ZOE: A Cloud-less Dialog-enabled Continuous Sensing Wearable Exploiting Heterogeneous Computation

Nicholas D. Lane[†], Petko Georgiev[§], Cecilia Mascolo[§], Ying Gao[‡]
[†]Bell Labs, [§]University of Cambridge, [‡]Intel Research

ABSTRACT

The wearable revolution, as a mass-market phenomenon, has finally arrived. As a result, the question of how wearables should evolve over the next 5 to 10 years is assuming an increasing level of societal and commercial importance. A range of open design and system questions are emerging, for instance: How can wearables shift from being largely health and fitness focused to *tracking a wider range of life events*? What will become the *dominant methods through which users interact with wearables* and consume the data collected? Are wearables *destined to be cloud and/or smartphone dependent* for their operation?

Towards building the critical mass of understanding and experience necessary to tackle such questions, we have designed and implemented ZOE – a match-box sized (49g) collar- or lapel-worn sensor that pushes the boundary of wearables in an important set of new directions. First, ZOE aims to perform multiple deep sensor inferences that span key aspects of everyday life (viz. personal, social and place information) on *continuously sensed data*; while also offering this data not only within conventional analytics but also through a speech dialog system that is able to answer impromptu casual questions from users. (*Am I more stressed this week than normal?*) Crucially, and unlike other rich-sensing or dialog supporting wearables, ZOE achieves this *without* cloud or smartphone support – this has important side-effects for privacy since all user information can remain on the device. Second, ZOE incorporates the latest innovations in system-on-a-chip technology together with a custom daughter-board to realize a three-tier low-power processor hierarchy. We pair this hardware design with software techniques that manage system latency while still allowing ZOE to remain energy efficient (with a typical lifespan of 30 hours), despite its high sensing workload, small form-factor, and need to remain responsive to user dialog requests.

Categories and Subject Descriptors

H.1.2 [**User/Machine Systems**]: Human information processing

General Terms

Design, Experimentation, Performance

Keywords

mobile sensing, wearable, dialog system, continuous sensing

1. INTRODUCTION

The wearable market is rapidly advancing both in terms of raw technology and user adoption rates. As is often the case during periods of considerable innovation, wearable technology continues to prompt questions at a faster rate than we are able to answer them. In this paper, we start to examine a series of wearable system design questions that we believe will be important for the evolution of this category of mobile sensing devices moving forward. Specifically, we investigate in the context of wearables:

1. Continuous Sensing of a Diverse Range of User Behaviors and Environments. Most wearables today largely focus on health and fitness. What is missing is a more comprehensive awareness of the lives of users because real people have existences outside of the gym. Increasingly sophisticated sensing algorithms are being developed on more powerful platforms, such as smartphones, that can gather deep insights into areas like our conversation patterns [46], incidents of stress [50], the places we live and work [33]. An understanding needs to be developed about how such sensing techniques can be integrated together into wearable-class hardware and run simultaneously on continuous sensor streams.

2. Cloud-free Operation by exploiting Heterogeneous Local Computation. The capabilities and sophistication of mobile energy-efficient processors progress at a relentless rate. For example, an iPhone 6 represents a 10-fold increase in computation over a 5-year old iPhone 3GS. Extensive research has given us a solid set of techniques and an understanding of cloud-offloading and cyber-foraging [18] – with recent work even exploring within the context of wearable specifically [30]. But benefits exist for privacy (as data never leaves the device) and energy with purely device-only solutions. Consumers are growing ever more wary of the remote processing of sensor data, that contains potentially sensitive personal information [16]. An important gap in our understanding is precisely how far we can push emerging wearable hardware to execute sensing and related algorithms without the assistance of remote computation. Options within mobile architectures continue to diversify (e.g., big.LITTLE technology, availability of FPGAs and DSPs, CPUs interacting with sensor-attached MCUs). To maximize these advances, we need to study how to best architect the sensor inference and user interaction workloads for wearables cross such heterogeneous hardware.

3. In-situ User Sensor Data Analysis with Near Real-time Responsiveness. Detailed analysis of user data from wearables is largely only computed and offered to the user in off-line fashion for post event consumption. For many activities which involve the "now", near real-time reporting can become pivotal: examples include the calculation of current levels of stress with respect to other days, simple queries for memory assistance – when did I last visit this place? Or the reporting on past sporting activity performance compared to the current activity. Supporting such forms of user experience requires investigation of how inference, analysis and user interaction methods can be provided on-demand. It also provokes questions: could any seldom used natural user interface methods, such as speech dialog systems, be made responsive and resource efficient enough for wearables?

We investigate these questions through the design and implementation of a novel wearable sensor – ZOE. The goal of ZOE is to continuously sense a comprehensive set of user behaviors and contexts that span three key areas of everyday life (specifically personal-, social- and place-oriented sensing); while also providing natural impromptu access to collected information with an interactive speech dialog subsystem (complemented with more conventional user interaction methods). Towards this goal, we have developed an integrated sensing module that incorporates a range of state-of-the-art sensing algorithms designed for devices more powerful than typical wearable hardware – critically, almost all of these algorithms have previously been studied in isolation. Together, with the demands of the dialog system, this forms a highly demanding sensing and interaction workload, parts of which require near real-time responsiveness.

We demonstrate this workload can be serviced with a wearable device – without cloud assistance – and still provide acceptable energy and responsiveness performance. This is possible through a custom wearable hardware prototype and a combination of workload optimizations and resource management algorithms. The prototype for ZOE is designed around the recently released Intel Edison SoC [5] that includes a dual-core 500 MHz CPU and 100 MHz MCU. In our design, Edison is coupled with another MCU present on a custom designed daughter-board to form a 3-tier computational hierarchy. Each tier in the hierarchy offers a different computation and energy trade-off that, when managed efficiently, enables ZOE to meet user experience needs.

The contributions of our work include:

- The study of a collection of diverse, complex sensing algorithms that are integrated together and performed simultaneously within the computation and energy constraints of wearable-class hardware.

- The proposal of a new, spontaneous way for wearable users to interact with their data through a verbal dialog. We demonstrate this is feasible even without cloud assistance while offering near real-time responsiveness.

- The design and prototyping of new wearable hardware that offers the heterogeneous computation necessary to meet the workload demands of ZOE. Through proposed performance techniques, we show the hardware capabilities of our prototype are efficiently used in support of sensing and interaction workloads.

- The systematic evaluation of the end-to-end design of ZOE using a variety of large-scale datasets and fine-grain energy and responsiveness measurements. Our results indicate ZOE can provide acceptable levels of inference accuracy, battery life and responsiveness.

2. ZOE DESIGN OVERVIEW

We begin our description of ZOE, with a comprehensive overview. After describing how this wearable can impact the everyday life of its users, we proceed to discuss the design goals that are natural extensions of such scenarios. We close with a comparison of ZOE to existing wearables of commercial, academic or industry lab origins.

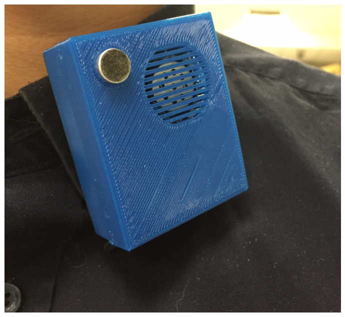


Figure 1: ZOE Prototype

2.1 Targeted User Experiences

ZOE offers many new capabilities not seen in today's wearables. By sketching usage scenarios for these capabilities we also highlight the technical challenges that must be solved.

Life with ZOE. As shown in Figure 1, ZOE is designed to be worn on a collar or lapel. At just 49g and with a typical lifetime of 30 hours, it is designed to be worn constantly during everyday activities. As such, ZOE aims to capture a broad set of user actions and ambient conditions, not just a single domain such as health and wellbeing. Shown in Table 1, the current prototype captures a diverse set of inferences from three main areas of life (viz. personal sensing, social sensing, place sensing). Because ZOE has built-in WiFi-host support any nearby computer or the user's own smartphone can be used to browse collected user information by simply connecting to the wearable (and being able to successfully authenticate), with ZOE acting as a local web-server (technique first proposed in [69]). Beyond such conventional interaction methods, users are also able to spontaneously ask spoken questions regarding their data and perform basic control operations (see Table 3). By providing both interaction methods users have conventional access to comprehensive analytics about their behavior, while still being able to informally gather short personal factoids whenever a question occurs to them. Significantly, ZOE can provide all of this functionality without the support of the cloud or off-loading computation to the user's smartphone. This addresses some aspects of user privacy concerns by allowing all data to remain on the device rather than data leaking to the cloud for analysis.

We envision many scenarios of potential use for ZOE. Within the sphere of personal use there are a number of natural application domains related to mHealth, life-logging, cognitive assistance, persuasive systems and the quantified-self movement. In more emerging fields there are applications of productivity, "life-hacking" and time accounting in which users want to optimize their activities by maximizing what they can accomplish each day. For example, with minor changes to the design presented in this paper, users would be able to understand which behavioral patterns lead to their most productive days or more simply track precisely how they spend their days in the office. When multiple ZOE devices are used in organizational and company settings, we expect they will be able to highlight, for instance, which teams work more closely together and how such co-operation relates to intra- and inter-team social interactions. Finally, ZOE can be a powerful tool in the hands of social scientists providing unobtrusively captured fine-grain behavioral data from which they can understand fundamental characteristics and relationships that shape our actions.

Technical Challenges. For ZOE to function as described above, the following challenges must be addressed.

1. Multi-domain User Sensing. To capture a broad understanding of a user's life will require multiple inference algorithms operating

Category	Subtype	Inferences
Personal	Transportation	{*motorized*, *non-motorized*}
	Phy. Activities	{*walk*, *stationary*, *run*, *other*}
	Stress Detection	{*stressed*, *neutral*}
Social	Social Interaction	{*conversation*, *other*}
	Conversation Analysis	estimators: *dominance*, *turns*
Place	Place Recognition	{*home*, *work*, *other*}
	Place Context	occupancy estimation
		estimators: *music*, *noise*, *chatter*

Table 1: Sensor inference capabilities of ZOE

in concert, often even simultaneously as they examine differing elements of the same event. Broadening the scope of user monitoring also requires more complex algorithms capable of mining deeper types of inferences.

2. Energy Efficiency. Wearables like ZOE work best when worn continuously day-after-day. This becomes impractical if they must be frequently recharged by users. Therefore wearables must be highly energy efficient and remain active for multiple days on a single charge. The design tension here, however, is that for similar reasons they also need to be small and light – which promotes small batteries. Furthermore, the need for additional sensing algorithms (see above) only applies more pressure to battery reserves.

3. Near Real-time Processing. The user experience of dialog interactions are highly time sensitive; for example, a typical pause before responding in normal conversations is around 500 ms [27]. The benefit of offering casual and impromptu questions or instructions disappears if user wait times are either excessive or even just inconsistent (leading to uncertainty – e.g., did it understand me?). Similarly, because ZOE provides not only longitudinal information but also the ability to query recent events or make comparisons (e.g., Am I being interrupted more today than normal?), sensor inferences must complete in a timely fashion, likely over time-scales of an hour or two.

2.2 Prototype Design

We will now describe in detail how ZOE overcomes the challenges just described. However, before diving into such details we briefly outline key aspects of our hardware design and software dataflow.

Hardware Prototype. As shown in Figure 2, the current hardware is only 49mm by 10mm with a depth of 57mm. Figure 1 shows ZOE within its 3D-printed case. These dimensions and device weight measurement (49g) assume the use of a 1500 mAh battery – as shown in all figures. Further details of our hardware design are discussed in §5.1.

The heart of our prototype is the Intel Edison [5] SoC that is paired with a custom daughter-board containing a programmable micro-controller, wireless charger, speaker and sensors (viz. gyroscope, accelerometer and microphone). The Edison itself has a further two processing units, the primary one being a dual-core 500 MHz Atom processor which is supported by a 100 MHz Quark processor. WiFi is also built-in directly to the Edison SoC which is required for both sensing and user interaction tasks. Essentially they form a three-tier processing hierarchy. Collectively, different configurations of these tiers offer a variety of energy usage and computational trade-offs that we leverage towards high levels of responsiveness and energy efficiency in ZOE.

Software Operation. ZOE is comprised of three core components illustrated in Figure 3. Specifically these are: Sensing Algorithms (§3), Dialog Subsystem (§4) and Performance Controller (§5). Each component relies on a secondary set of shared auxiliary modules (§6) – the most prominent of these is the User Knowledge

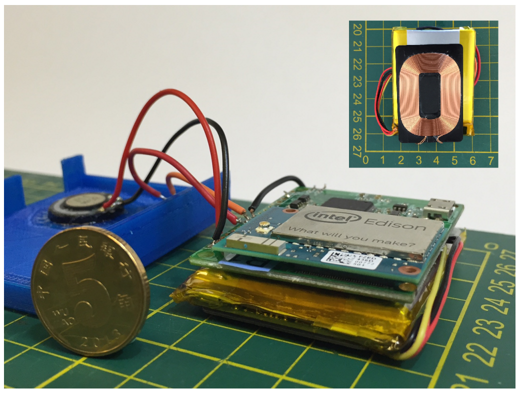


Figure 2: ZOE hardware showing the heterogeneous MCUs and CPU. (Inset image: Underneath prototype, showing recharging coil.)

Store. We now describe the operation of ZOE in terms of each of these core components.

Sensing Algorithms. We integrate into a single sensing module a broad set of accelerometer-, gyroscope-, WiFi- and microphone-based based algorithms that infer user behaviors and ambient contexts. Table 1 summarizes the specific inferences made by ZOE continuously throughout the day. Inferences are selected to provide well-rounded sensing coverage across key areas of the user's life. In our current prototype these domains include: *Personal Sensing* – which spans the actions, behaviors and the status of the user themselves; *Social Sensing* – that monitors and quantifies the interactions the user has with other people; and finally, *Place Sensing* – aiming to track locations of importance to users (home, work etc.) and characterize their conditions.

The stream of inferences generated by this component are organized into a local user knowledge store designed to support dialog queries. Using a few relatively simple inference processing mechanisms even single inferences generate multiple knowledge base entries. For example, as the user is detected to arrive home by ZOE, many behavioral data points are created, including: the duration of the commute, the distance, and a simple carbon footprint estimate. By storing this information in structured storage, supporting a diverse set of user inquires is simplified. Additional details are presented later in §6.

Dialog Subsystem. The purpose of this component is to recognize and respond to user queries for information and device commands (e.g., privacy settings) they issue. In many respects, the Sensing Algorithms component previously described shares many characteristics with the Dialog Subsystem. Both components track sensor data (only the microphone in this case) and react by performing different actions based on the content of observed data. For example, when a special activation keyword is spoken (e.g., the name of the system) then the complete sentence is analyzed so that the system can understand and respond. Our implementation includes optimizations based on fixed point math and fast methods for processing Gaussian Mixture Models (GMMs) on embedded hardware. However,

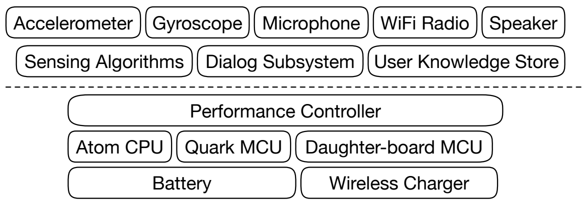


Figure 3: ZOE Architecture

Mobile Device	Dimensions	Weight	Lifetime	Battery	Cloud?	Inferences Made
ZOE	47 x 10 x 57mm	49g	30 hours	1500mAh	No	Physical Activity (Walking, Stationary, Running, Other); Stress; Transportation; Place; Place Context; Voicing; Conversation Analysis
MSP [23]	51mm x 36mm x NA	115g	10/20 hr	1800 mAH	No	Physical Activity (Walking, Running, Cycling, Stairs); Cooking; TV; Voicing; Eating; Working on a Computer; Elliptical Trainer; Elevator
Gabriel [30]	Glasss-sized	42g	1 hr	2.1Wh	Yes	Face Recognition; OCR; Object Recognition; AR; Motion/Activity
Microsoft Band [9]	19mm x 8.7 mm x NA	60g	48 hr	200mAh	Partial	Heart Rate; Sleep; Exercise Routine; Jogging; Hiking; Cycling
Jawbone Up3 [7]	220 x 12.2 x 3.0-9.3mm	29g	168 hr	38 mAh	No	Heart Rate; Sleep; Running; Steps; Calories; Automated Fitness Activity Recognition
Opo [35]	14cm^2	11.4g	93 hr	40mAh	No	Physical Proximity; Face-to-Face Interaction
LG G Watch R [8]	46.4 x 53.6 x 9.7mm	62g	23 hr [2]	410 mAh	Partial	Heart Rate; Sleep, Physical Activity; Voice Commands; Running; Steps; Calories

Table 2: Comparison of ZOE with recent and/or popular wearables from the literature and available commercially

How much time did I spend driving over the last week?
How many different places did I visit in the last four days?
Give me the number of times I was here this month?
Compared to the previous time I was here am I more stressed?
How does the amount of my cycling this week compare to last week?
How long did I talk to people this morning?
What fraction of my time do I spend listening to music?
On which day do I travel in a vehicle most?
Where do I spend more time chatting with people than usual?
Am I more stressed than normal on weekdays?
How many times was I feeling stressed this month?
If I leave for home now how long does it normally take me to get there?
When did I finish jogging?

Table 3: Representative sentences supported by Dialog Subsystem

we find that the most critical factor in determining the accuracy and latency of the Dialog Subsystem is the design of the language model and the composition of the vocabulary. Table 3 provides representative examples of our design decisions, under which interactions remain quasi-natural but limited in scope to device inferences and control. Responses are determined by the template-matching technique, each template is tied to handlers in code that dictates how to respond.

Performance Controller. This component collectively refers to the key design decisions that largely enable ZOE to operate for more than a day with a single battery charge and be responsive while still performing a rich set of sensing inferences. Our control plane leverages the built-in inter-processor communication mechanisms for synchronizing the operation of the three processor units (2 MCUs and Atom CPU). Critically, the MCUs offload the sensor sampling from the Atom and execute sensor data pre-filtering to control the workload generated for the CPU. The Quark MCU, for instance, filters silent samples and approaches the CPU for computation only when noise is detected in the user surroundings. This allows the CPU to remain in low-power sleep state for a large fraction of the day. Other functions falling under the umbrella term of performance control include speech activation (to spawn speech-related pipelines only when the context is human voice), owner triggered keyword spotting (to detect the commands issued by the owner), as well as dialog command prioritization over sensor processing to promote system responsiveness.

2.3 Comparisons to Existing Wearables

Table 2 surveys a range of wearables both from the literature and those commercially available. Devices in this table capture a number of wearable sub-categories. First, the very popular, but health-focused offerings from Jawbone and Microsoft. Second, examples of systems built on Google Glass (or glass-like) devices that primarily rely on vision techniques for inference. Third, very small single purpose devices like Opo [35] that have excellent battery life – in the case of Opo specializing in tracking face-to-face user interactions. Fourth, devices such as the Mobile Sensing Platform (MSP) [23] and the watch-style wearables represented in the table by the LG G Watch R [8]. Because these final two device classes are both multi-sensor and computationally capable, they therefore can support a variety of user inferences. However, watch wearable apps are still in their infancy and so the MSP is the closest wearable to ZOE in terms of breadth of *user-focused* activity recognition supported.

Inference Depth. One clear message from Table 2 is that *ZOE distances itself from other wearables due to the range of inferences it can support.* Commercial wearables, for example, only partially support the personal sensing inferences of ZOE and miss advanced microphone features such as stress detection – although ZOE is unable to track sleep, something that all of these health-targeting wearables can perform. Gabriel [30] is the closest comparison in shear inference depth because it leverages the vision capabilities of Google Glass to offer a number of deep context inferences. However, this device has much worse battery-life as well as being larger and heavier. The MSP is the nearest single device in terms of size and weight – supporting a few high-level user inferences similar to those of ZOE. But even acknowledging mobile technology advancements since 2008 (when the MSP v2, with which we compare, was developed), unlike ZOE this system does not include the performance control mechanisms needed to the maximize benefits of such hardware.

In fact, some of the best comparisons of inference breadth available are not designed for wearables but rather phones. For example, apps like PEIR [56] and CenceMe [54] support multiple inferences (such as activity, voicing, and transportation in the case of CenceMe) during their operation. Essentially ZOE provides levels of inference depth that exceeds even sensing apps for smartphones, and brings this experience to a different, miniaturized class of device: wearables.

Battery-Life and Form-Factor. ZOE has a high ratio of inference breadth to energy efficiency. It has a battery life comparable to the Microsoft Band [9] (30 hours vs 48 hours) but with a larger set of inferences; other devices, like Gabriel, that also support a number of inferences have lifespans of just 1 or 2 hours. Yet, ZOE does not approach the lifespans of the Jawbone Up3. In terms of form-factor, its size and weight are similar again to commercial devices like the Band. Although ZOE has a collar form factor and most devices in Table 2 are wrist-worn, the volumes of these devices are still comparable.

Interaction Model. None of the wearables we survey offer the same set of interaction modes as ZOE. All rely on cloud-based services to present wearable data to either the user's smartphone or desktop. None support the direct serving of data from the wearable to the user's devices like ZOE. Only one device (the Microsoft Band [9]) includes dialog support, but this was supported via a smartphone interacting with the cloud rather than the wearable operating in isolation. However, many devices do include screens for

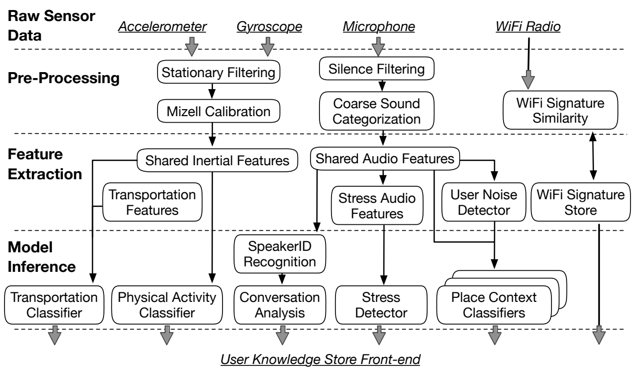


Figure 4: Sensing Algorithm Module

presenting data – although the lapel position of ZOE likely makes reading a screen difficult.

Leveraging Heterogeneous Computation. The general approach of MCU offloading of tasks from a CPU is not new for wearables. The recent MotoActv smartwatch [10] from Motorola, to name just one wearable, performs step counting on an MCU rather than the more energy consuming CPU. ZOE however relies on dual MCUs, with the Quark MCU itself having similar computational capacity as the MCU used by the Microsoft Band (ARM Cortex M4 MCU processor). Due to this design, and a number of performance controller optimizations (see §5), this architecture affords ZOE previously unseen features such as always-on audio sensing and user dialog support, all without cloud involvement. Other wearables include similarly rich computational options (the LG G Watch R, for example, incorporates a Qualcomm Snapdragon 400 SoC [11] complete with 4-core CPU and Hexagon DSP). But, such hardware is not yet being used to its full potential (in the case of the LG G Watch R its software only exploits 1 CPU core, and not the DSP [2]).

3. SENSING ALGORITHMS

In this section, we describe the sensing algorithms adopted in the current design of ZOE that span the categories of personal-, social- and place-related inference. Each form of sensing incorporated has already been validated in previously published work. The focus in ZOE is to combine these techniques into a single integrated sensing module that operates within hardware (and energy) constraints.

Overview. Figure 4 details the interaction between each signal processing and machine learning component needed to implement the collection of sensing algorithms in ZOE. A high degree of component sharing can be seen in the figure, especially between those components operating on data from the same sensor. Overall, the data-flow of each individual algorithm within the module is, at a high-level, very similar. *All sensors are continuously sampled* with this data feeding into pre-processing stages that perform operations like applying a low-pass filter or estimating the direction of gravity (see §6 for more details). Domain specific features are then extracted that project raw data into representations designed to accentuate patterns of similarity between features from the same logical class (intra-class distance); while doing the opposite (i.e., emphasizing the differences) between features of different logical classes (inter-class distance). Finally classification models determine which class the data belongs to (e.g., does the collected data correspond to the user driving or not driving).

Accelerometer and gyroscope processing algorithms (transportation mode and activity inferences) share pre-processing techniques and have a number of overlapping common features. Furthermore, both utilize the same classifier design and combine a static classifier with simple temporal smoothing. WiFi-radio processing is very simple in comparison and is not shared with any other algorithm. By far the most complex is audio processing. Two pipeline branches are encoded, one for human speech and one for background ambient sound. Shared pre-processing includes silence detection and coarse category classification {music, voicing, other}. Deep processing occurs along the voicing branch with stages including binary user-oriented speakerID and stress detection, while ambient audio is mined for various characteristics of the place (e.g., number nearby people). Sharing features is common in the voicing branch as are models in some cases (such as GMMs). Furthermore, social interaction metrics (that are somewhat distinct from the other classifiers included in the module) reuse whole components together – for example, conversation is detected using coarse classification, the presence of silence periods along with speaker ID to recognize that a user is actually involved in the conversation and simply not just nearby.

As previously mentioned, raw inferences from this module are expanded upon while being stored in the User Knowledge Store (§6). For example: frequency counts, durations, start and end-times. This explicitly captures multiple pieces of information each inference can hold about a person.

3.1 Personal Sensing

ZOE supports 3 forms of personal sensing; namely the detection of physical activities and transportation mode along with a recognizer of periods when the user is under stress.

Physical Activities. By extracting a series of 24 time- and frequency-domain features (adopted from [52]) from the accelerometer and gyroscope, the common user actions of {walk, stationary, run, other} are tracked. A 7-layer deep J.48 decision tree is trained that is then coupled to a 4-state Markov Model which smooths class transitions based on typical activity sequences. This classifier combination is common practice in many mobile sensing systems such as in [47]. Although relatively simple, this design has been shown to be sufficient for this combination of activities (e.g., [57]).

Transportation Mode. Most proposed transportation mode recognition techniques require data from either the GPS [63] or a WiFi-radio [41]. Instead ZOE incorporates less common accelerometer- and gyroscope-based approaches (e.g., [52, 32]) that can achieve similar levels of accuracy with a significantly lower energy overhead (as inertial sensors require much less power than GPS or a WiFi radio). In its current design this classifier only makes a binary decision {motorized, non-motorized}; although [32] indicates with further refinement finer-grain distinctions (trains, buses, cars) are possible. Again, we use a 4-second frame of 128 accelerometer samples and a superset of the features used for physical activity tracking are taken (adopted from [32]) with additional adopted features proposed first in [32]. To perform classification, we settle on an Adaboost ensemble learner with decision stumps acting as the weak learner (essentially two-level decision trees) – empirically we find that an ensemble of 20 stumps maximizes accuracy. The final inference is made after (again) smoothing with a two-state Markov model. In [32], this technique was found to outperform other candidate designs and, historically, it has been popular in many activity recognition settings (e.g., [47]).

Stress Periods. ZOE applies the findings of [50], and more loosely [61], towards detecting periods when the user is under stress. Whenever the user is detected to be speaking the voicing sample is analyzed and assigned into one of two categories: {*stressed*, *neutral*}. In addition to shared acoustic features that are also used by all audio classifiers in ZOE, stress detection uses additional features that focus on: pitch, speaking rate and a 17-dimension feature (TEO-CB-AutoEnv) that captures distinctive variations, especially in high-level talking styles ("anger", "loud"). For classification, as in [50], we use 16-component GMMs for both classes with a decision made by applying a likelihood function assuming equal priors. A single universal model is applied under all conditions and for all users, we leave albeit useful extensions to the [50] method involving model adaptation for future exploration.

3.2 Social Sensing

ZOE leverages two microphone pipelines, namely speech detection and speaker identification, as the basis for gathering a set of conversation-related metadata and inferences. The speech detection component is based on a simple decision tree and enables the detection of when and for how long a user socializes/interacts with other nearby people. The algorithm borrows elements from [51] and extracts features from 32ms sound frames every 1.28s when noise is detected. It has the dual role of both providing social inferences and acting as a trigger for other speech-related analysis.

The speaker identification module is used to detect whether the currently talking speaker is the owner of the wearable. This enables the monitoring of a wide set of additional conversation patterns including dominance (i.e. the fraction of time the owner is talking), turn taking periods, and even interactiveness (whether the user takes short turns frequently with other people) [37]. The speaker identification algorithm is adopted from [61] and relies on two Gaussian Mixture Models, one representing the targeted speaker and the other acting as a sink for other people. Similarly to the original pipeline, Perceptual Linear Predictive (PLP) coefficients are extracted every 30ms and accumulated over a window of 3s. We use a shorter window to capture the onset of turns.

3.3 Place Sensing

The final category of sensing performed by ZOE is based on significant places – i.e., locations that users frequently visit and spend the majority of their time. We utilize two types of place sensing: one that performs place recognition and another that analyzes the ambient conditions of a place.

Place Recognition. The most common method to recognize places is by comparing the signatures of WiFi scans (e.g., [40, 33]). A signature is simply a vector of WiFi access point SSIDs. If the signatures of two scans differ enough then they are assumed to be two distinct places; similarly, a user returning to the same places again (such as their home) is detected when the current signature matches prior visits. We perform WiFi scans periodically on a 15-minute duty-cycle, and use the Tanimoto coefficient to compute signature similarity. Because this approach is unable to determine place semantics (e.g., the meaning of the place) ZOE adopts a heuristic derived from [42] to identify the two most important user locations – their home and work locations. This is done based on the times and days of arrival and departure, along with visit frequency.

Place Context Analysis. To characterize the user's environment in different places, ZOE adopts a series of audio-based sensing techniques originally proposed in [70] and [67]. By combining the techniques of these two respective systems we are able to coarsely monitor: the number of people close to the user, and the intensity of background chatter, music and overall noise level. Three categories of place context (viz. music, noise, chatter) are recognized with category specific 4-class GMMs, each class corresponding to the level of intensity of the particular characteristic. Relatively common audio classification features are used, and these overlap with features already required for already described audio classifiers; the

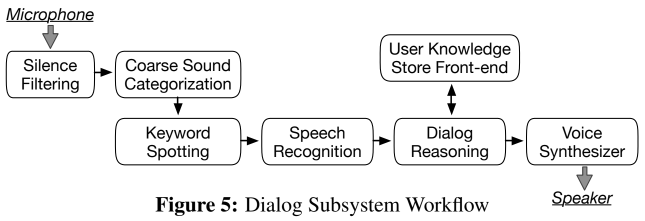


Figure 5: Dialog Subsystem Workflow

exception being a binary indicator that signals when the user is engaged in conversation that must be accounted for as it can skew the assessment of the ambient environment. In contrast, the estimation of occupancy is performed in a distinctly different manner, using an unsupervised counting algorithm from [70]. The features that feed into this counting algorithm include a 19-dimension MFCC feature vector along with pitch features that are used for gender disambiguation. A final occupancy estimate is produced by a clustering like process designed for this specific task.

4. DIALOG SUBSYSTEM

ZOE understands user instructions and requests through a purpose-built dialog system designed to operate within a series of hardware- and energy-based constraints.

Overview. Figure 5 illustrates the components that comprise the dialog subsystem, along with the dataflow between them. Operation begins with a continuous stream of microphone data from which audio frames containing non-silence and spoken words are extracted. This front-end is shared with the sensing algorithms detailed in §3. When voicing frames are identified then Keyword Spotting (KWS) is performed to recognize any sentences that begin with the dialog activation phrase (in the case of our prototype: "*ZOE*"). Any sentences beginning with the phrase are escalated to the Speech Recognition unit. At this stage, all words are attempted to be recognized through by a constrained vocabulary and language model that describes the boundaries of what ZOE is expected to understand. A template-based dialog reasoning phase occurs next that maps the sequence of inferred words for each sentence to a response or action from ZOE. Most responses are synthesized sentences spoken aloud by ZOE, the specific user information provided is determined by the keywords recognized in the dialog template. Alternatively, user instructions simply just alter a device setting, such as temporarily disabling sensing for reasons of user privacy.

Keyword Spotting. Robust recognition of the activation phrase (e.g., the name of the device) is critical to successful dialog support. False positive results will cause needless processing of irrelevant sentences. False negative results cause ZOE to ignore user speech input. To minimize this, we implement a state-of-the-art deep learning-based recognizer modeled on recent advances [21] in low-memory and low-complexity Deep Neural Networks (DNNs).

Within the KWS stage, and a continuous stream of sampled audio, certain frames that are identified as containing voicing are first represented as 40-dimension filter-bank energies (FBE) [20]. This variety of features is closely related to the MFCC and PLP features used in a number of the audio-related sensing described in §3. However, within the context of speech recognition FBE has been shown to be much more robust than its MFCC and PLP counterparts. Features are extracted from 30 ms audio frames that slide by 10 msec at a time. A 3-hidden-layer DNN is used that begins with 128 nodes corresponding closely to values of features that are ultimately linked to 2 output states in the final layer – one for each possible outcome – {keyword, no keyword}. The value of each node in the DNN is based on the values of all nodes in the prior

Processor	Operation
Quark MCU	Silence Filtering
Daughter-board MCU	Stationary Filtering, Mizell Calibration Physical Activities Classifier Transportation Classifier Shared Inertial Features Transportation Features
Edison CPU	Shared Audio Features, Stress Detector User Noise Detector, WiFi Similarity Conversation Analysis, SpeakerID Stress Features

Table 4: Allocation of ZOE workload to hardware components

layer and computed with activation functions uniquely parameterized on a node by node basis. As is standard practice, the final (2-state) layer uses softmax activation functions to result in a binary outcome given earlier inputs. Similarly, the proceeding layers have activation functions trained by a rectifier linear unit [24] (ReLU) selected for their performance in other related KWS systems.

Speech Recognition. To understand even a relatively limited vocabulary and constrained language model using a DNN-style solution similar to the one just described is not yet possible given the local computation available to ZOE. The length of processing delays – to name just one problem – would be unacceptable to users. Instead, we adopt proven speech recognition methods that target memory and computation constrained devices (e.g., [29]). We also carefully assemble our vocabulary and language model to sharply lower runtime evaluation (reported in §7.3) while still supporting $\approx$300 words and thousands of supported unique sentences.

Once KWS confirms the presence of the activation word within voicing speech audio segments, the beginning of that frame are passed to the Speech Recognition stage. We train an acoustic model using a 5-state HMM coupled to 39-dim. GMMs at each state corresponding to sub-word utterances. Multiple groups of candidate words for each sentence can be computed using the acoustic model. These candidates are further assessed with a bi-gram language model to judge the likelihood that particular word collections are correct given their sequential usage with an entire sentence.

Dialog Reasoning. We adopt the popular template-based approach to dialog interactions. Based on the probability of words in the recognized sentence it is matched to one of a fixed set of query templates by a fitness score. Below a fixed threshold the user is asked for clarification. For each template a query handler exists to generate requests to the User Knowledge Store (see §6). Parameters are determined by spoken keywords in the sentence; if one of them has an accuracy below a threshold, the user is asked to clarify. The logic in each query handler might dictate other processing to resolve certain keywords such as "normal" which will enact a separate query to determine the average for the user within some bound. As has been found in dialog system research, for close-world domains template matching can be highly effective but difficult to scale to general purpose usage – often due to its dependence on hand crafting of templates and template handlers.

Voice Synthesizer. ZOE replies to user dialog through a lightweight text-to-speech (TTS) synthesizer that incorporates the *formant synthesis* [26] technique. This approach can support a variety of languages and pronunciations without resorting to recordings of actual human voices; a critical factor in maintaining a low computational requirements along with a small memory footprint. Input to this component is simply plain text responses generated during Dialog Reasoning. Like most TTS systems words are then broken into the phonemes that describe how the letters in words should sound when spoken aloud. As a result, the generated speech is stilted and less natural (i.e., robotic sounding) than synthesizers with more onerous runtime resource requirements (based on real recordings). eSpeak uses a "formant synthesis" method. This allows many languages to be provided in a small size. The speech is clear, and can be used at high speeds, but is not as natural or smooth as larger synthesizers which are based on human speech recordings.

5. PERFORMANCE CONTROLLER

In this section, we outline the hardware and optimization design choices that heavily determine ZOE's energy profile.

5.1 Hardware Prototype

To provide context for the optimization control plane, we begin with a component-level description of the hardware. We briefly provide specifications of two primary modules (viz. Edison and daughter-board); additionally we describe how the 3-tier inter-processor communication functions.

Intel Edison SoC. The SoC and mount board is only 3.55 x 2.5 x 0.39 cm in size. Within the SoC are two programmable units. First, a dual-core Atom 22nm "Tangier" CPU that runs at 500 MHz and has access to 1GB of RAM. Second, a Quark processor operating at 100 MHz is currently configured as an MCU; both code and memory must occupy less than 192KB.

Custom Daughter-board. Designed to be roughly the same width of Edison, our daughter-board is around twice the height. All necessary sensors are either mounted or connected to this device. It also contains the power regulator components necessary to connect both a battery and the wireless recharging coil. Importantly, it also contains the 3rd processor (that acts as a 2nd MCU): an IT 8350 processor (from ITE) running at 48 MHz with 64KB of RAM.

Inter-Processor Communication. The operation of both MCUs (Quark and Daughter-board processor) is similar with respect to the signaling and message facilities enabling the communication with the primary CPU. Within the respective OSs of each MCU, a controller arbitrates incoming and outgoing messages transfered via IPC calls and shared memory. Within the MCU, code communication takes the forms of function calls. In the CPU, control nodes exist within the kernel that are also wrapped by libraries supporting interaction from within CPU-based source code.

5.2 Optimizations

We now elaborate on several design choices that are aimed at maximizing the power and performance benefits of the hardware components listed in the previous subsection. Table 4 summarizes the placement of primary ZOE components (shown in Figure 3) between computational tiers.

Sensor Pre-Processing. All microphone and accelerometer algorithms rely on a front-end of shared pre-processing routines residing partly in the MCUs or the Atom CPU. In the case of accelerometer-based ones (activities and transportation mode) these routines are placed on the daughter-board MCU and concern admission control and calibration. The admission control pre-processing is simple: by applying low-pass filter along with a threshold it seeks to recognize when the device is completely stationary; perhaps when a user leaves the device on a table. By filtering out these events the remainder of the accelerometer stages are only exercised when there is actually data to be processed. Calibration orients accelerometer values into gravitational units using parameters learned by applying the Mizell technique [55]. For the microphone, the front-end of shared stages are only comprised of two processes. The design for these two processes are adopted from [51]. First, an admission control separates silence from actual sound – a threshold-based decision is made based on the Root Mean Square (RMS) [64]. Second, for those microphone samples that contain non-silence, a coarse-grain category classifier is used to recognize frames containing voicing, music and other.

MCU Utilization. The proper use of the additional low-power processors is critical to maintaining acceptable levels of energy consumption. The MCUs excel at continuous sensor sampling at a low power cost and we take advantage of this by allocating to these units key filter-type computations conforming with the memory and runtime constraints. The daughter-board MCU samples the accelerometer without interruption at 50 Hz, whereas the Quark monitors the microphone continuously at 8KHz. The processing units liberate the Atom CPU from performing sensor sampling allowing it to go into a low-power standby state when the MCUs can handle the sensor events from the stream of incoming data.

The daughter-board MCU performs the full physical activity and transportation mode pipelines without assistance from the Atom CPU. The MCU needs to periodically communicate its accumulated inferences with the Atom CPU which is responsible for transferring the labeled data to the knowledge base. There is a communication overhead involving the transition of the CPU from sleep to high-power active state. To minimize the interactions between the two processing units we set the synchronization period to 20 minutes which is when the main CPU needs to be woken up to merge the inferences in the database.

Strategically, the simple silence filter is performed on the Quark so that whenever the environment is silent the energy hungry CPU is kept at bay. The performance limitations of the pre-release Quark MCU preclude us from deploying more complex pipelines such as speech detection; yet, the silence filter proves immensely useful given that a large proportion of the time users are in a silent context, e.g. a night's sleep. In the presence of noise, i.e. a non-silent frame is detected, the Atom CPU is approached with the next 1.28 secs of raw microphone data which is when the second-order speech/music/ambiance filter is applied.

6. IMPLEMENTATION

We conclude our description of ZOE by discussing implementation specific details.

Source and Binaries. The source tree of ZOE spans ≈18,000 lines-of-code written in C++ and Python – excluding large trunks of source code imported and modified for the components within the Dialog Subsystem (primarily for Speech Recognition) or for any model training toolkits utilized. The majority of the implementation runs on the Atom CPU of the Edison. Three separate cross-compilers are used – two for each respective MCU which build complete binary images and include MCU-specific OSs; with the last being the Atom where software is more conventionally compiled and installed (within a Linux-like OS, Yocto [15]). Complete binaries for each MCU are 48KB (daughter-board MCU) and 122KB (Quark MCU), respectively. In terms of workflow, the Atom CPU acts as the primary orchestrator within ZOE, but will sleep periodically (governed by the Performance Controller – §5) only to be woken by interrupts from either MCUs based on sampled data.

Sensor Algorithms. For many of the more standard algorithms such as GMM classification, Markov Models (MMs), Fast Fourier Transforms (FFTs) and a few common feature extractors, we port implementations from the HTK Speech Recognition Toolkit [4]. Others such as Decision Trees and Adaboost are ported from the Weka [31] toolkit. Lastly, for more domain specific features (MFCC, Teo-Auto-CB, WiFi Place signatures etc.) we have custom implementations based on algorithm descriptions. Off-line training steps utilize previously mentioned toolkits (HTK, Weka), along with the scikit-learn Python library [13].

User Knowledge Store and Front-end. Primarily to service dialog queries – but also to maximize the information readily available from inferences – ZOE maintains a User Knowledge Store. Instead of only storing the raw timestamped output of inferences, the front-end of the knowledge store generates information that is implied by individual or combinations of inferences; often, a single inference can generate dozens of new knowledge store records. Examples include estimating the calories burned during various forms of activities or the carbon footprint of transportation activity – both of which can be coarsely estimated based on commonly used reference material that maps calories to the time spent on certain activities [17], with similar references available for pollution and transport choices. Simple estimators are also built into the Knowledge Store for predicting, for example, the duration of events like commuting home at certain times of day. Finally, the Knowledge Store is encoded with simple common sense information such as: How many calories should a user burn everyday? This information is sourced from reference material.

Currently, effort has not been put into using the best predictors or estimators. It is simply a framework to collect and organize information. As a result, the Dialog Reasoning is made easier with essentially every dialog template being linked to code that understands how to generate a query to the Knowledge Store. Queries are parameterized by keywords spoken by the user. For example, when a user asks: "How many conversations did I have today?". The word "today" is a keyword that defines the timespan. Dialog Reasoning understands a limited number of such keywords. This component is implemented currently as a local Sqlite database with the front-end written in Python.

Dialog System. Our dialog system is a composite of code adapted from three open-source projects. First, the inference runtime of KWS that resides on ZOE hardware is a C++ DNN library implementation including many of the techniques described in [21]. To train this DNN we use Kaldi [59] and build the necessary glue to format the models generated. Second, we make extensive use of Pocketsphinx [36] for both the training of acoustic and language models, in addition to inference-time execution. Any necessary optimizations to speed up execution, were performed to the standard open source code-base. Third, to realize our TTS the implementation of eSpeak [3] was adopted and modified to support the audio driver of the Edison. Finally, template generation and links to the User Knowledge Base were built with Python as required.

7. EVALUATION

In this section, we provide key insights with respect to the system energy profile, the sensing algorithm accuracy and the dialog subsystem performance characteristics. The main findings can be summarized as follows:

- ZOE lasts 30 hours with a typical sensing/dialog load. A key enabler for this is the 3-tier design which offers a more than 30% relative improvement in the battery lifetime to alternatives.
- The dialog subsystem is 87% accurate and is able to respond within a second of finishing a 4-second long voice command.
- The oldest inferences a user may get are at most 20 minutes old which equals the accelerometer personal sensing synchronization period between the MCU and the Atom CPU. The sound-related inferences performed by the sensing algorithms are obtained largely in real-time on the Atom CPU.

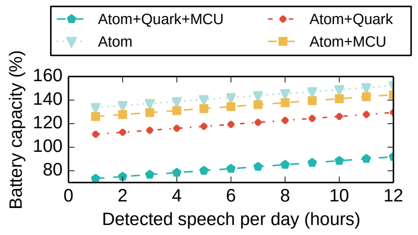


Figure 6: Percentage of the battery needed to handle the sensor processing of a daily workload (8h of silence, 1h to 12h of speech, 20 questions per day). Values higher than 100% indicate the battery needs to be recharged at least one more time during the day to sustain the workload.

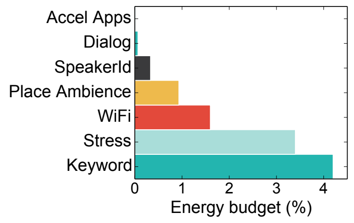


Figure 7: Breakdown of the energy consumed by the various system component computations for the workload scenario of 4.5h of speech, 11.5h of ambient context, 15min WiFi duty cycle and 20 dialog commands per day. The reported energy budget reflects the additional energy of the raw processing on top of what the Atom CPU consumes in an active state.

7.1 Energy Consumption

Here we detail the energy consumed by ZOE assuming a variety of workload conditions and different mobile device costs. The most notable results are 1) *our design is practical* as it allows the user to wear the device for well over a day with a single battery charge; and 2) compared to other off-the-shelf mobile devices, a *dedicated wearable* proves to be the only solution to have a reasonable energy drain and sustain common sensing workloads for at least a day.

System Lifetime. In this part of the analysis we show the total system energy consumption and the benefits of incorporating additional sensor processing units (Intel Quark and an MCU). We compare our heterogeneous design (Atom CPU + Quark co-processor + MCU) against three baseline models that exclude either the Quark, the MCU or both of the additional processing chips. In this experiment we generate synthetic sensing workloads by varying the amount of encountered speech. This is a prime determiner of the sensing workload because the heavier sensor pipelines are triggered by the presence of human voice: dialog subsystem interactions, stress detection, as well as device owner recognition and conversation analysis. We fix the time users are in a silent environment to 8 hours which roughly amounts to a night's sleep. We also assume 20 questions per day on average submitted to the dialog system as life logging inquiries. The battery used for the reported measurements is a typical 1500mAh Lithium-ion model with 3.7 volts.

In Figure 6 we plot the percentage of the battery capacity required by ZOE to process a day worth of sensing as a function of the number of hours users spend in conversations. The first critical observation is that our system design is the only among the alternatives that is able to make the battery last at least one day with a single charge. With 4.5 hours of speech, which is found to be an average amount of time users spend in conversations throughout the day [45], the system needs 78% of the battery capacity to handle a 24-hour processing load. This leaves us with *a system lifetime of* 30 *hours*. In contrast, the 2-tier solutions as well as the bare bones Atom-only deployment require 109% or more of the battery to handle the same workload which means that the Edison wearable needs to be recharged before the day expires. Overall, *the* 3*-tier design offers more than* 30% *improvement in the system longevity* and this is largely attributed to the fact that part of the sensor processing is offloaded to the dedicated low-power co-processors.

When either of the two additional chips (Quark or MCU) is left out, the sensor sampling that would otherwise be performed there will need to be delegated to the general-purpose Atom CPU. This prevents the CPU from going to low-power standby mode and instead keeps the processor active with an average current draw of 63mA. When the MCU is removed, the accelerometer sampling keeps the CPU busy, whereas when the Quark is not used, the microphone sampling maintains the CPU awake. Both of the 2-tier designs do not offer significant energy improvement over the Atom-only deployment because 1) their limited memory and performance capabilities allow them to run only simple pipelines such as silence filtering that on the Atom run in less than 2ms, and 2) in both cases the Atom CPU is burdened with sampling one or both of the sensors where the energy hog is maintaining a high-power active state. As shown in Figure 7, the energy budget for the additional processing of the system components remains cumulatively below 12%, meaning that the majority of the system energy is spent in keeping the CPU active to do the processing in the non-silent periods.

Effect of False Atom Wake-Ups. As discussed in the previous paragraph, keeping the Atom CPU awake is a major source of energy consumption which is why it is vital that the times the CPU is unnecessarily woken up to do processing are minimized. Such false wake-ups may occur when the silence filter running on the Quark co-processor errs and flags a silent frame as non-silent, i.e. the filter introduces a false negative with respect to silence. To evaluate the likelihood of such false wake-ups given our implementation of the RMS threshold-based silence filter, we assemble a dataset with 10 minutes of noise, including speech, and 10 minutes of silence. The silent samples are collected from smartphone recordings, whereas the noise samples reuse part of the ambient context dataset [68].

In Table 5 we list the confusion matrix for the silence filter applied on the prepared 20-minute dataset. The main result is that the false wake-up rate remains relatively low at 2.76% which ensures that the energy overhead of unnecessarily waking up the Atom is kept acceptably low. Depending on the chosen silence filter threshold, the false wake-up rate can be further reduced at the expense of decreasing the 92.74% accuracy with which noise is detected. We believe the current set-up provides a reasonable trade-off between accuracy and false wake-up energy overhead.

Actual/Predicted	**Silence**	**Noise**
Silence	97.23 %	2.76%
Noise	7.26%	92.74 %

Table 5: Confusion matrix for the silence filter running on the Intel Quark.

Projected Lifetime for Other Mobile Devices. In this next subsection we investigate the following research question: *can we deploy ZOE on another mobile device such as a smartphone and achieve similar or better battery life*? We perform back-of-the-envelope energy analysis for several other mobile devices including a smartphone and Google Glass. The smartphone and Edison energy measurements are performed with a Monsoon Power Monitor attached to the corresponding device. For the smartphone case, we port the sensing algorithms through the Android Native Development Kit to a Qualcomm Snapdragon 800 Mobile Development Platform (MDP) [12], a dedicated device for testing and debugging Android applications. It features the popular Snapdragon 800 SoC available in many high-end smartphone models (e.g., Google Nexus 5, Samsung Galaxy Note 3). We used energy estimates for Google Glass provided by [49] and without loss of generality assumed the pipeline processing runtimes to be similar to what is measured on the smartphone MDP. Given this set-up we report the projected energy budget required by the system to sustain a daily workload with 4.5 hours of speech in Table 6.

Out of the mobile devices, the custom-built Intel Edison wearable with the larger 1500mAh battery is the only one to be able to not fully drain the battery in a day. The second best alternative, a smartphone with a sizable 3200mAh, battery needs 151% of the capacity to process the sensing workload presented in the previous subsection. The massive energy drain can again be attributed to keeping the CPU active during the continuous sensor sampling. This energy profile is unacceptable given that the smartphone can and will be used for other tasks throughout the day such as phone calls, text messages, camera photos, web browsing, games and social networking applications. Last on the energy efficiency list is Google Glass, which according to our coarse projected measurements will need multiple recharges in a day (more than 7) to meet ZOE's sensing demands. This can be explained by both the relatively smaller battery and higher power consumption needed by a powerful CPU that supports augmented reality applications. To summarize, *among the evaluated options, the only feasible solution for supporting the application scenarios induced by ZOE proves to be a dedicated wearable with the lower power characteristics of an Intel Edison board*.

Mobile Device	Battery Specs	Energy Budget (%)
Intel Edison	1500mAh	78%
Smartphone (MDP)	3200mAh	151%
Intel Edison	400mAh	295%
Smartphone (MDP)	1500mAh	321%
Google Glass	2.1Wh (568mAh)	744%

Table 6: Energy expenditure required by ZOE to meet a daily workload if it was running on alternative devices. Table entries are sorted by energy budget. Measurements for the MDP and Edison are made with a Monsoon Power Monitor, and power estimates for Google Glass are from [49].

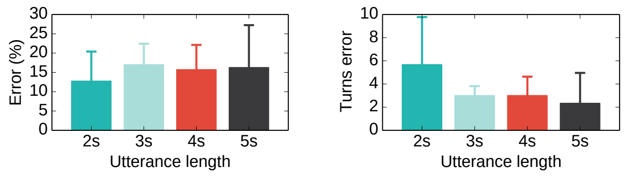
7.2 Sensing Algorithm Accuracy

In this subsection we discuss the accuracy of the adopted sensing pipelines that underlie the activity inferences performed by ZOE. The basic components of the personal and place sensing pipelines are largely unchanged from their original versions and here we briefly summarize their accuracy on the datasets we have access to while also detailing any differences. In addition, we provide preliminary results about the effectiveness with which the social sensing subsystem infers conversation patterns: device owner identification, dominance and turn taking.

Most of the algorithms accuracies are in line with previously published results: these include physical activity recognition [52], motorized vs. non-motorized vehicle detection [32], stress detection with a universal model [50], place recognition [40] and place ambient context analysis [68]. For correctness, we report results on the listed datasets but detailed evaluation of how the algorithms fare in more challenging conditions can be found in the original publications. One of the differences we find is the speaker identification [61] accuracy which on our dataset reaches 95% (94%) when 5 (3) seconds of audio data is used for the speech feature extraction. This is likely attributed to two factors: 1) we use up to 9 minutes of training data for each user (and 1 minute is held out for testing); and 2) the recordings are made in a clean environment.

Sensing Pipeline	Sensor	Dataset	Accuracy
Physical Activity	Accel	10 mins of activities	82%
Transportation Mode	Accel	150h transportation data [32]	85%
Stress Periods	Mic	Emotional Prosody [48]	71%
Place Recognition	WiFi	LifeMap [22]	90%
Place Context	Mic	local business ambience [68]	74%
Speaker Id (5s)	Mic	23 people, 230 mins of speech	95%
Speaker Id (3s)	Mic	23 people, 230 mins of speech	94%

Table 7: Accuracy of the pipelines.



(a) Dominance (b) Turn taking

Figure 8: Conversation pattern errors as a function of the utterance length used to identify the wearable owner.

Social Sensing Accuracy. We recall that we adopt the speakerID pipeline as a tool to monitor conversation patterns. Examples are inferring the percentage of time a person is talking (dominance levels) or the periods and durations of their turns. Here we shed light on how effective speaker recognition is in deriving such inferences. We perform a preliminary evaluation on a dataset of several conversations each lasting between 4 and 5 minutes. In all of the conversations one of the speakers is designated as a device owner with a personalized speaker identification model. The rest of the participants are unknown and their speech data is not used in the training of the sink GMMs representing other people. Given this setup, we replay the conversations and perform owner identification once every few seconds.

We repeat the above mentioned experiment for utterance lengths ranging from 2 to 5 seconds inclusive and in Figure 8 report the observed errors when speaker identification is adopted to estimate two conversation patterns: the speaker dominance levels and turn taking. An interesting result is that *using shorter periods of audio data for user recognition appears to reduce the dominance errors but significantly increases the turn taking uncertainty*. This phenomenon can be explained by the fact that shorter audio segments increase the temporal precision of understanding when someone is talking even when some errors are introduced, hence the higher dominance accuracy. However, those same errors tend to largely overestimate the number of turns.

Overall, *the algorithm for detecting dominance is off by 12% to 16% depending on the utterance length which is sufficient for coarse estimations of total speaking time*. On the other hand, the mean turn taking errors of 2 to 3 missed or additionally detected turns per conversation are comparatively large given that the average number of turns the target speaker takes is 7. That speaker identification algorithm on 3 to 5 second long utterances does not prevent us from roughly inferring conversation dynamics but may not be suitable for finer grained turn monitoring as has also been highlighted by the authors of SocioPhone [46]. Their proposed multi-device volume topography analysis for the same inference task has limited applicability in our setting, however, because the Edison wearable is supposed to be a self-contained personal sensing assistant and in general we cannot rely on other people owning the same device or having the same set of applications installed on their smartphone/mobile device.

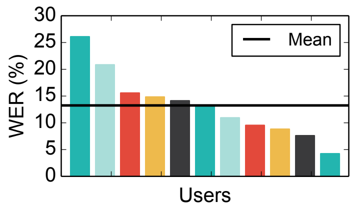


Figure 9: Word error rate (WER) of the 11 users on whom the speech recognition subsystem was tested.

7.3 Dialog Accuracy and Responsiveness

In this subsection we elaborate on the accuracy with which the dialog system interprets the commands issued by users as well as the level of system responsiveness to those commands. The latter is measured along two dimensions which we evaluate separately: the time required to decode a voice command and the staleness of the inferences when presenting aggregate results to users.

Dialog Accuracy. We begin by discussing the word-level accuracy of the custom-built dialog language model as a preliminary indication to how well ZOE interprets commands. To evaluate this we performed a controlled experiment on 11 speakers, 3 of which were female and 7 of which were non-native English speakers. The participants were required to read aloud a set of 25 command sentences spanning the type of questions supported by ZOE, e.g. *"What fraction of my time do I spend listening to music?"*. We recorded the audio files in an office environment with a sampling rate of 8kHz and fed them as input to the CMU Pocket Sphinx batch analysis tool. The word error rate (WER) is reported in Figure 9. Overall, we observe a mean WER of 13.24% across users which corresponds to an accuracy approaching 87%. We note that although the reported numbers are showing the word-level error rate, the majority of confused words are non-keywords which suggests the command-level accuracy is not critically impacted. A key component enabling these reasonably high figures is the custom-built language model that targets a domain-specific restricted vocabulary of a few hundred words related to daily activities. We find that the accuracy may drop by as much as 30% or 40% when a generic language model with a large dictionary is used.

Language Model	Dictionary Size	Real-time Factor
ZOE's custom	300 words	0.23x
CMU Sphinx HUB4	6000 words	1.98x
CMU Sphinx HUB4	133000 words	2.15x

Table 8: Dialog system runtime responsiveness as a function of the language model and dictionary size. The real-time factor shows how much time on average it takes ZOE to decode an utterance relative to its length.

Dialog Responsiveness. A major factor determining the real-time responsiveness of ZOE is the time it takes the speech recognition subsystem to decode the words in the voice commands. In Table 8 we show the real-time delay of the decoding process running on the Edison's Atom CPU as a function of the language model and number of words in the vocabulary. The key result is that *using a custom-built model with a domain-specific dictionary is necessary to reach the real-time performance of the system.* The real-time factor of 0.23x means that an utterance of length 4 seconds, for example, is transcribed in 0.92s so that a result can be delivered to the user within a second of completing the voice inquiry. In contrast, using larger general-purpose dictionaries significantly increases the transcription time so that with 6K words, for instance, the same 4-second command will be decoded in almost 8 seconds after issuing it, hardly a practical outcome.

Sensing Pipeline	Runtime (s)	Inference Window (s)
Physical Activity	0.002	4.000
Transportation Mode	0.002	4.000
Stress Periods	0.353	1.280
Place Context	0.025	1.280
Speaker Identification	0.105	5.000
Keyword Spotting	0.236	1.000

Table 9: Runtime of the sensing pipelines as measured on the Intel Atom 500 MHz CPU and the MCU for the accelerometer ones. The inference window shows the amount of time raw data is accumulated before the pipeline is run on the sensor samples.

Freshness of Inferences. A critical aspect shaping the life logging experience is what the amount of time is between the arrival of a sensor event and obtaining an inference. In general, sensor events trigger data processing requests which are queued in a FIFO manner and the actual time of the processing may not be immediate. Depending on how busy the system is the accumulated requests may be temporarily ignored. In our case, this scenario arises when the user initiates a voice command and the Keyword Spotting application detects the system name. ZOE servers the user with priority in which case sensor tasks accumulate in the queue. However, as shown in Table 9, *all of the sensing pipelines run with sub-second latency for a small fraction (up to 25%) of the time sensor data is gathered.* In effect, even in the busiest scenario for the system when speech is detected, the keyword spotting, speaker identification and stress detection can run all together and in real time. This property allows *inferences on the data to be obtained immediately in the absence of dialog commands.* The worst case is when the user initiates a series of commands, one after another without pauses in between, which is when the sensor data will be queued. Since long command sequences are highly unlikely, the sub real-time nature of the pipelines allows for the system to catch up with the sound inferences within a minute of successively issuing a few commands. Finally, the freshness of the WiFi localization and accelerometer inferences are determined by the WiFi scan duty cycle (15 min) and the update period when the MCU synchronizes accelerometer inferences with the Atom (once every 20 min).

8. DISCUSSION AND LIMITATIONS

We briefly survey a variety of issues related to the design and evaluation of ZOE.

Absence of End-to-End User Study. ZOE currently uses features of the Edison SoC (such as, MCU access) not publicly available. Because of this we had access to only two pre-release versions of this SoC, this precluded any meaningful user study. We plan to build a number of ZOE devices in the coming months as soon as this situation changes. Instead, the focus and contribution of this work is the systematic study of the hardware capabilities under the workload ZOE presents, along with series of proposed techniques that allows this workload to achieve acceptable levels of energy efficiency and responsiveness.

Of particular interest in future user studies will be user reaction to wearable-based dialog as a means of interaction. Dialog systems are generally gaining popularity and user acceptance in more powerful form factors, such as smartphones; along with always-on home targeted appliances like Amazon's Echo [1]. But clearly there will be times such interaction will not be appropriate, such as in public or within social groups (e.g., a meeting). While certainly the responses of ZOE may not be suitable for others to hear, users may also be uncomfortable with what the questions they ask indicate about themselves and their interests.

Is Custom Hardware Warranted? It is likely that a ZOE-like software system could have been designed for a number of mobile platforms. However, overall it would have failed to meet key technical challenges described in §2.1. For example, as we discussed in §7.1 the energy efficiency on alternative platforms like a smartwatch is significantly worse than ZOE's due to lack of low-power MCUs. More broadly, the mobile industry is shifting to increase use of specialized computation; for example, Apple's A7 motion co-processor [6] for interpreting primarily inertial sensors, and more complex speech [14] and vision algorithms appearing in Integrated Circuits providing leaps in performance that software advances alone struggle to match.

Battery Lifetime. Currently, ZOE has a lifetime comparable to the more sensor-rich wearables already on the market (such as the Microsoft Band). It is also likely that the energy efficiency of ZOE will improve dramatically as soon as two key available, but still inaccessible, Edison features are opened. First, the Edison CPU can operate at a range of higher and lower clock speeds, yet currently this is fixed at 500 MHz. This speed is unnecessary for some ZOE operations (and wasteful) while others would benefit from a temporary increase; because the CPU consumes most of the device energy, clock speed control will likely offer significant gains. Second, the Edison-located MCU is based on a processor that is much more capable than the tasks it currently performs. Due to limitations in early Edison compiler support, we are only able to perform a few basic inference stages on it, a situation that will be changed subsequently.

Size and Form-factor. As shown in Table 2, ZOE achieves an impressive ratio of inference capabilities to size/weight relative to comparison devices. We also expect to be able to decrease the footprint of ZOE by half in the next version by relatively simple layout optimizations to the daughter-board, allowing it to match the dimensions of the Edison. The more challenging issue in doing this will be to cope with a similar reduction in battery capacity that such changes would entail.

9. RELATED WORK

ZOE touches upon a wide range of topics, here we discuss the most salient prior work in each area.

Smartphone Sensing and Activity Recognition. The design of ZOE contributes to the area of human activity inference by investigating how a diverse set of existing techniques can be combined into a single integrated sensing module; in turn this module can run on a platform and form-factor where few have been previously seen individually (and certainly not being all supported simultaneously). Along with the recent explosion of techniques targeting smartphones (e.g., [58, 46, 66]), there is a long history of platform agnostic algorithms from activity recognition (e.g., [19]). However, even though many share building block components (features, related classes) few studies exist that investigate their usage together (e.g., [39]). Towards wearables with a more complete understanding of the life of users, ZOE incorporates more than 10 distinct inferences from three different sensors that have previously been often studied in isolation.

The few mobile systems that seek a more complete monitoring of user behavior and environment often focus on health (e.g., [25, 57]). Other systems that have targeted a diverse set of inferences have focused on a single sensor – the microphone being a popular one [51, 28]. DSP.Ear [28], for example, also supports sensing algorithms that cover a variety of activities but lacks the place- and social- inferences of ZOE, and ignores modalities other than audio.

Wearables and Personal Sensing Hardware. As surveyed in §2.3, ZOE distinguishes itself, particularly from commercial systems, largely because of the depth and range of its user inferences and exploration of features like dialog systems, heterogeneous computation and constraints like operating without a cloud. Wearable systems that seek the type of breadth as ZOE are almost always based on vision or audio data because of the richness of the two modalities. ZOE relies on multiple sensors but makes strong use of the microphone which sets itself apart in terms of sensing algorithms with wearables like [30][53][34]. Examples of microphone-based wearables include [62] but this system has a specific narrow focus on internal body sounds captured with a specially designed microphone. Instead, ZOE has a number of commonalities with the MSP [23] that was also a heavy user of the microphone. However, as described in §2.3 while ZOE monitors general user activities, as the MSP does, it also provides: detailed measurements of user state and social interactions; in addition to providing a dialog system through which to offer the user this information.

Heterogeneity and Mobile Resource Bottlenecks. Energy has been a considerable focus of a variety of mobile sensing systems [44, 43, 38]. Towards such challenges, low-power processors and multiple-tiers of computational units are commonly used approaches [18, 65]. ZOE adds to this knowledge by exploring the use of dual-MCUs to allow a comparatively very heavy sensing and interaction workload to remain feasible, especially in terms of energy. Architecturally similar approaches include [58, 60] that combine high- and low-power processors for energy efficient sensing. Even commercially, many smartphones today use DSPs and other processors to continuously sense and wait for one specific keyword to be spoken (a keyword) before initiating a dialog. Although the core of the techniques we use have appeared in such systems, it is the first time they have been applied to many of the more complex sensing algorithms (e.g., seen in our social sensing monitoring) and especially when they are integrated into a single system.

10. CONCLUSION

In this paper, we have presented the design, implementation and evaluation of ZOE – a prototype wearable that leverages the type of heterogeneous high-performance computation increasingly available to this class of mobile device. Through a mixture of hardware- and software-based approaches, ZOE offers a first-of-its-kind user experience by enabling the continuous sensing of a range of deep inferences along with a near real-time dialog system through which users can access captured information. All of this is provided without cloud assistance, which provides significant privacy assurances to the user by removing the need for third-party remote computation. We hope ZOE acts as an important reference example for future investigations into the design of wearables, and also provokes further study of: heterogeneous computation in wearables; in-situ dialog-based access to information for users; and, the limits of cloud-free wearable functionality.

Acknowledgments

This work was supported by Microsoft Research through its PhD Scholarship Program. We would also like to thank the anonymous reviewers and our shepherd, Jeremy Gummeson, for helping us improve the paper.

11. REFERENCES

[1] Amazon Echo. http://www.amazon.com/oc/echo.

[2] Android Wear hardware review: Sometimes promising, often frustrating. http://tinyurl.com/m9zb3yz.

[3] eSpeak. http://espeak.sourceforge.net.

[4] HTK Speech Recognition Toolkit. http://htk.eng.cam.ac.uk/.

[5] Intel Edison. http://www.intel.com/content/www/us/en/do-it-yourself/edison.html.

[6] iPhone 5s M7 Motion Coprocessor. https://www.apple.com/iphone-5s/specs/.

[7] Jawbone Up 3. http://jawbone.com/store/buy/up3.

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

[9] Microsoft Band. http://www.microsoft.com/Microsoft-Band/.

[10] Motorola MotoActv. https://www.motoactv.com.

[11] Qualcomm Snapdragon 400. https://www.qualcomm.com/products/snapdragon/processors/400.

[12] Qualcomm Snapdragon 800 MDP. https://developer.qualcomm.com/mobile-development/development-devices/snapdragon-mobile-development-platform-mdp.

[13] Scikit-Learn Python Library. http://scikit-learn.org/stable/.

[14] Sensory. http://www.sensory.com/products/integrated-circuits/.

[15] Yocto Project. https://www.yoctoproject.org/.

[16] Your Samsung SmartTV Is Spying on You, Basically. http://www.thedailybeast.com/articles/2015/02/05/your-samsung-smarttv-is-spying-on-you-basically.html.

[17] B. E. Ainsworth, W. L. Haskell, M. C. Whitt, M. L. Irwin, A. M. Swartz, S. J. Strath, W. L. O'Brien, D. R. Bassett, K. H. Schmitz, P. O. Emplaincourt, D. R. Jacobs, and A. S. Leon. Compendium of Physical Activities: an update of activity codes and MET intensities. *Medicine & Science in Sports & Exercise*, 32(9):498–516, Sept. 2000.

[18] R. K. Balan, D. Gergle, M. Satyanarayanan, and J. Herbsleb. Simplifying cyber foraging for mobile devices. In *Proceedings of the 5th International Conference on Mobile Systems, Applications and Services*, MobiSys '07, pages 272–285, New York, NY, USA, 2007. ACM.

[19] L. Bao and S. S. Intille. Activity recognition from user-annotated acceleration data. In *Pervasive computing*, pages 1–17. Springer, 2004.

[20] A. Biem, E. Mcdermott, and S. Katagiri. A discriminative filter bank model for speech recognition. In *Proceedings of the IEEE, ICASSP-96*, pages 545–548, 1995.

[21] G. Chen, C. Parada, and G. Heigold. Small-footprint keyword spotting using deep neural networks. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, ICASSP'14, 2014.

[22] J. Chon and H. Cha. Lifemap: A smartphone-based context provider for location-based services. *IEEE Pervasive Computing*, 10(2):58–67, Apr. 2011.

[23] T. Choudhury, G. Borriello, S. Consolvo, D. Haehnel, B. Harrison, B. Hemingway, J. Hightower, P. P. Klasnja, K. Koscher, A. LaMarca, J. A. Landay, L. LeGrand, J. Lester, A. Rahimi, A. Rea, and D. Wyatt. The mobile sensing platform: An embedded activity recognition system. *IEEE Pervasive Computing*, 7(2):32–41, Apr. 2008.

[24] L. Deng and D. Yu. Deep learning: Methods and applications. Technical Report MSR-TR-2014-21, January 2014.

[25] T. Denning, A. Andrew, R. Chaudhri, C. Hartung, J. Lester, G. Borriello, and G. Duncan. Balance: towards a usable pervasive wellness application with accurate activity inference. In *Proceedings of the 10th workshop on Mobile Computing Systems and Applications*, page 5. ACM, 2009.

[26] T. Dutoit. *An introduction to text-to-speech synthesis*, volume 3. Springer, 1997.

[27] J. Edlund, M. Heldner, and J. Hirschberg. Pause and gap length in face-to-face interaction. In *INTERSPEECH 2009, 10th Annual Conference of the International Speech Communication Association, Brighton, United Kingdom, September 6-10, 2009*, pages 2779–2782, 2009.

[28] P. Georgiev, N. D. Lane, K. K. Rachuri, and C. Mascolo. Dsp.ear: Leveraging co-processor support for continuous audio sensing on smartphones. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, SenSys '14, pages 295–309, New York, NY, USA, 2014. ACM.

[29] K. Gupta and J. D. Owens. Three-layer optimizations for fast gmm computations on gpu-like parallel processors. In *ASRU*, pages 146–151. IEEE, 2009.

[30] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan. Towards wearable cognitive assistance. In *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '14, pages 68–81, New York, NY, USA, 2014. ACM.

[31] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, Nov. 2009.

[32] S. Hemminki, P. Nurmi, and S. Tarkoma. Accelerometer-based transportation mode detection on smartphones. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, SenSys '13, pages 13:1–13:14, New York, NY, USA, 2013. ACM.

[33] J. Hightower, S. Consolvo, A. LaMarca, I. Smith, and J. Hughes. Learning and recognizing the places we go. In *Proceedings of the 7th International Conference on Ubiquitous Computing*, UbiComp'05, pages 159–176, Berlin, Heidelberg, 2005. Springer-Verlag.

[34] S. Hodges, L. Williams, E. Berry, S. Izadi, J. Srinivasan, A. Butler, G. Smyth, N. Kapur, and K. Wood. Sensecam: A retrospective memory aid. In *UbiComp 2006*, pages 177–193. Springer, 2006.

[35] W. Huang, Y.-S. Kuo, P. Pannuto, and P. Dutta. Opo: A wearable sensor for capturing high-fidelity face-to-face interactions. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, SenSys '14, pages 61–75, New York, NY, USA, 2014. ACM.

[36] D. Huggins-Daines, M. Kumar, A. Chan, A. W. Black, M. Ravishankar, and A. I. Rudnicky. Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 1, pages I–I. IEEE, 2006.

[37] D. Gatica-Perez, and D. K. Heylen. Multi-modal analysis of small-group conversational dynamics. *Multimodal Signal Processing*, 2012.

[38] N. D. Lane, Y. Chon, L. Zhou, Y. Zhang, F. Li, D. Kim, G. Ding, F. Zhao, and H. Cha. Piggyback CrowdSensing (PCS): energy efficient crowdsourcing of mobile sensor data by exploiting smartphone app opportunities. In *Proceedings of the 11th ACM Conference on Embedded Network Sensor Systems*, SenSys '13, pages 1–14, New York, NY, USA, 2013. ACM.

[39] Y. Ju, Y. Lee, J. Yu, C. Min, I. Shin, and J. Song. Symphoney: A coordinated sensing flow execution engine for concurrent mobile sensing applications. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, SenSys '12, pages 211–224, New York, NY, USA, 2012. ACM.

[40] D. H. Kim, Y. Kim, D. Estrin, and M. B. Srivastava. Sensloc: Sensing everyday places and paths using less energy. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, pages 43–56, New York, NY, USA, 2010. ACM.

[41] J. Krumm and E. Horvitz. Locadio: Inferring motion and location from wi-fi signal strengths. In *MobiQuitous*, pages 4–13. IEEE Computer Society, 2004.

[42] J. Krumm and D. Rouhana. Placer: Semantic place labels from diary data. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, pages 163–172, New York, NY, USA, 2013. ACM.

[43] K. Lin, A. Kansal, D. Lymberopoulos, and F. Zhao. Energy-accuracy trade-off for continuous mobile device location. In *Proceeding of the 8th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, pages 285–298 ACM, 2010.

[44] S. Nath. ACE: exploiting correlation for energy-efficient and continuous context sensing. In *Proceeding of the 10th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '12, pages 29–42 ACM, 2012.

[45] Y. Lee, C. Min, C. Hwang, J. L. 0001, I. Hwang, Y. Ju, C. Yoo, M. Moon, U. Lee, and J. Song. Sociophone: everyday face-to-face interaction monitoring platform using multi-phone sensor fusion. In *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '13, pages 499–500. ACM, 2013.

[46] Y. Lee, C. Min, C. Hwang, J. Lee, I. Hwang, Y. Ju, C. Yoo, M. Moon, U. Lee, and J. Song. Sociophone: Everyday face-to-face interaction monitoring platform using multi-phone sensor fusion. In *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '13, pages 375–388, New York, NY, USA, 2013. ACM.

[47] J. Lester, T. Choudhury, N. Kern, G. Borriello, and B. Hannaford. A hybrid discriminative/generative approach for modeling human activities. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, IJCAI'05, pages 766–772, San Francisco, CA, USA, 2005. Morgan Kaufmann Publishers Inc.

[48] M. Liberman, K. Davis, M. Grossman, N. Martey, and J. Bell. Emotional prosody speech and transcripts. 2002.

[49] R. LiKamWa, Z. Wang, A. Carroll, F. X. Lin, and L. Zhong. Draining our glass: An energy and heat characterization of google glass. In *Proceedings of 5th Asia-Pacific Workshop on Systems*, APSys '14, pages 10:1–10:7, New York, NY, USA, 2014. ACM.

[50] H. Lu, D. Frauendorfer, M. Rabbi, M. S. Mast, G. T. Chittaranjan, A. T. Campbell, D. Gatica-Perez, and T. Choudhury. Stresssense: Detecting stress in unconstrained acoustic environments using smartphones. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, pages 351–360, New York, NY, USA, 2012. ACM.

[51] H. Lu, W. Pan, N. D. Lane, T. Choudhury, and A. T. Campbell. 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, pages 165–178, New York, NY, USA, 2009. ACM.

[52] H. Lu, J. Yang, Z. Liu, N. D. Lane, T. Choudhury, and A. T. Campbell. The jigsaw continuous sensing engine for mobile phone applications. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, pages 71–84, New York, NY, USA, 2010. ACM.

[53] A. Mayberry, P. Hu, B. Marlin, C. Salthouse, and D. Ganesan. ishadow: Design of a wearable, real-time mobile gaze tracker. In *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '14, pages 82–94, New York, NY, USA, 2014. ACM.

[54] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell. Sensing meets mobile social networks: The design, implementation and evaluation of the cenceme application. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, SenSys '08, pages 337–350, New York, NY, USA, 2008. ACM.

[55] D. Mizell. Using gravity to estimate accelerometer orientation. In *Proceedings of the 7th IEEE International Symposium on Wearable Computers*, ISWC '03, pages 252–, Washington, DC, USA, 2003. IEEE Computer Society.

[56] M. Mun, S. Reddy, K. Shilton, N. Yau, J. Burke, D. Estrin, M. Hansen, E. Howard, R. West, and P. Boda. Peir, the personal environmental impact report, as a platform for participatory sensing systems research. In *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services*, MobiSys '09, pages 55–68, New York, NY, USA, 2009. ACM.

[57] Nicholas Lane, M. Lin, M. Rabi, X. Yang, A. Doryab, H. Lu, S. Ali, T. Choudhury, A. Campbell, and E. Berke. *Bewell: A smartphone application to monitor, model and promote wellbeing*. IEEE Press, 2011.

[58] S. Nirjon, R. Dickerson, J. Stankovic, G. Shen, and X. Jiang. smfcc: Exploiting sparseness in speech for fast acoustic feature extraction on mobile devices – a feasibility study. In *Proceedings of the 14th Workshop on Mobile Computing Systems and Applications*, HotMobile '13, pages 8:1–8:6, New York, NY, USA, 2013. ACM.

[59] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely. The kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011. IEEE Catalog No.: CFP11SRW-USB.

[60] B. Priyantha, D. Lymberopoulos, and J. Liu. Littlerock: Enabling energy-efficient continuous sensing on mobile phones. *IEEE Pervasive Computing*, 10(2):12–15, 2011.

[61] K. K. Rachuri, M. Musolesi, C. Mascolo, P. J. Rentfrow, C. Longworth, and A. Aucinas. Emotionsense: A mobile phones based adaptive platform for experimental social psychology research. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*, Ubicomp '10, pages 281–290, New York, NY, USA, 2010. ACM.

[62] T. Rahman, A. T. Adams, M. Zhang, E. Cherry, B. Zhou, H. Peng, and T. Choudhury. Bodybeat: A mobile system for sensing non-speech body sounds. In *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '14, pages 2–13, New York, NY, USA, 2014. ACM.

[63] S. Reddy, M. Mun, J. Burke, D. Estrin, M. Hansen, and M. Srivastava. Using mobile phones to determine transportation modes. *ACM Trans. Sen. Netw.*, 6(2):13:1–13:27, Mar. 2010.

[64] E. Scheirer and M. Slaney. Construction and evaluation of a robust multifeature speech/music discriminator. In *Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '97)-Volume 2 - Volume 2*, ICASSP '97, pages 1331–, Washington, DC, USA, 1997. IEEE Computer Society.

[65] J. Sorber, N. Banerjee, M. D. Corner, and S. Rollins. Turducken: Hierarchical power management for mobile devices. In *Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services*, MobiSys '05, pages 261–274, New York, NY, USA, 2005. ACM.

[66] W.-T. Tan, M. Baker, B. Lee, and R. Samadani. The sound of silence. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, SenSys '13, pages 19:1–19:14, New York, NY, USA, 2013. ACM.

[67] H. Wang, D. Lymberopoulos, and J. Liu. Local business ambience characterization through mobile audio sensing. In *23rd International World Wide Web Conference (WWW)*, 2014.

[68] H. Wang, D. Lymberopoulos, and J. Liu. Local business ambience characterization through mobile audio sensing. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 293–304, New York, NY, USA, 2014. ACM.

[69] R. Want, T. Pering, G. Danneels, M. Kumar, M. Sundar, and J. Light. The personal server: Changing the way we think about ubiquitous computing. In *Proceedings of the 4th International Conference on Ubiquitous Computing*, UbiComp '02, pages 194–209, London, UK, UK, 2002. Springer-Verlag.

[70] C. Xu, S. Li, G. Liu, Y. Zhang, E. Miluzzo, Y.-F. Chen, J. Li, and B. Firner. Crowd++: Unsupervised speaker count with smartphones. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, pages 43–52, New York, NY, USA, 2013. ACM.