unseen classes with very few (or zero) labelled training instances [295]. This can be achieved, for instance, by representing previously unseen classes as probability distributions over, *e.g.,* a neural network output layer representing a set of learned categories [155, 270]. Previously mentioned strategies such as the mapping of activity classes to a semantic space can alternatively also be used for incremental and one-shot learning [267, 276, 189]. These approaches differ from semi-supervised learning because they allow models to predict previously unseen classes.

Listen Learner also continues to improve its model over time as it observes more data using *incremental learning*. Incremental learning refers to a class of learning algorithms that can accommodate new data to continuously improve and extend a model's knowledge without fully retraining the model [207]. This has been explored in many domains, including computer vision [156, 214], audio event recognition [71], natural language processing [35, 300], and activity recognition [177, 227].

SoundSense [165] is similar to Listen Learner, in that it also provides a platform for audio event discovery. SoundSense was delivered on a mobile platform and starts with a pre-existing set of classes that it uses to bootstrap recognizers for new classes. SoundSense uses a Bayesian classifier and Hidden Markov Model (HMM) to learn new audio events, which results in assumptions about the distribution of audio events (*e.g.,* can be modeled by a Gaussian) that I find limiting. The model also requires a number of parameters to be set that may be difficult to do so well without *a priori* knowledge. Listen Learner's goal of enabling accurate acoustic HAR in a fixed environment (*e.g.,* by smart speaker in a room) is better served by the class discovery approaches discussed in this paper, as they allow our system to constantly update its model in its deployment environment.

PROOF-OF-CONCEPT IMPLEMENTATION

I implemented Listen Learner as an end-to-end system that automatically generates acoustic event classifiers over time. Here, I describe the sensing hardware, the data processing pipeline, and the self-supervised learning algorithm.

Hardware

The prototype consists of both a deployed sensor device (analogous to smart-speaker), and a processing server (on which the self-supervised learning algorithm is executed). Specifically, I built sensing device using a Raspberry Pi 3 Model B+ with a 4-mic microphone array (seeedstudio.io). I also connect a speaker using a 3.5 mm audio jack. I leverage the microphone array for beamforming and for computing acoustic direction-of-arrival (part of our feature set). I set the microphone sampling rate to 16 kHz 16-bit integer linear PCM. The sensing device is configured to connect to WiFi and upload featurized audio to our data processing server (2016 12-core Mac Pro, 64GB RAM).

Cluster-Classify Algorithm

I designed a self-supervised algorithm for identifying and generating classifiers for novel activity classes (Figure 13-3). Similar to traditional methods for training activity recognition algorithms, the goal of the learning process is to produce a model capable of discriminating between different activity classes based on incoming sensor data. However, we sought to minimize the assumptions of the model and user effort by inverting the annotation and training process.

To do this, we introduce a *cluster-classify* algorithm that learns an ensemble model by iteratively clustering unknown samples, and then training classifiers on the resulting cluster assignment. This allows for a “one-shot” interaction for the user to label parts

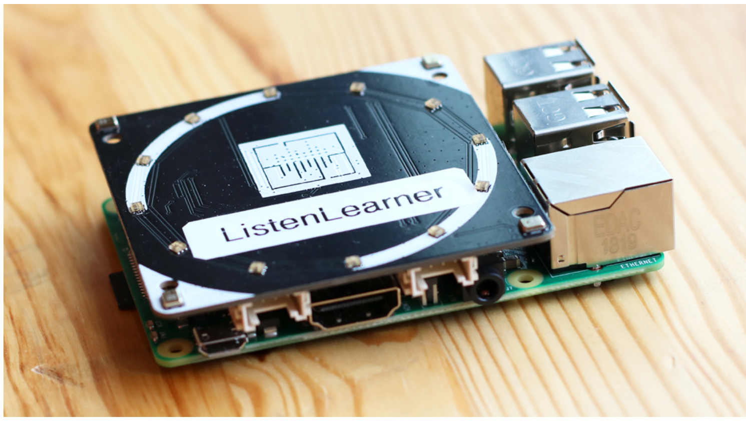


Figure 13-2. The experimental hardware platform. I used a variant of a Raspberry Pi 3 B+ with a 4-mic microphone array for beamforming and computing acoustic direction of arrival.

of the ensemble model when they are activated. The algorithm implementation uses freely available scientific and machine learning packages available for Python. We use Keras for embedding computation (keras.io), fastcluster and SciPy for clustering [183, 121], and Scikit Learn for classification [199].

Segmentation

For my prototype implementation, I decided not to cluster silence. While many environments have persistent background audio events (*e.g.,* HVAC) that can be used to identify the deployment location, I ignore these in the interest of computational efficiency. I segment audio events using an adaptive threshold that triggers when the microphone input level becomes 1.5 standard deviations higher than the mean of the last minute. I also employ hysteresis to further smooth our thresholding scheme.

Featurization

As mentioned earlier, Listen Learner uses bottleneck feature representations extracted from the last hidden layer of a VGG-architecture [246] deep CNN audio model [106]. This model was initially trained on the YouTube-8M dataset, and further augmented using a library of professional sound effects [142]. I construct 96x64 log-mel spectrogram patches as input to the CNN by using a non-overlapping 960 ms sliding window over audio input. For example, an audio clip of a faucet running for 9.6 seconds would produce 10 featurized embeddings. In the prototype sensing device hardware,

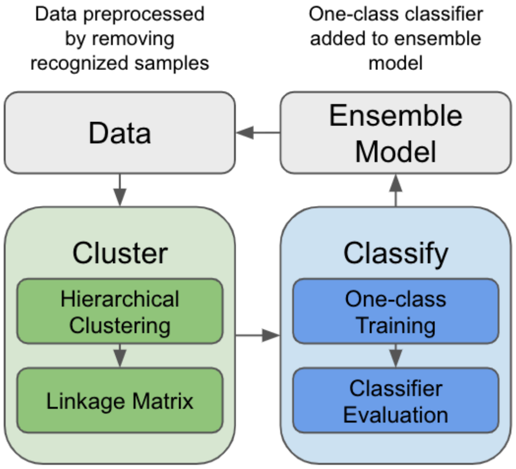


Figure 13-3. This flow diagram shows the stages of the cluster-classify algorithm for self-supervised learning.

this computation takes an additional 1 second per 960 ms of audio. While the computation latency causes some input frames to be dropped, I did not find this limiting for classification performance due to the sustained nature of human activity. The choice of using deep neural network embeddings, which can be seen as learned low-dimensional representations of input data [88], is consistent with the manifold assumption (*i.e.,* that high-dimensional data roughly lie on a low-dimensional manifold [37]). By performing clustering and classification on this lower-dimensional learned representation, the algorithm more easily discovers and recognizes novel classes.

Data Management

For research purposes, my coauthors and I chose to aggregate data collected from the sensing devices on a central server, which also permitted us to run heavier algorithms. I transmit and store only featurized data, which is computed on the sensing board. The algorithm has access to a data pool, which contains featurized data that is not yet recognized by the system. Data is added to the pool in batches (*e.g.*, after the end of each day, or after n samples).

Cluster Step

Using unsupervised clustering methods, we first try to infer the location of class boundaries within the input data. Our approach is supported by the cluster assumption, which states that if points are in the same cluster, they are likely to belong to the same class and that the decision boundary between classes should lie in a low-density region [37]. For our implementation, we choose to use a hierarchical agglomerative clustering (HAC) algorithm known as Ward's method [280]. Using the linkage matrix produced by the algorithm, we take all clusters merged with size $n_{min} \leq n \leq n_{max}$ as candidate clusters representing classes of audio events. Note that these candidate clusters may overlap with one another, but we evaluate all possible groupings of data to find the best representation of classes.

Classify Step

While the clustering algorithm separates data into clusters by minimizing the total within-cluster variance, we also sought to evaluate clusters based on their *classifiability*. Following the clustering step, we use the unsupervised one-class support vector machine (SVM) algorithm that learns the decision boundary for novelty detection [237]. For each candidate cluster, a one-class SVM is trained on the cluster's data points, and its F1 score is computed with all samples in the data pool.

Model Construction

Traditional clustering algorithms seek to describe the input data by providing a cluster assignment, but this alone cannot be used to discriminate unseen samples. Thus, to facilitate the system's inference capability, we construct an ensemble model using the one-class SVMs generated from the previous step. We adopt an iterative procedure for building our ensemble model by selecting the first classifier with an F1 score exceeding the threshold, θ_{clf} and adding it to the ensemble. When a classifier is added, we run it on the data pool and mark samples that are recognized. We then restart the *cluster-classify* loop until either 1) all samples in the pool are marked or 2) a loop does not produce any more classifiers.

Incremental Learning

Listen Learner is designed for longitudinal deployment, where more data is revealed to the system over time. As described earlier, the data pool grows over time as more audio is captured from the environment. When a new data is added, we re-run our algorithm.

Of course, there are computational and data storage limits. As a practical compromise, in the current implementation, we only store audio samples within a fixed time window (*e.g.,* one-week's worth of data). When new data is received beyond this threshold, the oldest samples are discarded. Other methods are also possible, depending on the hardware or desired behavior (*e.g.*, random subsampling of the data pool, or a replacement scheme that discards data based on accuracy rather than age).

Audio Directionality

The sensing hardware includes a microphone array, allowing additional directionality information, which serves as an additional sensing modality. We chose to represent this using the x and y components of a normalized unit vector. Techniques for handling multimodal data can generally be categorized into two approaches: early-fusion, where different sensor values are concatenated together at the model input, and late-fusion, which performs integration by combining the outputs of modality-specific models.

During the clustering step, we use a late-fusion approach that performs clustering twice on the data: once using only directional information and once using only audio embedding. Clusters from both views are provided to the classify step, with directional information considered first. Because unsupervised clustering such as Ward's clustering relies on distance metrics that weights each dimension equally, the relatively small number of directionality features would have negligible impact in cluster formation if the features were concatenated together. In contrast, the classification stage allows the classifier to determine relative feature merit by learning weights. Thus, we use an early-fusion approach when training one-class SVMs by fusing input from audio embeddings with directionality vectors.

User Interaction

The last step of our process is to seek labels from the user for the model that our system has generated. There are many different possible approaches that depend on the platform. Examples include voice-based conversation agents [200, 222, 221], text response for screen-based hardware, and push notifications for mobile devices. As a proof of concept, the prototype device I designed acts like a smart speaker that queries the user using a simple speech interface. When an unlabeled class in the model is activated, the system asks the user, *"what was that sound?"* immediately after the sound event. I use Google voice transcription service to recover the class label from the user. For this implementation, the system only accept labels if they are a single word, or else it prompts the user again.

HYPERPARAMETERS & ALGORITHM BEHAVIOR

As noted in our implementation, there are series of parameters that can be adjusted to alter the system's behavior. Namely, they are: θ_{clf} (the classifier acceptance threshold), n_{min} (minimum cluster size), n_{max} (maximum cluster size), and the ν parameter (controls number of support vectors [237]) of the one-class SVM. I conduct a series of preliminary studies to inform the design of the algorithm and its hyperparameters.

Tuning Metrics

Traditional clustering metrics focus on evaluating cluster assignments as descriptions of classes. These approaches include calculating cluster purity [119], conditional entropy [214], and information-based approaches [269]. By running the generated model on a dataset, it is possible to generate a cluster assignment and apply these metrics. To better reflect the algorithm's purpose of discriminating unseen samples, we use our own objective function that takes into account the classification performance of the classifiers on unseen data. We formulate our objective measure for evaluating a set of hyperparameters using the following equation.

$$\mu = \alpha \cdot \mathrm{F1}_{micro} + (1 - \alpha) \cdot r_{accept}$$

α represents the weighted average between the F1 score of accepted samples and the accept rate. Equivalently, α can be adjusted to influence the algorithm's behavior.

Tuning Procedure

I empirically determined the hyperparameter settings for our algorithm using two datasets of audio events and environmental sounds. Because these offline datasets contain ground-truth labels, they allow us to find the optimal settings for the system. Specifically, I use the ESC-10 subset of the ESC-50 dataset (10 classes, 400 clips) [206] and UrbanSound8K (10 classes, 8732 clips) [228]. Furthermore, I randomly subsample 3000 0.96-second windows from the UrbanSound8K dataset to simulate a realistic usage period (10 event classes x 10 events per day x 30 days). Each dataset is shuffled and split into a tuning set (25%), used for hyperparameter tuning, and an evaluation set (75%), used later for our formal evaluation.

I further divide the tuning set into three partitions for training (60%), holdout (20%), and testing (20%). The algorithm is first run using hyperparameters on the training set. The extracted unlabeled classifiers are labelled by randomly selecting samples from the holdout set (without replacement) and taking the ground truth label of the first instance that was recognized. This simulates the proof-of-concept system's labelling strategy of asking the user for a label the first time it is triggered. The ensemble model is then evaluated on the test set to calculate $\mathrm{F1}_{\mathrm{micro}}$ and $\mathrm{r}_{\mathrm{accept}}$. This process is repeated 10 times for each hyperparameter combination, and the mean is used by the objective function.

Finally, for hyperparameter search, I use a parameter-free black-box optimizer [128] to maximize the objective function, and the optimizer runs for 50 iterations.

Algorithm Behavior

By optimizing the hyperparameters according to different metrics, I can influence the system's behavior. Below I describe three example "settings" for the implementation:

Relaxed (low α) - This setting aims to ignore as few data points as possible at the cost of classifier accuracy. By lowering the required "confidence" value, a larger number of classes are detected.

Balanced (medium α) - The balanced setting produces an intermediate behavior that seeks to accept a moderate number of samples with usable levels of accuracy.

Conservative (high α) - Accepts new classes only when very confident. This results in more events being unclustered, and thus ignored, but recognition is more accurate.

These settings were acquired by manually fine-tuning the output of the black-box hyperparameter optimizer. For the ESC-10 dataset, I use α values of 0.4, 0.75, and 0.9 for Relaxed, Balanced, and Conservative, respectively. For the UrbanSound8K dataset, I use α values of 0.6, 0.8, and 0.9 for Relaxed, Balanced, and Conservative, respectively. The inverse relationship between F1 accuracy and acceptance rate is shown in Figures 13-4 and 13-5.

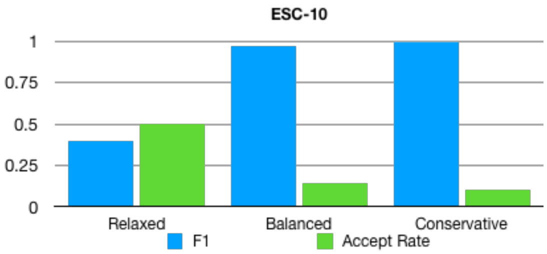


Figure 13-4. This chart shows the F1 accuracy and the accept rate of hyperparameter profiles on the tuning set of ESC-10.

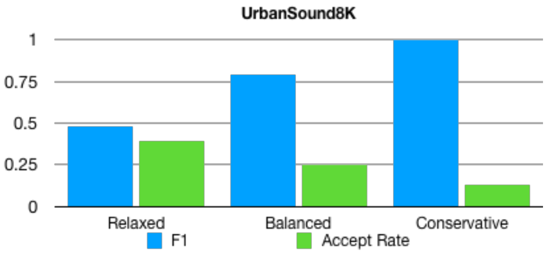


Figure 13-5. This chart shows the F1 accuracy and the accept rate of hyperparameter profiles on the tuning set of UrbanSound8K.

In-The-Wild Investigation

In addition to the preliminary experiments for hyperparameter tuning, I also performed a 10 day-long in-the-wild data collection. Because the primary motivation of Listen Learner is to support low-burden personalized activity recognition by learning the specific properties of environments and activities of interest to the user, I wanted to run the system under real-world conditions to characterize the types of data present in entirely uncontrolled environments.

As an added benefit, this gave me an opportunity to test elements specific to our implementation such as: 1) the hardware and software stack and 2) the ability to test directionality information with our algorithm. The in-the-wild investigation was conducted across a period of one and a half weeks at 7 locations (7 rooms, 5 buildings). Specifically, they include a mix of high-activity and low-activity environments: Office, Basement, Kitchen 1, Bathroom 1, Living Room, Kitchen 2, and Bathroom 2. Consent was obtained from both the owners of the spaces and any temporary visitors. Signs were posted at deployment locations instructing visitors to either sign a consent form if they agree to participate in the data collection or unplug the sensing device for the duration of their stay.

I was primarily interested in investigating the performance of Listen Learner in real-world scenarios and validating the behavior of the three hyperparameter tunings. For each behavior setting, I take the mean value between the ESC-10 and UrbanSound8K hyperparameter value for use on the collected dataset. An average of 41.9 (SD=55.7), 27.9 (SD=27.3), 9.9 (SD=7.5) classes were discovered by algorithm using the Relaxed, Balanced, and Conservative settings, respectively. It is possible that multiple classes are generated by the same object and would be given the same label by the user (*e.g.,* microwave running, microwave door closing). I show room-specific results in Figure 13-6. This shows that the system is able to effectively discover classes in real-world environments and hyperparameters tuned using the objective function produces consistent behavior across datasets.

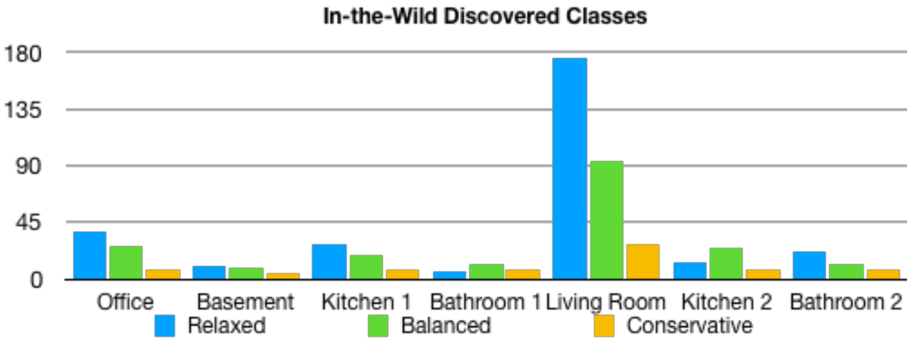


Figure 13-6. Number of discovered classes by room and their hyperparameter profiles. Notice the variation in activity and number of classes discovered.

The in-the-wild investigation was useful for characterizing the system's performance in realistic scenarios. However, without any reliable ground truth data, it was impossible to quantify the system's classification and discovery performance. I use these insights to inform the design of a formal evaluation by collecting ground truth as a part of the procedure.

EVALUATION

In this section, I describe the methodology used for testing the feasibility of Listen Learner. This formal evaluation uses similar metrics as the ones used during the preliminary experiments. I quantify performance by categorizing model output into three possibilities: 1) Correct (recognized a sound belongs to an audio class and classifies it correctly), 2) Incorrect (recognized a sound event occurred but misclassifies it), and 3) Ignored (sound event ignored by the system). Additionally, I was also interested in the time and amount of data needed by the system to reach a usable level of performance. Thus, I also take into consideration the amount of elapsed time required with respect to model accuracy.

Datasets

I conduct our evaluation with three datasets of varying sizes and number of classes. Two of the datasets (ESC-10 and UrbanSound8K) were previously used for hyperparameter tuning experiments, but we use a temporally-separated split for this

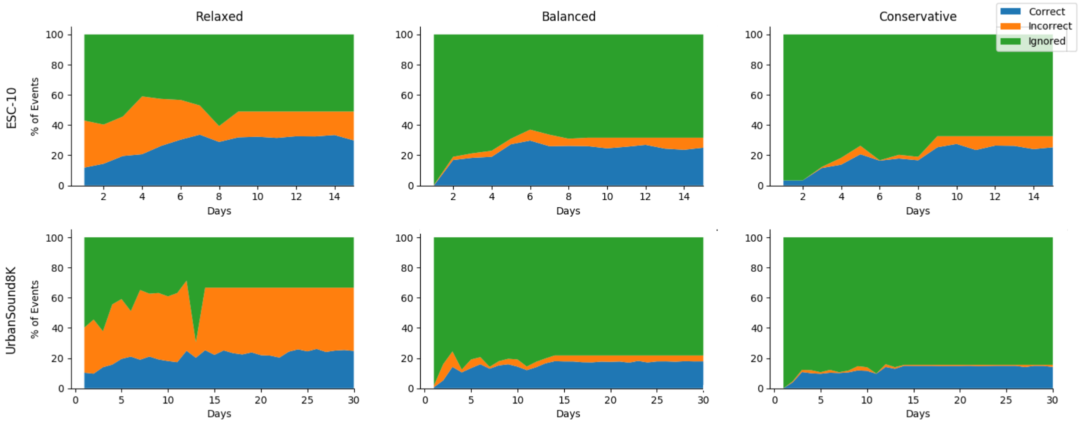


Figure 13-7. Evaluation results from offline datasets. This chart shows the number of correct, incorrect, and ignored samples of the system over time. We simulate the passage of time on offline datasets by selecting 100 random samples per day. A high number of correct events (blue) and low number of incorrect events (orange) denotes high accuracy. A small number of ignored events (green) denotes high accept rate.

evaluation (*i.e.*, separate from split used for hyperparameter tuning). The downloaded datasets are recorded with different microphones in different environments, and thus, are not representative of the type of data I envision for the system. Therefore, in addition to these offline datasets, I also collected a real-world dataset of labelled audio from 6 environments over a one-week period using our sensing hardware. Specifically, the environments are: an apartment bathroom, an apartment kitchen, a bathroom, a fabrication workshop, an electronics workshop, and an office.

For each environment, I selected 5-7 events of interest and recorded 5 20-second clips each of these events in use each day. Data collection was performed in a controlled setting when no other people were present in the environment. Mobile objects or actions that could be performed in different locations (*e.g.,* speech, electric toothbrush) had their locations randomized across recording sessions. Meanwhile, the position and orientation of the recording device was kept constant across sessions. This data collection process was repeated for one week. In total, we collected 1295 audio clips (432 minutes) from this deployment, resulting in 26970 featurized samples.

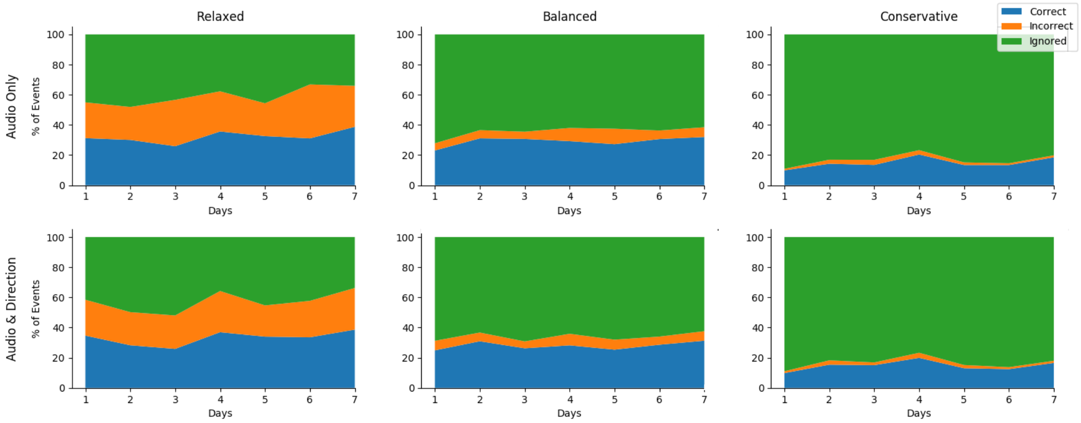


Figure 13-8. Evaluation results from our real-world dataset. This chart shows the number of correct, incorrect, and ignored samples of our system over time. We compare system performance with and without directional information.

Procedure

I follow a similar evaluation procedure as the one used during our hyperparameter tuning experiments. Specifically, the entire dataset was divided into training (60%), holdout (20%), and test set (20%). I simulate the passage of time by gradually expanding the portion of the training set used by the *cluster-classify* algorithm. For the two downloaded datasets, I approximate the passage of a day by selecting the next 100 samples (using timestamps from the dataset).

The *cluster-classify* algorithm is run on the cumulative pool of samples collected up until the current day. Labels for the unknown classifiers generated by the algorithm are determined by using the ground truth of the first recognized instance randomly drawn from the holdout set. The mean F1 and accept rates of 10 repetitions are used. Finally, I calculate performance metrics on the test set. To summarize, I conduct the evaluation on three datasets of varying sizes and number of classes (two offline datasets, and one dataset collected from a controlled lab study).

RESULTS

In this section, I discuss the results of the evaluation in terms of previously defined metrics. Further, I investigate how much data is required for Listen Learner to discover new classes and provide usable accuracy by evaluating the F1 score and accept rate over time. In addition, I test aspects specific to our implementation such as the inclusion of directional data.

Accuracy

The accuracy over time of Listen Learner can be seen in Figure 13-7 and 13-8. The Conservative setting for the UrbanSound8K achieved the highest final F1 score (0.91), while the Relaxed setting had the lowest final F1 score (0.37). System performance on the real-world dataset achieved F1 scores of 0.59, 0.84, and 0.88 for the Relaxed, Balanced, and Conservative settings, respectively. Specific to our real-world dataset, the apartment kitchen environment achieved the highest accuracy after the one-week period. Both the Balanced and Conservative settings achieved F1 scores of 1.0, with accept rates of 0.39 and 0.16, respectively. The lowest accuracy within the real- world dataset occurred in the apartment bathroom (Relaxed, $F1_{micro}$=0.42 and r_{accept}=0.73).

In general, the addition of more audio samples leads to higher accuracy, due to the ability to form larger clusters and train more robust classifiers. However, since the *cluster-classify* algorithm evaluates a candidate classifier's F1 score on the pool of collected samples as a criterion for acceptance, the addition of rare audio events or skewed distribution may bias this score. Examples of this are visible at the 4th day of ESC-10 and 13th day of UrbanSound8K. In all tested datasets, there is a convergence point at which processing more data does not lead to significant improvements in either accuracy or accept rate. Depending on variation present within the dataset, the point at which convergence occurs may vary.

Accept Rate

Similar to the trends shown in the accuracy results, the accept rates for various hyperparameters remained consistent to their tuned behaviors. Our real-world dataset reached accept rates of 0.66, 0.38, and 0.20 for the Relaxed, Balanced, and Conservative settings, respectively. Specific to our dataset, the Relaxed setting run on the apartment kitchen environment achieved a 0.85 accept rate with a F1 score of 0.64. On the other hand, the apartment bathroom environment had the lowest accept rate ($F1_{micro}$=1.0, r_{accept}=0.03) when processed with the Conservative setting. On a dataset

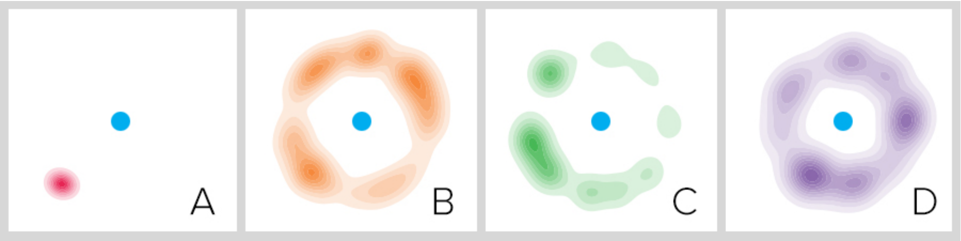


Figure 13-9. This visualization shows the audio direction of select audio classes: coffee grinder (A), speech (B), faucet (C), and refrigerator door. The center dot represents the placement of the sensing device.

level, the highest accept rate was reached by the Relaxed setting on UrbanSound8K (0.67), while the Conservative setting for the same dataset had the lowest rate (0.16).

Our analysis highlights tradeoffs between our two performance metrics (F1 score and accept rate). Specifically, our analysis of room-specific performance shows that some environments (*e.g.,* apartment kitchen) results in more audio classes discovered than others (*e.g.*, apartment bathroom). As the accuracy threshold for adding new classes to the model is relaxed, a greater number of events can be recognized. At the same time, the overall robustness of the model is improved by accepting only high accuracy classes. In the discussion section, I propose different interaction designs that can complement either a more conservative or more relaxed setting.

Effect of Sound Direction

The use of custom sensing hardware for collecting the real-world dataset allowed me to integrate directionality information for identifying and classifying different objects of interest. I first inspected the distribution of direction measurements by class. Figure 13-9 shows the distribution of the direction information for four classes in our data. In some cases, directionality information can be valuable for approximating the relative orientation of stationary objects to the sensing device (*e.g.,* Figure 13-9A) with respect to mobile ones (*e.g.,* Figure 13-9B). However, there are also cases where directional information does not form clean clusters, even for stationary objects (*e.g.,* Figure 13-CD). I hypothesize that this is due to acoustic phenomena in the environment such as multipath propagation.

Figure 13-8 compares the performance of our algorithm with and without directionality information. I find that in general, the inclusion of directionality does not increase the performance of the system by facilitating higher classification accuracy (p=0.47) or accept rate (p=0.59). I find that in some rooms (bathroom), the inclusion of direction can lead to increases in F1 score of up to +0.16. I hypothesize that many of the tested audio classes in that particular environment (*e.g.* faucet, urinal, toilet) produced similar sounds which could be more easily differentiated using directional information. However, these results imply that, in general, directional measurements do not provide significant performance gains.

Another assumption of using directionality information is that the position and orientation of the recording device itself does not change over the data collection period. Our data collection procedure takes this into account, simulating the placement of a smart speaker whose location largely remains unchanged. Using IMUs integrated in some consumer smart speakers, such as the Apple Homepod [11], any movement can be detected, triggering the system to recalibrate.

DISCUSSION & DESIGN IMPLICATIONS

The results from my evaluation provide insights on the value of such a system, and it can inform the design of interactions that can be used with one-shot acoustic activity recognition. I also discuss possible improvements and additional use-cases.

Discovery-Based vs. Directed

My evaluation of Listen Learner quantified its ability to 1) discover new audio classes in its surrounding environment and 2) correctly classify new events using its self-supervised model. Although I presented multiple tuning settings that can be adjusted for different requirements, I believe the Balanced setting is most relevant for eventual use of Listen Learner as a solution for low-burden, high accuracy acoustic activity recognition. One drawback of Listen Learner's purely discovery-based approach is the inability to explicitly include a class of interest in the model. This problem can be addressed by biasing the audio event segmentation so that it captures sounds similar to a user-provided example. For example, the user can say *"pay attention to this sound",* and the system can use the similarity of the acoustic embedding of that event as a threshold for comparison. Further, the clustering step can also be modified to be a centroid-based approach, where user-provided examples form initial cluster seeds, which are then refined through additional iterations and more data.

Integration with Pre-trained Models

A significant amount of time may elapse before audio classes discovered by Listen Learner are added to the classifier and labelled by the user. To address this issue, one can augment the system using a pre-trained model containing a generic set of default classes. In the absence of any discovered classes, the system is still able to provide some level of functionality to the user through the pre-trained model output. In turn, Listen Learner can gradually discover and replace the pre-trained classes with those trained specifically on its deployment environment.

Interaction Strategies

The inclusion of a pre-trained model also introduces new interaction possibilities (Figure 13-10). While the proof-of-concept implementation requires users to respond to an open-ended query (*e.g.,* "What was that sound?") that is triggered when an unlabeled class is activated, there are many other methods for eliciting user input to label, confirm, and disambiguate classes.

One approach is to have users label classes by correction. Using a pre-trained model, cursory labels can first be assigned to unlabeled classes, and users can be queried for confirmation (*e.g.,* "was that a microwave?"). Likewise, when a data point falls within an ambiguous region, the system can ask a disambiguation query. (*e.g.,* "Was that a microwave or a faucet?"). This streamlines the processing needed to extract label names from a conversational response and can also be used to verify the correctness of a labelled classifier.

Different interaction strategies can also stem from the algorithm behavior induced by the hyperparameter settings. While I mostly consider the Balanced setting when discussing Listen Learner's value, the performance metrics of other settings can used to alter the types of interactions with the system. For example, the relaxed setting has the potential to discover more event classes in the environment but is also prone to incorrect classifications. In this scenario, a “accept early then verify” approach could be used that requires multiple verifications before a new class is integrated.

Privacy Preservation

While acoustic approaches to activity recognition afford higher resolution sensing and better classification accuracy, the capture and transmitting of audio data, especially spoken content raises privacy concerns. In a commercial version of this approach, all data could be retained on the device (which would require computationally capable and

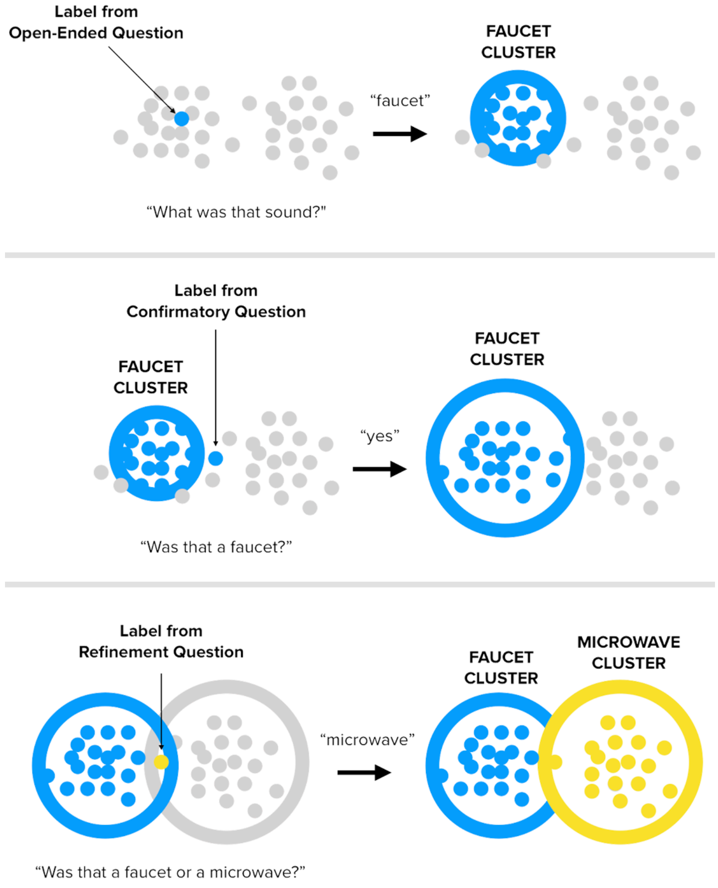


Figure 13-10. Interaction implications made possible through Listen Learner. I show examples of open-ended queries (top). confirmatory queries (middle), and disambiguation queries (bottom).

power-hungry hardware). Alternatively, heavier processing of anonymized features could occur in the cloud, but user labels of model classes are stored locally. This architecture has many benefits, such as better energy efficiency and the ability to use specialized hardware to accelerate the machine learning algorithm. Note that some drawbacks traditionally associated with server-sided data processing, such as transmitting labelled data to a third-party server, do not apply in this case, as the transmitted training data do not contain any activity labels.

LIMITATIONS AND FUTURE WORK

The biggest limitations of this activity discovery approach are its inability to explicitly include classes of interest and the relatively high computational cost of model training compared to traditional supervised machine learning approaches. As described earlier, it is possible to incorporate algorithmic changes that support learning through directed examples or include pre-trained models to facilitate the recognition specific class recognition. While computational cost is relatively high, I believe the low interaction burden to the user is more important for personalized activity recognition. Moreover, our *cluster-classify* algorithm does not need to be run in real-time and can be scheduled to run periodically (*e.g.,* run overnight to process data from each day).

I also envision several directions for future work, specifically the interaction design for one-shot labelling systems such as Listen Learner. In my proof-of-concept implementation, I use a simple speech-based approach that only allows users to respond with a single word. However, this leaves little room for personalized naming and results in unnatural responses. As the primary goal of Listen Learner is to maximize activity recognition accuracy while minimizing user burden, additional work in determining elicitation methods for consistent, high-quality labels.

Another interesting avenue for future exploration is improving the software and algorithm of the system to allow more accurate discovery of events. I intend to investigate audio separation (blind signal separation) techniques to further increase the efficiency of the algorithm. Currently, the system assumes that incoming audio events are segmented and non-overlapping. However, in a realistic environment, it is common for multiple events to occur simultaneously, making it difficult to segment audio based on the current adaptive amplitude threshold. In addition, as described earlier, other methods of clustering and classification can be used to better support the consideration of user-provided examples and multi-modal input.

Finally, sensor fusion approaches are possible. The evaluation results show that the current hardware's support for directional information does not provide much additional performance to the system. The addition of other sensing hardware (*e.g.,* motion sensor, thermometer) can better complement the audio input modality to expand the set of activities accessible to Listen Learner.

CONCLUSION

In this Chapter, I present Listen Learner, a system that seeks to enable high accuracy, low effort acoustic activity recognition using one-shot labelling. First, I motivate and describe the hardware and software components of my proof-of-concept implementation that gradually discovers new event classes from its environment with no user training involved. I designed the system to support a tunable parameter that prioritizes either the number of discovered classes or the model's classification accuracy and evaluate each setting on two downloaded datasets and a real-world dataset collected using a custom sensing hardware. My results show that Listen Learner provides accuracy levels suitable for common activity recognition use-cases and can augment or complement existing systems, further reducing user burden, and bringing the vision of context-aware interactions even closer to reality.

14. CONCLUSION

THESIS CONTRIBUTIONS

In this dissertation, I have described eight projects that enable context-driven implicit sensing and interaction, which are 1) highly practical, 2) supports high-fidelity sensing and human-level semantics, and 3) are designed with privacy preservation as a first-order primitive. By combining novel sensing with machine learning, my work transforms raw signals into intelligent abstractions that can power rich, context-sensitive applications, unleashing the potential of next-generation computing platforms. Digitizing the physical environment through context-awareness has many high-impact applications, from specific domains such as elder care, health monitoring, and empowering people with disabilities, to much broader applications such as smart infrastructures, robotics, and novel interactive experiences for consumers.

More broadly, this dissertation has made contributions in the domain of interactive sensing, ubiquitous computing, and generally, in human-computer interaction research. My work on EM-Sense (Chapter 4), ViBand (Chapter 5) and the follow-up work on fine-grained hand activity sensing (Chapter 6) contributes to an understanding of what can be uniquely and accurately sensed from the vantage point of a wearable device. It highlights the promise of emerging use-cases that leverage nuanced yet practical contextual sensing, transforming the user's arms and hands into an input and sensing platform—an Arm v2.0. This thread of research illuminates the value in knowing what activities the hands are engaged in, better supporting more assistive, more accommodating and more delightful computational experiences.

Likewise, my exploration on general-purpose sensing, as discussed in Zensors (Chapter 8), Synthetic Sensors (Chapter 9), and the deeper exploration on sensor constellations (Chapter 10) illustrates a thread of research that pushes the notion of highly practical and high-fidelity contextual awareness into rooms, living spaces, and buildings. It builds on the progress laid out by infrastructure-mediated sensing, and extends it into a *one-sensor, many facets* scheme, which brings the promise of smart environments more closer to reality. Through real-world deployments, I uncover the strengths and limitations of these techniques, helping illuminate accuracy implications across different ecological, technical, and human-centered constraints.

Finally, my dissertation addresses the limitations of previous ubiquitous sensing and machine learning approaches, most of which fundamentally incur heavy data collection and training burden. To build robust context-sensing models, large amounts of data are required, which is time consuming, error-prone, and difficult to scale. In Ubicoustics (Chapter 11) and Listen Learner (Chapter 12), I reduce data collection and training burden through a data-driven approach. Both projects leverage sensors that already exist; they are bootstrapped by large amounts of pre-existing, high-quality data (Chapter 11), and I apply transfer learning and one-shot learning (Chapter 12) schemes so that they are fine tuned for activity recognition. As a result, these projects reduce the need for users to manually train the system, enabling human-centric experiences that are plug-and-play and capable of learning over time.

FUTURE DIRECTIONS

Altogether, the research projects in my dissertation highlight context-driven interactive experiences, and in future work, I envision expanding this exploration to other domains (*e.g.*, sensing for robotics). Moving forward, I am extremely excited to explore related areas of interest. For example, I want to scale-up in physical scope – my work has traditionally explored room-scale domains, but I want to expand this to *buildings, neighborhoods, and cities*. This is an inherently interdisciplinary endeavor, and I look forward to bringing a new set of tools for researchers and practitioners in areas such as architecture and urban planning.

Additionally, as sensors and devices proliferate, I am excited to further explore how to mitigate and balance their effect on *privacy*. My training in human-computer interaction will allow me to tackle this problem from both social and computational perspectives. This includes working with social scientists to create better privacy models that limit the impedance mismatch between data fidelity and user trust. Equally important is working with designers and machine learning experts to relentlessly push AI more closer to the "edge," and build better differential privacy models.

Finally, I am also excited to explore a broader view of sensing, for example, investigating better approaches to support *sensor data visualization* and *sensemaking*.

As previously mentioned, encoding sensor data with human-level semantics will play a key role in unlocking context-driven applications, and I want to design, build, and study systems that empower users to make better sense of large volumes of sensor data.

ACKNOWLEDGMENTS

MENTORS AND COLLABORATORS

I am indebted to my mentors who constantly challenge my views and have influenced my thinking throughout graduate school. First, I would like to thank my super star advisor, Chris Harrison, for his unwavering guidance and support throughout my Ph.D. I would also like to thank my thesis committee for their insightful feedback and support—Scott Hudson, Niki Kittur, and Shwetak Patel. I would also like to thank Jeff Bigham and Anind Dey for being my "honorary advisors" throughout graduate school. I would also like to thank industry mentors at Google: Alex Kauffman, Murphey Stein, Boris Smus, Emre Karagozler and Ivan Poupyrev. I am extremely grateful to Mira Dontcheva for showing me the ropes and believing in me when I was just starting out in research. I would also like to thank Eytan Adar and Mark W. Newman for their valuable research guidance early on. And I would like to thank my high-school mentor, Chris Chiu, who imbued in me the values of hard work, kindness, and being fun-loving.

Science succeeds through resilient team effort, and I am grateful to have worked side-by-side with many brilliant collaborators. To my lab mates Robert Xiao and Yang Zhang— I would not be where I am without you both. To Patrick Carrington, Ting-Hao "Kenneth" Huang, Sai Swaminathan, Cole Gleason, Anhong Guo, Karan Ahuja, Yasha Iravantchi, and Jason Wu—thank you for inspiring me with your brilliance and hard work. A big thank you to my amazing collaborators: Mayank Goel, Xiang 'Anthony' Chen, Stephen Oney, Jason Wiese, Walter Lasecki, Julia Schwarz, Adrian deFreitas, Sven Meyer. I would also like to thank the amazing system builders at CMU's Software Engineering Institute: Abhijit Hota, Sudershan "Sudo" Boovaraghavan, Chen Chen, and Yuvraj Agarwal. Last but not the least, I am grateful for the lifelong friends I have made at Disney Research— Moshe Mahler, Eric Brockmeyer, Joana Vaz Parker, Joana Campos, Joao Guerreiro, Iolanda Leite, André Pereira, Marynel Vasquez, Adam Stambler, Sean and Jenn Hyde, Jimmy Krahe, John Mars, Michelle Ma, Alanson Sample, and Jack Yang.

RESEARCH SUPPORT

I would like to thank Google, Disney Research, Yahoo!, Qualcomm, and Adobe Research for their generous gifts and strong support of my research. I'm grateful for the resources provided by the Human-Computer Interaction Institute and the School of Computer Science at Carnegie Mellon University. Thanks to Lauren Hardwig and JaRon Pitts, who have been extremely invaluable to our lab. A big thank you to Queenie Kravitz, who is the pillar of the CMU HCII PhD program. I would also like to thank Byron Spice, Karen Harlan, Sue Cribbs, and George Darakos for help with media and external relationships. Likewise, I would like to thank Cindy Chepanoski for her incredible help in streamlining technology transfer and licensing.

HCII FAMILY

The Human-Computer Interaction Institute at CMU is one of a kind, and I'm extremely lucky to have been a part of this family. Thank you to Jodi Forlizzi, Anind Dey, and Justine Cassell for steering the ship during my time as a student. I'm grateful for the staff who have helped me along the way: Ryan Reis, Carolyn Buzzeli-Stumpf, Lindsay Olshenke, Jessica Stanley, Justin Puglisi, Ebony Dickey. To all the faculty who taught me invaluable lessons, either through classes, hallway discussions, or through osmosis: Amy Ogan, Jessica Hammer, Laura Dabbish, Aaron Steinfeld, Jen Mankoff, Brad Myers, Bob Kraut, John Zimmerman, Sara Kiesler, Dan Siewiorek, Jim Morris, Jason Hong, Steven Dow, Lining Yao, Geoff Kauffman, Ken Kodeinger, Skip Shelley, Bhiksha Raj, Anthony Rowe, Ruslan Salakhutdinov. And to all the amazing students who have adopted me as part of their cohort, have lent a hand, or inspired me along the way: Stacey Kuznetsov, Beka Gulotta, Sam Finkelstein, Jen Marlow, Derek Lomas, Haiyi Zhu, Gabi Marcu, Nesra Yannier, Sauvik Das, Erik Harpstead, Kelly Rivers, Kerry Chang, Rogu Kang, Tatiana Vlahovic, Dan Tasse, Jenni Olsen, Chris MacLellan, Caitlin Tenison, David Gerritsen, Brandon Taylor, Nikola Banovic, Xu Wang, Anhong Guo, Anna Kasunic, Judeth Choi, Nathan Hahn, Qian Yang, Amy Cook, Alexandra To, Toby Li, Julian Ramos, Franceska Xhakaj Mary Beth Kerry, Ken Holstein, Cole Gleason, Laura Licari, Joseph Seering, Rushil Khurana, Kristin Williams, Michael Rivera, Siyan Zhao, Judith Uchidiuno, Michael Madaio, Fannie Liu, Felicia Ng, Zhang Yao, Michal Luria, Steven Dang, Joselyn McDonald, Haojian Jin, Michael Liu, Kareem Bedri, Joseph Chang, Yasmine Kotturi, Julia Cambre, Samantha Reig, Lynn Kirabo, Stephanie Valencia, Meghan Hoffman, Lea Albaugh, Sujeath Pareddy, Andrew Kuznetsov, Evi Bernitsas, Humphrey Yang, Zeyu Yan, and Zhi Tan. I'm sure I'm missing someone (sorry), but thank you 100x!

REFERENCES

1. Abbott, R.E, Hadden, S.C. 1990. Product Specification for a Nonintrusive Appliance Load Monitoring System. EPRI Report #NI-101, 1990.
2. Abowd, G. D. and Mynatt, E. D. Designing for the human experience in smart environments. Smart environments: technologies, protocols, and applications (2004).
3. Abowd, G.D. 2012. What next, ubicomp?: celebrating an intellectual disappearing act. In Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp '12). ACM, New York, NY, USA, 31-40.
4. Abowd, G.D. Dey, A.K., Brown, P.J., Davies, N., Smith, M., Steggles, P. 1999. Towards a Better Understanding of Context and Context-Awareness. In Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing (HUC '99), Hans-Werner Gellersen (Ed.). Springer-Verlag, London, UK, UK, 304-307.
5. Abu-El-Haija, S. et. al. 2016. Youtube-8M: A large-scale video classification benchmark. https://arxiv.org/abs/1609.08675
6. Akl, A., Feng, C. and Valaee, S. A Novel Accelerometer-Based Gesture Recognition System. IEEE Transactions on Signal Processing '11, 12: 6197-6205.
7. Aloysius, N., Geetha, M. 2017. A review on deep convolutional neural networks. 2017 International Conference on Communication and Signal Processing (ICCSP), Chennai, 2017, pp. 0588-0592.
8. Alpenfels, E. J. 1955. The Anthropology and Social Significance of the Human Hand. In Artificial Limbs. Volume 2, Number 2. 4-21.
9. Amento, B., Hill, W. and Terveen, L. The sound of one hand: a wrist-mounted bio-acoustic fingertip ges-ture interface. In Proc. CHI EA '02, 724-725.
10. Ananthanarayan, S., Sheh, M., Chien, A., Profita, H., Siek, K. 2013. Pt Viz: towards a wearable device for visualizing knee rehabilitation exercises. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13). 1247-1250. https://doi.org/10.1145/2470654.2466161
11. Apple, Inc. Homepod. https://www.apple.com/homepod/, Retrieved June 9, 2019.
12. Apple, Inc. Impulse Response Utility, User Manual. 2009. https://documentation.apple.com/en/impulseresponseutility/usermanual, Retrieved June 9, 2019.
13. Arroyo, E., Bonanni, L., Selker , T. 2005. Waterbot: exploring feedback and persuasive techniques at the sink. In Proceedings of the SIGCHI Conference on Human Factors in

Computing Systems (CHI '05). ACM, New York, NY, USA, 631-639. DOI=http://dx.doi.org/10.1145/1054972.1055059

14. AT&T Labs. AT&T research on the next secure data transfer device: your body. June 1, 2012. http://innovationblogarchive.att.com/innovation/story/a7782685

15. Aytar, Y., Vondrick, C., Torralba, A. 2016. SoundNet: Learning sound representations from unlabeled video. *Advances in Neural Information Processing Systems (NIPS '16)*.

16. Babenko, A., and Lempitsky, V. 2015. Aggregating Deep Convolutional Features for Image Retrieval. *In IEEE International Conference on Computer Vision* (ICCV), Dec. 2015, pp. 1269-1277. DOI: 10.1109/ICCV.2015.150. arXiv: 1510.07493.

17. Bao, L., Intille, S. S. 2004. Activity Recognition from User-Annotated Acceleration Data. In Proceedings of Pervasive Computing '14.

18. BBC Sound Effects. https://shop.prosoundeffects.com/products/bbc-complete-sound-effects-library, Retrieved on April 4, 2018.

19. Beckmann, C., Consolvo, S., La-Marca, A. 2004. Some Assembly Required: Supporting End-User Sensor Installation in Domestic Ubiquitous Computing Environments. In Proceedings of Ubiqui-tous Computing (UbiComp '04), Nottingham, UK, September 7-10, 2004. 107-124. DOI=http://dx.doi.org/10.1007/978-3-540-30119-6_7

20. Bellamy, J.C. Digital Telephony, 3rd Edition. Chapter 6.1 Digital Modulation, pp. 279-308.

21. Beltran, A., Erickson, V.L., Cerpa, A.E. 2013. ThermoSense: Occupancy Thermal Based Sensing for HVAC Control. In Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings (BuildSys'13). ACM, New York, NY, USA. Article 11, 8 pages. DOI=http://dx.doi.org/10.1145/2528282.2528301

22. Ben-Menahe, A. 1995. "A Concise History of Main-stream Seismology: Origins, Legacy, and Perspectives". In Bulletin of the Seismological Society of America, Seismological Society of America, 85 (4): 1202–1225.

23. Bernaerts, Y., Druwé, M., Steensels, S., Ermeulen, J. and Schöning, J. The office smartwatch: development and design of a smartwatch app to digitally augment interactions in an office environment. In Proc. DIS '14, 41-44.

24. Bernstein, M.S., Brandt, J., Miller, R.C., Karger, D.R. Crowds in two seconds: enabling realtime crowd-powered interfaces. In Proc. UIST '11.

25. Berque, D., Burgess, J., Billingsley, A., Johnson, S., Bonebright, T.L., Wethington, B. 2011. Design and evaluation of persuasive technology to encourage healthier typing behaviors. In Proceedings of the 6th International Conference on Persuasive Technology: Persuasive Technology and De-sign: Enhancing Sustainability and Health (PERSUASIVE '11), Curtis P.

Haugtvedt and Agnis Stibe (Eds.). ACM, New York. Article 9, 10 pages. DOI: https://doi.org/10.1145/2467803.2467812

26. Bigham, J.P., Jayant, C., Ji, H., Little, G., Miller, R.C., Tatarowicz, A., White, B., White, S., Yeh, T. 2010. VizWiz: nearly real-time answers to visual questions. In Proceedings of the 23nd annual ACM symposium on User interface software and technology (UIST '10). ACM, New York, NY, USA, 333-342. DOI=http://dx.doi.org/10.1145/1866029.1866080
27. Blum, A., Mitchell, T. 1998. In Proceedings of the eleventh annual conference on Computational learning theory. ACM, 92-100.
28. Boyle, M., Edwards, C., Greenberg, S. 2000. The effects of filtered video on awareness and privacy. In Proceedings of the 2000 ACM confer-ence on Computer supported cooperative work (CSCW '00). ACM, New York, NY, USA, 1-10. DOI=http://dx.doi.org/10.1145/358916.358935
29. Boyle, M., Greenberg, S. 2005. The language of privacy: Learning from video media space analysis and design. In ACM Trans. Comput.-Hum. Interact. 12, 2 (June 2005), 328-370. DOI=http://dx.doi.org/10.1145/1067860.1067868
30. Brookfield, S. D., Preskill, S. 1999. Discussion as a way of teaching: Tools and techniques for democratic classrooms. San Francisco: Jossey-Bass.
31. Buchler, M.C. 2002. Algorithms for Sound Classification in Hearing Instruments. PhD thesis, ETH Zurich.
32. Buettner, M., Prasad, R., Philipose, M. and Wetherall, D. Recognizing daily activities with RFID-based sen-sors. In Proc. UbiComp '09, 51-60.
33. Bussmann, J.B.J., Martens, W. L. J. Tulen, J. H. M., Schasfoort, F. C., van den Berg-Emons, H. J. G., Stam, H.J. 2001. Measuring daily behavior using ambulatory accelerometry: The Activity Monitor. In Behavior Research Methods, Instruments, & Computers. August 2001, Volume 33, Issue 3, pp 349-356.
34. Cao Wireless Sensor Tags: Monitor and Find Every-thing from the Internet. Last accessed: January 20, 2017. http://www.caogadgets.com/
35. Carlson, A. et al. 2010. Toward an architecture for never-ending language learning. In *Proceedings of Twenty-Fourth AAAI Conference on Artificial Intelligence.* 1306-1313.
36. Chang, Y., Paruthi, G., Newman, M.W. 2015. A field study comparing approaches to collecting annotated activity data in real-world settings. In Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15). ACM, New York, NY, USA, 671-682. DOI: http://dx.doi.org/10.1145/2750858.2807524
37. Chapelle, O., Scholkpof, B., Zien, A. 2006. Semi-supervised learning. *IEEE Transactions on Neural Networks* 20, 3. IEEE, 542-542.

38. Chatterjee, S. et al. 2016. mCrave: continuous estimation of craving during smoking cessation. In Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '16). ACM, New York, NY, USA, 863-874. DOI: http://dx.doi.org/10.1145/2971648.2971672

39. Chen, K.Y., Cohn, G., Gupta, S. and Patel. S.N. uTouch: sensing touch gestures on unmodified LCDs. In Proc. of CHI '13. 2581-2584.

40. Choudhury, T., Borriello, G., Consolvo, S., Haehnel, D., Harrison, B., Hemingway, B., Hightower, J., Klasnja, P., Koscher, K., LaMarca, A., Landay, J.A., LeGrand, L., Lester, J., Rahimi, A., Rea, A., Wyatt, D. 2008. The Mobile Sensing Platform: An Embedded Activity Recognition System. In IEEE Pervasive Computing 7, 2 (April 2008), 32-41. DOI=http://dx.doi.org/10.1109/MPRV.2008.39

41. Clark, H.H., Brennan, S.E. 1991. Grounding in communication. Perspectives on Socially Shared Cognition. pp. 127 - 149. L. Resnick, J. Levine and S. Teasley, Editors. American Psychological Society, Washington, DC.

42. Clarkson, B. 2002. Life Patterns: Structure from Wearable Sensors. Ph.D. Thesis, Massachusetts Institute of Technology.

43. Clarkson, B., Sawhney, N. and Pentland, A. Auditory context awareness in wearable computing. In Workshop on Perceptual User Interfaces, November 1998.

44. Cohen, J. A coefficient of agreement for nominal scales. Educational and Psychological Measurement 20(1), 1960.

45. Cohn, G., Gupta, S., Froehlich, J., Larson, E., Patel, S.N. 2010. GasSense: Appliance-Level, Single-Point Sensing of Gas Activity in the Home. In Pervasive Computing: 8th International Conference, Pervasive 2010, Helsinki, Finland, May 17-20, 2010. Springer. 265-282. DOI=http://dx.doi.org/10.1007/978-3-642-12654-3_16

46. Cohn, G., Morris, D., Patel, S. and Tan, D. Humantenna: using the body as an antenna for real-time whole-body interaction. In Proc. of CHI'12. 1901-1910.

47. Cohn, G., Gupta, S., Froehlich, J., Larson, E. and Patel, S. GasSense: Appliance-Level, Single-Point Sensing of Gas Activity in the Home. In Proc. of Pervasive '10. 265-282.

48. Cohn, G., Morris, D., Patel, S.N. and Tan, D.S. Your noise is my command: sensing gestures using the body as an antenna. In Proc. of CHI '11. 791-800.

49. Cohn, G., Stuntebeck, E., Pandey, J., Otis, B., Abowd, G.D. and Patel, S.N. SNUPI: sensor nodes utilizing powerline infrastructure. In Proc. UbiComp '10. 159-168.

50. Consolvo, S., McDonald, D.W., Toscos, T., Chen, M.Y., Froehlich, J., Harrison, B., Klasnja, P., LaMarca, A., LeGrand, L., Libby, R., Smith, I., Landay., J.A. 2008. Activity sensing in the

wild: a field trial of ubifit garden. In Proceedings of the SIGCHI Conference on Hu-man Factors in Computing Systems (CHI '08). ACM, New York, NY, USA, 1797-1806.

51. Cornelius, C., Peterson, R., Skinner, J., Halter, R. and Kotz, D. A wearable system that knows who wears it. In Proc. MobiSys '14, 55-67.

52. Csikszentmihalyhi, M., Larson, R. 1987. Validity and Reliability of the Experience-Sampling Method. Journal of Nervous and Mental Disease. 175:526-536.

53. Cutkosky, M.R. 1989. On grasp choice, grasp models, and the design of hands for manufacturing tasks. In IEEE Transactions on Robotics and Automation, vol. 5, no. 3. 269-279.

54. Dalal, S., Alwan, M., Seifrafi, R., Kell, S., Brown, D. A rule-based approach to the analysis of elders activity data: Detection of health and possible emergency conditions. In Proceedings of AAAI Fall 2005 Symposium. 2545-2552, 2005.

55. De Castro Lopo, E. libsamplerate (software). http://www.mega-nerd.com/SRC/index.html

56. De Freitas, A.A., Dey, A.K. 2015. The Group Context Framework: An Extensible Toolkit for Opportunistic Grouping and Collaboration. In Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '15).

57. Dementyev, A. and Paradiso, J.A. WristFlex: Low-power gesture input with wrist-worn pressure sensors. In Proc. UIST '14, 161-166.

58. Deng, J., Krause, J., Fei-Fei, L. Fine-grained crowdsourcing for fine-grained recognition. In Proc. CVPR '13.

59. Dey, A.K. 1998. Context-aware computing: The CyberDesk project. In the Proceedings of the AAAI 1998 Spring Symposium on Intelligent Environments (AAAI Technical Report SS-98-02), pp. 51-54, Palo Alto, CA, AAAI Press. March 23-25, 1998.

60. Dey, A.K. 2000. Providing Architectural Support for Building Context-Aware Applications. Ph.D. Dissertation. Georgia Institute of Technology, Atlanta, GA, USA. AAI9994400.

61. Dey, A.K., Hamid, R., Beckmann, C., Li, I., Hsu, D. 2004. a CAPpella: programming by demonstration of context-aware applications. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '04). ACM, New York, NY, USA, 33-40. DOI=http://dx.doi.org/10.1145/985692.985697

62. Deyle, T., Palinko, S., Poole, E.S. and Starner, T. Hambone: A Bio-Acoustic Gesture Interface. In Proc. ISWC '07, 1-8.

63. Dialog Semiconductor. IoT Sensor Development Kit. Last accessed: January 20, 2017. http://www.dialog-semiconductor.com/iotsensor

64. Dietz, P. and Leigh, D. DiamondTouch: a multi-user touch technology. In Proc. of UIST '01. 219-226.

65. Dimoulas, C., Kalliris, G., Papanikolaou, G. and Kalampakas, A. Long-term signal detection, segmentation and summarization using wavelets and fractal dimension: A bio-acoustics application in gastrointestinal-motility monitoring. Comput. Biol. Med. 37, 4 (April 2007), 438-462.

66. Do, T.M.T., Kalimeri, K., Lepri, B., Pianesi, F. and Gatica-Perez, D. Inferring social activities with mobile sensor networks. In Proc of ICMI '13. 405-412.

67. Doan, H.G., Vu, H., Tran, H. 2015. Recognition of hand gestures from cyclic hand movements using spatial-temporal features. In Proceedings of the Sixth International Symposium on Information and Communication Technology (SoICT 2015). ACM, New York, NY, USA, 260-267. DOI=http://dx.doi.org/10.1145/2833258.2833301

68. Dollar, A.M., 2014. Classifying Human Hand Use and the Activities of Daily Living. The Human Hand as an Inspiration for Robot Hand Development. Volume 95, Springer Tracts in Advanced Robotics. 201-216.

69. Doyle, P., 2005. Echo and Reverb: Fabricating Space in Popular Music Recording. 2005. Wesleyan. ISBN 978-0819567949.

70. EchoFlex: Clean Tech Lighting & Temperature Con-trols. Last accessed: January 20, 2017. http://www.echoflexsolutions.com/

71. Elizalde, B., Badlani, R., Shah,A., Kumar, A., Raj, B. 2017. Never-Ending Learner of Sounds. In *NIPS Workshop on Machine Learning for Audio*, 2017. https://arxiv.org/pdf/1801.05544

72. Elkan, C., Noto, K. 2008. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (KDD '08). ACM, New York, NY, USA, 213-220. DOI: https://doi.org/10.1145/1401890.1401920

73. Enlightened. Redefining Smart Buildings. Last ac-cessed: January 20, 2017. http://www.enlightedinc.com

74. Ephrat, A., Mosseri, I., Lang, O., Dekel, T., Wilson, K., Hassidim, A., Freeman, W.T., Rubinstein, M. 2018. Looking to Listen at the Cocktail Party: A Speaker-Independent Audio-Visual Model for Speech Separation. https://arxiv.org/abs/1804.03619

75. Epstein, D.A., Kang, J.H., Pina, L.R., Fogarty, F., Munson, S.A. 2016. Reconsidering the device in the drawer: lapses as a design opportunity in personal informatics. In Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '16). ACM, New York, NY, USA, 829-840. DOI: http://dx.doi.org/10.1145/2971648.2971656

76. Eronen, A.J. et al. 2006. Audio-based context recognition. In *IEEE Transactions on Audio, Speech, and Language Processing* 14, no. 1 (2006): 321-329.

77. Fails, J., Olsen, D. 2003. A design tool for camera-based interaction. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03). ACM, New York, NY, USA, 449-456. DOI=http://dx.doi.org/10.1145/642611.642690

78. Fails, J.A. and Olsen, D. Light widgets: interacting in everyday spaces. In Proc. IUI '02.

79. FitBit. Healt and Activity Index. Last Accessed: April 2, 2017. https://www.fitbit.com/activity-index

80. Fogarty, J., Hudson, S.E., Atkeson, C.G., Avrahami, D., Forlizzi, J., Kiesler, S., Lee, J.C., Yang, J. 2005. Predicting human interruptibility with sensors. ACM Trans. Comput.-Hum. Interact. 12, 1 (March 2005), 119-146. DOI: https://doi.org/10.1145/1057237.1057243

81. Fogarty, J., Au, C. and Hudson, S.E.: Sensing from the Basement: A Feasibility Study of Unobtrusive and Low-Cost Home Activity Recognition. In Proc. UIST ’06, 91-100.

82. Foggia, P., Petkov, N., Saggese, A., Strisciuglio, N., Vento, M. 2015. Reliable detection of audio events in highly noisy environments. In *Pattern Recognition Letters 65* (2015): 22-28.

83. Font, F., Roma, R., Serra, X. 2013. Freesound technical demo. In Proceedings of the 21st ACM international conference on Multimedia (MM '13). ACM, New York, NY, USA, 411-412. DOI: https://doi.org/10.1145/2502081.2502245

84. Franklin, M. J., Kossmann, D., Kraska, T., Ramesh, S. and Xin, R. CrowdDB: answering queries with crowdsourcing. In Proc. SIGMOD '11.

85. Froehlich, J.E., Larson, E., Campbell, T., Haggerty, C., Fogarty, J., Patel, S.N. 2009. HydroSense: infrastructure-mediated single-point sensing of whole-home water activity. In Proceedings of the 11th international conference on Ubiquitous computing (UbiComp '09). ACM, New York, NY, USA, 235-244. DOI=http://dx.doi.org/10.1145/1620545.1620581

86. Fry, W.J. (1958). Biological and medical acoustics. The Journal of the Acoustical Society of America, 30(5), 387-393.

87. Fukui, R., Watanabe, M., Gyota, T., Shimosaka, M. and Sato, T. Hand shape classification with a wrist contour sensor: development of a prototype device. In Proc. Ubicomp '11, 311-314.

88. Goodfellow, I., Bengio, Y., Courville, A. 2016. Deep learning. Vol. 1. Cambridge: MIT Press.

89. Google, Inc. Compatibility Definition for Android 6.0. October 16, 2015. http://source.android.com/compatibility/android-cdd.pdf

90. Grosse-Puppendahl, T., Herber, S., Wimmer, R., Englert, F., Beck, S., von Wilmsdorff, J., Wichert, R., and Kuijper, A. Capacitive near-field communication for ubiquitous interaction and perception. In Proc. of UbiComp '14. 231-242.

91. Guo, A., Chen, X.A., Qi, H., White, S., Ghosh, S., Asakawa, C., Bigham, J.P. 2016. VizLens: A Robust and Interactive Screen Reader for Interfaces in the Real World. In Proceedings of the

29th Annual Symposium on User Interface Software and Technology (UIST '16). ACM, New York, NY, USA, 651-664. DOI: https://doi.org/10.1145/2984511.2984518

92. Gupta, S., Chen, K., Reynolds, M.S., Patel, S.N. 2011. LightWave: using compact fluorescent lights as sensors. In Proceedings of the 13th international conference on Ubiquitous compu-ting (UbiComp '11). ACM, New York, NY, USA, 65-74. DOI=http://dx.doi.org/10.1145/2030112.2030122
93. Gupta, S., Reynolds, M.S., Patel, S.N. 2010. ElectriSense: single-point sensing using EMI for electrical event detection and classification in the home. In Proceedings of the 12th ACM international conference on Ubiquitous computing (UbiComp '10). ACM, New York, NY, USA, 139-148. DOI=http://dx.doi.org/10.1145/1864349.1864375
94. Haché, G., Lemaire, E.D and Baddour, N. Wearable mobility monitoring using a multimedia smartphone platform. In IEEE Trans. on Instrumentation and Measurement '11. 3153-3161.
95. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. and Witten, I. The WEKA Data Mining Software: An Update. SIGKDD Explor., 11(1), 2009.
96. Hall, M.A., Correlation-based Feature Subset Selection for Machine Learning. 1998.
97. Hansen, P. 2001. Recent Bio-acoustical Publications, 1999 and earlier. Part 1: Invertebrates - birds. Bio-acoustics 11(3), 223-262
98. Hara, K., Sun, J., Moore, R., Jacobs, D., Froehlich, J.E. Tohme: Detecting Curb Ramps in Google Street View Using Crowdsourcing, Computer Vision, and Ma-chine Learning. In Proc. UIST '14.
99. Harrison, C., Tan, D. and Morris, D. Skinput: Appropriating the Body as an Input Surface. In Proc. CHI '10, 453-462.
100. Harrison, C., Xiao, R. and Hudson, S.E. Acoustic barcodes: passive, durable and inexpensive notched identification tags. In Proc. UIST '12. 563-568.
101. Hart, G. 1992. Nonintrusive Appliance Load Monitoring. In Proceedings of the IEEE EPRI Information and Automation Technology Conference, Washington, DC, June 26-28, 1992. 80(12), 1870-1891. DOI=http://dx.doi.org/10.1109/MIM.2016.7477956
102. Hart, G. Advances in Nonintrusive Appliance Load Monitoring. In Proceedings of EPRI Information and Automation Conference '91.
103. Hartmann, B., Abdulla, L., Mittal, M., Klemmer, S.R. 2007. Authoring sensor-based interactions by demonstration with direct manipulation and pattern recognition. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07). ACM, New York, NY, USA, 145-154. DOI=http://dx.doi.org/10.1145/1240624.1240646
104. Hartmann, B., Wu, L., Collins, K., Klemmer, S.R. 2007. Programming by a sample: rapidly creating web applications with d.mix. In Proceedings of the 20th annual ACM symposium on

User interface software and technology (UIST '07). ACM, New York, NY, USA, 241-250. DOI=http://dx.doi.org/10.1145/1294211.1294254

105. He, K., Zhang, X., Ren, S., Sun, J. 2016. "Deep residual learning for image recognition." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770-778. 2016.

106. Hershey, S., Chaudhuri, S., Ellis, D.P.W., Gemmeke, J.F., Jansen, A., Moore, R.C., Plakal, M. 2017. CNN architectures for large-scale audio classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (ICASSP), 2017, pp. 131-135. IEEE.

107. Hill, J., Horton, M., Kling, R., Krishnamurthy, L. 2004. The platforms enabling wireless sensor networks. In Communications of the ACM 47, 6 (June 2004), 41-46. DOI=http://dx.doi.org/10.1145/990680.990705

108. Hinton, G., Salakhutdinov, R. 2006. Reducing the Dimensionality of Data with Neural Networks. In Science 313 (5786): 504-507 (July 2006). DOI= http://dx.doi.org/10.1126/science.1127647

109. Hodges, S., Thorne, A., Mallinson, H. and Floerke-meier, C. Assessing and optimizing the range of UHF RFID to enable real-world pervasive computing appli-cations. In Pervasive Computing '07. 280-297.

110. Holz, C., and Baudisch, P. Fiberio: A Touchscreen That Senses Fingerprints. In Proc. of UIST'13. 41-50.

111. Hoque, E., Stankovic, J. 2012. AALO: Activity recognition in smart homes using Active Learning in the presence of Overlapped activities. In *Proceedings of 2012 6th International Conference on Pervasive Computing Technologies for Healthcare (Pervasive Health) and Workshops.* IEEE, 139-146.

112. Hudson, S.E., Fogarty, J., Atkeson, C., Avrahami, D., Forlizzi, J., Kiesler, S., Lee, J., Yang, J. 2003. Predicting human interruptibility with sensors: A Wizard of Oz feasibility study. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03). 257-264. DOI=http://dx.doi.org/10.1145/642611.642657

113. Hudson, S.E. and Smith, I. Techniques for addressing fundamental privacy and disruption tradeoffs in awareness support systems. In Proc. CSCW '96.

114. Iglewicz, B. and Hoaglin, D. (1993), Volume 16: How to Detect and Handle Outliers., The ASQC Basic Refer-ences in Quality Control: Statistical Techniques, Ed-ward F. Mykytka, Ph.D., Editor.

115. International Code Council, Inc. (2011). 2012 International Building Code (IBC). Country Club Hills, IL.

116. InvenSense, Inc. MPU-6500 Product Specification Re-vision 1.0. September 18, 2013. https://store.invensense.com/datasheets/invensense/MPU_6500_Rev1.0.pdf

117. InvenSense, Inc. MPU-6500 Register Map And De-scriptions Revision 2.1. September 16, 2013. https://www.invensense.com/wp-content/uploads/2015/02/MPU-6500-Register-Map2.pdf

118. Ioffe, S., Szegedy, C. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In ICML, volume 37 of JMLR Workshop and Conference Proceedings. 448-456.

119. Jansen, A. et al. 2017. Large-scale audio event discovery in one million youtube videos. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 786-790.

120. Jansen, A. et al. 2018. Unsupervised learning of semantic audio representations. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 126-130.

121. Jones, E., Oliphant, T., Peterson, P. et al. 2001. Scipy: Open source scientific tools for Python. http://www.scipy.org/, Accessed April 4, 2019.

122. Jung, P.G., Lim, G., Kim, S. and Kong, K. A Wearable Gesture Recognition Device for Detecting Muscular Activities Based on Air-Pressure Sensors. IEEE Trans. on Industrial Informatics, 11(2), 485-494. Feb. 2015.

123. Kastrinaki, V., Zervakis, M., Kalaitzakis, K. A survey of video processing techniques for traffic applications. Image and Vision Computing 21(4), 2003.

124. Khan, A., Mellor, S., Berlin, E., Thompson, R., McNaney, R., Olivier, P., Plötz, T. 2015. Beyond activity recognition: skill assessment from accelerometer data. In Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15). ACM, New York, NY, USA, 1155-1166. DOI: http://dx.doi.org/10.1145/2750858.2807534

125. Khan, M., Acharya, B. and Verm, S. Comparison be-tween different illumination independent change detec-tion techniques. In Proc. ICCCS '11.

126. Kim, D., Hilliges, O., Izadi, S., Butler, A., Chen, J., Oikonomidis, I. and Olivier, P. Digits: freehand 3D interactions anywhere using a wrist-worn gloveless sen-sor. In Proc. UIST '12, 167-176.

127. Kim, Y., Schmid, T., Charbiwala, Z. and Srivastava, M.B. ViridiScope: design and implementation of a fine-grained power monitoring system for homes. In Proc. of UbiComp ’09. 245-254.

128. King, D. E. 2009. Dlib-ml: A Machine Learning Toolkit. *Journal of Machine Learning Research* Vol. 10. 1755-1758.

129. Klingensmith, N., Bomber, J., Banerjee, S. 2014. Hot, cold and in between: enabling fine-grained environmental control in homes for efficien-cy and comfort. In Proceedings of the 5th international conference on Future energy systems (e-Energy '14). ACM, New York, NY, USA, 123-132. DOI=http://dx.doi.org/10.1145/2602044.2602049

130. Knocki: Turn Any Surface into a Remote Control. http://knocki.com/

131. Kopec, J.A., Esdaile, J.M. 1990. Bias in case-control studies: A review. Journal of epidemiology and community health. 44 (3): 179-86

132. Krogman, W.M. 1942. The Anthropology of the hand. Ciba Symposia, Vol. 4 (4). 1294.

133. Kuznetsov, S., Paulos, E. 2010. UpStream: motivating water conservation with low-cost water flow sensing and persuasive displays. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10). ACM, New York, NY, USA, 1851-1860. DOI=http://dx.doi.org/10.1145/1753326.1753604

134. Lam, M., Mirshekari, M., Pan, S., Zhang, P., Noh, H.Y. 2016. Robust occupant detection through step-induced floor vibration by incorporating structural characteristics. In *Dynamics of Coupled Structures*, Volume 4, pp. 357-367. Springer, Cham.

135. Lane, N.D., Georgiev, P., Qendro, L. 2015. DeepEar: robust smartphone audio sensing in unconstrained acoustic environments using deep learning. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. ACM, New York, NY, USA, 283-294. DOI: https://doi.org/10.1145/2750858.2804262

136. Lange, B.M., Jones, M.A. and Meyers, J.L. Insight lab: an immersive team environment linking paper, displays, and data. In Proc. CHI ’98, 550-557.

137. Laput, G., Harrison, C. 2019. Exploring the Efficacy of Sparse, General-Purpose Sensor Constellations for Wide-Area Ubiquitous Sensing. Under Review. In Proceedings of the ACM Journal on Interactive, Mobile, Wearable, and Ubiquitous Technologies (IMWUT UbiComp '19).

138. Laput, G., Harrison, C. 2019. Investigating Fine-Grained Hand Activities Using Commodity Smartwatches. To appear in Proceedings of the 37th Annual ACM Conference on Human Factors in Computing Systems (CHI '19).

139. Laput, G., Harrison, C. 2019. SurfaceSight: A New Spin on Touch, User, and Object Sensing for IoT Experiences. In Proceedings of the 37th Annual ACM Conference on Human Factors in Computing Systems (CHI '19).

140. Laput, G., Brockmeyer, E., Hudson, S.E., Harrison, C. 2015. Acoustruments: Passive, Acoustically-Driven, Interactive Controls for Handheld Devices. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15).

141. Laput, G. Adar, E., Dontcheva, M., Li, W. 2012. Tutorial-based interfaces for cloud-enabled applications. In Proceedings of the 25th annual ACM symposium on User interface software and technology (UIST '12). ACM, New York, NY, USA, 113-122.

142. Laput, G., Ahuja, K., Goel, M., Harrison, C. 2018. Ubicoustics: Plug-and-Play Acoustic Activity Recognition. In Proceedings of the 31st Annual ACM Symposium on User Inter-face Software and Technology (UIST '18). ACM, New York, NY, USA, 213-224.

143. Laput, G., Dontcheva, M., Wilensky, G., Chang, W., Agarwala, A., Linder, J., Adar, E. 2013. PixelTone: a multimodal interface for image editing. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13). ACM, New York, NY, USA, 2185-2194.

144. Laput, G., Xiao, R., Chen, X.A, Hudson, S.E., Harrison, C. 2014. Skin buttons: cheap, small, low-powered and clickable fixed-icon laser projectors. In Proceedings of the 27th annual ACM symposium on User interface software and technology (UIST '14). ACM, New York, NY, USA, 389-394.

145. Laput, G., Chen, X.A., Harrison, C. 2016. SweepSense: Ad Hoc Configuration Sensing Using Reflected Swept-Frequency Ultrasonics. In Proceedings of the 21st International Conference on Intelligent User Interfaces (IUI '16).

146. Laput, G., Lasecki, W.S., Wiese, J., Xiao, R., Bigham, J.P., Harrison, C. 2015. Zensors: Adaptive, Rapidly Deployable, Human-Intelligent Sensor Feeds. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Com-puting Systems (CHI '15). ACM, New York, NY, USA, 1935-1944. DOI: http://dx.doi.org/10.1145/2702123.2702416

147. Laput, G., Xiao, R., Harrison, C. 2016. ViBand: High-Fidelity Bio-Acoustic Sensing Using Commodity Smartwatch Accelerometers. In Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16). ACM, New York, NY, USA, 321-333. DOI: https://doi.org/10.1145/2984511.2984582

148. Laput, G., Yang, C., Xiao, R., Sample, A. and Harrison, C. EM-Sense: Touch Recognition of Uninstrumented, Electrical and Electromechanical Objects. In Proc. UIST '15, 157-166.

149. Laput, G., Zhang, Y., Harrison, C. 2017. Synthetic Sensors: Towards General-Purpose Sensing. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17). ACM, New York, NY, USA, 3986-3999. DOI: https://doi.org/10.1145/3025453.3025773

150. Lasecki, W., Gordon, M., Koutra, D., Jung, M., Dow, S., Bigham, J. Glance: Rapidly Coding Behavioral Video with the Crowd. In Proc. UIST ’14.

151. Lasecki, W.S., Song, Y., Kautz, H., and Bigham, J.P. Real-time crowd labeling for deployable activity recognition. In Proc. of CSCW ’13. 1203-1212.

152. Lee, H., Pham, P., Largman, Ng, A.Y. 2009. Unsupervised feature learning for audio classification using convolutional deep belief networks. *Advances in neural information processing systems (NIPS)*. 1096-1104.

153. Lee, S.W., Mase, K. 2002. Activity and Location Recognition Using Wearable Sensors. In IEEE Pervasive Computing 1, 3 (July 2002), 24-32. DOI=http://dx.doi.org/10.1109/MPRV.2002.1037719

154. Lee, M.L., Dey, A.K. 2015. Sensor-based observations of daily living for aging in place. In Personal Ubiquitous Computing. 19, 1 (January 2015), 27-43. DOI: http://dx.doi.org/10.1007/s00779-014-0810-3

155. Fei-Fei, L., Fergus, R., Perona, P. 2006. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence.* IEEE, 594-611.

156. Li, L.J., Fei-Fei, L. 2010. Optimol: automatic online picture collection via incremental model learning. *International Journal of Computer Vision* 88, 2. Springer, 147-168.

157. Li, H., Ye, C. and Sample, A. IDSense: A Human Object Interaction Detection System Based on Passive UHF RFID. In Proc. CHI '15, 2555-2564

158. Liang, D., Thomaz, E. 2019. Audio-Based Activities of Daily Living (ADL) Recognition with Large-Scale Acoustic Embeddings from Online Videos. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 1, Article 17 (March 2019), 18 pages. DOI: https://doi.org/10.1145/3314404

159. Libelium. WaspMote Event Module. Last accessed: January 20, 2017. http://www.libelium.com/products/waspmote/

160. Lien, J., Gillian, N., Karagozler, M.E., Amihood, P., Schwesig, C., Olson, E., Raja, H., Poupyrev, I. 2016. Soli: ubiquitous gesture sensing with millimeter wave radar. ACM Trans. Graph. 35, 4, Article 142 (July 2016), 19 pages. DOI: https://doi.org/10.1145/2897824.2925953

161. Lin, F.X., Ashbrook, D., White, S. 2011. RhythmLink: securely pairing I/O-constrained devices by tapping. In Proceedings of the 24th annual ACM symposium on User interface software and tech-nology (UIST '11). ACM, New York, NY, USA, 263-272. DOI=http://dx.doi.org/10.1145/2047196.2047231

162. Logan, B., Healey, J., Philipose, M., Tapia, E.M., Intille, S. 2007. A long-term evaluation of sensing modalities for activity recognition. In Proceedings of the 9th international conference on Ubiquitous computing (UbiComp '07). Springer-Verlag, Berlin, Heidelberg, 483-500.

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

164. Lowe, D.G. Object recognition from local scale-invariant features. In Proc. ICCV ‘99, 1150-1157.

165. Lu, H., Pan,W., Lane, N.D., Choudhury, T., Campbell, A.T. 2009. SoundSense: scalable sound sensing for people-centric applications on mobile phones. In *Proceedings of the 7th international conference on Mobile systems, applications, and services* (MobiSys '09). ACM, New York, NY, USA, 165-178. DOI=http://dx.doi.org/10.1145/1555816.1555834

166. Lukowicz, P., Ward, J.A., Junker, H., Stager, M., Troster, G., Atrash, A., Starner, T. 2004. Recognizing Workshop Activity Using Body Worn Microphones and Accelerometers. In *International Conference on Pervasive Computing*. Springer, 18-32.

167. Maekawa, T., Kishino, Y., Sakurai, Y. and Suyama, T. Recognizing the use of portable electrical devices with hand-worn magnetic sensors. In Proc. Pervasive '11. 276-293

168. Maekawa, T., Kishino, Y., Yanagisawa, Y. and Sa-kurai, Y. Recognizing handheld electrical device usage with hand-worn coil of wire. In Proc. Pervasive'12. 234-252.

169. Marcus, A., Karger, D., Madden, S., Miller, R., Oh, S. Counting with the crowd. In Proc. VLDB ’12.

170. Matrix One. The World’s First IoT App Ecosystem. Last accessed: January 20, 2017. http://matrix.one/

171. Maynes-Aminzade, D., Winograd, T., Igarashi, T. Eyepatch: prototyping camera-based interaction through examples. In Proc. UIST '07.

172. McFee, B., Humphrey, E., Bello, J.P. 2015. A Software Framework for Musical Data Augmentation." In *Proceeding International Society for Music Information Retrieval Conference (*ISMIR 2015), pp. 248-254.

173. McLoughlin, I., Zhang, H., Xie, Z., Song, Y., Xiao, W. 2015. Robust sound event classification using deep neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing,* 23, no. 3 (2015): 540-552.

174. McNeill, D. 1985. So You Think Gestures Are Nonverbal? In Psychological Review 92, No. 3. 350-371.

175. Medwin, H., Clay C.S. 1998. Fundamentals of Acous-tical Oceanography, Academic Press.

176. Mierswa, I., Morik, K. 2005. Automatic feature extraction for classifying audio data. *Machine learning* 58,2-3. Springer, 127-149.

177. Minnen, D., Starner, T., Essa, I., Isbell, C. 2006. Discovery characteristic actions from on-body sensor data. In *Proceedings of 2006 10th IEEE International Symposium on Wearable Computers*. IEEE, 11-18.

178. Morris, D., Saponas, T.S., Guillory, A., Kelner, I. 2014. RecoFit: using a wearable sensor to find, recognize, and count repetitive exercises. In Proceedings of the SIGCHI Conference on

Human Factors in Computing Systems (CHI '14). ACM, New York, NY, USA, 3225-3234. DOI: https://doi.org/10.1145/2556288.2557116

179. Moschetti, A., Fiorini, L., Esposito, D., Dario, P., Cavallo, F. 2016. Recognition of Daily Gestures with Wearable Inertial Rings and Bracelets. In Sensors (Basel, Switzer-land) vol. 16,8 1341. 22 Aug. 2016. DOI:10.3390/s16081341

180. Moschetti, A., Fiorini, L., Esposito, D., Dario, P., Cavallo, F. 2017. Daily activity recognition with inertial ring and bracelet: An unsupervised approach. In Proceedings of IEEE International Conference on Robotics and Automation (ICRA), Singapore, 2017, pp. 3250-3255. DOI: 10.1109/ICRA.2017.7989370

181. Movassaghi, S., Abolhasan, M., Lipman, J., Smith, D., Jamalipour, A. 2014. Wireless Body Area Networks: A Survey. In Comm. Surveys & Tutorials. 1658 - 1686.

182. Mujibiya, A., Cao, X., Tan, D.S., Morris, D., Patel, S.N. and Rekimoto, J. The sound of touch: on-body touch and gesture sensing based on transdermal ultra-sound propagation. In Proc. ITS '13, 189-198.

183. Mullner, D. et al. 2013. Fastcluster: Fast hierarchical, agglomerative clustering routines for R and Python. *Journal of Statistical Software* 53, 9. Foundation for Open Access Statistics, 1-18.

184. Music, J., Murray-Smith, J. 2010. Virtual hooping: teaching a phone about hula-hooping for fitness, fun and rehabilitation. In Proceedings of Mobile HCI (MobileHCI '10). ACM, New York, NY, USA, 309-312. DOI=http://dx.doi.org/10.1145/1851600.1851654

185. Naik, G.R., Wang, W. 2014. Blind Source Separation: Advances in Theory, Algorithms and Applications. In *Signals and Communication Technology*, Springer 2014. ISBN 9783642550164.

186. Nair, V., Hinton, G.E. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10). 807-814.

187. Network Sound Effects. https://www.sound-ideas.com/Product/199/Network-Sound-Effects-Library, Retrieved on July 12, 2018.

188. Nguyen, L.T., Zeng, M., Tague, P., Zhang, J. 2015. I did not smoke 100 cigarettes today!: avoiding false positives in real-world activity recognition. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (UbiComp '15). ACM, New York, NY, USA, 1053-1063. DOI: https://doi.org/10.1145/2750858.2804256

189. Nguyen, L.T., Zeng, M., Tague, P., Zhang, J. 2015. Recognizing new activities with limited training data. In *Proceedings of the 2015 ACM International Symposium on Wearable Computers* (ISWC '15). ACM, New York, NY, USA, 67-74. DOI: https://doi.org/10.1145/2802083.2808388

190. Notion. Wireless Home Monitoring System. Last accessed: January 20, 2017. http://getnotion.com/

191. Ogata, M. and Imai, M. SkinWatch: skin gesture interaction for smart watch. In Proc. AH '15, 21-24.

192. Ogata, M., Sugiura, Y., Makino, Y., Inami, M. and Imai, M. SenSkin: adapting skin as a soft interface. In Proc. UIST '13, 539-544.

193. Paglin, M.D., Hobson, J.R. and Rosenbloom, J. (1999). The Communications Act: A Legislative History of the Major Amendments, 1934-1996. Pike & Fische.

194. Pan, S., Mirshekari, M., Zhang, P., Noh, H.N. 2016. "Occupant traffic estimation through structural vibration sensing," In *Proc. SPIE 9803, Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2016*, 980306 (20 April, 2016). DOI= http://dx.doi.org/10.1117/12.2222024

195. Parascandolo, G., Heittola, T., Huttunen, H., Virtanen, T. 2017. Convolutional recurrent neural networks for polyphonic sound event detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25, 6: 1291-1303. DOI: https://doi.org/10.1109/TASLP.2017.2690575

196. Pascoe, J. 1998. Adding generic contextual capabilities to wearable computers. In the Proceedings of the 2nd IEEE International Symposium on Wearable Computers (ISWC'98), pp. 92-99, Pittsburgh, PA, IEEE. October 19-20, 1998.

197. Patel, S.N., Reynolds, M.S., Abowd, G.D. 2009. Detecting Human Movement by Differential Air Pressure Sensing in HVAC System Duct-work: An Exploration in Infrastructure Mediated Sensing. In Proceedings of Pervasive '08. Springer-Verlag, Berlin, Heidelberg, 1-18. DOI=http://dx.doi.org/10.1007/978-3-540-79576-6_1

198. Patel, S.N., Robertson, T., Kientz, J.A., Reynolds, M.S. and Abowd, G.D. At the flick of a switch: detecting and classifying unique electrical events on the residential power line. In Proc. of UbiComp '07. 271-288.

199. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E. 2011. Scikit-learn: Machine Learning in Python. J. Mach. Learn. Res. 12 (Novem-ber 2011), 2825-2830.

200. Pejovic, V., Mirco, M..2014. "InterruptMe: designing intelligent prompting mechanisms for pervasive applications." In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pp. 897-908. ACM, 2014.

201. Peltonen, V., Tuomi, J., Klapuri, A., Huopaniemi, J. and Sorsa, T. Computational auditory scene recogni-tion. In IEEE Acoust, Speech, and Signal Proc. 1520-6149.

202. Perkowitz, M., Philipose, M., Fishkin, K., Patterson, D.J. 2004. Mining models of hu-man activities from the web. In Proceedings of the 13th international conference on World Wide Web (WWW '04). ACM, New York, NY, USA, 573-582. DOI=http://dx.doi.org/10.1145/988672.988750

203. Perng, J.K., Fisher, B., Hollar, S. and Pister, K.S. Acceleration sensing glove. In Proc. ISWC '09, 178-180.

204. Phan, H.P., Hertel, L., Maass, M., Mertins. A. 2016. Robust audio event recognition with 1-max pooling convolutional neural networks. https://arxiv.org/abs/1604.06338

205. Philipose, M. Large-scale human activity recognition using ultra-dense sensing. The Bridge, National Academy of Engineering. 35, 4 (2005).

206. Piczak, K.J. 2015. ESC: Dataset for Environmental Sound Classification. In *Proceedings of the 23rd ACM international conference on Multimedia* (MM '15). ACM, New York, NY, USA, 1015-1018. DOI: https://doi.org/10.1145/2733373.2806390.

207. Polikar, R., Upda, L., Upda, S.S., Honavar, V. 2001. Learn++: An incremental learning algorithm for supervised neural networks. *IEEE transactions on systems, man, and cybernetics, part C (applications and reviews)* 31, 4. IEEE, 497-508.

208. Qian, Z., Sagers, R.D. and Pitt, W.G. (1999), Investi-gation of the mechanism of the bio-acoustic effect. In J. Biomed. Mater. Res., 44: 198–205.

209. Randell, C. and Muller, H. Context awareness by analysing accelerometer data. In Proc. ISWC '00. 175-176

210. Ranjan, J., Yao, Y., Griffiths, E. and Whitehouse, K. Using mid-range RFID for location based activity recognition. In Proc. of UbiComp '12. 647-648.

211. Rantz, M.J. 2011. Using sensor networks to detect urinary tract infections in older adults. In IEEE 13th International Conference on e-Health Networking, Applications and Services, Columbia, MO, 2011, pp. 142-149.

212. Ravi, N., Dandekar, N., Mysore, P., Littman, M. 2005. Activity recognition from accelerometer data. In Proceedings of the 17th conference on Innovative applications of artificial intelligence - Volume 3 (IAAI'05), Bruce Porter (Ed.), Vol. 3. AAAI Press 1541-1546.

213. Ravichandran, R., Saba, E., Chen, K.Y., Goel, M., Gupta, S., Patel, S.N. 2015. Wi-Breathe: Estimating respiration rate using wireless signals in natural settings in the home. In IEEE International Conference on Pervasive Computing and Communications (PerCom), St. Louis, MO, 2015, pp. 131-139.

214. Rebuffi, S.A. et al. 2017. iCaRL: Incremental Classifier and Representation Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2001-2010.

215. Reed, I.S. and Solomon, G. Polynomial Codes over Certain Finite Fields. J. Soc. Ind. Appl. Math. 8(2), 300-304 (1960).

216. Rekimoto J. and Ayatsuka, Y. CyberCode: designing augmented reality environments with visual tags. In Proc. DARE '00, 1-10

217. Rekimoto, J. Gesturewrist and gesturepad: Unobtrusive wearable interaction devices. In Proc. ISWC '01, 21-27.

218. Relayr Wunderbar. Last accessed: January 20, 2017. https://relayr.io/wunderbar/

219. Ren, X. 2009. Egocentric recognition of handled objects: Benchmark and analysis. In Proc. CVPR Workshops ’09, 2160-7508.

220. Rose, J. 2008. Producing Great Sound for Film and Video. 3rd Edition. Focal Press and Elsevier Inc. ISBN 978-0-240-80970-0.

221. Rosenthal, S.L., Veloso, M., Dey, A.K. 2012. “Acquiring accurate human responses to robots’ questions.” *International journal of social robotics* 4, no. 2 (2012): 117-129.

222. Rosenthal, S.L., Dey, A.K.. 2010. “Towards maximizing the accuracy of human-labeled sensor data.” In *Proceedings of the 15th international conference on Intelligent user interfaces*, pp. 259-268. ACM, 2010.

223. Rossing, T.D., Moore, F.R., Wheeler, P.A.. 2001. The Science of Sound. 3rd Edition. Pearson. ISBN 978-0805385656.

224. Rowe, A., Gupta, V., Rajkumar, R. 2009. Low-power clock synchronization us-ing electromagnetic energy radiating from AC power lines. In Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys '09). ACM, New York, NY, USA, 211-224. DOI=http://dx.doi.org/10.1145/1644038.1644060

225. Roy, N. and Choudhury, R. Ripple II: Faster Communication through Physical Vibration. In Proc. NSDI '16, USENIX Association, 671-684.

226. Ryan, N., Pascoe, J., Morse, D. 1998. Enhanced reality fieldwork: the context-aware archaeological assistant. Computer Applications and Quantitative Methods in Archaeology. V. Gaffney, M. van Leusen and S. Exxon, Editors. Oxford.

227. Sagha, H. et al. 2013. Robust activity recognition combining anomaly detection and classifier retraining. In *Proceedings of 2013 IEEE International Conference on Body Sensor Networks*. IEEE, 1-6.

228. Salamon, J., Bello, J.P. 2015. Feature learning with deep scattering for urban sound analysis. In *Proc. 23rd European Signal Processing Conference* (EUSIPCO), 2015. IEEE.

229. Salber, D., Dey, A. K., Abowd, G. D. The context toolkit: aiding the development of context-enabled ap-plications. In Proc. CHI '99.

230. Salton, G., Buckley, C. 1988. Term-weighting approaches in automatic text retrieval. In Information Processing & Management. 24 (5): 513–523. DOI=http://dx.doi.org/10.1016/0306-4573(88)90021-0

231. Samsung Electronics. SmartThings: Smart Home, Intelligent Living. Last accessed: January 20, 2017. https://www.smartthings.com

232. Saponas, T.S., Tan, D.S., Morris, D. and Balakrish-nan, R. Demonstrating the feasibility of using forearm electromyography for muscle-computer interfaces. In Proc. CHI '08, 515-524.

233. Saponas, T.S., Tan, D.S., Morris, D., Balakrishnan, R., Turner, J. and Landay, J. A. Enabling always-available input with muscle-computer interfaces. In Proc. UIST '09, 167-176.

234. Sato, M., Poupyrev, I. and Harrison, C. Touché: en-hancing touch interaction on humans, screens, liquids, and everyday objects. In Proc. CHI '12, 483-492.

235. Scherer, D., Müller, A.C., Behnke, S. 2010. Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition. In Artificial Neural Networks (ICANN), 20th International Conference on. Thessaloniki, Greece: Springer. 92-101.

236. Schilit, B.N., Adams, N.I., Want, R. 1994. Context-aware computing applications. In the Proceedings of the 1st International Workshop on Mobile Computing Systems and Applications, pp. 85-90, Santa Cruz, CA, IEEE. December 8-9, 1994.

237. Scholkopf, B., Williamson, R.C., Smola, A.J., Shawe-Taylor, J., Platt, J.C. 2000. Support vector method for novelty detection. *Advances in neural information processing systems (NIPS)*. 582-588.

238. Scott, J., Bernheim, A.J.B., Krumm, J., Meyers, B., Hazas, M., Hodges, S., Villar, N. 2011. PreHeat: controlling home heating us-ing occupancy prediction. In Proceedings of the 13th international conference on Ubiquitous computing (UbiComp '11). ACM, New York, NY, USA, 281-290. DOI=http://dx.doi.org/10.1145/2030112.2030151

239. Sears WallyHome: Smart Home Sensing and Moisture Detection. Last accessed: January 20, 2017. https://www.wallyhome.com/

240. Seeed Studio. ReSpeaker 2-Mic PHAT. https://www.seeedstudio.com/ReSpeaker-2-Mics-Pi-HAT-p-2874.html, Retrieved on July 18, 2018

241. Sen.se Mother. The Universal Monitoring Solution. Last accessed: January 20, 2017. https://sen.se/mother/

242. Settles, B. 2009. Active learning literature survey. *CS Technical Reports*. University of Wisconsin-Madison Department of Computer Sciences.

243. Shipman, F.M., Gutierrez-Osuna, R., Monteiro, C.D.D. 2014. Identifying Sign Language Videos in Video Sharing Sites. In ACM Trans. Access. Comput. 5, 4, Article 9 (March 2014), 14 pages. DOI: http://dx.doi.org/10.1145/2579698

244. ShotSpotter. www.shotspotter.com, Retrieved on April 3, 2018.

245. Simmonds, J. and MacLennan, D. 2005. Fisheries Acoustics: Theory and Practice, second edition. Blackwell Press.

246. Simonyan, K., Zisserman, A. 2014. Very Deep Convo-lutional Networks for Large-Scale Image Recognition. In arXiv. eprint arXiv:1409.1556. 2014arXiv1409.1556S

247. Smith, J.R., Fishkin, K.P., Jiang, B., Mamishev, A., Philipose, M., Rea, A.D., Roy, S. and Sundara-Rajan. K. RFID-based techniques for human-activity detection. In Commun. ACM. Vol. 48-9, 2005. 39-44.

248. SoundSnap. https://www.soundsnap.com, Retrieved on July 12, 2018.

249. Stager, M., Lukowicz, P., Perera, N., von Buren, T., Troster, G., Starner, T. 2003. Soundbutton: Design of a low power wearable audio classification system. In *Proceedings of the Seventh IEEE International Symposium on Wearable Computers* (ISWC '03). IEEE.

250. Standards for the Diagnosis and Management of Patients with Chronic obstructive pulmonary disease (COPD). American Thoracic Society. http://www.thoracic.org/copd/patients_general.asp

251. Starner, T., Schiele, B., Pentland, A. 1998. Visual con-textual awareness in wearable computing". In Proc. of the Second International Symposium on Wearable Computers (Cat. No.98EX215), 1998, pp. 50-57.

252. Stojanov, T., Ding, X. 2015. Operators Skill Level Evaluation Method for Balancing of an Apparel Assembly Line. Int. J. Productiv. Manage. As-sess. Technol. 3, 1 (January 2015), 1-12. DOI=http://dx.doi.org/10.4018/IJPMAT.2015010101

253. Stork, J.A.. Spinello, L., Silva, J., Arras, K.O. 2012. Audio-based human activity recognition using non-markovian ensemble voting. In *RO-MAN*, 2012 IEEE, pp. 509-514. IEEE.

254. Stowell, D., Giannoulis, D., Benetos, E., Lagrange, M., Plumbley, M.D. 2015. Detection and classification of acoustic scenes and events. In *IEEE Transactions on Multimedia*, 17, no. 10 (2015): 1733-1746.

255. Tang, A., Greenberg, S., Fels, S. 2008. Exploring video streams using slit-tear visualiza-tions. In Proceedings of the working conference on Ad-vanced visual interfaces (AVI '08). ACM, New York, NY, USA, 191-198. DOI=http://dx.doi.org/10.1145/1385569.1385601

256. Tapia, E.M., Intille, S.S., Larson, K. 2004. Activity Recognition in the Home Using Simple and Ubiquitous Sensors. In Pervasive Computing: Second International Conference, PERVASIVE 2004, Linz/Vienna, Austria, April 21-23, 2004. Springer. 158-175. DOI=http://dx.doi.org/10.1007/978-3-540-24646-6_10

257. Tapia, E.M., Intille, S.S., Larson, K. 2007. Portable wireless sensors for object usage sensing in the home: challenges and practicalities. In Proceedings of the 2007 European conference on Ambient intelligence (AmI'07). Springer-Verlag, Ber-lin, Heidelberg, 19-37.

258. Taylor, J.S. 1985. Vibration syndrome in industry: Dermatological viewpoint. Am. J. Ind. Med., 8: 415-432. doi:10.1002/ajim.4700080423

259. Temko, A., Nadeu. C. 2009. Acoustic event detection in meeting-room environments. *Pattern Recognition Letters* 30, 14. Elsevier, 1281-1288.

260. Texas Instruments. SimpleLink SensorTag. Last accessed: January 20, 2017. http://www.ti.com/ww/en/wireless_connectivity/sensortag2015/gettingStarted.html

261. The Federal Communications Commission. Code of Federal Regulations, Title 47, Part 15.

262. Theano Development Team. “Theano: A Python framework for fast computation of mathematical ex-pressions”. In arXiv e-prints, abs/1605.02688.

263. Theodoridis, S., Koutroumbas, K. 2010. Pattern Recognition & Matlab Intro (4th ed.). Academic Press, Inc., Orlando, FL, USA.

264. Torralba, A., Murphy, K.P., Freeman, W.T. and Rubin, M.A. Context-based vision system for place and object recognition. In Proc. ICCV ‘03, 273-280.

265. Torres-Martinez, E., Paules, G., Schoeberl, M., Kalb, M. 2003. "A Web of Sensors: Enabling the Earth Science Vision". In Acta Astronautica, Volume 53, Issues 4-10. pp. 423–428.

266. Tzanetakis, G., Cook, P. 2002. Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing* 10, 5. IEEE 293-302.

267. Ul Alam, M.A., U Roy, N. 2017. Unseen Activity Recognitions: A Hierarchical Active Transfer Learning Approach. In *Proceedings of 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 436-446.

268. van den Oord Aaron van den Oord, et al. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499.*

269. Vinh, N.X., Epps, J., Bailey, J. 2010. *Journal of Machine Learning Research* 11, Oct. 2837-2854.

270. Vinyals, O. et al. 2016. Matching networks for one shot learning. *Advances in neural information processing systems (NIPS).* 3630-3638.

271. Vogler Christian Vogler and Dimitris Metaxas. 1998. ASL recognition based on a coupling between HMMs and 3D motion analysis. In Proc. of the Sixth International Conference on Computer Vision, IEEE Cat. No.98CH36271, Bombay, 1998, pp. 363-369.

272. Von Ahn, L. and Dabbish, L.. Labeling images with a computer game. In Proc. CHI '04.

273. Vondrick, C., Patterson, D., and Ramanan, D. Efficiently Scaling Up Crowdsourced Video Annotation. Inter-national Journal of Computer Vision.

274. Wah, C. Crowdsourcing and its applications in computer vision. University of California, San Diego. 2006.

275. Walton, A. Life Expectancy of a Smartphone. Retrieved from chron.com, September 21, 2014.

276. Wang, W., Miao, C., Hao, S. 2017. Zero-shot human activity recognition via nonlinear compatibility based method. In *Proceedings of the International Conference on Web Intelligence* (WI '17). ACM, New York, NY, USA, 322-330. DOI: https://doi.org/10.1145/3106426.3106526

277. Wang, E.J., Lee, T.J., Mariakakis, A., Goel, M., Gup-ta, S. and Patel, S.N. MagnifiSense: inferring device interaction using wrist-worn passive magneto-inductive sensors. In Proc. UbiComp '15, 15-26.

278. Wang, Y., Liu, J., Chen, Y., Gruteser, M., Yang, J. and Liu, H. E-eyes: device-free location-oriented activity identification using fine-grained WiFi signatures. In Proc. of MobiCom '14. 617-628.

279. Want, E., Hopper, A., Falcão, V., Gibbons, J. 1992. The active badge location system. ACM Trans. Inf. Syst. 10, 1 (January 1992), 91-102.

280. Ward Joe H Ward Jr. 1963. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association* 58, 301. Taylor & Francis Group, 236-244.

281. Ward, A.M.R. 1998. Sensor-driven computing. PhD dissertation. Computer Laboratory, University of Cambridge. Cambridge, UK.

282. Ward, J.A., Lukowicz, P., Tröster, G. and Starner, T.E. Activity recognition of assembly tasks using body-worn microphones and accelerometers. IEEE Trans. on Pattern Analysis & Mach. Intelligence '06. 1553-1567.

283. Ward, J.A., Lukowicz, P., Tröster, G., Starner, T.E. 2006. Activity recognition of assembly tasks using body-worn microphones and accelerometers. In IEEE Transactions on Pattern Analysis and Machine Intelligence. 28, 10 (2006), 1553–1567.

284. Watanabe, H., Terada, T., Tsukamoto, M. 2013. Ultrasound-based movement sensing, gesture-, and context-recognition. In Proceedings of the 2013 International Symposium on Wearable Computers (ISWC '13). ACM, New York, NY, USA, 57-64. DOI=http://dx.doi.org/10.1145/2493988.2494335

285. Way, D. and Paradiso, J. A usability user study concerning free-hand microgesture and wrist-worn sensors. In Proc. BSN '14, 138-142.

286. Weiser, M. 1991. The computer for the 21st Century. Scientific American 265(3): pp. 66-75. September 1991.

287. Weiser, M., Brown, J.S. 1997. The coming age of calm technology. Beyond Calculation: The Next Fifty Years of Computing. pp. 75-85. Peter J. Denning and Robert M. Metcalfe, Editors. Springer- Verlag, New York.

288. Wen, H., Rojas, J.R. and Dey, A. Serendipity: Finger Gesture Recognition using an Off-the-Shelf Smart-watch. In Proc. CHI '16, 3847-3851.

289. Willet, P. 1988. Recent trends in hierarchic document clustering: A critical review. In Information Processing & Management. 24 (5): 577-597. DOI=http://dx.doi.org/10.1016/0306-4573(88)90027-1

290. Wilson, D.H., Atkeson, C. 2005. Simultaneous tracking and activity recognition (STAR) using many anonymous, binary sensors. In Proceedings of the Third international conference on Pervasive Computing (PERVASIVE'05), Hans-W. Gellersen, Roy Want, and Albrecht Schmidt (Eds.). Springer-Verlag, Berlin, Heidelberg, 62-79. DOI=http://dx.doi.org/10.1007/11428572_5

291. Wilson, F.R. 1998. The Hand: How its use shapes the brain, language, and human culture. Random House, Inc.

292. Wolf, C. 1942. The Human Hand. Methuen, London.

293. Wopereis Iwan G. J. H. Wopereis, Jeroen J. G. van Merriënboer, and Paul A. Kirschner. 2010. Improvising in music: a learning biography study to reveal skill acquisition. In Proceedings of the 9th International Conference of the Learning Sciences - Volume 2 (ICLS '10), Vol. 2. International Society of the Learning Sciences 419-420.

294. Wyatt, D., Philipose, M., Choudhury, T. 2005. Unsupervised activity recognition using automatically mined common sense. In Proceedings of the 20th national conference on Artificial intelligence - Volume 1 (AAAI'05), Anthony Cohn (Ed.), Vol. 1. AAAI Press 21-27.

295. Xian, Y. et al. 2018. Zero-shot learning – a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence.* IEEE.

296. Xu, C., Pathak, P.H. and Mohapatra, P. Finger-writing with Smartwatch: A Case for Finger and Hand Gesture Recognition using Smartwatch. In Proc. HotMobile '15, 9-14.

297. Yang, J., Gao, J., Zhang, Y., Waibel, A. 2001. Towards automatic sign translation. In Proceedings of the first international conference on Human language technology research (HLT '01). 1-6. DOI=http://dx.doi.org/10.3115/1072133.1072223

298. Yatani, K., Truong, K.N. 2012. BodyScope: a wearable acoustic sensor for activity recognition. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing* (UbiComp '12). ACM, New York, NY, USA, 341-350. DOI=http://dx.doi.org/10.1145/2370216.2370269

299. Yeo, H., Flamich, G., Schrempf, P., Harris-Birtill, D., Quigley, A. 2016. Radar-Cat: Radar Categorization for Input & Interaction. In Proceedings of the 29th Annual Symposium on User

Interface Software and Technology (UIST '16). ACM, New York, NY, USA, 833-841. DOI: https://doi.org/10.1145/2984511.2984515

300. Zhang, B., Su, J., Xu, X. 2006. A class-incremental learning method for multi-class support vector machines in text classification. In *Proceedings of 2006 International Conference on Machine Learning and Cybernetics*. IEEE, 2581-2585.

301. Zhang, O., Srinivasan, K. 2016. Mudra: User-friendly Fine-grained Gesture Recognition using WiFi Signals. In Proceedings of the 12th International on Conference on emerging Net-working Experiments and Technologies (CoNEXT '16). ACM, New York, NY, USA, 83-96. DOI: https://doi.org/10.1145/2999572.2999582

302. Zhang, Y. and Harrison, C. Tomo: Wearable, Low-Cost, Electrical Impedance Tomography for Hand Gesture Recognition. In Proc. UIST '15, 167-173.

303. Zhao, N., Dublon, G., Gillian, N., Dementyev, A. and Paradiso, J.A. EMI Spy: Harnessing electromagnetic interference for low-cost, rapid prototyping of proxemic interaction. In Proc. Wearable and Implantable Body Sensor Networks 2015, 1-6.

304. Zhao, W., Chellappa, R., Phillips, P. and Rosenfeld, R. Face recognition: A literature survey. ACM Comput. Surv. 35(4), 2003.

305. Zhao, Y., LaMarca, A. and Smith, J.R. A battery-free object localization and motion sensing platform. In Proc. UbiComp '14. 255-259.

306. Zheng, F., Zhang, G., Song, Z. (2001), "Comparison of Different Implementations of MFCC," J. In Computer Science & Technology, 16(6): 582–589

307. Zhong, L., El-Daye, D., Kaufman, B., Tobaoda, N., Mohamed, T. and Libschner, M. OsteoConduct: wire-less body-area communication based on bone conduction. In Proc. BodyNets '07. ICST, Article 9, 8 pages.

308. Zimmer, T., & Beigl, M. AwareOffice: Integrating Modular Context-Aware Applications. In Proc. ICDCSW’06.

309. Zimmerman, T.G. 1996. Personal area networks: near-field intrabody communication. IBM Syst. J. 35, 3-4 (September 1996), 609-617.