Enabling Context-Awareness in Mobile Systems via Multi-Modal Sensing

by

Xuan Bao

Department of Computer Science
Duke University

Date: ____________________
Approved:

____
Romit Roy Choudhury, Supervisor

____
Bruce Maggs

____
Landon Cox

____
Alexander Varshavsky

Dissertation submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in the Department of Computer Science
in the Graduate School of Duke University
2013

Abstract

Enabling Context-Awareness in Mobile Systems via Multi-Modal Sensing

by

Xuan Bao

Department of Computer Science
Duke University

Date: ____________________
Approved:

____
Romit Roy Choudhury, Supervisor

____
Bruce Maggs

____
Landon Cox

____
Alexander Varshavsky

An abstract of a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science in the Graduate School of Duke University
2013

Abstract

The inclusion of rich sensors on modern smartphones has changed mobile phones from simple communication devices to powerful human-centric sensing platforms. Similar trends are influencing other personal gadgets such as the tablets, cameras, and wearable devices like the Google glass. Together, these sensors can provide a high-resolution view of the user's context, ranging from simple information like locations and activities, to high-level inferences about the users' intention, behavior, and social interactions. Understanding such context can help solving existing system-side challenges and eventually enable a new world of real-life applications.

In this thesis, we propose to learn users' context via multi-modal sensing. The intuition is that human behaviors leave footprints on different sensing dimensions - visual, acoustic, motion and even in cyber space. By collaboratively analyzing these footprints, the system can obtain valuable insights about the user. The analysis results can lead to a series of applications including capturing life-logging videos, creating automatic online content ratings and even enabling new ways for human-object interactions. Through these applications, we show that a wide spectrum of previously "invisible" behaviors can potentially be captured and revealed by tapping into the rich set of sensors embedded in modern commercial mobile devices.

To my mother, father and Bolin

Contents

List of Tables

List of Figures

Acknowledgements

I would like to take this opportunity to thank all the people that have helped me along this five-year journey of my Ph.D. I would like to first express my most sincere thanks to my advisor Romit Roy Choudhury. I came to Duke as an ECE master student with a vague idea of doing research. It was Romit that introduced me to the real world of research. I can still vividly remember my first meeting with him. He drew a chart on a sheet of paper explaining how wireless and mobile fit into different layers of system design and what skill sets a student needs to acquire to conduct research in each direction. He also introduced me to a set of interesting research papers to read. And even from the very beginning, he took the time to meet me regularly and gave me advice regarding selecting research topics and refining project ideas. During the five years, there are many such moments that I will never forget – the late night preparation of my first conference presentation, the long philosophical discussion about choosing future careers, and of course the countless nights of preparing paper submissions. Step by step, Romit has guided me through this wonderful journey of my Ph.D. I am very grateful for having such an amazing advisor and I thank him for his extensive patience, detailed guidance and endless support at so many levels.

Second, I would like to thank Duke Computer Science, and Electrical and Computer Engineering departments for providing all the generous support for my PhD study. I will especially thank my committee members Bruce Maggs and Landon Cox for their patience and support. I really appreciate that they take the time to participate in my defense in the early morning (6AM PST).

I would also like to thank my many mentors during my internships. Especially, I need to thank Alexander Varshavsky for his frank advice that helped me during my internship, my job search and my career choices.

I am also fortunate to have the opportunity to work with a group of excellent lab-mates, especially my senior peers – Souvik Sen, Justin Manweiler, and Ionut Constandache. From them, I have learned knowledge and skills both about research and beyond. Our friendship helped me survive all the ups and downs that naturally come with the long journey of PhD study.

There are many others without whom I would not be able to come this far. I need to thank my colleagues and co-authors who stand side by side with me during all the impossible deadlines. I need to thank my family and my wife for supporting me during my entire education. It is only with the help of all these great people that I am able to achieve what I have done and accomplish this thesis.

1 Introduction

The information explosion in the mobile era makes users' attention the most precious resource. Therefore, knowing the users' context becomes vital for distilling and presenting useful information to the user. Previous research work has extensively investigated inferring users' context from individual sensing dimensions. For example, the computer vision community has examined various techniques to understand users' facial expression by analyzing the movement of facial muscles. The researchers in signal processing have investigated how emotions impact human voices in terms of pitch, pause and even semantic expressions. Similarly, people from gesture recognition, text mining, and web analysis all have contributed to understanding human behaviors from different aspects.

Our work attempts to combine multiple information sources to provide even richer inferences about the users' behaviors. We propose that a wide spectrum of previously "invisible" behaviors can be captured and revealed by tapping into the rich set of sensors embedded in modern commercial mobile devices. Moreover, by exmining these behaviors, we are able to understand users' intentions and reactions in various scenarios. These inferences range from low-level user behaviors like gesture interactions to high-level user attributes such as attention and interests. We show that these inferences can be useful in a series of applications – enabling touch-sensitiveness on passive surfaces, understanding user's reactions towards online content, and capturing groups of users' interests for generating life-logging video highlights.

Successfully translating this vision into usable systems entails many challenges. For example, one challenge could be how to summarize raw data from each individual dimensions to valuable information towards an application Our work has borrowed extensively from existing research in different fields and mostly concentrated on one key challenge: how can we synthesize information from different dimensions together to understand a specific type of user behavior, especially when these information are of drastically different nature. There are several solutions to this problem ranging from using ensemble classifiers, designing specific semi-supervised learning algorithms, and to analyzing information from groups of users. We will discuss these solutions in more detail later. The rest of the introduction elaborates on the background in mobile computing and our contributions.

1.1 Background - Opportunities in Mobile Computing

Mobile devices are evolving. The inclusion of rich sensors (camera, microphone, accelerometer, gyroscope, etc.) in smartphones enables these devices to effectively observe, analyze and understand the users' context. In the future, this trend of expanding sensing capabilities is expected to continue with many new sensors (barometer, NFC, RFID readers, etc.) being embedded into mobile devices. In other words, these mobile devices are transforming from simple communication tools to powerful sensing/computing platforms. Fully equipped with powerful CPUs, large storage, multiple network interfaces and a rich set of sensors, these devices have the hardware that can potentially change the horizon of computing.

Users' behaviors have also changed during this new mobile era. Studies have shown that people nowadays are using their mobile devices all day long and are carrying their devices to almost all occasions. As IEEE spectrum describes this, "Not only is it always on, it is always somewhere on us". As the result, these mobile devices have become people's companions to various activities and can potentially be faithful observants of their daily lives.

Therefore, on one hand, the devices are now equipped with the sensing/computing capability to acquire and analyze information from multiple dimensions; on the other hand, users' new usage patterns have provided the devices perfect opportunities to continuously observe their activities from a close range. These two emerging trends together present us an unprecedented opportunity to obtain a high-resolution view of the users context, ranging from simple information like locations and activities, to high-level inferences about the user's intention, behavior, and social interactions. Understanding such context can help solving existing system-side challenges and eventually enable a new world of real-life applications.

1.2 Research Contribution

We propose to analyze users' intention, activities and interests via multi-modal sensing. Our intuition is that users' actions leave a wide spectrum of footprints on different sensing dimensions. Users' physical movements can be captured by motion sensors. Their mood may be reflected in their facial expressions or be modulated in their voices – detectable respectively from the embedded camera and microphone. Similarly, information sources such as users' web browsing history, email content and even calendar schedules can all provide useful hints regarding users' broad context.

In this thesis, we introduce three major projects related to enabling and exploiting context-awareness. Particularly, we attempt to show that different levels of users' behaviors can all be captured by the rich set of sensors embedded in mobile devices. These behaviors range from low-level activities such as gesture interactions to higher level information such as users' interests and attentions.

The first project targets to understand lower level user activities – gesture interactions with passive surfaces. The goal is to enable touch-sensitiveness on passive surfaces via sensing users' gesture interactions. We explore how the vibrations generated from gesture interaction can be analyzed to understand which region of the surface the user has been interacting with. When the interaction can be localized, we can use this information to effectively turn the surfaces into touch-sensitive input devices.

The second project is an example of understanding higher level user interest. This project attempts to enable automatic online content rating by sensing users' reactions. We explore how information from multiple sensing dimensions can be used to infer users' reactions when they watch movies from mobile devices. Moreover, the information can be analyzed collaboratively to estimate users' opinions towards the content and eventually provides both personalized ratings and semantic descriptions. The ratings and labels can be used as quality indicators for users to later make informed choices.

The last project investigates how group behaviors can be collaboratively examined to understand users' attentions and interests in social gatherings. The intuition is that whenever interesting events happening within a social gathering, people tend to switch their attention to focus on this new event. For example, people may all start looking at a similar person or object. Such behaviors can be captured by collaboratively analyze sensing information from multiple devices. The next section elaborates on each individual project.

1.3 Overview of Individual Research Projects

Understand Gesture interactions: Giving Passive Surfaces the Sense of Touch

This project is an example that shows how smartphone technology can help innovate flexible and promising UI designs. The vision is to turn passive surfaces into interactive interfaces by attaching a smartphone to them. Any interaction with the surfaces can generate unique vibration signal patterns that can be captured by the motion sensors inside the smartphones. The phone can then analyze such signals and use the final results for multiple purposes (e.g., virtual keyboard, gaming boards).

Understand User's Interest: Automatic Content Rating via Reaction Sensing

This project designs a system for automatically rating content - mainly movies and videos - at multiple granularities. Our key observation is that the rich set of sensors available on today's smartphones and tablets could be used to capture a wide spectrum of user reactions while users are watching movies on these devices. Examples range from acoustic signatures of laughter to detect which scenes were funny, to the stillness of the tablet indicating intense drama. Moreover, unlike in most conventional systems, these ratings need not result in just one numeric score, but could be expanded to capture the user's experience. In our experiments, encouraging results show consistent correlation between the user's actual ratings and those generated by the system. With more rigorous testing and optimization, our system could be a candidate for real-world adoption.

Understand User's Interests in Groups: Mobile Phone based Video Highlights via Collaborative Sensing

Event coverage is a well established notion in conventional sensor networks, where any event, such as a gas leak or a moving vehicle, can be detected, localized, and tracked over time. In this work, we use smartphones as the sensing platform and extend this notion to coverage of life events happening at social gatherings. We envision a futuristic application where mobile phones collaboratively form groups based on social context (e.g., people engaging in the same conversation), sense their ambience and recognize socially "interesting" events. Whenever an event is detected, the phone with a good view of the event triggers a video recording, and later, the video-clips from different phones are "stitched" into a video highlights of the occasion. With new devices like Google glasses looming on the horizon, this line of research becomes even more promising than ever before.

2

Giving Passive Surfaces the Sense of Touch

2.1 Introduction

Motion sensors were introduced in mobile devices primarily to enable new experiences in gaming. While the gaming industry has certainly benefited, systems and application developers have exploited these sensors in unanticipated ways. The Siri app, for example, gets activated when users lift their iPhones to their ears – accelerometers on the phone help identify the specific "lift" gesture. Localization systems differentiate places, such as Starbucks from WalMart, based on the user's movement patterns within these stores Azizyan et al. (2009a). A PhonePen tool Agrawal et al. (2011) demonstrates the ability to write English alphabets via hand gestures – users can switch TV channels by writing "CNN" in the air. These and many other creative apps Goel et al. (2012); Partridge et al. (2002) have enriched the mobile computing market, justifying the inclusion of additional sensors, such as gyroscopes. We aim to harness these inertial sensors in today's smartphones, and together with machine learning techniques, enable new capabilities for future app developers. Our key idea is that we can enable touch-sensitivity in an ordinary laptop screen simply by attaching a smartphone. Moreover, we show that the same approach may potentially be extended to other surfaces too, enabling a new spectrum of touch-sensitive applications. We also compare TouchSense with recently proposed acoustic-based methods and show how motion and acoustic sensing may potentially be combined to create a even more powerful and accurate system.We present our intuition next, followed by applications, challenges, and opportunities.

Consider Figure 2.1(a), where an off-the-shelf Android smartphone (with a 3-axis accelerometer and a gyroscope) has been attached to the back of a Lenovo laptop screen. When a user taps with her finger on the laptop's display (Figure 2.1(b)), it creates a response on both the inertial sensors. On the accelerometer, this response will be in the form of linear acceleration, while for the gyroscope, one can expect some degree of angular rotation (along the vertical axis on the plane of the display). In such a set up, we submit the following hypothesis: *the combined response on the accelerometer and the gyroscope will vary based on the location of the finger tap, but will remain almost the same for multiple taps on the same location.* If such a hypothesis can be proven, then it should be possible to develop a touch-sensitive laptop display at zero cost. A smartphone application would only need to obtain the location of the finger tap and feed it back to the laptop (perhaps through WiFi or Bluetooth). The laptop could perform the corresponding task, agnostic of whether the tap came from the mouse or a finger.

One could readily envision generalizing the central idea to other surfaces. As an example, a smartphone could be attached to an easel that has a poster mounted on it; when a visitor taps on a specific part of the poster, the smartphone can compute the location of the tap and play a relevant video. Another example could be in collaborative gaming where multiple users place a sensor-enabled tablet in the center of a table, and tap their ends of the table to interact with the game. Of course, these are visions of the future – we have not attempted to support such applications

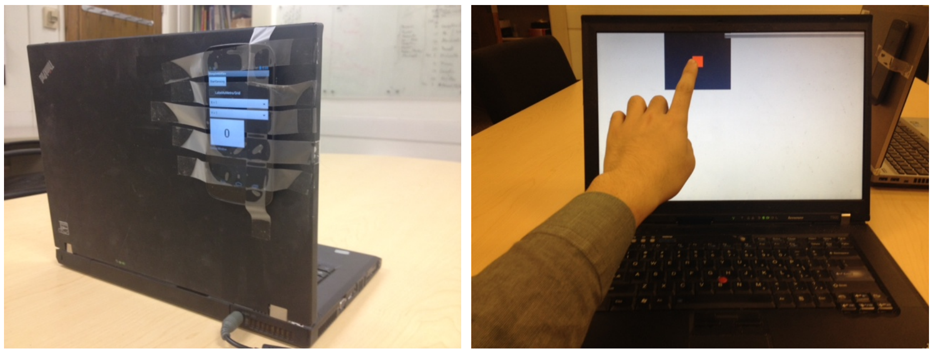


FIGURE 2.1: System set up: (a) a smartphone taped to the back of the laptop display; (b) a user taps on the laptop screen.

running on different types of surfaces. Instead, we have focused on the specific case of laptop screens, as a first step towards understanding the problem landscape and building a stable, functional prototype. We call our system *TouchSense*.

Building even this constrained prototype entails a number of challenges: (1) The core problem of modeling motion behaviors (based on a large number of features), and learning a fingerprint from it, is a non-trivial problem. The raw inertial sensor measurements are correlated, multi-dimensional and noisy. Since the gestures are physical actions, no two gestures are identical, and there is natural variation even within repetitions. (2) Models that scale across users without extensive per-user training compound the problem, as do different laptops. (3) Finally, users do not tap with consistency – a fatigued user after a day's work may tap softer than when she comes into the office in the morning. TouchSense copes with these challenges at the expense of lower resolution. Test results derived from 16 different users and two different laptop models (Lenovo and HP) demonstrate 84% tap detection accuracy for a grid of size 3×4. When errors occur, they are somewhat encouraging, i.e., the computed grid location is frequently adjacent to the one that the user actually tapped. Finally, even with a grid of size 4×6, the accuracy degrades gracefully to 65%. We also found that with two phones attached to the laptop, the opportunities magnify, leading to a 7% gain in accuracy.

Our contributions may be summarized as follows:

- *We identify an opportunity to make passive surfaces touch-sensitive via the use of off-the-shelf smartphones.* We exploit the idea that touches at different locations can cause distinct vibrations and rotations that can be sensed by the phone's motion sensors.

- *We adopt a robust ensemble-based supervised machine learning approach to make TouchSense usable across different users and laptop models.* Our experimental results demonstrate promise, allowing for on-screen gestures like tapping on large-sized icons, double-tapping to open applications, and swiping to scroll webpages.

- *We explore the potential of extending our approach to different surfaces and combining it with acoustic methods.* The early results show promising performance on rigid surfaces such as conference tables.

The rest of the paper expands on each of these contributions, beginning with overview and design details, followed by prototype implementation, and performance evaluation. We conclude the paper with a summary and a brief discussion on future work.

2.2 TouchSense Design

TouchSense extracts a fingerprint from the inertial motion sensors of off-the-shelf smartphones, attached to the back of a laptop screen. This is of course a proof of concept, but if the fingerprinting techniques indeed scale for real world deployments, the actual packaging can be far more sophisticated. In fact, laptop manufacturers could embed motion sensors directly into laptop displays as a cost-effective alternative to capacitive touch screens.

Figure 2.2 shows the consequence of repeated taps on the smartphone's 3-axis accelerometer and gyroscope. Observe that the linear acceleration is along the Z axis, perpendicular to the plane of the screen. The gyroscope fluctuations are dominantly along the X-axis, indicating a rotation around the vertical direction on the plane of the screen (also called *roll*). A slight rotation also emerges on the Y-axis, implying that the top of the phone rotates away from the user (also called *pitch*). Clearly, these rotations are an outcome of the torque and force, respectively, imposed by the taps. Evident from the figure, these per-tap signal patterns are clear and consistent.

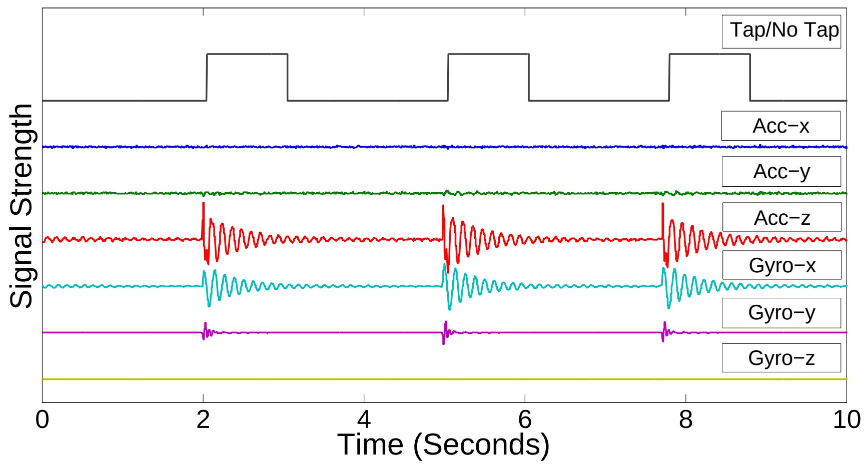


FIGURE 2.2: Raw smartphone sensor data for three taps from a phone attached to the laptop. True tap events (Tap/No Tap) shown along with associated accelerometer and gyroscope signals.

The general workflow of the TouchSense is depicted in Figure 2.3. First, an interaction detection module running on the phone detects that some form of interaction with the laptop screen has occurred. This module then polls the accelerometer and gyroscope readings at $200Hz$. When an interaction is indeed confirmed, the sensor streams are segmented and fed to the following stages. We discuss each of these stages next.

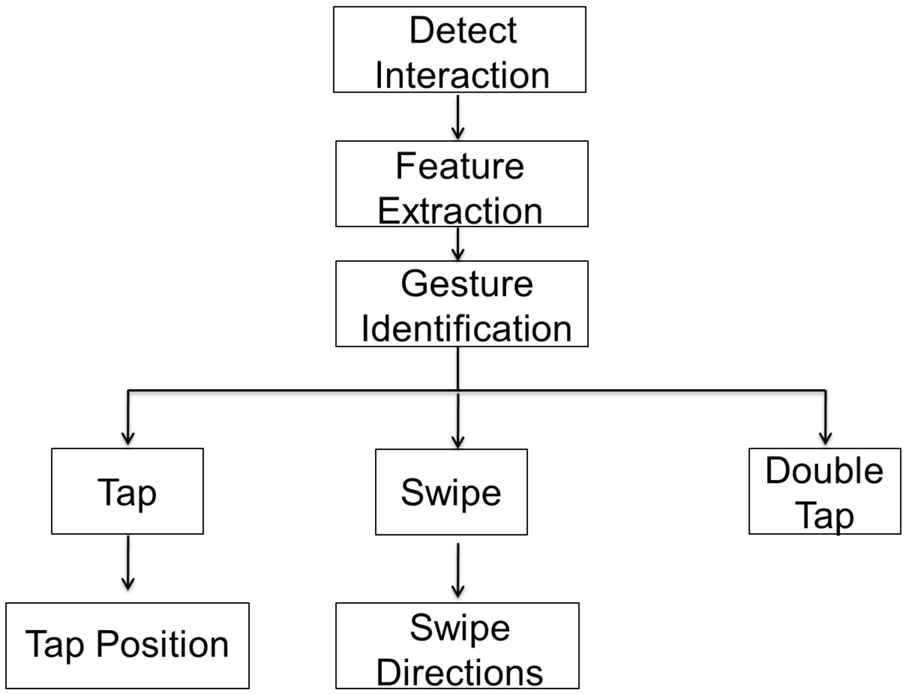


FIGURE 2.3: A schematic showing the TouchSense system work flow. The text describes the system and each component in more detail.

2.2.1 Detecting Interactions

Besides tapping on the screen, the user is likely to type on the keyboard, tap on the table, or interact with objects around the laptop. This module is tasked to avoid false alarms from such interactions, and only select signals that correspond to screen taps or swipes. Towards this filtering task, TouchSense employs thresholds on signal amplitude and looks into the active signal dimensions. Non-screen interactions are either too soft or too strong, and not exactly on the accelerometer/gyroscope dimensions induced by screen interactions. Thus, when a signal satisfies the filtering constraints, say at time t_i, TouchSense extracts the subsequent time-window of the signal till the point the signal has faded. The fading time is bounded by $1s$, hence, our experiments ensure that distinct screen interactions are separated by at least that duration. Of course, in the case of double-taps, they happen within the same signal window. This signal window, also called a *segment*, is passed on to the subsequent stages for feature extraction and gesture classification.

2.2.2 Feature Extraction

The goal of the feature extraction module is to extract useful and informative features from segmented stream of accelerometer and gyroscope readings. The intuition is that taps on different regions of the screen cause the screen to vibrate/oscillate distinctly. For example, tapping on the upper left corner of the screen causes it to vibrate more significantly than the bottom right corner (a signal magnitude effect). Also, because the taps propagate vibrations across the screen, certain parts of the screen may vibrate earlier than other parts (a signal temporal effect). Other physical effects of gestures such as taps and swipes on the screen include subtle rotations and oscillations of the screen, and the corresponding induced correlations between multiple sensing dimensions (e.g., taps causing correlated vibrations and rotations). Thus, it should be possible to create distinct ’fingerprints’ of taps at any particular region. While the fingerprint itself is the model we will learn (which we describe in detail in the next section), the goal of the features we extract is to provide a rich enough space of observations (in other words, to capture all the information related to the event) so that we can then use these features to create accurate models. Since the machine learning models we employ are relatively robust to large numbers of noisy features (we describe these models in the next section), our goal with feature extraction is to capture as much relevant information as possible.

The raw data we collect for each gesture event is a set of sensor time series (see Figure 4.14). Classical timeseries analysis suggests that the salient features of any particular timeseries can be described by summarizing the time and frequency domains of that time series. Because a gesture may produce distinctive correlated effects on the three axes of a sensor, we also want to extract features from the

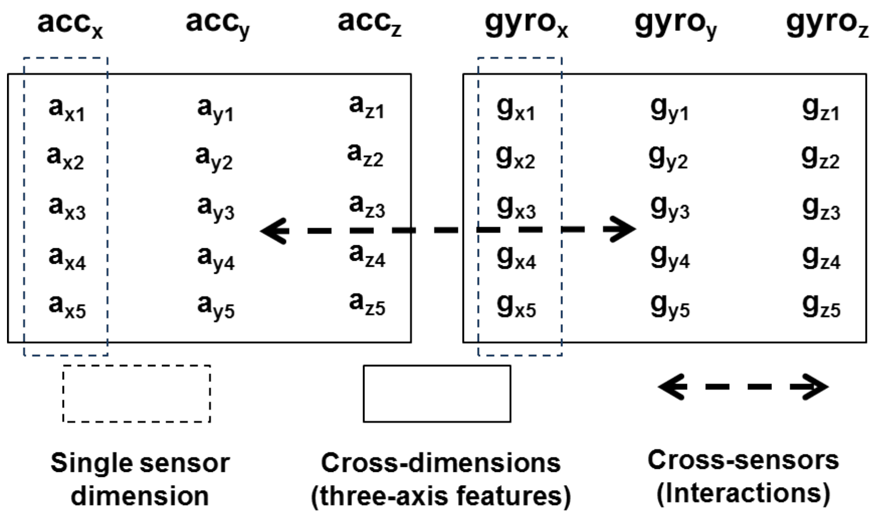

acc$_x$
acc$_y$
acc$_z$
gyro$_x$
gyro$_y$
gyro$_z$
a$_{x1}$ a$_{y1}$ a$_{z1}$
a$_{x2}$ a$_{y2}$ a$_{z2}$
a$_{x3}$ a$_{y3}$ a$_{z3}$
a$_{x4}$ a$_{y4}$ a$_{z4}$
a$_{x5}$ a$_{y5}$ a$_{z5}$
g$_{x1}$ g$_{y1}$ g$_{z1}$
g$_{x2}$ g$_{y2}$ g$_{z2}$
g$_{x3}$ g$_{y3}$ g$_{z3}$
g$_{x4}$ g$_{y4}$ g$_{z4}$
g$_{x5}$ g$_{y5}$ g$_{z5}$
Single sensor dimension
Cross-dimensions (three-axis features)
Cross-sensors (Interactions)

FIGURE 2.4: Schematic showing the form of the accelerometer and gyroscope data we observe. The boxes around the rows and columns are groupings of the measurements we use to extract features, see the text for details.

joint multidimensional (x,y,z) time series for each sensor. Hence, at a high-level, we extract two types of features: column features and matrix features (see Figure 4.14). Column features are extracted from the individual components of each sensor axis, while matrix features capture the correlation between the three-axis accelerometer and gyroscope vectors. In total, we use 250 features for each tap. The features we use span both the time domain and frequency domains. The underlying physical event of a tap motivates our choice of features, which are related to quantities like the amount of energy in the tap signal, the rate of change of the sensor measurements etc.

Time Domain Features:

Time domain features consist of three categories:

(i) *Single sensor dimension features.* These are event features that we extract per axis of the accelerometer and gyroscope (classical single timeseries features). So, for example, features we extract from the z-component of the accelerometer etc. We pick features that capture the broad temporal sequence of the event as a whole, as well as various summary statistics from the event. This is useful because it allows the classifiers to create models on important/sensitive sensor dimensions directly. To this end, we extract (1) a fixed length cubic spline interpolation of the vector itself, (2) extreme values (min/max) of the vector, (3) moments of the vector including the mean, standard deviation and higher order moments including the skewness (to measure the asymmetry of the vector components) and the kurtosis (to measure the 'peaked'-ness of the vector components), and (3) a fixed length spline interpolation of the first order numerical derivative of the signal.

(ii) *Three-axis accelerometer and gyroscope features* These are event features that we extract from the joint multidimensional (x,y,z) measurements of the gyroscope and the accelerometer (separately). These features are also important because we measure accelerations and rotations as multidimensional (three-axis) vectors, and thus treating each dimension/axis independently, we lose information in the correlations between the dimensions. Treating the event data from the three axes of the gyroscope (for example) as a matrix, we extract (1) various matrix norms – 1-norm (maximum absolute column sum of the matrix), infinity norm (maximum absolute row sum of the matrix), Frobenius norm (square root of the squared sum of the entries in the matrix). These matrix norms provide various summary statistics of the data in the matrix. (2) squared l_2 norm (can be viewed as the norm of the combined vector) of the rows in the matrix (3) features based on the magnitude of the acceleration and angular rotation vectors (an estimate of taps' energy).

(iii) *Features based on interactions between sensor dimensions.* These event features capture information present in the correlation between different sensor dimensions, as well as the different sensors themselves. For example, examining both the x and y axis event measurements from the gyroscope can help decide whether the screen made a subtle clockwise or counter-clockwise rotation during the event. Interaction features we extract include (1) the angle between accelerometer and gyroscope vectors and the rate these angles change (a concise representation of the correlation between acceleration and rotation) (2) The Pearson correlation coefficients between all pairs of sensor dimensions (which provides an estimate of the amount of linear correlation between any pair of sensor dimensions) . In summary, these features analyze sensor dimensions in different combinations and capture the correlations between them.

Frequency Domain Features:

In order to capture signal in the events more easily identified in the frequency domain, we also extract various frequency domain features. These features include the (1) Fast Fourier Transform (FFT) for all six sensor dimensions (3-axis accelerometer and gyroscope) , and (2) power spectrum computed from the FFT vectors.

Cross Device Features:

Cross device features are designed for scenarios involving more than one measurement device, for example, in the case where there are multiple phones mounted on the laptop screen or laptop manufacturers mount several motion sensors on the back of the screen. This features are useful, for example, in helping determine where someone tapped by analyzing the relative signal strengths detected by the two devices. The device spatially closer to where the tap occurred would then presumably observe a larger magnitude signal than the device further away. An analogous opportunity exists with respect to analyzing the temporal difference for an event between the observation sequences at the two devices since the device closer to the tap would 'feel' the event slightly faster than the device that is spatially further away. For these cross device interaction features, we analyze column vectors on the same sensor dimension from the two devices. The features include (1) the element-wise ratio between the values in these vectors (2) the norm of the difference between these vectors, and (3) the angles between these vectors.

2.2.3 Gesture Identification

After feature extraction, the next step is to learn models that will accurately predict events, in our case, distinguish between gestures, locate taps and identify swipe directions. We model all of these problems as supervised classification problems. Our overall strategy for classification is to use an ensemble classifier, which combines multiple classifiers together in a non-linear fashion. Using an ensemble is well known to be more robust and potentially more accurate than using any single classifier Breiman (2001); Caruana et al. (2004); Jahrer et al. (2010). In our work, we use three powerful individual classifiers to generate a large set of candidate/base classifiers which we ensemble. These three classifiers are Support Vector Machines (SVM), Bagged Decision Trees and Random Forests.

Support Vector Machine (SVM)

We use both linear and rbf kernel SVM (libLinear and libSVM, Schoelkopf et al. (1998)). We vary the tuning parameters (regularization coefficient, and kernel bandwidth parameter for the rbf) in order to obtain a set of SVM classifiers for both the linear and non-linear cases. We find the kernel classifiers are quite a bit more accurate for our application, and believe this is because our problem likely has strongly non-linear decision boundaries.

Bagged Decision Trees

A set of bagged decision trees is an ensemble classifier by itself that is composed of a collection of decision trees. Each component decision tree is trained on a bootstrap sample of the original data, instead of training it on all the training data. The diversity in the different bootstrap samples allows the ensemble to mitigate the impact of poor samples and improves the predictive performance. The bagged tree ensemble classifier itself is a simple majority vote from the component trees. In our experiments, several such bagged decision trees are used by varying the number of trees in the model.

Random Forests

Random forests are another ensemble decision tree based classifier Breiman (2001). Unlike bagged decision trees, the decision trees in random forest not only are learned on a subset of samples but they also on a select subset of features from the whole set of features. Thus random forests ensemble together an even more diverse collection of decision trees and also employ a majority vote for the final classifier.

Ensemble - Training, Validation and Test

For any event we are trying to classify, the final ensemble classifier takes the output from all the component classifiers (the various SVM, Bagged Decision Tree and Random Forest models we learned) as inputs, and aims to produce an improved and more robust estimate of the label (tap position, gesture etc.) as its output. To learn this final ensemble classifier, we divide the data into three datasets: a training set, a validation set, and a test set. The training set is used to train each individual classifier. Then, the trained model is used to predict the labels for the examples in the validation set. These form the inputs/features for the final ensemble classifier, which is trained using the validation set (true) labels. We use random forests for our ensemble learning method mainly due to their excellent predictive performance and (relative) absence of tuning parameters. After all the classifiers are learned (both individual and final ensemble random forest), we can make predictions for test data. We do this for any test example by (1) obtaining the individual classifier predictions, and then, (2) combining the individual predictions using the final ensemble random

Table 2.1: Experiment/Users/Equipment

Experiment	Users	Laptop
Exp.1 Gesture	1-8 (120 gestures$\times$3)	Lenovo
Exp.2 Swipe	1-8 (30 swipes$\times$4)	Lenovo
Exp.3 Tap (12 grids)	1-8 (60 taps$\times$12)	Lenovo
Exp.3 Tap (12 grids)	9-16 (30 taps$\times$12)	HP
Exp.3 Tap (24 grids)	1-8 (30 taps$\times$24)	Lenovo
Exp.4 Two Device	9-16 (30 taps$\times$12)	HP

forest to obtain our final predictions.

2.3 Evaluation

We evaluated the accuracy of TouchSense in a series of experiments, testing the limits of our pseudo touchscreen with different input techniques. Experiment 1 evaluated TouchSense's ability to recognize *taps*, *double taps* and *swipes*. Experiment 2 evaluated TouchSense's ability to recognize *swipe* directions (*left*, *right*, *up*, *down*. Experiment 3 evaluated TouchSense's ability to detect the location of taps. Experiment 4 evaluated whether using multiple phones improves TouchSense's accuracy. Experiment 5 evaluated TouchSense on a rigid surface. Finally, Experiment 6 compared TouchSense with the acoustic approach.

In summary, we show that TouchSense can distinguish between tap, double tap and swipe gestures with near perfect accuracy ($M > 99\%$), recognize swipe directions precisely ($M = 90\%$) and detect tap locations well ($M = 84\%$ on 3$\times$4 grid and $M = 65\%$ on 4$\times$6 grid), achieving good accuracy ($M = 63\%$) even when no training data is available for a user. In addition, TouchSense's accuracy improves with more sensors ($M = 88\%$), is consistent across users and isn't affected by the laptop type. Finally, TouchSense achieves 96% accuracy on a conference table and could be combined with acoustic sensing to achieve 97% accuracy.

2.3.1 Data Collection

TouchSense performs its inferences using gyroscope and accelerometer readings collected from a mobile phone attached to the back of the laptop display. Unless stated otherwise, all experiments were conducted using sensor data from a single phone, attached to the top left corner of the laptop. We tried a variety of different locations in pilot studies and found that when the phone was placed there, TouchSense achieved the best results.

Accelerometer and gyroscope sensor data was sampled at 200Hz from a Samsung Galaxy Nexus phone. Our machine learning algorithms were run off-line, on a server, using MATLAB routines and machine learning packages. Evaluation was performed using five-fold cross validation. In total, we collected labeled data for more than $19,560$ gestures.

Overall, 16 users participated in our experiments. Two laptops were used. Table 2.1 summarizes the participants and the laptops used in each experiment. More details are presented in the later experiment sections.

2.3.2 Detecting Interactions

As a first step, TouchSense needs to identify that a gesture has occurred and differentiate actual gestures from external events (e.g., tapping on table). To study the effectiveness of TouchSense at the task, we conducted a brief experiment where 3 users were asked to perform TouchSense-supported gestures (*tap*, *double tap*, and *swipes*) as well as non-supported gestures (typing on keyboard and tapping on the desk). Adding these spurious events allowed us to evaluate both how often TouchSense misses supported events and produces false positives. We spaced out events and "non-events" between 2 to 4 seconds to ensure there was no overlap between events. In total, 75 events were collected in a quiet office setting, of which 45 were TouchSense-supported gestures and 30 were not. TouchSense did not miss any events and reported no false positives.

2.3.3 Experiment 1: Distinguishing Gestures

The purpose of this experiment was to evaluate whether we could distinguish between three basic gestures that form the basis for many touch-based interfaces: *tap, double tap, swipe.*

Eight right-handed volunteers (2 female) ranging in age from 21 to 27 (median 25) were recruited from a local university. All participants had experience with touchscreens. Participants were asked to sit in front of a laptop and perform specific gestures when prompted. Since this experiment was conducted on a 15.4" Lenovo T500 laptop, we refer to this group of participants as the *Lenovo group.*

We wrote a simple application to visually prompt the user to provide input (*tap*, *double tap*, or *swipe*). For taps and double taps, a red square was shown on the middle of the screen. Every 3 seconds, the square would turn green, at which point participants were instructed to tap or double tap it. For swipes, 4 squares were shown on the screen (Figure 2.5). Every 3 seconds, two of the squares would turn green and users were asked to swipe from one green square to the other one.

All data was collected for a particular input (*tap*, *double tap*, or *swipe*) before moving on to the next one. 120 samples of each input were collected from each user, totaling $120 \times 3 = 360$ input events. The 120 swipe events consisted of 30 swipes in each of the four cardinal directions: *left*, *right*, *up* and *down.* For this portion of the evaluation, the four directional swipes were aggregated into one group of 120 swipe events.

Figure 2.6 plots representative accelerometer and gyroscope readings collected from this process for different inputs. Clearly, different gestures produce very distinct responses.

Figure 2.7 shows that TouchSense can distinguish between gestures with near-

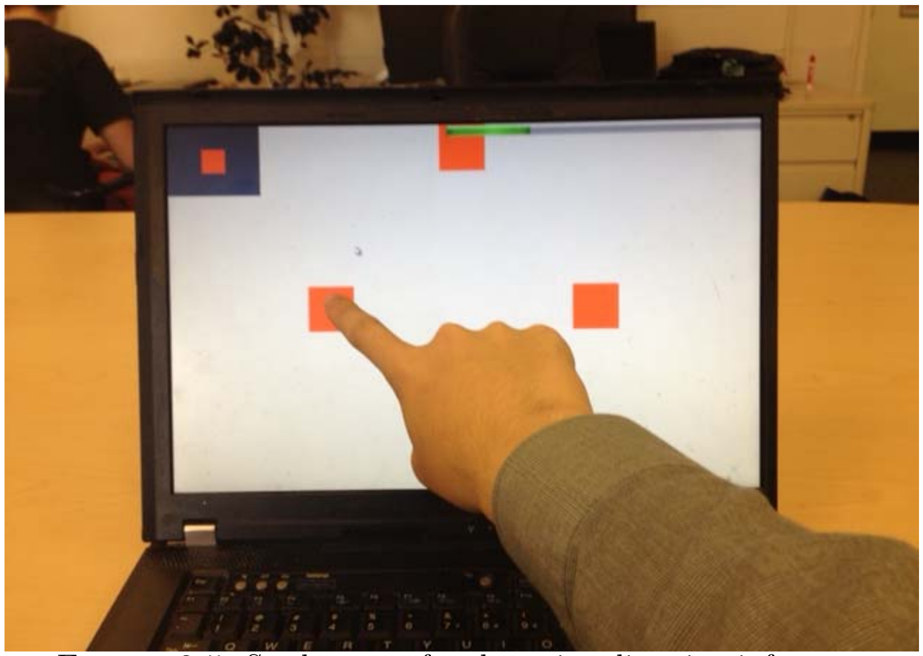


FIGURE 2.5: Study setup for the swipe direction inference.

perfect accuracy. Since TouchSense could differentiate between different gestures well, we decided to study swipes and taps in more detail.

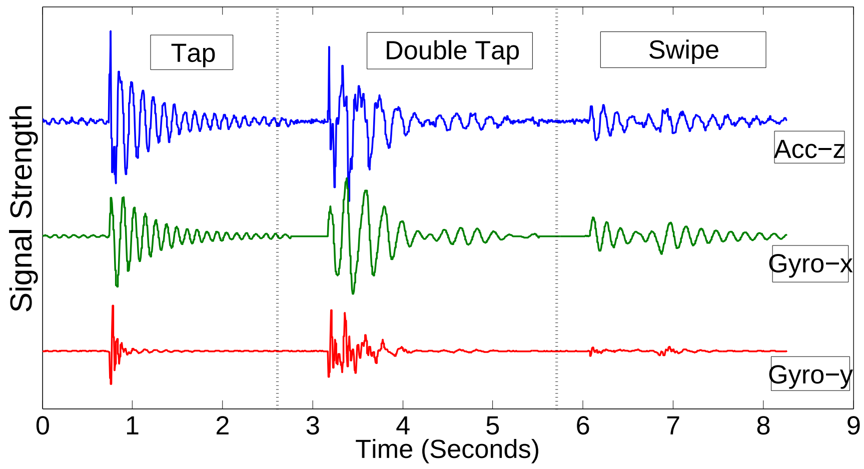


FIGURE 2.6: Sample sensor readings generated by taps, double taps and swipes. Shown here are the most visibly responsive sensor dimensions.

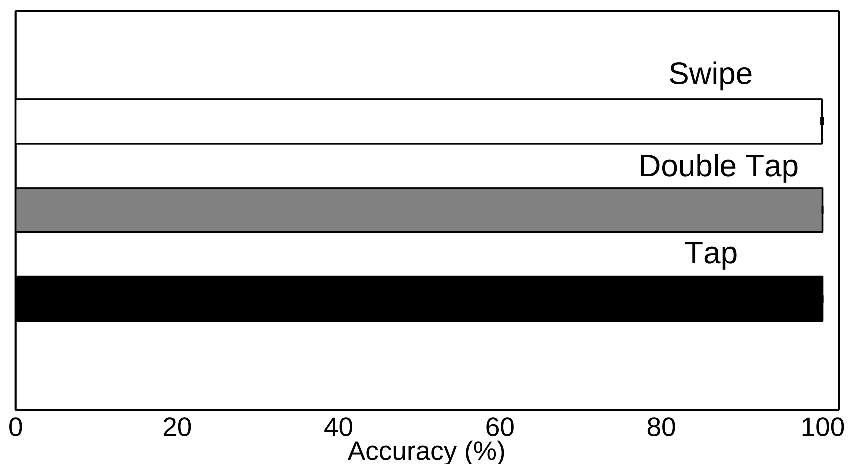


FIGURE 2.7: TouchSense's accuracy in differentiating between taps, double taps and swipes. Error bars show 95% confidence interval.

2.3.4 Experiment 2: Detecting Swipe Directions

The purpose of this experiment was to evaluate how well *TouchSense* distinguishes between swipe directions: *left*, *right*, *up* and *down*. We selected these directions since they are commonly used on touchscreen devices such as tablets and smartphones, for example, for scrolling. For this experiment, we used the swipes collected in Experiment 1 (120 swipes per user).

Figure 2.8 shows TouchSense's accuracy in distinguishing swipe directions. The accuracies range from 82% to 91%, with a mean accuracy of 86%. We do not see a significant difference in accuracy between the different directions.

2.3.5 Experiment 3: Recognizing the Location of Taps

In this five-part experiment, we studied the performance of TouchSense in identifying different tap locations on our pseudo-touchscreen. We looked at: (a) absolute accuracy, (b) differences between laptops, (c) coarse vs. fine grained localization of taps, (d) training the classifier with other users' data and (e) the relative importance of the gyroscope and accelerometer sensors.

Recall that our approach is based on supervised learning, which is highly depen-

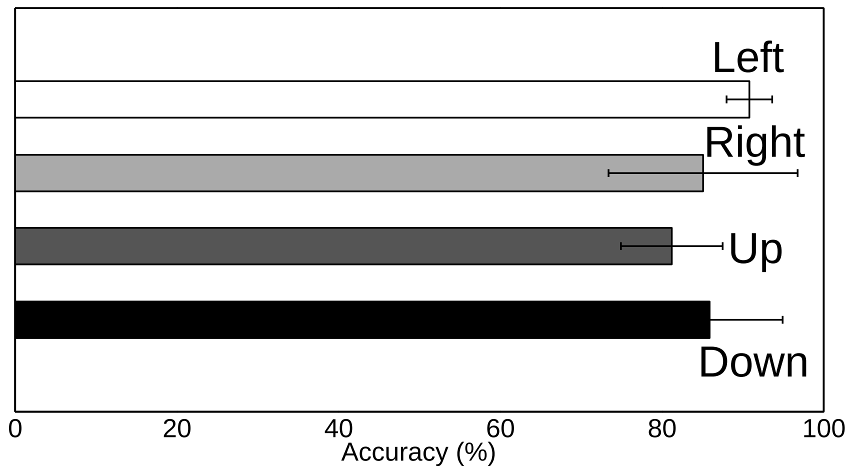


Figure 2.8: Mean accuracy in distinguishing between different swipe directions. Error bars report 95% confidence interval.

dent on the number of samples available for training the classifier. Generally speaking, the larger the number of taps we collect from each user, the higher TouchSense's accuracy, up to a saturation point. Since our machine learning models are flexible, larger amounts of training data allow these models to adapt to subtle characteristics of the signal.

How many samples are needed?

As a preliminary part of this experiment, we wanted to get a sense of the limitations of our approach by training our classifier with a large amount of training data. In particular, we were curious about where accuracy would taper off with respect to the number of training samples. We used the same data collection technique we used in Experiments 1 and 2. However, this time participants were asked only to tap on the screen when prompted by the application. For this experiment, our application prompted the participant to tap on one of the 12 rectangles (a 3×4 grid) drawn on the laptop's screen. Each rectangle measured 5.5cm high, 8cm wide, . This layout is reflected in Figure 2.10. We chose this arrangement as it resembles the startup screen of Windows 8.

We collected data from 3 participants (75min. per person). The current tile was marked by a red square. Every three seconds, the square would turn green, prompting the user to tap it. All 90 taps were collected from each square, before the application advanced to the next square. Squares advanced from left to right, top to bottom.

Figure 2.9 shows that accuracy increases with the number of training samples up to about 60 samples, where it levels off. Based on this result, we collected 60 taps per tile from subsequent users.

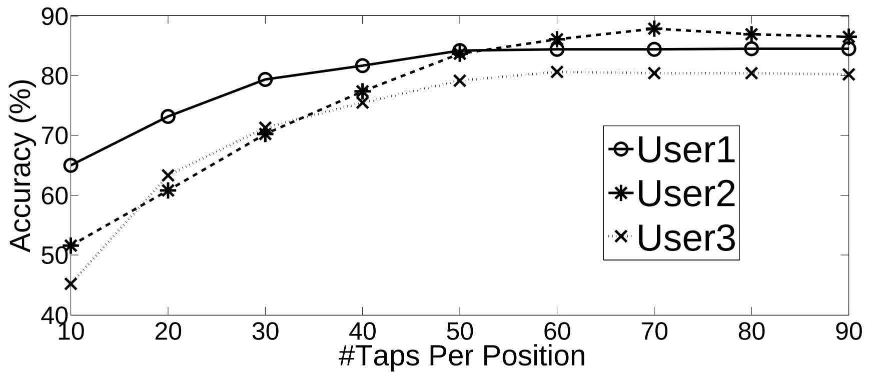


Figure 2.9: Tap localization accuracy as a function of number of training samples for three users. Accuracy tapers off after 60 training samples.

Using the same experimental setup, we collected data from another 5 right-handed participants (2 female), aged 21 to 27 (median 25), giving us a total of 8 participants. We used 60 taps per person to train TouchSense. Data collection lasted approximately 45min per user.

Figure 2.10 shows the mean accuracy for each tile with an overall accuracy of 84%. The accuracy tends to be lower in the central tiles, which is to be expected since the higher number of neighbors increases potential misclassification. There also appears to be a slight lowering of accuracy for tiles located on the right side of the screen. This is reasonable, since our phone is mounted on the left corner of the

Table 2.2: Confusion matrix for tap localization on the 3 × 4 grid. The first column shows the average tile inference accuracy. The 2nd, 3rd, 4th, and 5th columns show the confusion percentage with, respectively, the 1st, 2nd, 3rd, and 4th most confused tiles. Tiles are indexed by the row and column number of their position on the screen.

Avg acc (%)	1st conf	2nd conf	3rd conf	4th conf
1,1: 89.35	**2,1**: 2.49	**3,2**: 2.13	**3,1**: 1.27	**1,2**: 1.25
1,2: 82.00	**2,2**: 9.98	**3,2**: 2.62	**2,3**: 1.29	**1,3**: 1.28
1,3: 78.58	**1,4**: 7.05	**2,4**: 4.98	**2,3**: 2.39	**2,2**: 2.16
1,4: 78.00	**2,4**: 10.90	**1,3**: 5.44	**3,4**: 1.56	**3,2**: 1.31
2,1: 84.00	**3,1**: 5.46	**2,3**: 2.82	**1,1**: 2.22	**3,2**: 2.19
2,2: 74.00	**1,2**: 7.88	**3,2**: 6.98	**2,3**: 6.62	**2,4**: 1.52
2,3: 72.00	**2,2**: 8.08	**2,4**: 6.75	**1,3**: 5.46	**3,2**: 2.81
2,4: 68.51	**1,4**: 13.22	**2,3**: 6.49	**3,4**: 2.86	**3,2**: 2.62
3,1: 82.00	**3,2**: 5.97	**2,1**: 4.10	**3,3**: 3.69	**2,2**: 1.72
3,2: 86.50	**3,3**: 9.72	**3,1**: 3.51	**3,4**: 0.43	-
3,3: 85.00	**3,2**: 7.44	**3,4**: 3.89	**3,1**: 1.31	**2,3**: 0.89
3,4: 84.51	**3,3**: 7.25	**3,1**: 3.46	**2,3**: 1.84	**1,3**: 0.69

screen[1].

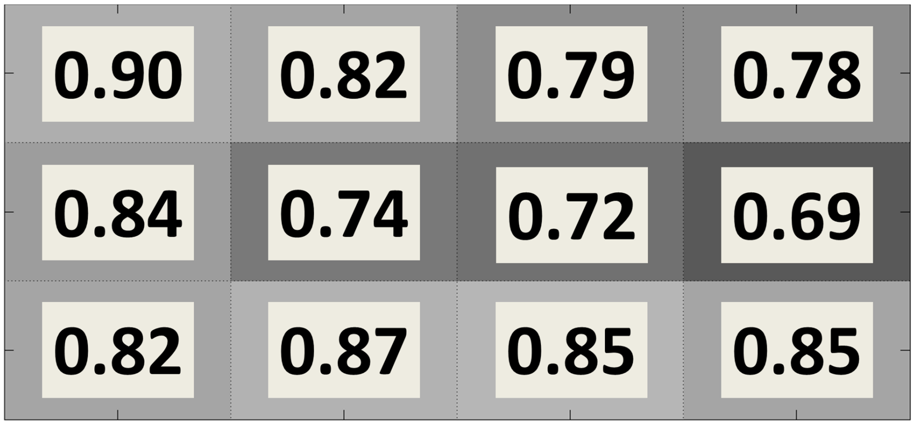


FIGURE 2.10: Tap localization accuracy for each tile on the laptop screen for the case of 12-tile grid (3×4).

Table 2.2 displays the mean classification accuracy, along with the percentage of confusions with the top-most confused tiles. These results show that our misclassifications are spatially correlated.In other words, when a mistake is made in locating a tile, it is likely made due to confusion with a nearby tile. Knowledge of this fact may help an application to derive an effective UI to mask misclassifications. For example,

[1] Note that our preliminary experiments showed that attaching the phone to the center of the screen resulted in lower overall accuracy than our chosen position. With the phone mounted in the center, accuracy in the corner and edge tiles dropped.

if an application's UI does not require the resolution offered by a 3×4 grid a UI designer can place touch-enabled titles in regions with higher inference accuracy.

Training with fewer samples

Although training with 60 samples obtains accuracy that is close to the limits of our algorithm, we were curious about how accuracy would degrade with fewer training samples since it would reduce the required training time for each user.

For this set of users, we compared the accuracy of a classifier trained on the first 30 samples to one trained on all 60. Figure 2.11 shows the difference in accuracy between training with 60 taps and 30 taps per tile for the same set of users. Training the classifier with 60 taps results in a statistically significantly higher accuracy ($t(7) = 3.97$, $p < .01$). However, training with 30 taps still achieves an average accuracy of 80%. Since using 30-tap training effectively halves the training time to roughly 25min, we believe it could be a compelling choice for certain kinds of applications.

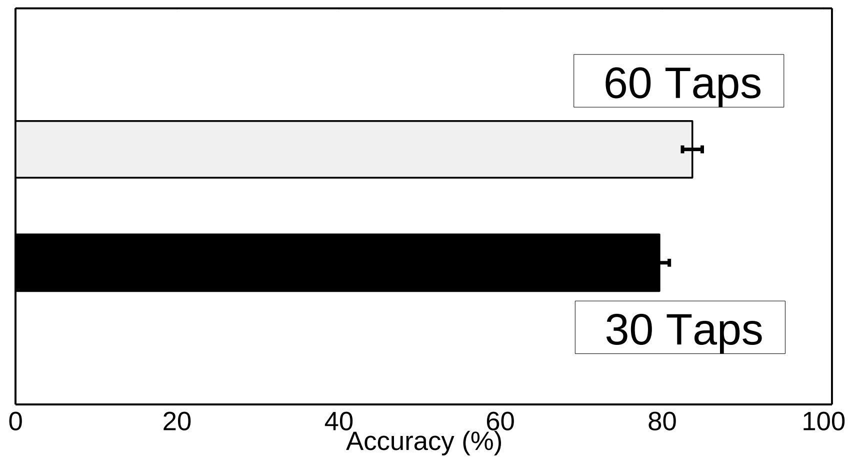


FIGURE 2.11: Mean tap location inference accuracy when training with different number of samples. Error bars report 95% confidence interval.

Does the laptop matter?

We next examined whether the particular model of the laptop would affect Touch-Sense's performance. We used the same procedure used in the earlier part of this experiment to collect data from an additional 8 right-handed volunteers, aged 27 to 34 (median 30), but on a different laptop. 30 taps were collected from each participant for each of the 12 tiles, this time on a 15.6” HP Elite8560P. We refer to this group as the *HP group*.

Figure 2.12 shows the accuracy the system achieves for each user in both *HP* and *Lenovo* groups. The difference in performance between the two groups was not statistically significant. At the very least, Figure 2.13 suggests that TouchSense's performance is not tied to a single piece of equipment.

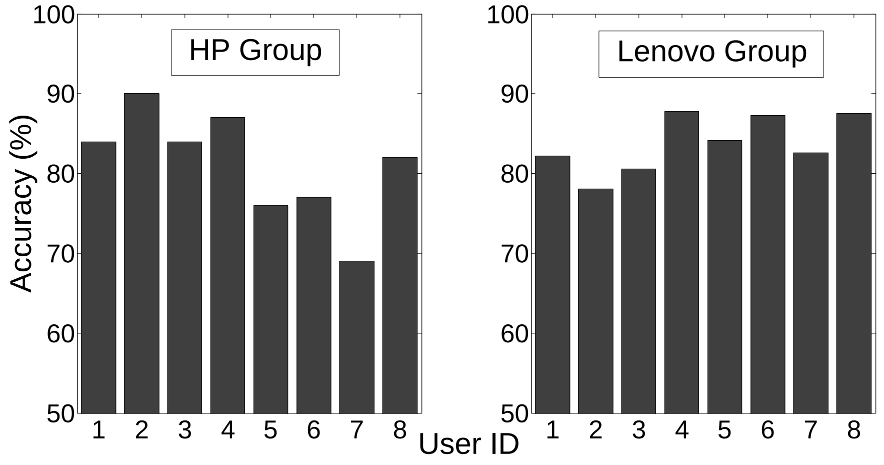


FIGURE 2.12: Accuracy for all 16 users on the 12-tile tap localization experiment. Users are grouped based on which laptop they used (*Left:* HP group, *Right:* Lenovo group).

Coarse vs. Fine-grained Localization

Having established TouchSense's performance on the 12-tile grid, we attempted to examine its limitations in terms of localization granularity. In this experiment, we examined whether our approach would work on a finer-grained 4×6 arrangement of tiles. We used the same experimental setup as in the previous section with participants from the *Lenovo* group. The only difference was that we used a 4×6 grid instead of a 3×4 grid. Each tile was 4.5cm high, 5.4cm wide. To keep the data

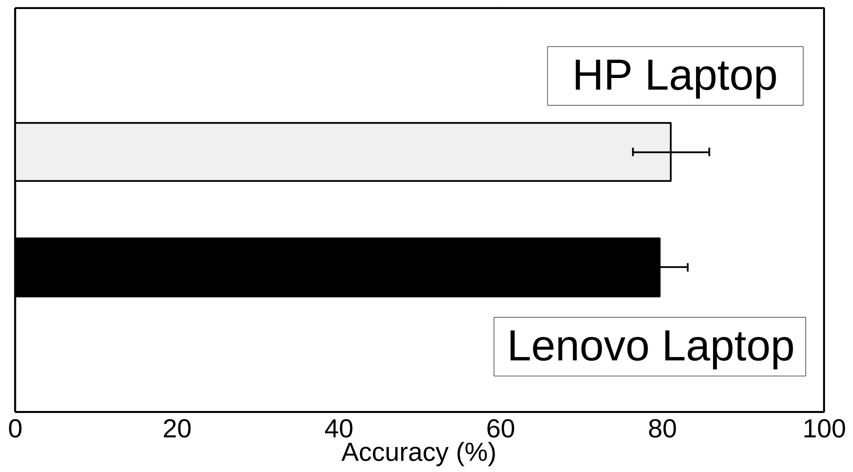


FIGURE 2.13: Mean tap location inference accuracy using data collected from the two laptops: HP and Lenovo. Error bars show 95% confidence interval.

collection time reasonable, we collected 30 taps per position for each of the 24 tiles. Data collection lasted approximately 45min per user. Figure 2.14 compares overall accuracy between the coarse-grained and fine-grained conditions. As expected, overall accuracy for the fine-grained condition is significantly lower than that of the coarse-grained condition $t(7) = 9.2537$, $p < .001$.

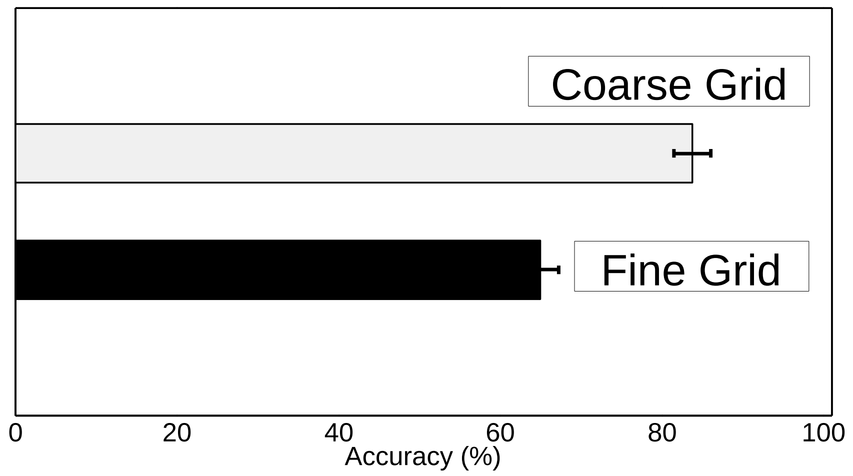
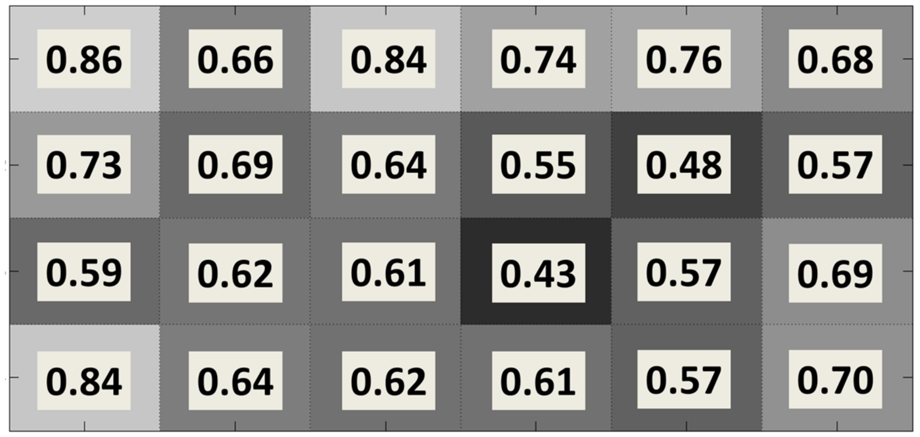


FIGURE 2.14: Mean tap localization accuracy for two different granularities: 3×4 (coarse grid) vs. 4×6 (fine grid). Error bars show 95% confidence interval.

Figure 2.15 shows the mean accuracy at a per tile level, with an The average accuracy per tile is 65%. Similar to with the 12-tile grid, accuracy trends lower for center tiles and tiles farther from the top left corner.



0.86	0.66	0.84	0.74	0.76	0.68
0.73	0.69	0.64	0.55	0.48	0.57
0.59	0.62	0.61	0.43	0.57	0.69
0.84	0.64	0.62	0.61	0.57	0.70

FIGURE 2.15: Figure showing the tap location detection accuracy per tile mapped to the 24-tile grid (4×6).

Training With Other Users' Data

The approach we have described so far relies on collecting training data from a user in order to infer tap locations for that user. Clearly, limiting or eliminating this training data collection requirement is desirable. To explore this limitation, we conducted a set of experiments where we evaluated how TouchSense could infer a particular user's tap locations without collecting training data from that user. Instead, we train the system on data collected from other users. We then examine how TouchSense's performance changes as we add data for that user to the training set.

We randomly selected a user to "leave out," training the classifier with data from the other 7 users. Then, we measured TouchSense's accuracy as we added that user's data into the training set. This was measured with 0, 5, 10, and 15 tap samples for the user. We did this test for all users, leaving one user out at a time. Figure 2.16 shows TouchSense's accuracy when aggregated across all users. The results show that with no training from a particular user, TouchSense achieves 62% accuracy. As training samples are added, accuracy gradually increases.

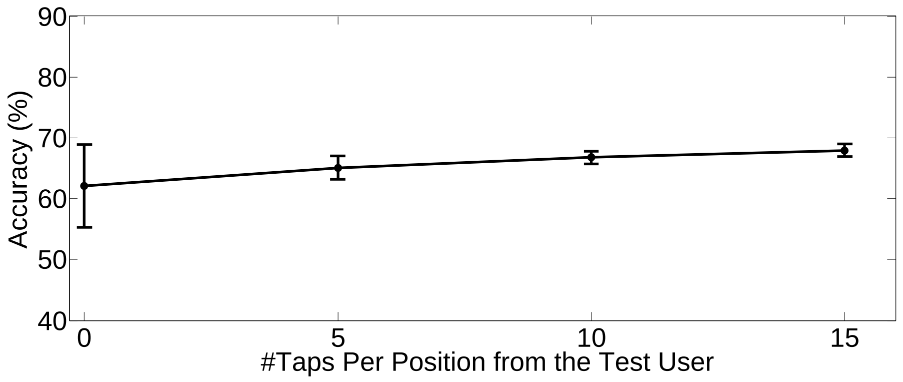


FIGURE 2.16: Plot shows the mean tap location inference accuracy when training with other users' data. Results are aggregated across all users. Accuracy improves as a user's data is added to the training set.

Relative Importance of Gyroscope and Accelerometer

Our results so far are based on using sensor data collected from both the accelerometer as well as the gyroscope. To test how each sensor contributed to overall accuracy, we re-ran the tap localization experiment on the *Lenovo* group's 3×4 grid data, using only one sensor at a time. Figure 2.17 shows that a classifier using only the gyroscope (78% accuracy), performs almost as well as one that uses both the gyroscope and the accelerometer (80% accuracy). Using the only the accelerometer yields 60% accuracy.

2.3.6 Experiment 4: Increasing accuracy with more phones

The purpose of this experiment was to examine whether using two phones would improve accuracy over the single phone case, and if so, by how much.

For this experiment, a similar setup was used as in Experiments 1–3, but with an additional Samsung Galaxy Nexus phone attached to the bottom right corner of the laptop display. 30 taps were collected for each tile of a 3×4 grid from the 8 users of the *HP* group. We chose the bottom right location to minimize the area on the screen with no nearby phones. With the additional sensing, we expected accuracy to improve.

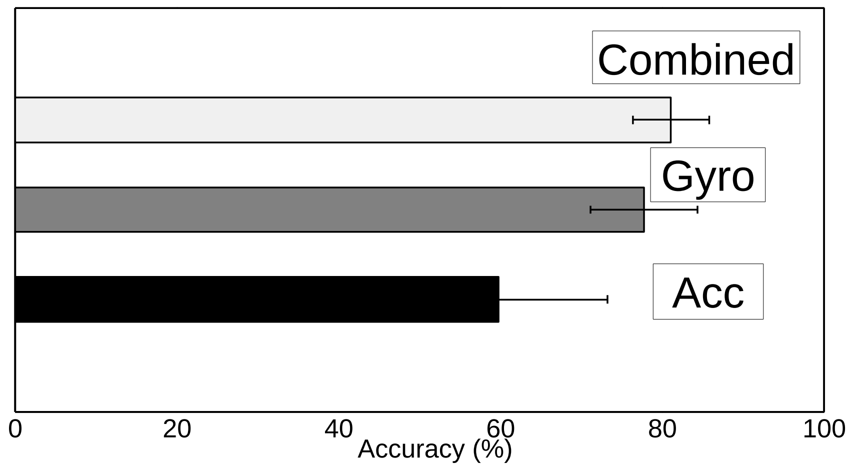


FIGURE 2.17: Localization accuracy when using only the accelerometer, only the gyroscope, and the combination of the two.

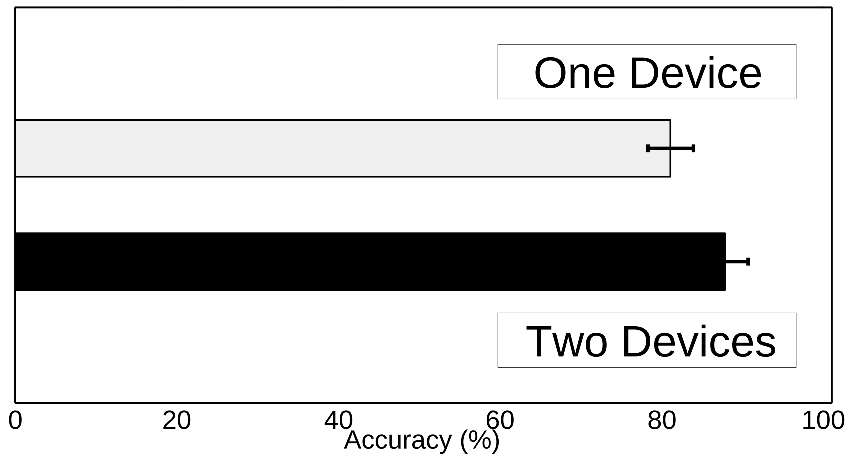


FIGURE 2.18: Mean tap location inference accuracy for one and two device scenarios. Error bars show 95% confidence interval.

Figure 2.18 shows the accuracy of the two different approaches when aggregated over all users. Using two devices led to a significant increase in accuracy ($t(7)$ = -2.1514, $p < .05$). On average, using two devices increases the accuracy by 7%.

2.3.7 Experiment 5: Using TouchSense on a rigid surface

The purpose of this experiment is to examine the applicability of using TouchSense on a rigid surface, in our case, a conference table. The ability to differentiate between tap positions on a table could be used for controlling devices or sharing content between participants. For this experiment, we chose the sharing scenario. We assume eight participants are sitting across a wooden conference table ($1.2m * 2.4m$). One user places her phone on the table and shares a file with one of the other users by tapping on a table along the direction toward the file sharing target. See our experimental setup in Figure 2.19.

To test how well TouchSense differentiates between various positions we conducted four sets of experiments. In each experiment, a person tapped 30 times per location on one of the seven positions marked by the red crosses. The TouchSense bar in Figure 2.20 shows that TouchSense works well, achieving 96% mean accuracy. Although these initial results are very encouraging, we plan to conduct more tests to quantify TouchSense's accuracy on other kinds of rigid surfaces.

2.3.8 Experiment 6: Comparing with the acoustic approach

The purpose of this experiment is to compare TouchSense with detecting tap positions using an acoustic approach. For fair comparison, we used the embedded microphone in the smartphone to capture the acoustic events. Our implementation uses both time domain and frequency domain acoustic features. The time domain features include the mean and variance of the acoustic signal within 0.1 second of when the tap is detected. The frequency domain features include the FFT features up to $3kHz$, aggregated into 10 bins.

We repeated the experiment described in Experiment 5, this time capturing both acoustic and motion data. Figure 2.20 shows that an acoustic method in a quiet setting achieves 94% accuracy. Combining acoustic features with TouchSense features improves the overall accuracy further to 97%.

Unfortunately, the acoustic approach does not work in a noisy setting, when, for example, people talk while tapping. In this case, it is hard to separate tapping signals from the background noise as we illustrate in Figure 2.21. The figure shows

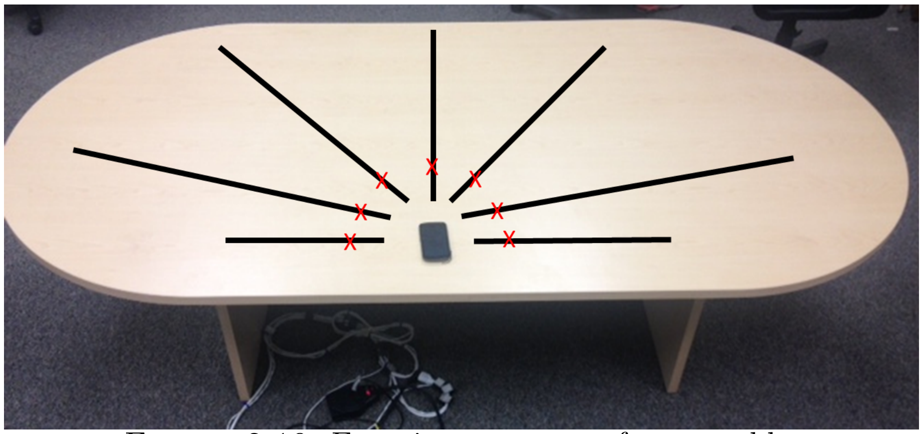


FIGURE 2.19: Experiment on a conference table.

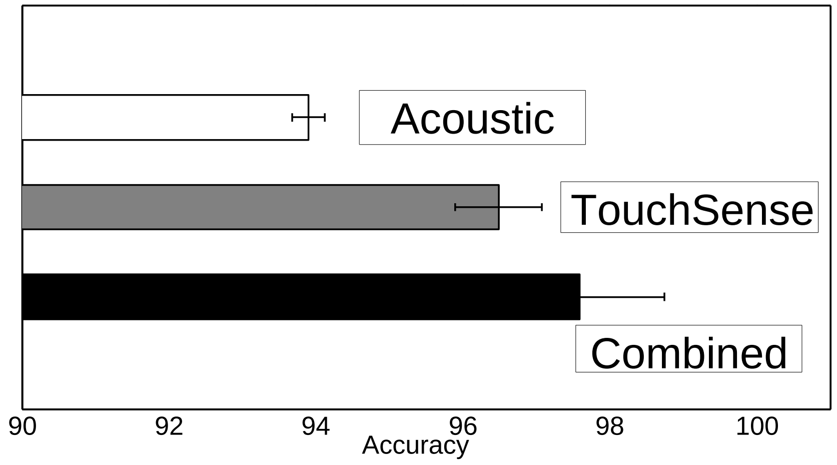


FIGURE 2.20: Comparison with an acoustic-based method.

audio signals acquired in both quiet and noisy settings. The gray curve in the figure is the acoustic signal and the red circles are the identified taps. Since it is much harder to separate taps from the noise, the acoustic method performs very poorly. We conclude that combining TouchSense with the acoustic method may be beneficial if one uses the acoustic features obtained in a quiet setting.

2.4 Related Work

Past research has looked at ways to enable touch-input on arbitrary surfaces. These approaches generally fall into three categories based on whether they require instru-

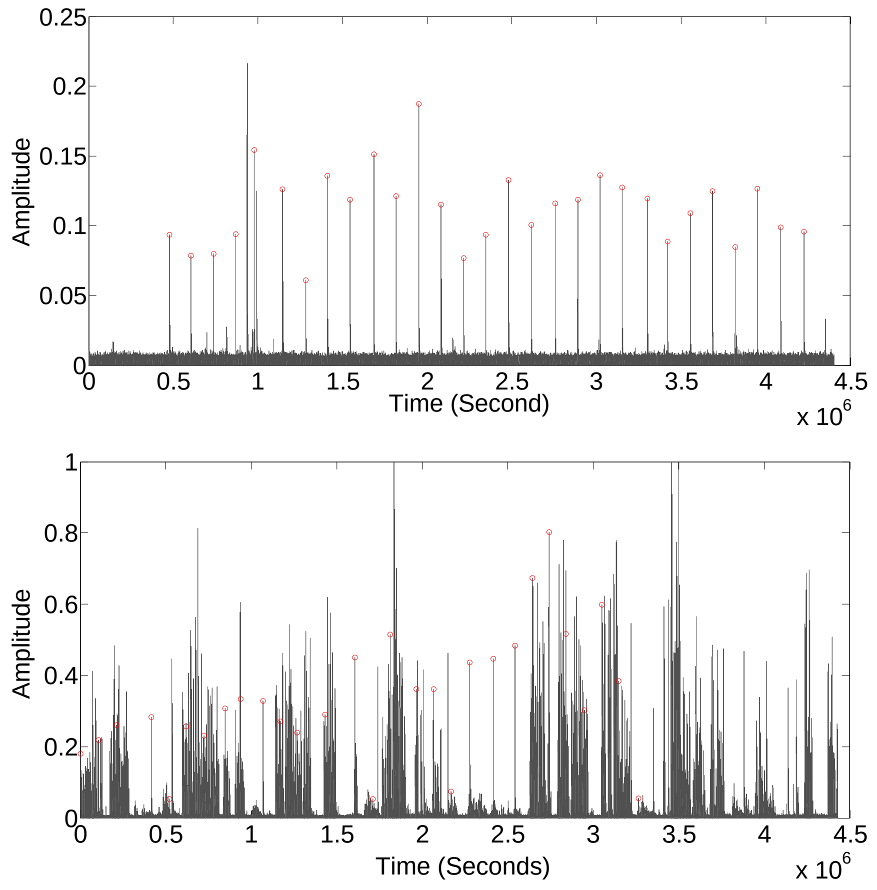


Figure 2.21: Taps in both quiet (top) and noisy (bottom) settings. The noise comes from users' conversation.

mentation of the environment: Vision-based, Acoustic and Electrical, and Sensor-based Techniques.

2.4.1 Vision-based Techniques

Projection-camera systems are a popular way of enabling input on surfaces, without requiring instrumentation. The Microsoft Surface and *Touchlight* use multiple cameras to detect contact with a table or wall Wilson (2004). Alternatively, Han's approach requires only a single camera by using the surface as an infrared waveguide and detecting when a user's finger frustrates the surface Han (2005). *SixthSense* uses a wearable projector-camera setup Mistry and Maes (2009), detecting when a user's hand interacts with a projected image. *SideSight* uses infrared LEDs and optical sensors to enable interaction at the sides of a mobile device Butler et al. (2008). Vision can also be used without infrared if configured to detect the user's skin tone Kane et al. (2009) or his shadows Cowan and Li (2011). The usage of depth cameras can enable curved surfaces for touch-input Wilson (2010). Though cameras can enable a variety of surfaces with touch-input detection, they involve expensive computation. Additionally, vision-based solutions can be sensitive to changing environments and require that they be mounted a distance away from the surface.

2.4.2 Acoustic and Electrical Techniques

Acoustic sensing overcomes some of the limitations of vision-based approaches by augmenting the surface with an audio sensor. Paradiso Paradiso et al. (2002) and Ishii Ishii et al. (1999) mounted multiple microphones on a surface and used time of flight to detect touch events. *Scratch Input* Harrison and Hudson (2008) and *TapSense* Harrison et al. (2011) use a classifier to distinguish between different types of touch events on a given surface, either from different materials or different finger interaction. *Skinput* detects touch input on the user's arm by using an armband of acoustic sensors Harrison et al. (2010). *Touché* adds touch-input to objects by using custom circuitry in their swept frequency capacitive sensing technique Sato et al. (2012). These approaches are effective but can require non-standard hardware. We have also experimented with acoustic sensing and found that combining it with motion sensing improves the accuracy of either technique alone. More importantly, in contrast with previous acoustic- and electrical-based work that only distinguishes between different types of interactions, TouchSense also attempts to locate the positions of interactions.

2.4.3 Sensor-based Techniques

Most relevant to *TouchSense* is the *TapPrints* project Miluzzo et al. (2012). Using the accelerometer and gyroscope, *TapPrints* detects where a user is tapping on a mobile phone or tablet. Using machine learning techniques, *TapPrints* demonstrates good detection accuracy. While *TapPrints* focuses on the security implications of using motion sensor data to snoop sensitive information such as passwords, *TouchSense* focuses on enabling touch-input detection on laptop screens.

2.5 Conclusion and Future Work

Anecdotal evidence suggests that toddlers are getting so used to touch screens on smartphones and tablets, that they are beginning to tap passive surfaces, such as laptop screens, TVs, and even books, and expecting results out of them. The TouchSense project was inspired by these observations. Our central goal is to inexpensively enhance passive surfaces with a sense of touch, even if at a much coarser granularity than real capacitive screens. As a first step, we develop a proof of concept using off-the-shelf smartphones taped to laptop displays. We find that the position of the tap on the laptop screen generates distinct motion fingerprints on the accelerometer and gyroscope of the phone. Results show that different kinds of taps and swipes can be recognized, while tap localization can be achieved at reasonable granularity, enabling simple gaming, scrolling, and icon selection applications.

Additional work is certainly necessary before TouchSense is ready for deployment – diversity induced by varying screen tilts, disparate phone hardware, and different phone positions, needs to be accommodated. Finer resolution is also needed to support a larger set of applications. Nonetheless, we believe that the techniques from this research bode promise. We see the viability of a future in which children books come instrumented with inertial sensors, facilitating new kinds of human-surface interactions.

3

Automatic Content Rating via Reaction Sensing

3.1 Introduction

Online content ratings serve as "quality indicators" to help a user make more informed decisions. While these ratings have been effective, we believe that there is room for improving the value and experience with ratings. Our observations are two-fold: (1) Today's ratings are most often a simple number, such as a "4 star" for a Netflix movie, a 87% red-tomato by Flixster, or simply 23 Likes for videos in YouTube. These numbers may be viewed as a *highly-lossy compression* of the viewer's experience, that often leaves the new user asking for more. (2) Eliciting a carefully considered rating from users is difficult, partly due to the lack of incentives. Providing a brief review can take up a good amount of user's time. Once a user has watched the video, she may not be willing to make this time investment. We envision that content rating systems of the future will require minimal user participation and yet provide rich, informative ratings. Figure 3.1 shows an example – a movie thumbnail could not only have a star rating, but also a tag-cloud of user reactions, and even short clips indexed by these reactions (such as, all scenes that were hilarious).

This paper makes an attempt to realize this vision through a system called *Pulse*. The opportunity arises from the growing number of sensors that are entering the mobile platform, especially smartphones and tablets. We hypothesize that when users watch a movie on these devices, a good fraction of their reactions leave a footprint on various sensing dimensions. For instance, if the user frequently turns her head and talks – detectable through the front facing camera and microphone – one could infer the user's lack of attention to that movie. Other kinds of inferences may arise from laughter detection via the microphone, the stillness of the device from the accelerometer, variations in orientation from gyroscope, fast forwarding of the movie, etc. Pulse learns the mapping between the sensed reactions and these ratings. Later, the knowledge of this mapping is applied to users to automatically compute their ratings, especially when they do not provide one. The sensed information is also used to create a tag-cloud of reactions, expected to offer a "break-up" of the different emotions evoked by the movie. If one wishes, she may also be able to watch a set of short clips that pertain to any of these emotions. Pulse can provide them since it logs user reactions for each segment, across many users. The result is like a customized trailer Jacob and Steglich (2010), one per user reaction.

The core ideas in Pulse may generalize to a variety of applications: (1) The timeline of a movie can be annotated with reaction labels (e.g., funny, intense, warm) so that viewers could jump ahead to desired segments. (2) The advertising industry may use Pulse to offer free or subsidized movies in exchange for more targeted ads. A user who reacts to a particular scene could be presented with corresponding ads. (3) It may be feasible to create an automatic highlights of a movie, perhaps consisting of all action scenes. (4) Finally, Pulse may offer educational value to film institutes and mass communication departments – students can use reaction logs as case studies from real-world users.

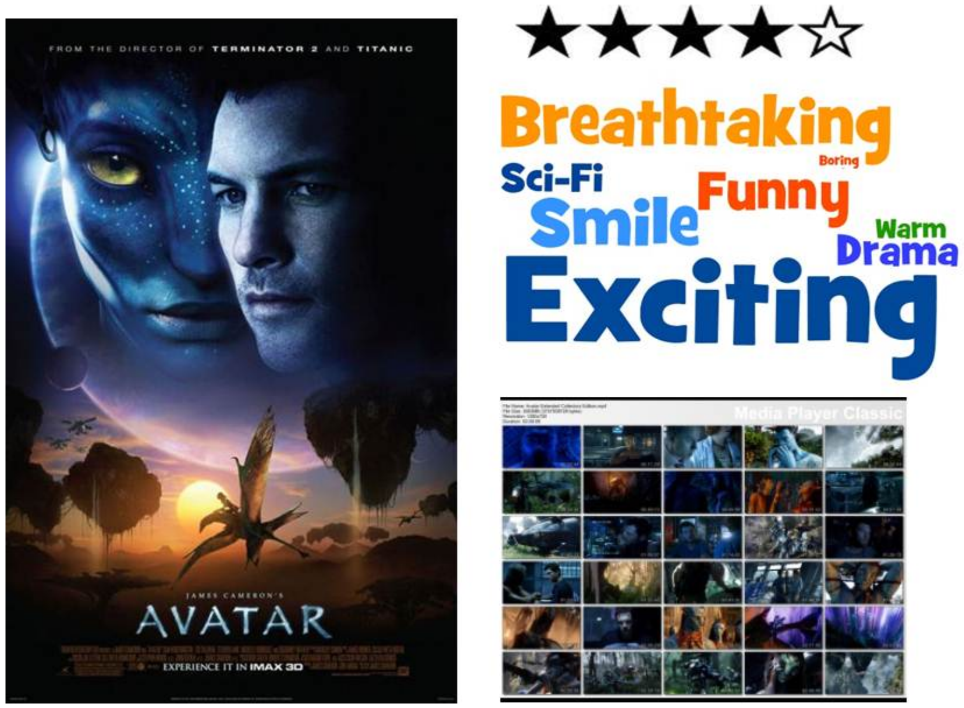


FIGURE 3.1: Envisioned movie ratings for the future – a conventional 5-star rating; a tag-cloud of user reactions; movie clips indexed by these reactions.

Of course, translating Pulse to reality, and enabling these applications, entails a number of challenges. The viewer's head pose, lip movement, and eye blinks need to be detected and monitored over time to infer reactions Cherubini et al. (2010). The user's voice needs to be separated from the sounds of the movie (which may be audible if the user is not wearing headphones), and classified as either laughter or speech. Patterns in accelerometers and gyroscopes need to be identified and translated to user focus or distractions. Finally, the function that translates reactions to ratings needs to be estimated through machine learning, and the learnt parameters used to generate semantic labels as a summary about the movie Paolucci et al. (2008); Teevan et al. (2009).

This paper incorporates these ideas into a Samsung tablet running the Android OS, and distributes these tablets to real users for evaluation. Results indicate that

Pulse's final ratings are consistently close to the user's ratings (mean gap of 0.46 on a 5 point scale), while the reaction tag-cloud reliably summarizes the dominant reactions. The highlights feature also extracted the appropriate segments, while the energy footprint remained small and tunable. A small-scale user study generated an enthusiastic response to Pulse.

The main contributions may be summarized as follows.

- **We identify an opportunity to automatically rate content at a few different granularities.** Our approach requires minimal user participation and harnesses multi-dimensional sensing available on modern tablets and smartphones.

- **We design a practical system, Pulse, that senses user reactions and translates them to an overall system rating.** In addition, we process the raw sensor information to produce rating information at variable granularities – a tag-cloud and a reaction-based highlight.

- **We develop Pulse on Android based Samsung Galaxy tablets and evaluate it with 11 volunteers, each of whom watched 4 to 6 movies**. Results show that the average gap between human and system ratings is 0.46 (on a 5 point scale). The tag-cloud exhibits similarity to the user's true reactions, thereby capturing reasonably, the user's overall experience.

The rest of the paper expands on each of these contributions, beginning with a high level overview, and followed by design, implementation, and evaluation.

3.2 System Overview

Pulse has been implemented on Android tablets and focuses specifically on movies and videos. Figure 3.2 envisions the high level architecture. This section briefly describes the three main modules, namely (1) Reaction Sensing and Feature Extraction (RSFE), (2) Collaborative Labeling and Rating (CLR), and (3) Energy Duty-Cycling (EDC).

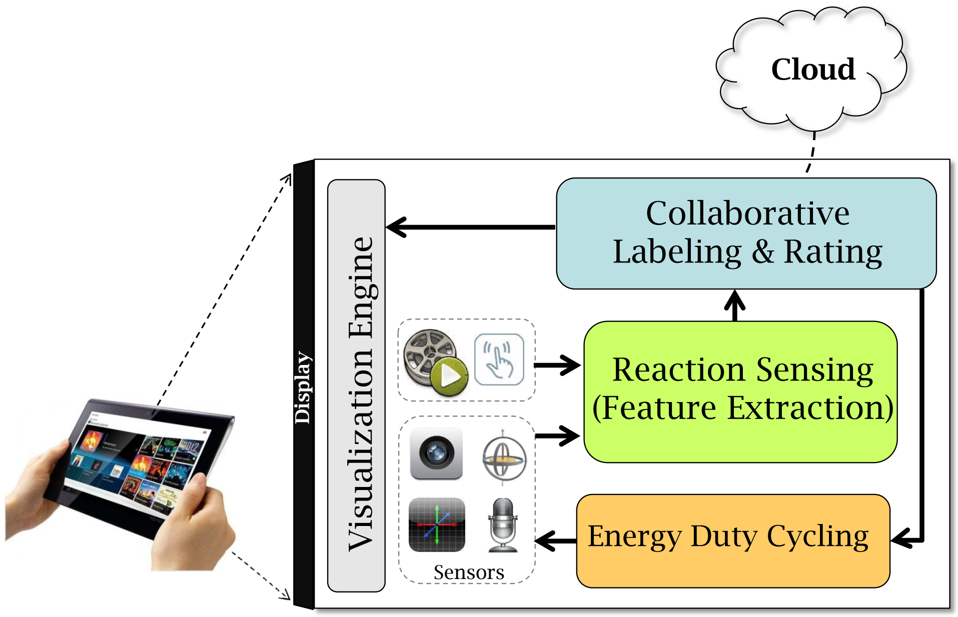

Cloud
Collaborative Labeling & Rating
Reaction Sensing (Feature Extraction)
Energy Duty Cycling
Visualization Engine
Display
Sensors

FIGURE 3.2: Architectural overview of Pulse.

(1) Reaction Sensing/Feature Extraction (RSFE)

When a user watches a video via the Pulse media player, all relevant sensors are activated, including the (front-facing) camera, microphone, accelerometer, gyroscope, and available location sensors. The raw sensor readings are forwarded to the RSFE module, which is tasked to distill out the features from them. These features inlude visual features collected through the front-facing camera, acoustic features extracted from embedded microphone, motion features (acceleration, rotation) captured by motion sensors, and control operations (e.g., fast forward) detected by the media player. RSFE collects all these features and forwards them to the collaborative labeling and rating (CLR) module.

Visual: The inputs from the front-facing camera are processed to first detect a face, and then track its movement over time. Since the user's head can temporarily move out of the tablet camera, the face is tracked even when it is partly visible. The visual sub-module extracts sophisticated features related to the face, eyes, and lips, and then feeds them to the learning module.

Acoustics: The acoustic sub-module is tasked to identify when the user is laughing or talking, which offer useful information about the corresponding segments in the movie. The challenge here is that users often use the tablet's in-built speakers while watching a movie, and this sound gets recorded back by the microphone. Pulse uses speech enhancement techniques to separate the user's voice from the movie sounds.

Motion: Motion sensors also provide insight into the user's reactions. Of interest are motions like stillness of the tablet (perhaps during an intense scene), or frequent jitters and random fluctuations (perhaps when the user's attention is less focused).

Control Operations: In addition to sensory inputs, Pulse also exploits how the user alters the natural playback of the movie. For instance, moving back the slider to a recent time-point may indicate stronger interest in a scene; forwarding the slider multiple times may suggest impatience. RSFE collects all these features and forwards them to the collaborative labeling and rating (CLR) module.

(2) Collaborative Labeling and Rating (CLR)

Content storage and streaming, especially with movies and videos, is moving towards the cloud based model. The ability to assimilate content from many cloud users naturally offer insights into behavior patterns of a collective user base Sarwar et al. (2001) – Netflix, Amazon, Hulu, are examples of service providers that leverage this approach to provide recommendation and personalization. Pulse is also positioned to benefit from access to the cloud. In particular, Pulse employs *collaborative filtering* methods where ratings are used across users to help improve accuracy. With more labeled data from users, Pulse will improve in its ability to learn and predict user ratings.

Sensing user reactions and exporting to the cloud raises privacy concerns Li et al. (2010), especially with face detection. However, we observe that none of the raw sensor readings need to be shared. Upon approval from the user, only ratings and semantic labels (or any subset of them with which the user is comfortable) can be exported. In the degenerate case, Pulse uploads the final star rating and discards the rest. This mimics today's systems, except that the rating will be determined automatically.

(3) Energy Duty-Cycling (EDC)

When the tablet is connected to a power-outlet, the EDC module is not necessary. In fact, we find that Pulse's energy consumption is marginal compared to the energy consumed by the tablet's display and CPU, while playing the movie. However, when running on smartphones, EDC's task is to minimize the energy consumption due to sensing. As mentioned earlier, the key idea is to sense each user during non-overlapping time segments, and then "stitch" the user reactions to form the overall rating. The evaluation section presents measurement results.

Figure 3.3 shows how the different sub-modules lead up to the final rating. The RSFE module processes the raw sensor readings and extracts features to feed to CLR.

The CLR module processes each (1 minute) segment of the movie to create a series of "*semantic labels*" as well as "*segment ratings*". Techniques such as collaborative filtering, Gaussian process regression (GPR), and support vector machines (SVM) are employed to address different types of challenges. Finally, the segment ratings are merged to yield the final "star rating" while the semantic labels are combined to create a tag-cloud. Thus, from the raw sensor values to the final star rating, Pulse distills information at various granularities to generate the final summary of the user's experience. We begin the technical discussion with the RSFE module.

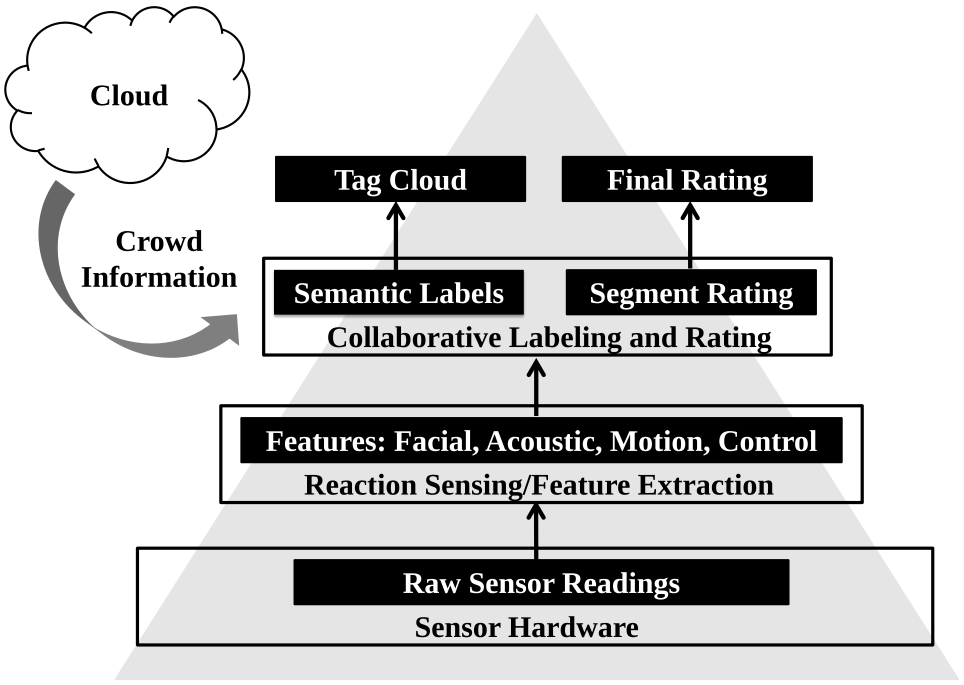

Cloud
Crowd Information
Tag Cloud
Final Rating
Semantic Labels
Segment Rating
Collaborative Labeling and Rating
Features: Facial, Acoustic, Motion, Control
Reaction Sensing/Feature Extraction
Raw Sensor Readings
Sensor Hardware

FIGURE 3.3: The RSFE and CLR modules distill raw sensor readings to a rating, tag-cloud, and video trailers

3.3 System Design: RSFE

We discuss the design of the Reaction Sensing and Feature Extraction module (RSFE).

3.3.1 Reaction Features: Visual

Pulse records visual information from the front camera to assess human reactions. Several prior efforts have attempted to achieve this using techniques involving face detection, eye tracking, and lip tracking Morris et al. (2002). However, our application presents a few unique challenges and opportunities compared to the traditional scenarios. First, the front facing camera on a mobile device usually does not capture the user's face from an ideal angle. In the case of our tablet, the top-mounted camera usually captures a tilted view of the face and eyes, requiring us to compensate for a rotational bias. Second, due to relative motion between the user and the tablet, the user's face may frequently move out of the camera view, either fully or partially. This derails contour matching methods, making continuous face detection difficult. Third, practical issues such as users wearing spectacles adds to the complexity. Fortunately, however, the field of view of the tablet is usually limited, making it easier to filter out unknown objects in the background, and extract the dominant user's face. Also, for any given user, particular head-poses are likely to repeat more than others (due to the user's head-motion patterns).

Pulse employs a combination of face detection, eye tracking, and lip tracking, using techniques from contour matching, speeded up robust feature (SURF) detection Bay et al. (2006), and frame-difference based blink detection algorithms Morris et al. (2002). The flow of operations is as follows:

1. Pulse continuously runs a contour matching algorithm on each frame for face detection.
2. If a face is detected, the system runs contour matching for eye detection as well as lip detection, and identifies the SURF image keypoints in the region of the

face. These image keypoints may be viewed as small regions of the face that maintains similar image properties across frames.

3. Now, if a full face is not detected, Pulse still tracks keypoints similar to previously detected SURF keypoints – this allows detecting and tracking a partial face, which occurs frequently in real life.

4. Pipelined with the face detection, Pulse runs an algorithm to perform blink-detection and eye-tracking. The difference in two consecutive video frames are analyzed to identify a blink. Essentially, if the pixels that change across consecutive frames form two nearly-symmetric ellipses, then the pixels are likely to be the blink. For eye-tracking, contour matching-based techniques fail when users are wearing spectacles – blink-detection is effective here. In other words, even if the eyes are blurred by the spectacles, the blinks can approximate the eye positions.

Figure 3.4 shows an intermediate output of the algorithm. Here Pulse detects the face through the tablet camera, detects the eyes using blink detection, and finally tracks the keypoints.

Pulse draws out the following features: face position, eye position, lip position, face size, eye size, lip size, relative eye and lip position to the entire face, and the variation of each over the duration of the movie. We believe these features reasonably capture some of the reaction footprints useful for ratings Lang et al. (1993).

3.3.2 Reaction Features: Acoustic

The Pulse video player activates the microphone and records ambient sounds while the user is watching the movie – this sound file is the input to our acoustic sensing sub-module. The key challenge is to separate the user's voice from the movie soundtrack, and then classify the user's voice as either laughter or speech. Since

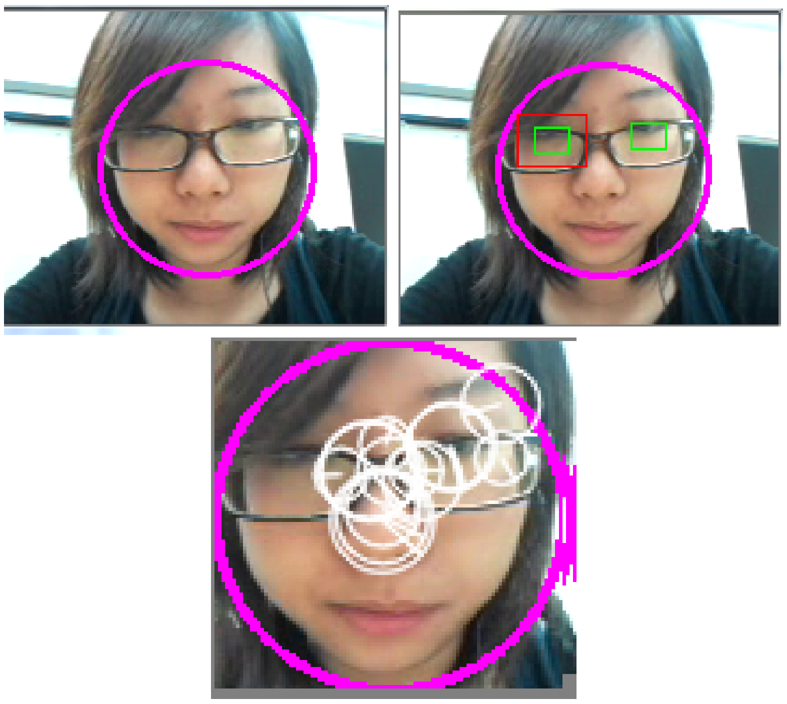


FIGURE 3.4: Visual sensing in Pulse: Face, eye, and blink detection for a user with spectacles.

the movie soundtrack played on the tablet's speakers can be loud, separation is not straightforward. We describe Pulse's approaches as follows.

Voice Detection

Given that the human voice exhibits a well-defined footprint on the frequency band (bounded by $4kHz$), Pulse's first approach was to extract this band using a low pass filter and then perform separation Sohn et al. (1999). However, the tablet already performs this filtering (to improve speech quality for phone calls). Figure 3.5 demonstrates this by comparing the Power Spectral Densities of the following: (1) the original movie soundtrack, (2) the sound of the movie recorded through the tablet microphone, and (3) the sound of the movie and human voice, recorded by the tablet microphone. Evidently, the recorded sounds drop sharply at around $4kHz$. At less than $4kHz$, the movie soundtrack with and without human voice are comparable,

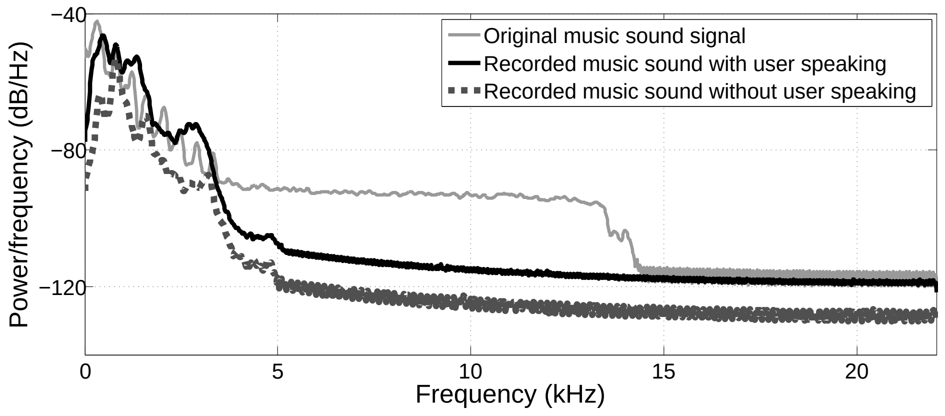


FIGURE 3.5: Comparing power spectral density – original and recorded soundtrack with human voice.

and therefore non-trivial to separate.

Pulse adopts two heuristic techniques to address the problem, namely (1) energy detection before and after speech enhancement and (2) per-frame spectral density comparison. We describe them here and show how they are applicable in different volume regimes.

(1) Energy Detection with Speech Enhancement:
Well established speech enhancement tools in literature can suppress noise and amplify the speech content in an acoustic signal. Pulse uses this to its advantage by measuring the (root mean square) signal energy before and after speech enhancement. For each frame, if the RMS energy diminishes considerably after speech enhancement, we regard this frame as noise. The simple intuition is that signals that contain speech will pass background noise suppression without being affected significantly; other noises should be reduced.

(2) Per-frame Spectral Density Comparison:

We observe that the power spectral density within $[0, 4]$ kHz is impacted by whether the user is speaking, laughing, or silent. In fact, the conversation from the movie can also impact this frequency regime. Figure 3.5 demonstrates an example case. Therefore, we compare the (per-frequency) amplitude of the recorded sound with the amplitude from the original soundtrack in each frame. If the amplitude of the recorded signal exceeds the soundtrack significantly, we deem that this video frame contains the user's voice.

Heuristic Selection based on Volume Regimes:
The two heuristics above perform differently depending on the volume of playback. Figure 3.6(a) reports their performance when the tablet volume is high – the dark horizontal lines in the top window represents the time windows when the user was actually speaking. The dark horizontal lines in the other two windows represent system detected speaking. Evidently, the second heuristic – per-frame spectral density comparison – exhibits better discriminative capabilities. This is because at high volumes, the human speech gets drowned by the movie soundtrack, and speech enhancement tools become unreliable. However, in low-volume cases, the soundtrack power is still low while the human voice is high, thereby allowing energy detection to identify the voice. Figure 3.6(b) shows this situation.

Laughter Detection

Pulse assumes that acoustic reactions during a movie are either speech or laughter – so, once human voice is detected, it needs to classified to one of the two categories. We use a support vector machine (SVM) and train it on the *Mel-Frequency Cepstral Coefficients* (MFCC) as the principle features. In sound processing, Mel-frequency cepstrum is a representation of the short-term power spectrum of a sound, and are commonly used in speech recognition McKinney and Breebaart (2003). To reduce

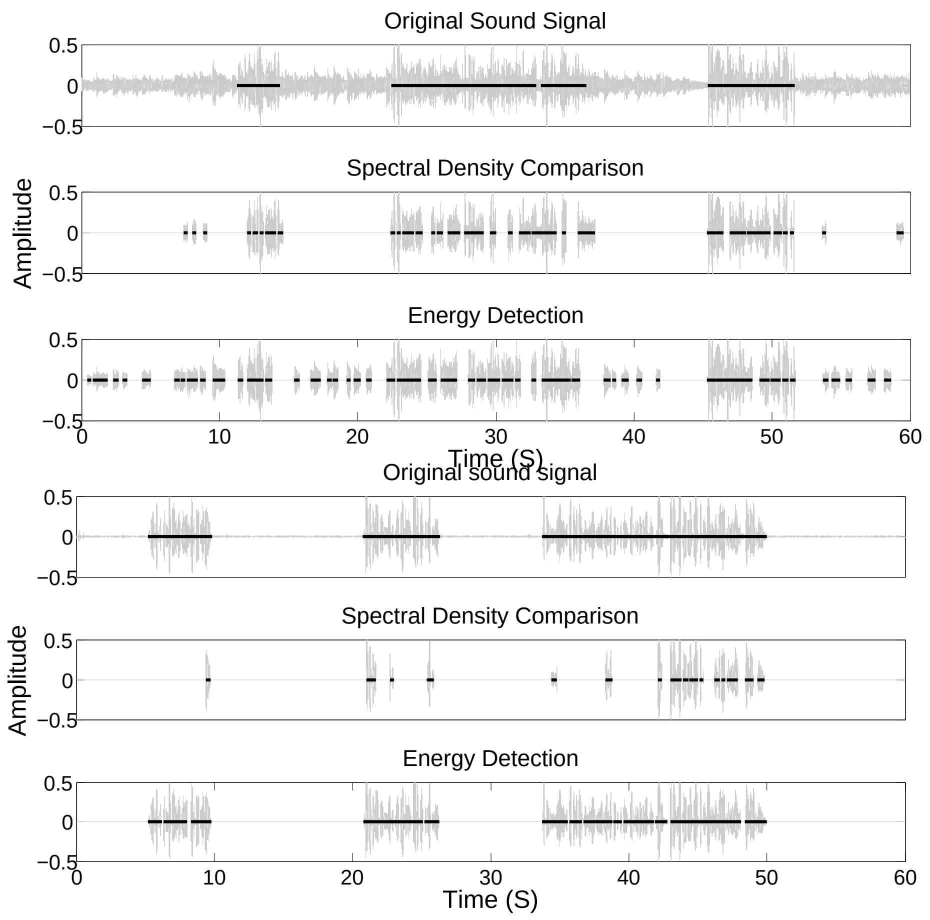


FIGURE 3.6: Comparison of voice detection. Top: High Volume; Bottom: Low Volume.

false positives, Pulse performs a simple outlier detection. If a frame is suspected as laughter, but the 4 preceding and following frames are not, then these outlier frames are eliminated. Figure 4.7 reports results showing high accuracy and few false positives.

3.3.3 Reaction Features: Motion

Accelerometer and gyroscope readings are also likely to contain information about the user's reactions. The mean of the sensor readings over the playback of the entire movie may capture the typical holding position/orientation of the device, while variations may be indicators of potential events. Figure 3.8 shows an example where

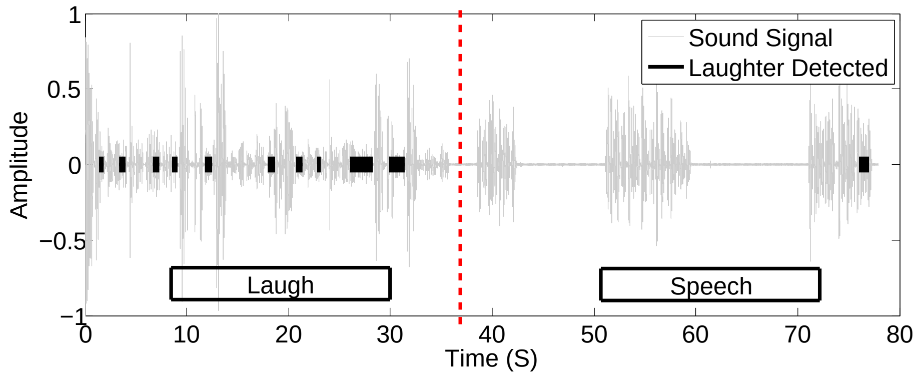


FIGURE 3.7: Discriminating laughter and speech from voice signals recorded by a tablet microphone.

mean and variance (after some smoothing) appear well correlated to when users change their ratings. It is possible that users are perform micro-movements at the beginning or end of logical segments, and the sensors seem to be capturing them. Pulse attempts to gain insights from these motion signatures.

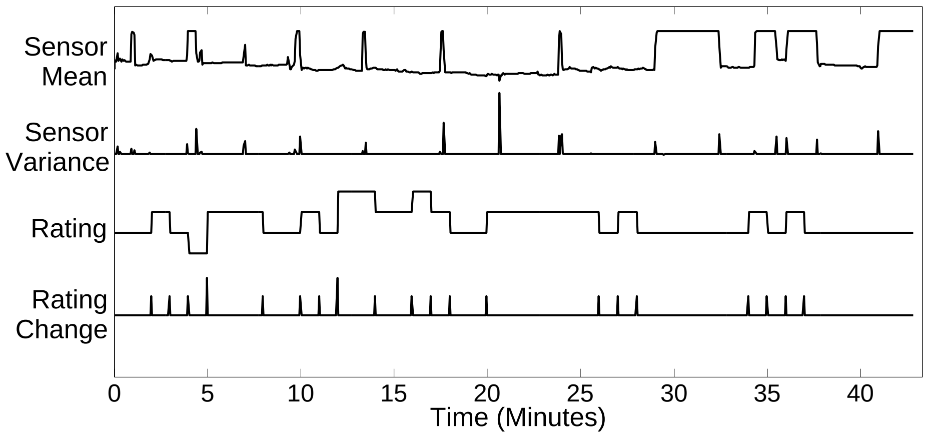


FIGURE 3.8: Motion correlates with rating changes.

3.3.4 Reaction Features: Touch Screen

Users tend to skip boring segments of a movie and, sometimes, may roll back to watch an interesting segment again. The information about how the user moved the slider can reveal the user's reactions for different movie segments. If Pulse observes a developing trend for skipping certain segments, or a trend in rolling back, the corresponding segments are assigned ratings proportionally (lower/higher).

3.4 System Design: CLR

This section describes the machine learning components in Pulse. The key goals are to model the sensed data and use the models to: (1) estimate segment ratings; (2) generate the final star rating from the segment ratings; (3) estimate semantic labels; (4) generate the tag-cloud from the semantic labels. To this end, Pulse requests multiple users to watch a movie, label different segments of the movie, and provide a final star rating as ground truth.

Ratings. *Segment ratings* are ratings for every short segment of the movie, necessary to compute the overall movie quality as well as to select enjoyable segments. A key challenge here is the ambiguity in how reaction features map to segment ratings. Laughter in a comedy movie may be a positive reaction, while laughter in a horror movie may mean the opposite. Some users may get excited and fidget in an intense scene, while others may watch it motionless. Pulse employs *Collaborative Filtering* and *Gaussian Process Regression* (GPR) to cope with such ambiguities (detailed later). To convert segment ratings to the *final rating*, Pulse uses a weighted averaging function.

Labels. *Semantic labels* are English labels assigned to each segment of the movie. CLR generates two types of such labels – *reaction labels* and *perception labels*. (1) *Reaction labels* are direct outcomes of reaction sensing, reflecting on the viewer's raw behavior while watching the movie (e.g., laugh, smile, focused, distracted, nervous, etc.). (2) *Perception labels* reflect on subtle emotions evoked by the corresponding scenes (e.g., funny, exciting, warm, etc.) While identifying reaction labels is straightforward, identifying perception labels is more challenging. Pulse employs a semi-supervised learning method combining Collaborative Filtering and SVM to predict perception labels. Then, Pulse aggregates all the predicted labels, counts their relative occurrences, and develops the tag-cloud description of the movie. The efficacy of prediction is quantified through cross-validation. The following subsection elaborates on the methodology and techniques.

3.4.1 Modeling and Prediction Challenges

We begin by describing our experimentation methodology, which will help explain the challenges we faced during modeling and prediction. Thereafter, we describe the solutions.

Experiment Methodology

To obtain labeled user data, we conducted a formative user study. We initially recruited 11 volunteers (4 females), aged 24–28. We provided volunteers with Android-based Samsung tablets pre-loaded with 6 movies (3 comedies, 2 dramas, and 1 horror), and asked them to watch only those movies they have not watched earlier. The volunteers were required to watch the movie using our Pulse video player, which activates and records sensor readings during playback. Because we needed data from natural settings, we let users watch movies at any place and time they chose; most users took the tablets home. We also provided a software tool that allowed users to rate the movie soon after they watched it. This tool scans through the movie minute by minute (like fast-forwarding) and allows volunteers to rate segments on a scale from 1 to 5 (1 being "did not like", 5 being "liked"). Volunteers also labeled some segments with "perception" labels, indicating how they perceived the attributes of that segment. The perception labels were picked from a pre-defined set – some examples are "funny", "scary", "intense". Finally, volunteers were asked to provide a final (star) rating for the overall movie, again on a scale of 1 to 5.

Challenges

Pulse's goal is to model user behavior from the collected labeled data, and use this model to predict (1) segment ratings, (2) perception labels, and (3) the final (star) rating for each movie. Note that this is a high bar for Pulse – predicting human judgment, minute by minute, is quite difficult. The difficulty gets exacerbated by 3 types of heterogeneities, described next.

(1) Heterogeneity in users behavior: Some users watch movies attentively, while others are more fidgety. Such diversities are common among users, and particularly so when observed through the sensing dimensions. As a result, a naive universal model trained from a crowd of users is likely to fail in capturing useful behavioral signatures for any specific user. In fact, such a model may actually contain little information since the ambiguity from diverse user-behaviors may mask (or cancel out) all useful patterns. For example, if half of the users hold their devices still when they are watching a movie intensely, while the other half happen to hold their devices still when they feel bored, a generic model learned from all this information will not be able to use "stillness" as a discriminator between intensity and boredom. Thus, a good one-fit-all model may not exist. To confirm this, we created a regression model for estimating segment ratings using all available labeled data. Figure 3.9 plots the cross-validation results for the leave-one-video-out test, comparing the model's estimated segment ratings vs. the actual user ratings. The results show that the model's estimates fail to track the actual user ratings, and mostly converges on the *mean* rating of training data.

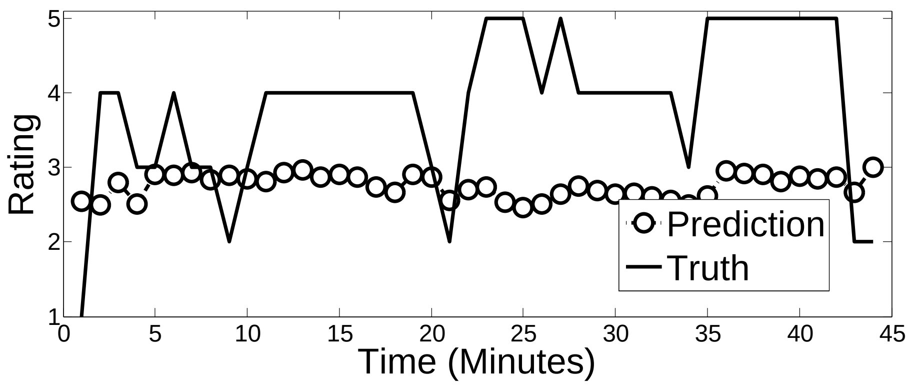


FIGURE 3.9: Poor results from regression when attempting to learn a model applicable to all users.

(2) Heterogeneity in environment factors: Even for the same user, her "sensed behavior" may differ from time to time due to different environmental factors. For instance, the behavior associated with watching a movie in the office may be substantially different from the behavior during a commute, which is again different from when at home. Figure 3.10 shows the gyroscope sensor data distribution from the same user watching two movies. The distribution clearly varies even for the same user, indicating that the way the user holds the device may not always be similar.

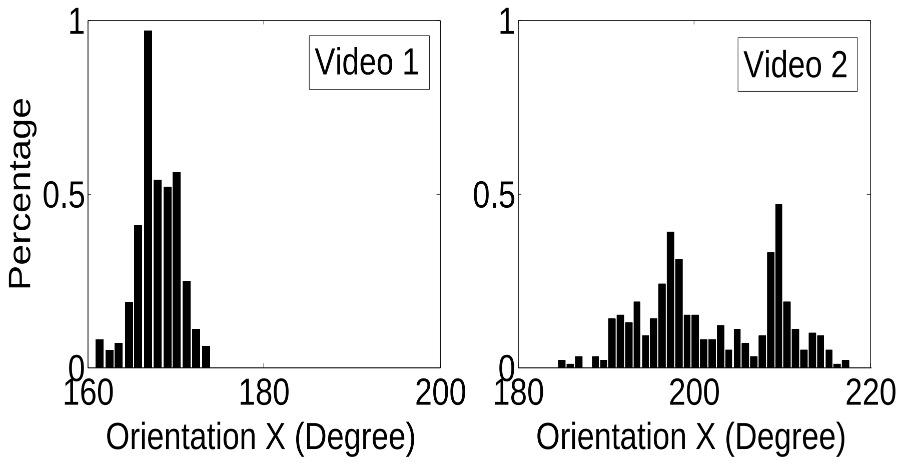


FIGURE 3.10: Orientation sensor data distribution

(3) Heterogeneity in user tastes: Finally, users may have different tastes, resulting in different ratings/labels given to the same movie scene. Some scenes may appear hilarious to one, and may not be so to another. Figure 3.11 shows the deviation in ratings given to the same scenes by 5 different users. Clearly, there is dissimilarity in taste.

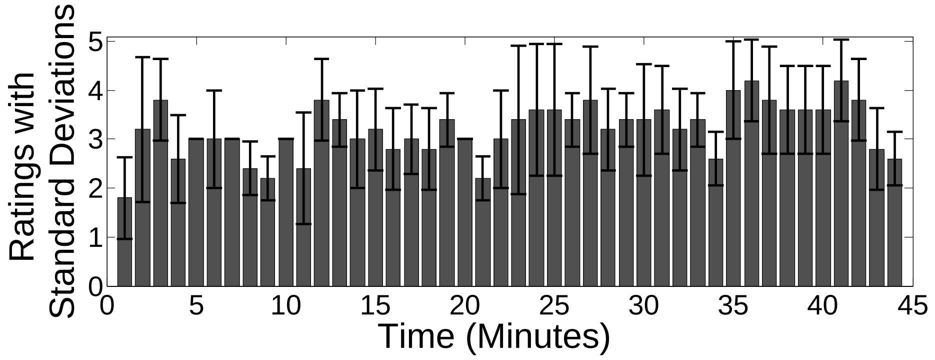


FIGURE 3.11: High Std. Dev. in ratings across users.

3.4.2 Pulse's Learning Approach

The heterogeneities described above highlight the core challenge – we need to develop a model that will capture the unique taste/behavior of a user under different environments. One (brute force) approach would be to train a series of per-user models, each tailored to a specific viewing environment and for a specific genre of a movie. However, it is nearly impossible to enumerate all such environments, and worse, the user would have to provide ratings and labels for all combinations of movie genres and environments. This is impractical.

Pulse overcomes this problem by basing its solution on the following intuition. Although users exhibit heterogeneity overall, their reactions to certain parts of the movie are remarkably similar (or coherent). Therefore, we analyze the collective behavior of multiple users to extract only these coherent signals – i.e., segments for which most users exhibit agreement in their reactions. Similarly, for perception labels, Pulse also learns from segments on which most users agree. Collaborative filtering techniques Sarwar et al. (2001) provide the ability to draw out these segments of somewhat "universal" agreement. We designed two separate semi-supervised learning methods – one for segment ratings and another for perception labels. For segment ratings, we combine *collaborative filtering* with *Gaussian Process Regression* (GPR). For perception labels, we combine *collaborative filtering* with support vector machines (SVM).

When a new user watches a movie, Pulse uses the sensed data from *only* the "universally agreed" segments to train a customized model, which is then used to predict the ratings and labels of the rest of the user's segments. In other words, Pulse bootstraps using ratings that are agreeable in general, and by learning how the new user's sensing data correlates with these agreeable ratings, Pulse learns the user's "idiosyncrasies" (which is the most difficult aspects of automatic content rating). Now, with knowledge of these idiosyncrasies, Pulse can "extrapolate" to other segments of the movie (that users did not agree upon), and predict the ratings for this specific user Ouyang and Li (2012). Figure 3.12 illustrates our method. From the ratings of users A, B, and C, Pulse learns that minute 1 is intense (I) and minute 5 is boring (B). Then, when user D watches the movie, his sensor readings during the first and the fifth minutes are used as the training data to create a personalized model. This model is then used to predict the 2^{nd}, 3^{rd}, and 4^{th} segment ratings.

Figure 3.13 shows that Pulse's approach works reasonably well, with Pulse's estimated ratings tracking the actual user ratings. We will discuss additional results – precision, recall, and fallout – on segment rating and label prediction in the evaluation section.

Besides coping with inherent heterogeneity of users, we observed additional chal-

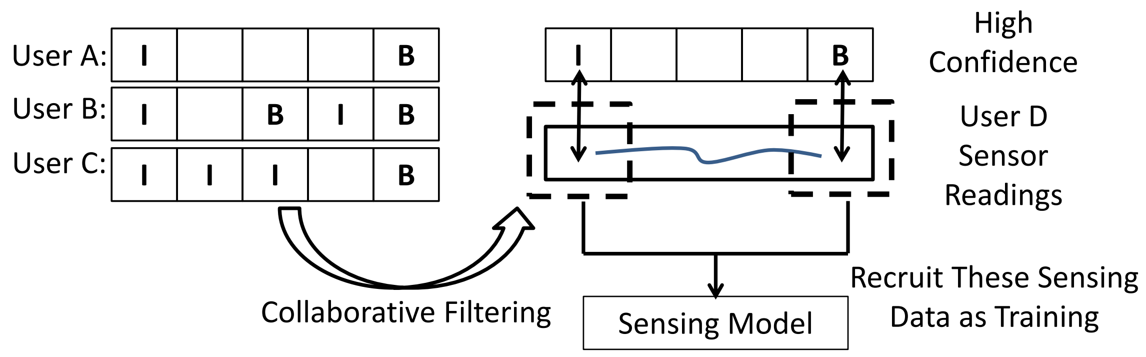


FIGURE 3.12: Pulse learns a custom model from high-confidence segments.

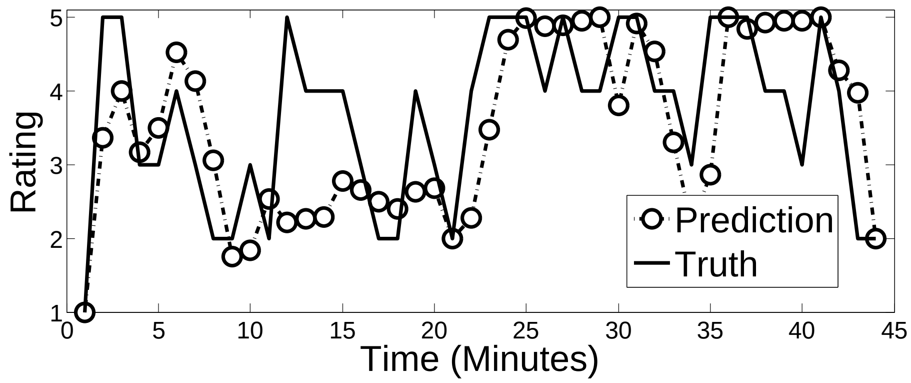


FIGURE 3.13: Collaborative filtering and GPR improve prediction – circles are the Pulse's predictions

lenges emerging from (1) *time-scale of ratings* and (2) *sparsity of labels*. The first problem arises from the mismatch between the time-scale of sensed reactions (a laughter lasts a few seconds) and the time-scale of human ratings (one for each minute). As a result, the human labels we obtain are not necessary labeling the specific sensor pattern, but rather an aggregate of useful and useless patterns over the entire minute. This naturally raises the difficulty for learning the appropriate signatures. The situation is similar for labels. It is unclear exactly which part within the 1-minute segment was labeled as hilarious, since the entire minute may include both "hilarious" and "non-hilarious" sensor signals. To cope with this, we assume that each 3 second window in the sensing data has the label of the corresponding minute. In our prediction, once Pulse yields a rating/label for each 3-second entry, we aggregate them back to the minute granularity, allowing us to compute both prediction accuracy and false positives.

The second problem relates to how labels gathered in each movie are sparse (volunteers did not label each segment, but opted to label only scenes that seemed worthy of labeling). As a result, we found 65.9% of the segments unlabeled. This warrants careful adjustment of the SVM parameters – otherwise SVM may classify all segments as "none of the valid labels", and appear to achieve high accuracy (since much of the data indeed has no valid label). Precisely recognizing and classifying the few minutes of the labeled segments, from thousands of minutes of recordings, is an ambitious task. We designed under these constraints while ensuring we do not over-fit – the next section reports on the results.

3.5 Evaluation

In this section, we demonstrate the feasibility of predicting (1) segment ratings, (2) final ratings, and (3) semantic labels, through multi-dimensional sensing.

3.5.1 Metrics

We adopt three measures commonly used in information retrieval, namely, *Precision*, *Recall*, and *Fallout*. These metrics essentially are methods to compute overlaps (and non-overlaps) between two sets of items. Consider the case of segment rating. One set is the set of movie segments that the user truly enjoyed (i.e., segments manually rated as 4 or 5) – we call this the *Human Selected* set. The other set contains segments that Pulse believes the user enjoyed – called the *Pulse Selected* set. Then, the 3 metrics can be defined as:

$$Precision = \frac{|\{\text{Human Selected} \cap \text{Pulse Selected}\}|}{|\{\text{Pulse Selected}\}|}$$

$$Recall = \frac{|\{\text{Human Selected} \cap \text{Pulse Selected}\}|}{|\{\text{Human Selected}\}|}$$

$$Fall - out = \frac{|\{\text{Non-Relevant} \cap \text{Pulse Selected}\}|}{|\{\text{Non-Relevant}\}|}$$

Higher values of precision and recall are better; the converse for fallout. These metrics apply for the semantic labels as well, where one set is provided by humans, and the other generated by Pulse.

3.5.2 Summary of Results

1. **Segment Rating:** Predicted segment ratings closely follow users' segment ratings, with an average error of 0.7 on a 5-point scale. This is a 40% improvement over estimations based on only distribution or collaborative filtering (the improvement is more pronounced in terms of *recall*). More importantly, Pulse is able to capture enjoyable segments with an average precision of 71%, an average recall of 63%, and a minor fallout of 9%.

2. **Final Rating:** Pulse's overall star rating demonstrates an average error of 0.46 in the 5 point scale.

3. **Label quality:** On average, Pulse covers 45% of the perception labels with a minor average fallout of 4%. We also observe an order of magnitude improvement over a pure SVM-based approach and modest gains over collaborative filtering. The reaction labels capture the audience's reactions well. The tag-clouds were received with enthusiasm (a qualitative feedback). Detailed results follow.

3.5.3 Performance of Segment Rating

To quantify Pulse's accuracy at predicting segment ratings on the 5-point scale, we compared the results of four prediction algorithms: Random, Collaborative Average, Collaborative High Confidence and Pulse. Random predicts the scores randomly. Collaborative average uses the average of all user's segment ratings as the prediction. Collaborative High Confidence assigns the average user's score for only those segments that were given consistent ratings by most users and assigns the scale's average score (3 in our case) to other segments. Finally, Pulse, uses collaborative filtering results as a starting point and exploits sensing data as described in previous sections to provide a more accurate prediction. Figure 3.14 plots the mean prediction errors of the four algorithms as black bars. Pulse outperforms other algorithms, achieving 0.7 mean error. This result shows the value that sensing data may bring to automatic segment rating.

Often, people are interested in finding good movies (rated above 3) and may not care whether a movie is rated 1.4 or 2.4. This observation can be used to optimize Pulse further by treating ratings from 1 to 3 as the same. In doing so, we are essentially reducing the resolution of expressing that a movie is not worth watching to a single score. The mean prediction errors of the four algorithms, when the rating score is reduced to a 3-point scale, are shown as grey bars in Figure 3.14. Here, Pulse again outperforms other algorithms, achieving 0.25 mean error.

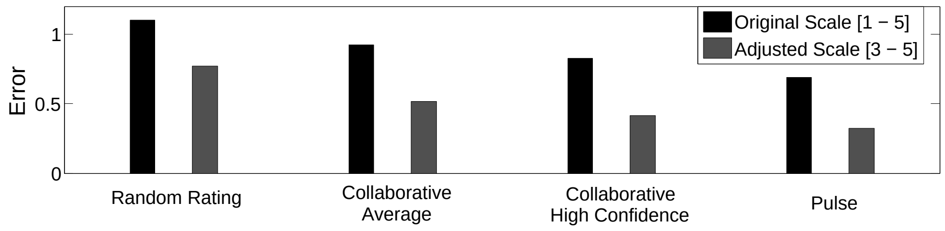


FIGURE 3.14: Mean Pulse segment rating error.

Segment Selection.

Pulse selects the "enjoyable" segments, i.e., ones rated as 4 and above, to generate a highlights of the movie. To evaluate whether Pulse's selection matches with the user's, we evaluate Pulse using precision, recall, and fallout. Figure 3.15 shows the average precision ranges from 57% to 80%, an average recall of 63%, and a minor fallout usually less than 10%. Pulse performed well on 2 comedy and 2 dramas, corresponding to the first four bars in each group. The performance was weaker in the remaining 2 movies (1 comedy and 1 horror).

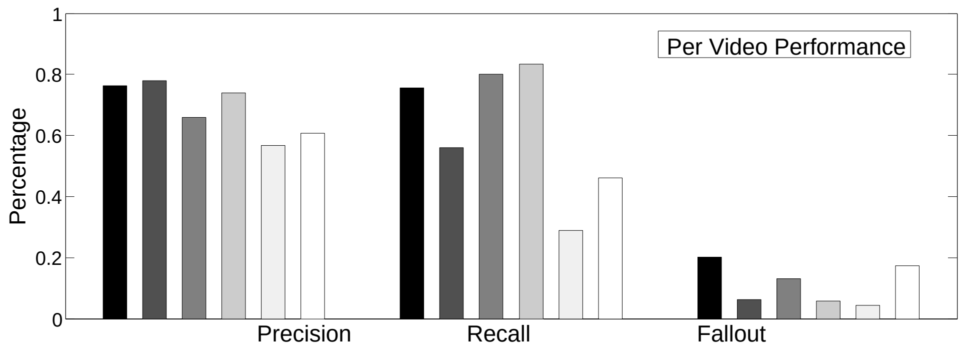


FIGURE 3.15: Precision/Recall/Fallout per video

Figure 3.16 shows the average per-user performance. Except for one outlier (the second user), the precision is above 50% with all recalls above 50%. Fallout ranges from 0 to 19%. Given the sparse labels we have, the accuracy we believe is reasonable – on average Pulse creates less than one false positive every time it includes five true positives. One may observe that the second user might be characterized as "picky" – the low precision, reasonable recall, and small fallout, suggest that she rarely assigns high scores. We note that all the above selections are personalized; a good segment for one user may be boring to another and Pulse can identify these inter-personal differences.

Figure 3.17 illustrates the break-up of contributions from collaborative filtering and sensing. The four bars show the number of true positives, total number of

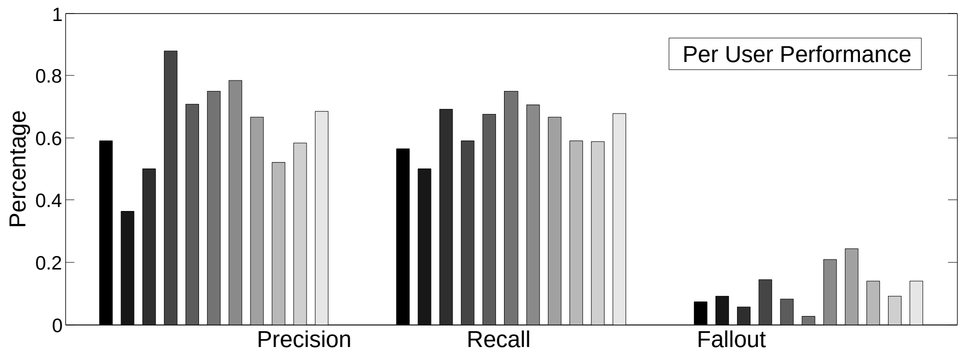


FIGURE 3.16: Precision/Recall/Fallout per user

positive samples (segments with ratings of 4 or 5), false positives, and total number of negative samples (segments with rating 1 to 3), respectively. As the figure illustrates, the contribution from sensing is substantial.

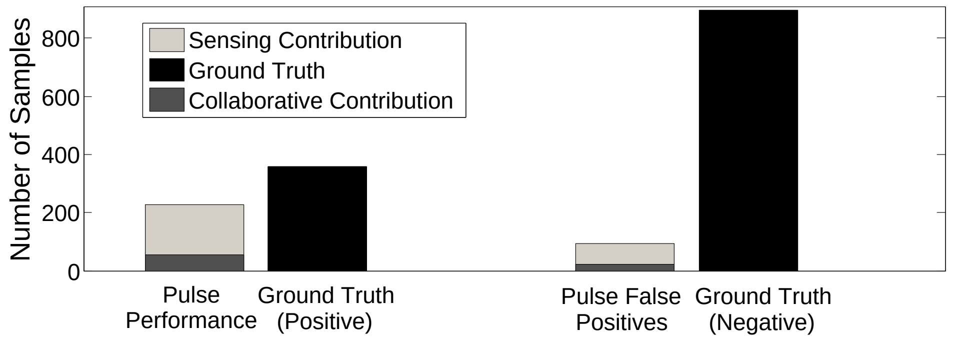
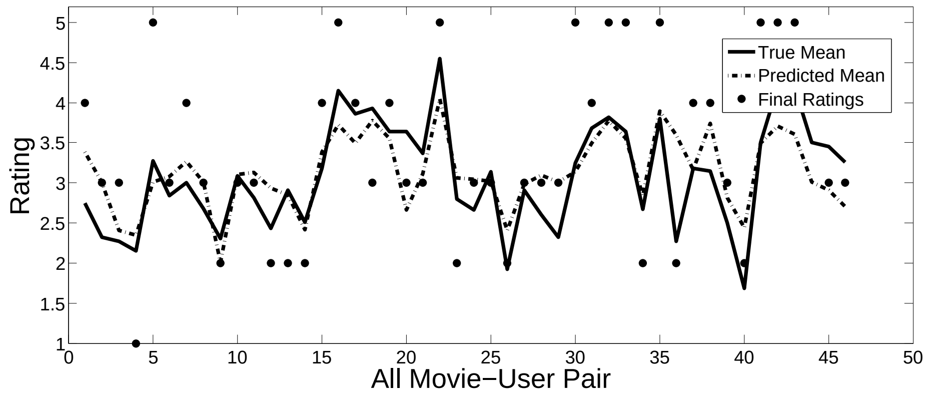


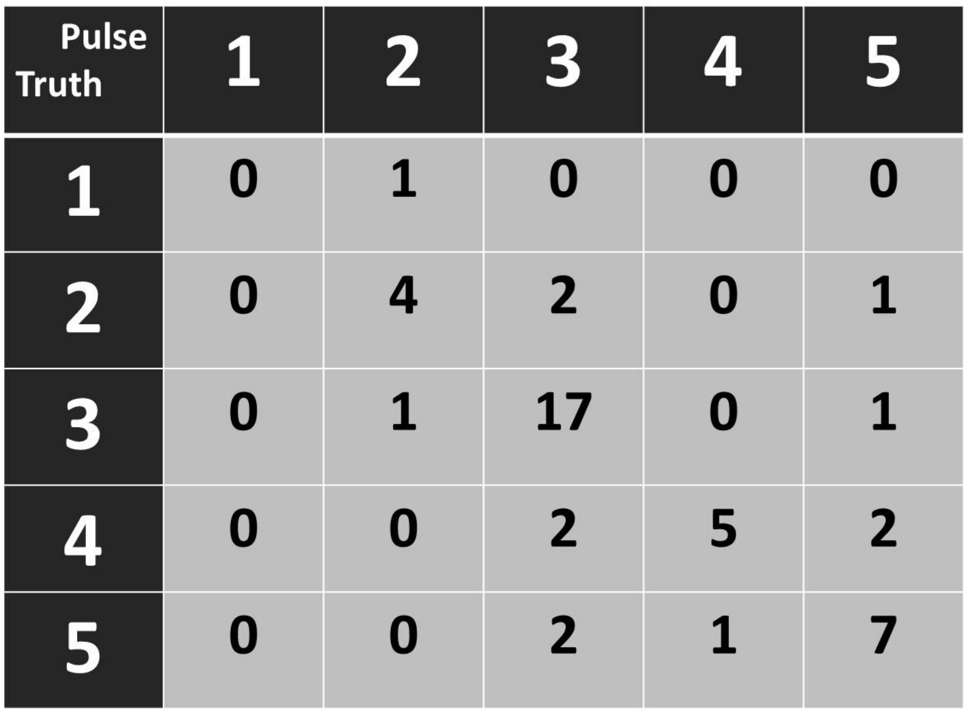
FIGURE 3.17: Break-up of contributions.

3.5.4 Performance of Final "Star" Rating

Pulse generates final ratings by thresholding the mean scores of per-minute segment ratings. This thresholding function essentially tries to learn how users map their mean scores for each segment to the final score. Figure 3.18(a) shows the means of predicted and true segment ratings (dashed and solid lines), as well as the true final rating. Pulse tracks the user's mean rating well. We observed that users are often conservative in rating the movie segments, but more generous with the final rating. Figure 3.18(b) shows Pulse's prediction of final ratings using a confusion matrix.

Higher values concentrate around the diagonal, indicating desired performance. We are aware that we may have over-fitted our data with the thresholds, and intend to investigate this more carefully in future.





Truth \ Pulse	1	2	3	4	5
1	0	1	0	0	0
2	0	4	2	0	1
3	0	1	17	0	1
4	0	0	2	5	2
5	0	0	2	1	7

FIGURE 3.18: (a) Mean segment ratings and corresponding users' final ratings. (b) Confusion matrix.

3.5.5 Performance of Label Quality

Pulse associates semantic labels to each movie segment and eventually generates a tag cloud for the entire movie. This section evaluates the efficacy to predict labels. Recall that our semantic labels consist of *reaction labels* and *perception labels*; we evaluate them separately. As a ground truth, we intend to know the user's reactions and perceptions at every time point. However, providing this information (while the user is watching the movie) would have interfered with their viewing experience. Therefore, we asked the users to provide the perception labels to each movie segment after watching the movie. For reaction labels, we recruited two volunteers to view the video recording from the tablet camera, and label the viewer's reactions – we used this as ground truth Ekman and Friesen (2003).

Reaction Label Quality

Reaction labels capture users' actions while watching a movie (e.g., laugh, smile, etc.). The (limited) vocabulary is shown in Table 3.1. Figure 3.19 shows the comparison between Pulse's prediction and the ground truth – the gray portion is ground truth while the black dots denote when Pulse detects the corresponding labels. Although Pulse sometimes mislabels on a per-second granularity, the general time frame and weight of each label is reasonably well captured.

Table 3.1: Label Vocabulary

Label Category	Vocabulary
Perception	Funny, Intense, Warm
Reaction	Laugh, Smile, Shaking, Focused, Distracted, Speaking

Perception Label Quality

Perception labels represent a viewer's perception of each movie segment (e.g., funny, warm, intense). Figure 3.20 shows the performance of perception label prediction for

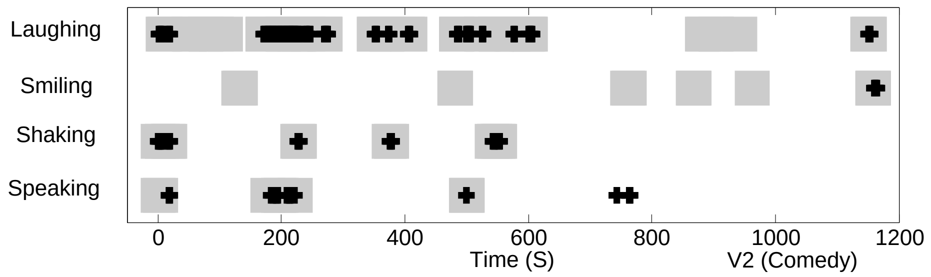


FIGURE 3.19: Reaction label prediction vs. groundtruth

each label, averaged over all users. These labels are hard to predict because (1) their corresponding behaviors can be subtle and implicit; (2) users provided these labels for few segments. Our performance is proportionally weaker: average precision is 50%, recall is 35%; however, fallout is satisfactory: 4%.

Figure 3.21 compares the performance between pure-SVM (using cross validation), collaborative filtering, and Pulse. From top to bottom, the figures show precision, recall, and fallout, respectively. Pulse demonstrates substantial improvement over SVM alone, but is comparable to collaborative filtering.

3.5.6 Tag Cloud and User Feedback

We attempted to visually summaries the results of Pulse using a tag cloud similar to Figure 3.1. The terms used within the tag-cloud combine perception and reaction labels, each weighted by its normalized occurrence frequency. We informally asked users who watched the episode (using Pulse) to comment on this tag-cloud. The feedback was resonantly enthusiastic, with comments like “very cool”, and “certainly useful information with zero extra burden”. Some users correctly pointed out that “a richer tag set is needed”.

3.5.7 Power Consumption

We measured the power consumption of Pulse on the Samsung tablets and Nexus S smartphones, using the Monsoon Power monitor. Figure 3.22 compares Pulse's performance with conventional media players – with no active sensors. Given the high playback and display power on tablets, Pulse-based sensing adds only 16% more energy. The energy burden is higher on smartphones, and would call for duty cycling the sensors, perhaps to only sense the decisive segments. We leave this to future work.

3.6 Related Work

Pulse builds on work that falls roughly in two areas: activity inferencing and multimedia annotation. We discuss some of the related work here.

Activity Inference: The large number of sensors on mobile devices have been leveraged as a rich sensing platform. Accelerometers are useful beyond motion Bao et al. (2004); microphones often effective in detecting environments Lu et al. (2009), and user's reaction Kennedy and Ellis (2004); front-facing cameras are valuable towards face/eye tracking in real-time video streams. Combined with machine learning and inferencing, these platforms are lending themselves to intent and context recognition Luther et al. (2008) The future is poised for more activities along these directions. Of course, such forms of continuous sensing causes substantial power drain. Existing proposals include offloading to the cloud Cuervo et al. (2010) or duty cycling techniques Wang et al. (2009). In future, efforts such as Little Rock could offload sensing to DSP chips, allowing the CPU to sleep Priyantha et al. (2011).

Multimedia Annotation: A powerful technique to annotating multimedia is to aggregate sensor data across multiple devices as a way of supersampling Honicky et al. (2008). TagSense is one example, using sensor data from multiple devices, to annotate images Qin (2011). Pulse uses a similar approach, but asynchronously aggregated across users. Recommender systems often annotate items using a set of known attributes – this maintains calibration across users, while capturing diversity in opinions Yu et al. (2009). Though these results tend to yield diverse results, they are resource intensive, require lots of time Rosenfeld and Morville (1998). We hope to address some of these issues with a sensor based approach, available free in today's devices.

3.7 Conclusion

Advances in personal sensing and machine learning are empowering machines to better understand human behavior. This paper guides this opportunity into an application that automatically rates content on behalf of human users. The core idea is to leverage device sensors, such as cameras, microphones, accelerometers, and gyroscopes, to sense qualitative human reactions while she is watching a video;learn how these qualitative reactions translate to a quantitative value; and visualize these learnings in an easy-to-read format. Thus, when using our system, a movie automatically gets tagged not only by a conventional star rating, but also with a tag-cloud of user reactions, as well as highlights of the movie for different emotions.

At this stage, Pulse is still a prototype with many limitations. On the technical side, the current label vocabulary is still limited. We intend to explore additional optimizations in machine learning to improve performance, while taking advantage of more sensors that enter the tablet platform. On the social side, pulse may raise privacy concerns especially for exporting information to the cloud. Though we do not have a clear solution to this problem yet, Pulse should certainly place most of its functionalities locally on the user's device and potentially only needs to upload ratings in the end. Different methods for data fusion can also help anonymize the data.

With these limitations, we still believe there is value in building a sensing-based automatic rating system. With the universe of content growing at a rapid pace, the need for associating meta data to content will become increasingly relevant. Pulse is an early attempt towards this goal, with direct applications in recommendations systems and information retrieval Cantador et al. (2008).

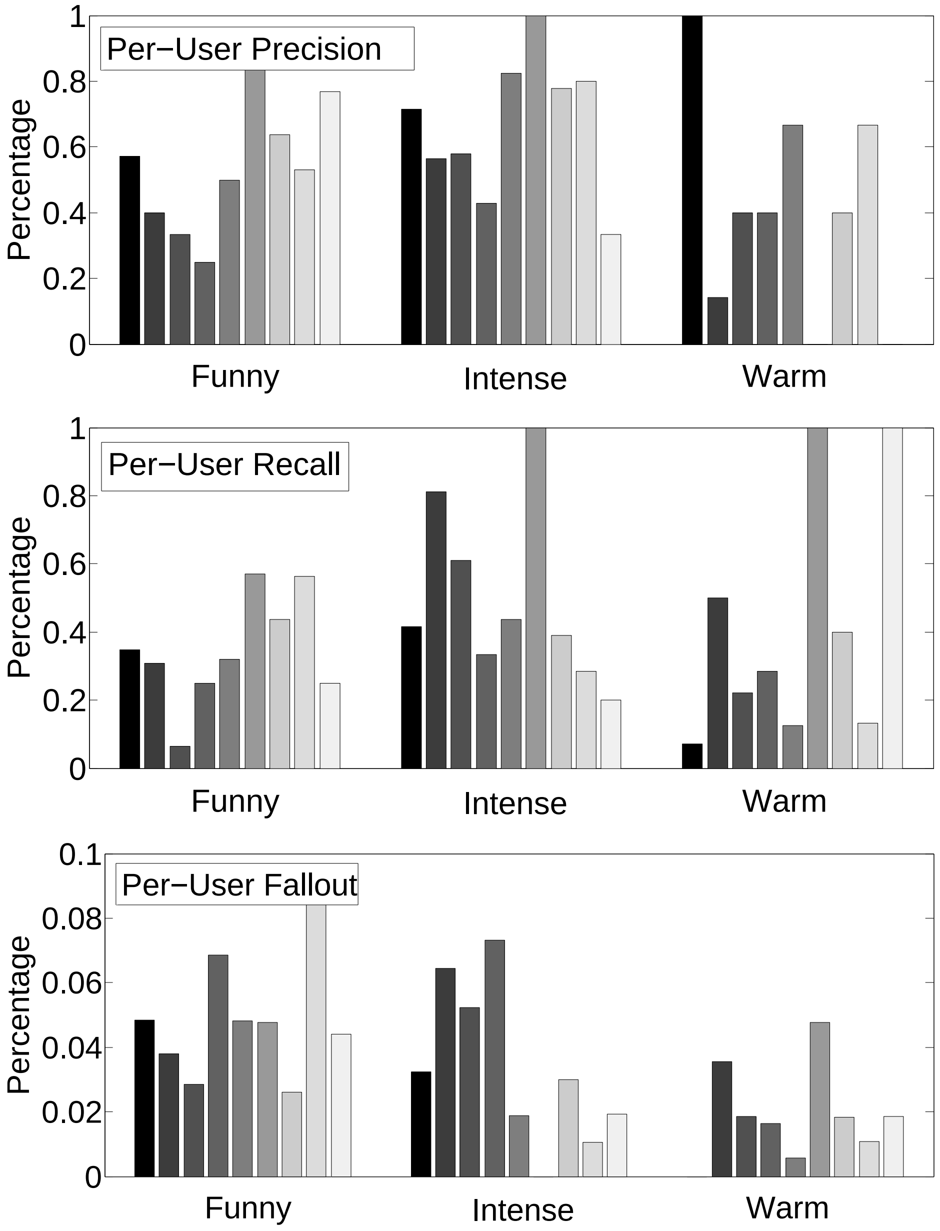


FIGURE 3.20: Performance for each label (averaged across users).

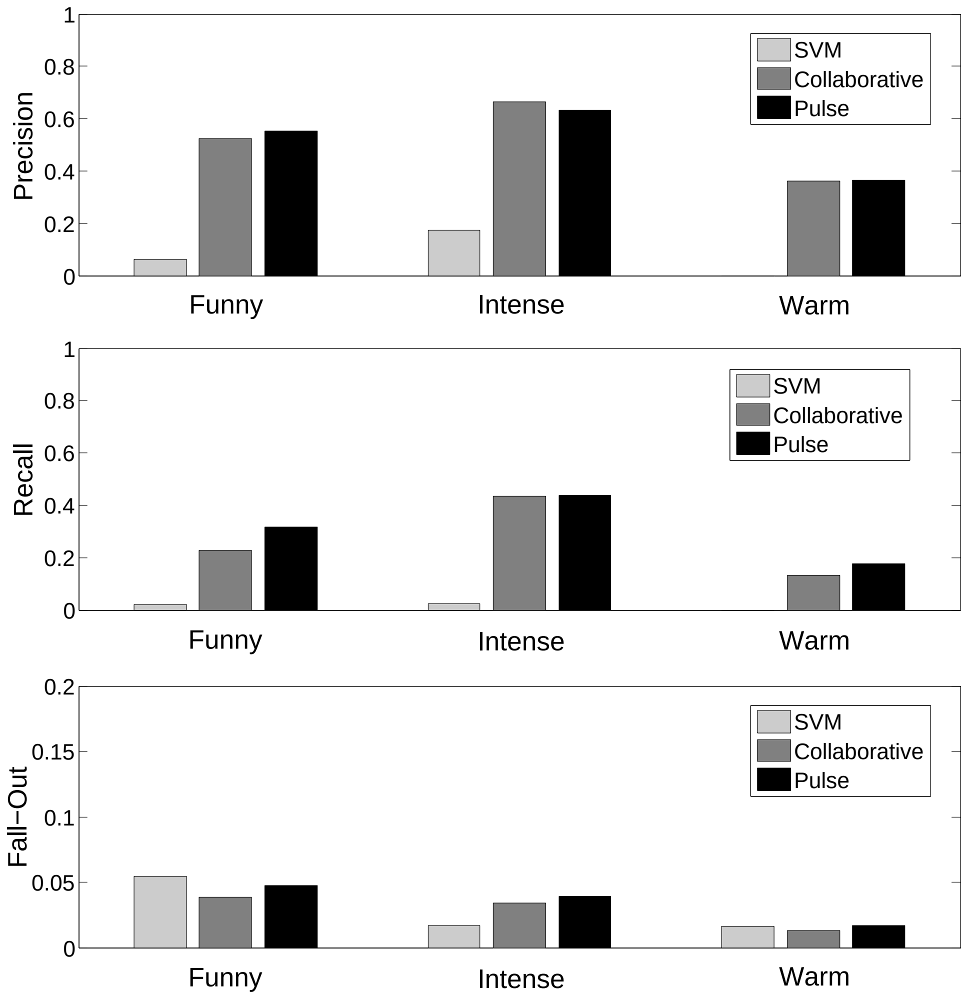


FIGURE 3.21: Performance comparison between SVM, collaborative filtering and our method (Pulse).

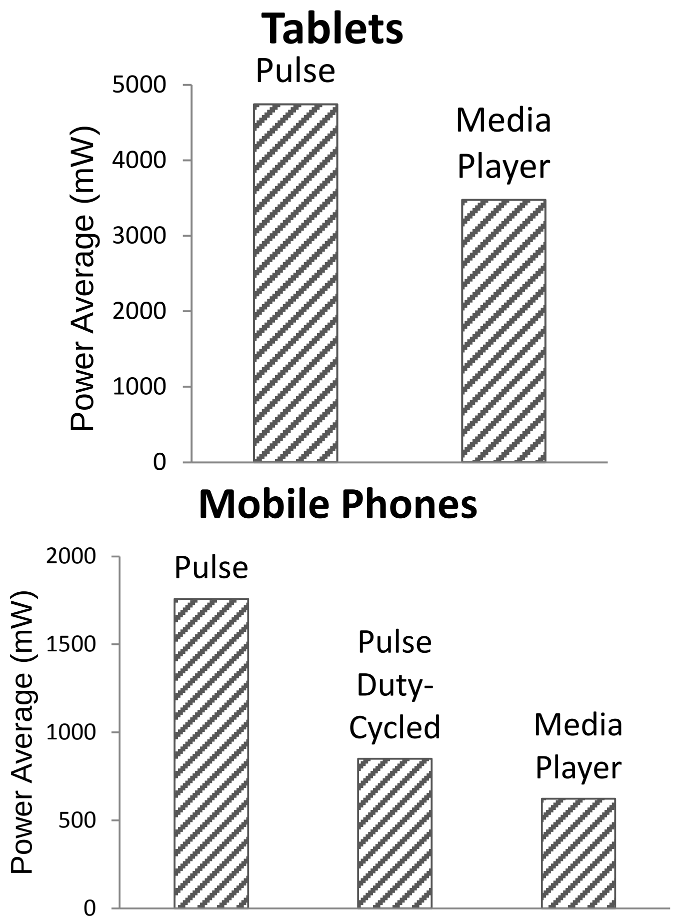


Figure 3.22: Power consumption comparison.

4 MoVi: Mobile Phone based Video Highlights via Collaborative Sensing

4.1 Introduction

The inclusion of multiple sensors on a mobile phone is changing its role from a simple communication device to a life-centric sensor. Similar trends are influencing other personal gadgets such as the iPods, palm-tops, flip-cameras, and wearable devices. Together, these sensors are beginning to "absorb" a high-resolution view of the events unfolding around us. For example, users are frequently taking geo-tagged pictures and videos Gaonkar et al. (2008); Torniai et al. (2007), measuring their carbon footprint Dada et al. (????), monitoring diets Reddy et al. (2007), creating audio journals and tracking road traffic Mohan et al. (2008); Lu et al. (2009). With time, these devices are anticipated to funnel in an explosive amount of information, resulting in what has been called as an information overload. Distilling the relevant content from this overload of information, and summarizing it to the end user, will be a prominent challenge of the future. While this challenge calls for a long-term research effort, as a first step, we narrow its scope to a specific application with a clearly defined goal.

We ask, assuming that people in a social gathering are carrying smart phones, *can the phones be harnessed to collaboratively create a video highlights of the occasion.* An automatic video highlights could be viewed as a distilled representation of the social occasion, useful to answer questions like “what happened at the party?” The ability to answer such a question may have applications in travel blogging, journalism, emergency response, and distributed surveillance.

This paper makes an attempt to design a Mobile Phone based Video Highlights system (MoVi). Spatially nearby phones collaboratively sense their ambience, looking for event-triggers that suggest a potentially “interesting” moment. For example, an outburst of laughter could be an acoustic trigger. Many people turning towards the wedding speech – detected from the correlated compass orientations of nearby phones – can be another example. Among phones that detect a trigger, the one with the “best quality” view of the event is shortlisted. At the end of the party, the individual recordings from different phones are correlated over time, and “stitched” into a single video highlights of the occasion. If done well, such a system could reduce the burden of manually editing a full-length video. Moreover, some events are often unrecorded in a social occasion, perhaps because no one remembered to take a video, or the designated videographer was not present at that instant. MoVi could be an assistive solution for improved social event coverage[1]. The video highlights created by MoVi can be used to cover the missed moments in manually taken videos.

A natural concern is: *phones are often inside pockets and may not be useful for recording events.* While this is certainly the case today, a variety of wearable mobile devices are already entering the commercial market Berry et al. (2007). Phone sen-

[1] This bears similarity to spatial coverage in sensor networks, except that physical space is now replaced by a space of social-events, that must be covered by multiple sensing dimensions.

sors may blend into clothing and jewelry (necklaces, wrist watches, shirt buttons), exposing the camera and microphones to the surroundings. Further, smart homes of the future may allow for sensor-assisted cameras on walls, and on other objects in a room. A variety of urban sensing applications is already beginning to exploit these possibilities Miluzzo et al. (2008); Mistry. (2009). MoVi can leverage them too.

Translating this vision into a practical system entails a range of challenges. Phones need to be grouped by social contexts before they can collaboratively sense the ambience. The multi-sensory data from the ambience needs to be scavenged for potential triggers; some of the triggers need to be correlated among multiple phones in the same group. Once a recordable event (and the phones located around it) is identified, the phone with the best view should ideally be chosen.

While addressing all these challenges is non-trivial, the availability of multiple sensing dimensions offers fresh opportunities. Moreover, high-bandwidth wireless access to nearby clouds/servers permits the outsourcing of CPU-intensive tasks Cuervo et al. (2010). MoVi attempts to make use of these resources to realize the end-goal of collaborative video recording. Although some simplifying assumptions are made along the way, the overall system achieves its goal reasonably well. In our experiments in real social gatherings, 5 users were instrumented with iPod Nanos (taped to their shirt pockets) and Nokia N95 mobile phones clipped to their belts. The iPods video-recorded the events continuously, while the phones sensed the ambience through the available sensors. The videos and sensed data from each user were transmitted offline to the central MoVi server.

The server is used to mine the sensed data, correlate them across different users, select the best views, and extract the duration over which a logical event is likely to have happened. Capturing the logical start and end of the event is desirable, otherwise, the video-clip may only capture a laugh and not the (previous) joke that may have induced it. Once all the video-clips have been shortlisted, they are sorted in time, and "stitched" into an automatic video highlights of the occasion. For a baseline comparison, we used a manually-created video highlights; multiple users were asked to view the full length iPod videos, and mark out events that they believe are worth highlighting. The union of all events (marked by different users) were also stitched into a highlights. We observe considerable temporal overlap in the manual and MoVi-created highlights (the highlights are 15 *minutes* while the full length videos are around 1.5 *hours*). Moreover, end users responded positively about the results, suggesting the need (and value) for further research in this direction of automatic event coverage and information distillation.

The rest of the paper is organized as follows. The overall system architecture is proposed in Section 4.2, and the individual design components are presented in Section 4.3. Section 4.4 evaluates the system across multiple real-life and mock social settings, followed by user-surveys and exit-interviews. Section 4.5 discusses the cross-disciplinary related work for MoVi. We discuss the limitations of the proposed system and future work in Section 4.6. The paper ends with a conclusion in Section 4.7.

4.2 System Overview

Figure 4.1 shows the envisioned client/server architecture for MoVi. We briefly describe the general, high level operations and present the details in the next sections. The descriptions are deliberately anchored to a specific scenario – a social party – only to provide a realistic context to the technical discussions. We believe that the core system can be tailored to other scenarios as well.

In general, MoVi assumes that people are wearing a camera and are carrying

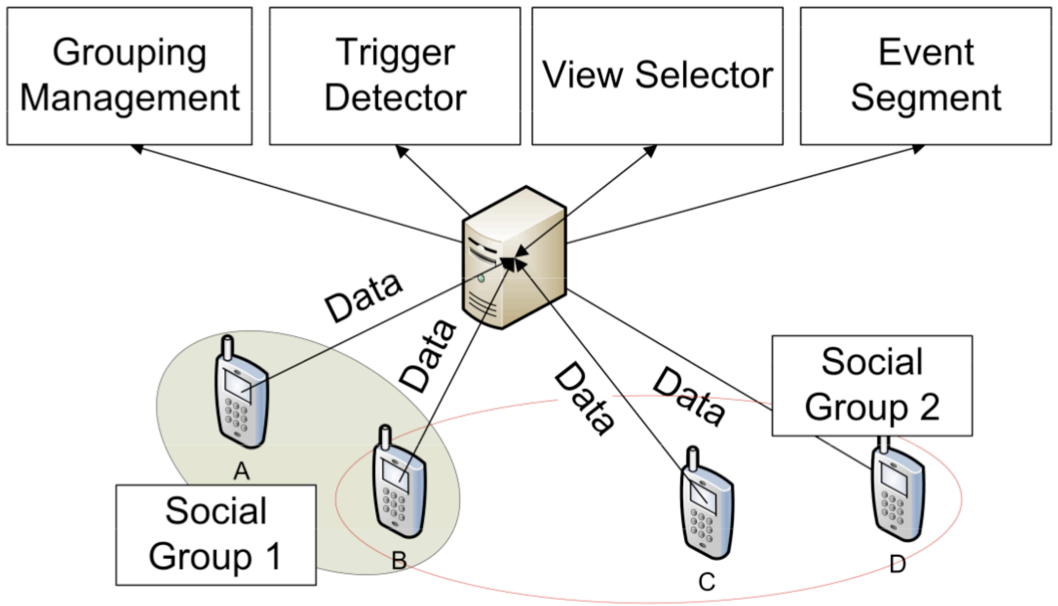


FIGURE 4.1: The MoVi architecture.

sensor-equipped mobile devices such as smart phones. The camera can be a separate device attached on a shirt-button or spectacles, or could even be part of the wearable phone (like a pocket-pen, necklace, or wrist watch Virpioja et al. (2007)). In our case, an iPod Nano is taped onto the shirt pocket, and the phone is clipped to a belt or held in the hand. Continuous video from the iPod and sensor data from the phone are sent to the MoVi server offline.

At the MoVi server, a *Group Management* module analyzes the sensed data to compute social groupings among phones. The idea of grouping facilitates collaborative-inferring of social events; only members of the same social group should collaborate for event identification. If real time operations were feasible, the Group Management module could also load-balance among the phones to save energy. Each phone could turn off some sensors and be triggered by the server only when certain events are underway. We are unable to support this sophistication in this paper – optimizing energy consumption and duty-cycling is part of our future work. A *Trigger Detection* module scans the sensed data from different social groups to recognize potentially interesting events. Once an event is suspected, the data is correlated with the data from other phones in that same group.

Confirmed of an event, the *View Selector* module surveys the viewing quality of different phones in that group, and recruits the one that is "best". Finally, given the best video view, the *Event Segmentation* module is responsible for extracting the appropriate segment of the video, that fully captures the event. The short, time-stamped video segments are finally correlated over time, and stitched into the video highlights.

Challenges

The different modules in MoVi entail distinct research challenges. We briefly state them here and visit them individually in the next section.

(1) The Group Management module needs to partition the set of mobile devices based on the social context they are associated to. A social zone could be a gathering around an ice-cream corner, a group of children playing a video game, or people on the dance floor. The primary challenges are in identifying these zones, mapping phones to at least one zone, and updating these groups in response to human movement. Importantly, these social groups are not necessarily spatial – two persons in physical proximity may be engaged in different conversations in adjacent dinner tables.

(2) The Event Detection module faces the challenge of recognizing events that are socially "interesting", and hence, worth video recording. This is difficult not only because the notion of "interesting" is subjective, but also because the space of events is large. To be detected, interesting events need to provide explicit clues detectable by the sensors. Therefore, our goal is to develop a rule-book with which (multi-modal) sensor measurements can be classified as "interesting". As the first step towards developing a rule book, we intend to choose rules shared by different events. Our proposed heuristics aim to capture a set of intuitive events (such as laughter, people watching TV, people turning towards a speaker, etc.) that one may believe to be socially interesting. Details about event detection will be discussed in Section 4.3.2.

(3) The View Selection module chooses the phone that presents the best view of the event. The notion of "best view" is again subjective, however, some of the obviously poor views need to be eliminated. The challenge lies in designing heuristics that can achieve reliable elimination (such as ones with less light, vibration, or camera obstructions), and choose a good candidate from the ones remaining. Details regarding our heuristics will be provided in Section 4.3.3.

(4) The Event Segmentation module receives a time-stamped event-trigger, and scans through the sensor measurements to identify the logical start and end of that event. Each social event is likely to have an unique/complex projection over the different sensing dimensions. Identifying or learning this projection pattern is a challenge.

MoVi attempts to address these individual challenges by drawing from existing ideas, and combining them with some new opportunities. The challenges are certainly complex, and this system is by no means a mature solution to generating automated highlights. Instead it may be viewed as an early effort to explore the increasingly relevant research space. The overall design and implementation captures some of the inherent opportunities in collaborative, multi-modal sensing, but also exposes unanticipated pitfalls. The evaluation results are limited to a few social occasions, and our ongoing work is focused on far greater testing and refinement. Nevertheless, the reported experiments are real and the results adequately promising to justify the larger effort. In this spirit, we describe the system design and implementation next, followed by evaluation results in Section 4.4.

4.3 System Design and Basic Results

This section discusses the four main modules in MoVi. Where suitable, the design choices are accompanied with measurements and basic results. The measurements/results are drawn from three different testing environments. (1) A set of students gathering in the university lab on a weekend to watch movies, play video games, and perform other fun activities. (2) A research group visiting the Duke SmartHome for a guided-tour. The SmartHome is a residence-laboratory showcasing a variety of research prototypes and latest consumer electronics. (3) A Thanksgiving dinner party at a faculty's house, attended by the research group members and their friends.

4.3.1 Social Group Identification

Inferring social events requires collaboration among phones that belong to the same social context. To this end, the scattered phones in a party need to be grouped socially. Interestingly, physical collocation may not be the perfect solution. Two people in adjacent dinner tables (with their backs turned to each other) may be in physical proximity, but still belong to different social conversations (this scenario can be generalized to people engaging in different activities in the same social gathering). Thus people should not video-record just based on spatial interpretation of a social event. In reality, a complex notion of "same social context" unites these phones into a group – MoVi tries to roughly capture this by exploiting multiple dimensions of sensing. For instance, people seated around a table may be facing the same object in the center of the table (e.g., a flower vase), while people near the TV may have a similar acoustic ambience. The group management module correlates both the visual and acoustic ambience of phones to deduce social groups. We begin with the description of the acoustic methods.

(1) Acoustic Grouping

Two sub-techniques are used for acoustic grouping, namely, ringtone and ambient-sound grouping.

Grouping through Ringtone. To begin with an approximate grouping, the *MoVi* server chooses a random phone to play a short high-frequency ring-tone (similar to a wireless beacon) periodically. The ring-tone should ideally be outside the audible frequency range, so that it is not interfered by human voices and also not annoying to people. With Nokia N95 phones, we were able to play narrow-bandwidth tones at the edge of the audible range and use it with almost-inaudible amplitude [2]. The single-sided amplitude spectrum of the ringtone is shown in Figure 4.2. The target is to make the ringtone exist only on $3500Hz$. This frequency is high enough to avoid being interfered by indoor noises.

Phones in the same acoustic vicinity are expected to hear the ringtone[3]. To detect which phones overheard this ringtone, the MoVi server generates a frequency-domain

[2] Audible range differs for different individuals. Our choice of frequency, $3500Hz$, was limited by hardware. However, with new devices such as the iPhone, it is now possible to generate and play sounds at much higher frequencies.

[3] We avoid bluetooth based grouping because the acoustic signals are better tailored to demarcate the context of human conversations while bluetooth range may not reflect the social partition among people. However, in certain extremely noisy places, bluetooth can be used to simplify the implementation.

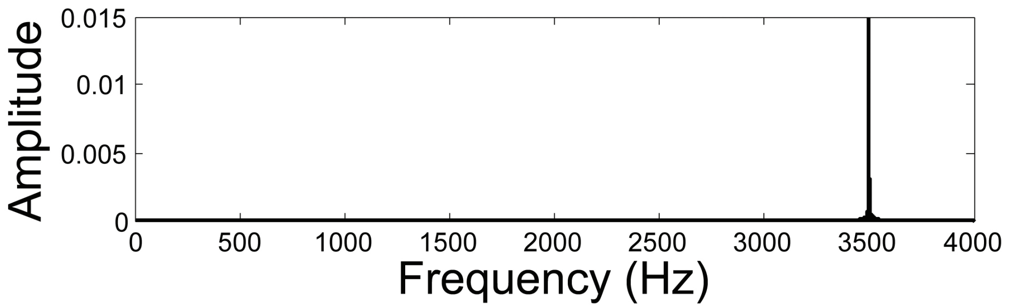


FIGURE 4.2: Single-sided amplitude spectrum of the ringtone

representation of the sounds reported at each phone (a vector, $\overrightarrow{S}$, with 4000 dimensions), and computes the *similarity* of these vectors with the vector generated from the known ring-tone ($\overrightarrow{R}$). The similarity function, expressed below, is essentially a weighted intensity ratio after subtracting white noise (Doppler shifts are explicitly addressed by computing similarity over a wider frequency range).

$$Similarity = \frac{Max\{\overrightarrow{S}(i) | 3450 =< i <= 3550\}}{Max\{\overrightarrow{R}(i) | 3450 =< i <= 3550\}}$$

Therefore, high similarities are detected when devices are in the vicinity of the ringtone transmitter. The overhearing range of a ringtone defines the auditory space around the transmitter.

Figure 4.3 shows the similarity values over time at three different phones placed near a ring-tone transmitter. The first curve is the known transmitted ringtone and other three curves are the ones received. As shown in Figure 4.3, the overheard ringtones are in broad agreement with the true ringtone. All phones that exhibit more than a threshold similarity are assigned to the same acoustic group. A phone may be assigned to multiple acoustic groups. At the end of this operation, the party is said to be “acoustically covered”.

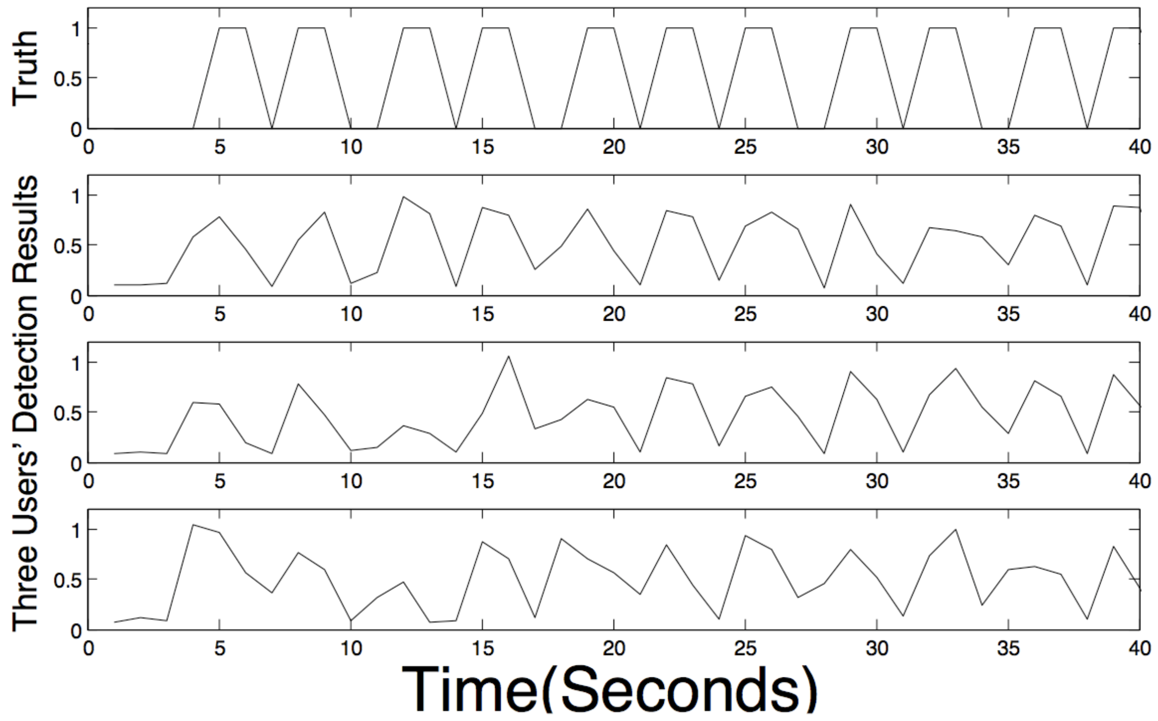


FIGURE 4.3: Ringtone detection at phones within the acoustic zone of the transmitter.

Grouping through Ambient Sound. Ringtones may not be always detectable, for example, when there is music in the background, or other electro-mechanical hum from machines/devices on the ringtone's frequency band. An alternative approach is to compute similarities between phones' ambient sounds, and group them accordingly. Authors in Nakakura et al. (2009) address a similar problem – they use high-end, time-synchronized devices to record ambient sound, and compare them directly for signal matching. However, we observed that mobile phones are weakly time-synchronized (in the order of seconds), and hence, direct comparison results will yield errors. Therefore, we classify ambient sound in stable classes using an SVM (Support Vector Machine) on MFCC (Mel-Frequency Cepstral Coefficients), and group phones that "hear" the same classes of sound. We describe the process next.

For classification, we build a data benchmark with labeled music, human conversation, and noise. The music data is a widely used benchmark from Dortmund University Homburg et al. (2005), composed of 9 types of music. Each sample is ten seconds long and the total volume is for around 2.5 hours. The conversation data set is built by ourselves, and consists of 2 hours of conversation data from different male and female speakers. Samples from each speaker is around ten minutes long. The noise data set is harder to build because it may vary entirely based on the user's background (i.e., the test may arrive from a different distribution than the training set). However, given that MoVi is mostly restricted to indoor usage, we have incorporated samples of A/C noises, microwave hums, and the noise of phone grazing against trousers and table-tops. Each sample is short in length but we have replicated the samples to make their size equal to other acoustic data.

MFCC (Mel-Frequency Cepstral Coefficients) Logan (2000); Rabiner and Juang (1993) are used as features extracted from sound samples. In sound processing, Mel-frequency cepstrum is a representation of the short-term power spectrum of a sound. MFCC are commonly used as features in speech recognition and music information retrieval. The process of computing MFCC involves four steps: (1) We divide the audio stream into overlapping frames with 25ms frame width and 10ms forward shifts. The overlapping frames better capture the subtle changes in sound (leading to improved performance), but at the expense of higher computing power. (2) Then, for each frame, we perform an FFT to obtain the amplitude spectrum. However, since each frame has a strict cut-off boundary, the FFT causes leakage. We employ the Hann window technique to reduce spectral leakage. Briefly, Hann window is a raised cosine window that essentially acts as a weighting function. The weighing function is applied to the data to reduce the sharp discontinuity at the boundary of frames. This is achieved by matching multiple orders of derivatives, and setting the value of the derivatives to zero Harris (1978). (3) We then take the logarithm on the spectrum,and convert the log spectrum to Mel (perception-based) spectrum. Using Mel scaled units Logan (2000) is expected to produce better results than linear units because Mel scale units better approximate human perception of sound. (4) We finally take the Discrete Cosine Transform (DCT) on the Mel spectrum. In Logan (2000), the author proves that this step approximates principal components analysis (PCA), the mathematically standard way to decorrelate the components of the feature vectors, in the context of speech recognition and music retrieval.

After feature extraction, classification is performed using a two-step decision, using support vector machines (SVM), a machine learning method for classification Chang and Lin (2001). Coarse classification tries to distinguish music, conversation, and ambient noise. Finer classification is done for classes within conversation and music McKinney and Breebaart (2003). Classes for conversation include segregating between male and female voices, which is useful to discriminate between, say, two social groups, one of males, another of females. Similarly, music is classified into multiple genres. The overall cross validation accuracy is shown in Table 4.1. The reported accuracy is tested on the benchmarks described before. Based on such classification, Figure 4.4 shows the grouping among two pairs of phones – <A,B> and <A,C> – during the Thanksgiving party. Users of phones A and C are close friends and were often together in the party, while user of phone B joined A during some events as in. Accordingly, A and C are more often grouped as in Figure 4.4(b) while user A and B are usually separated (Figure 4.4(a)).

Table 4.1: Cross Validation Accuracy on Sound Benchmarks

Classification Type	Accuracy
Music, Conversation, Noise	98.4535%
Speaker Gender	76.319%
Music Genre	40.3452%

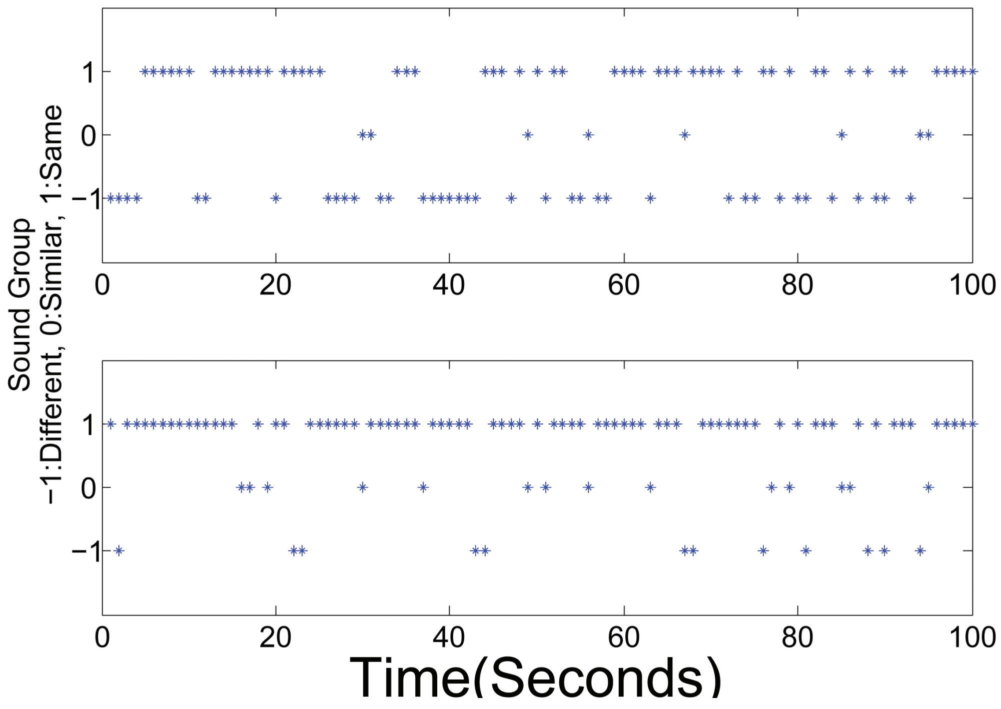


FIGURE 4.4: Grouping based on acoustic ambience: (a) users A and B's acoustic ambiences' similarity. (b) users A and C's acoustic ambiences' similarity.

(2) Visual Grouping

As mentioned earlier, acoustic ambience alone is not a reliable indicator of social groups. Similarity in visual ambience, including light intensity, surrounding color, and objects, can offer greater confidence on the phone's context Azizyan et al. (2009b). We describe our visual grouping schemes here.

Grouping through Light Intensity. In some cases, light intensities vary across different areas in a social setting. Some people may be in an outdoor porch, others in a well-lit indoor kitchen, and still others in a darker living room, watching TV. We implemented light-based grouping using analogous *similarity* functions as used with sound. However, we found that the light intensity is often sensitive to the user's orientation, nearby shadows, and obstructions in front of the camera. To achieve robustness, we conservatively classified light intensity into three classes, namely, bright, regular, and dark. Most phones were associated to any one of these classes; some phones with fluctuating light readings, were not visually-grouped at all. Figure 4.5 illustrates samples from three light classes from the social gathering at the university.

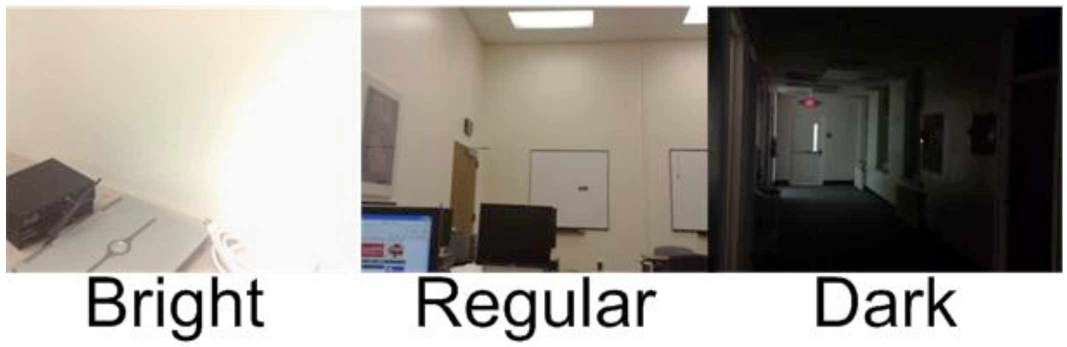


FIGURE 4.5: Grouping based on light intensity – samples from 3 intensity classes.

Grouping through View Similarity. A second way of visual grouping pertains to similarity in the images from different phone cameras. Multiple people may simultaneously look at the person making a wedding toast, or towards an entering celebrity, or just towards the center of a table with a birthday cake on it. MoVi intends to exploit this opportunity of common view. To this end, we use an image generalization technique called spatiogram Birchfield and Rangarajan (2005). Spatiograms are essentially color histograms encoded with spatial information. Briefly, through such a representation, pictures with similar spatial organization of colors and edges exhibit high similarity. The second order of spatiogram can be represented as:

$$h_I(b) = \langle n_b, \mu_b, \sigma_b \rangle, b = 1, 2, 3 \cdots B$$

where n_b is the number of pixels whose values are in the b^{th} bin (each bin is a range in color space), and μ_b and σ_b are the mean vector and covariance matrices, respectively, of the coordinates of those pixels. B is the number of bins. Figures 4.6(a) and (b) show the view from two phones while their owners are playing a multi-player video-game on a projector screen. Both cameras capture the screen as the major part of the picture. Importantly, the views are from different instants and angles, yet, the spatiogram similarities are high. Comparing to the top two pictures, the views in Figure 4.6(c) and (d) are not facing the screen, therefore exhibiting a much lower view similarity.

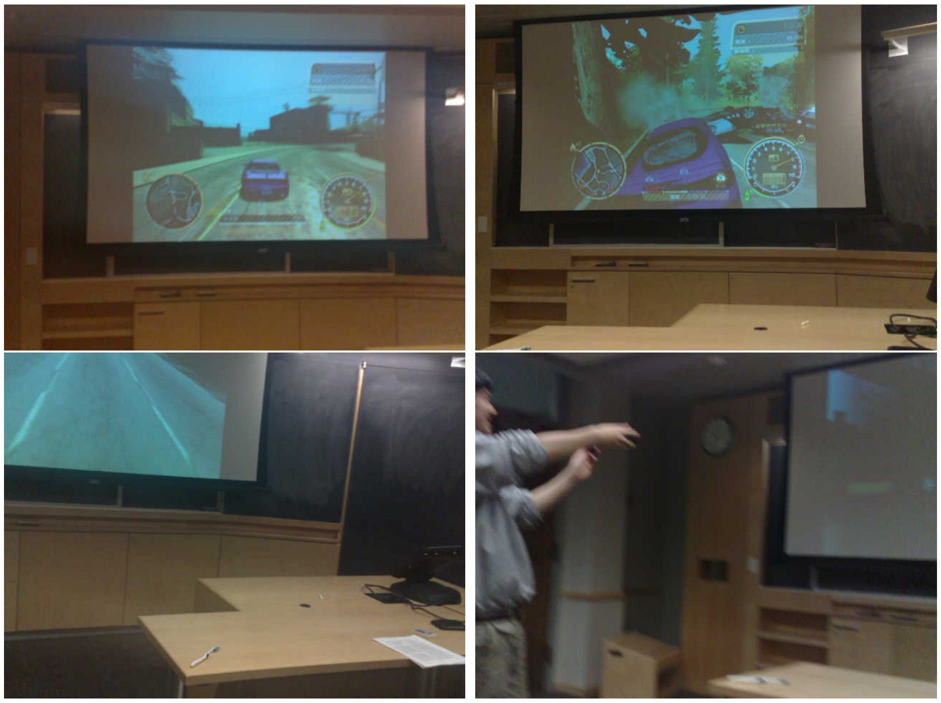


FIGURE 4.6: Grouping based on view similarity – top two phones (a, b) are in the same group watching video games, while the bottom two (c, d) are in the same room but not watching the games..

The MoVi server mines through the acoustic and visual information (offline), and combines them to form a single audio-visual group. View similarity is assigned highest priority, while audio and light intensity are weighed with a lower, equal priority. This final group is later used for collaboratively inferring the occurrence of events. Towards this goal, we proceed to the discussion of event-triggers.

4.3.2 Trigger Detection

From the (recorded) multi-sensory information, the MoVi server must identify patterns that suggest events of potential social interest. This is challenging because of two factors. First, the notion of interesting is subjective; second, the space of social events (defined by human cognition) is significantly larger than what today's sensing/inferring technology may be able to discern. We admittedly lower our targets, and try to identify some opportunities to detect event-triggers. We design three categories, namely (1) Specific Event Signature, (2) Group Behavior Pattern, and (3) Neighbor Assistance.

(1) Specific Event Signatures

These signatures pertain to specific sensory triggers derived from human activities that, in general, are considered worth recording. Examples of interest include, laughter, clapping, shouting, whistling, singing, etc. Since we cannot enumerate all possible events, we intend to take advantage of collaboration using triggers related to group behavior instead of relying heavily on specific event signatures. Therefore, as a starting point, we have designed specific acoustic signatures only for laughter Kennedy and Ellis (2004) using MFCC. Validation across a sample of 10 to 15 minutes of laughter, from 4 different students, offered evidence that our laughter-signature is robust to independent individuals. Negative samples are human conversation and background noise. Figure 4.7 shows the distribution of self-similarity between laughter-samples and cross similarity between laughter and other negative samples. In other words, the laughter samples and negative samples form different clusters in the 12 dimensional space. We achieved a cross-validation accuracy of 76% on our benchmark.

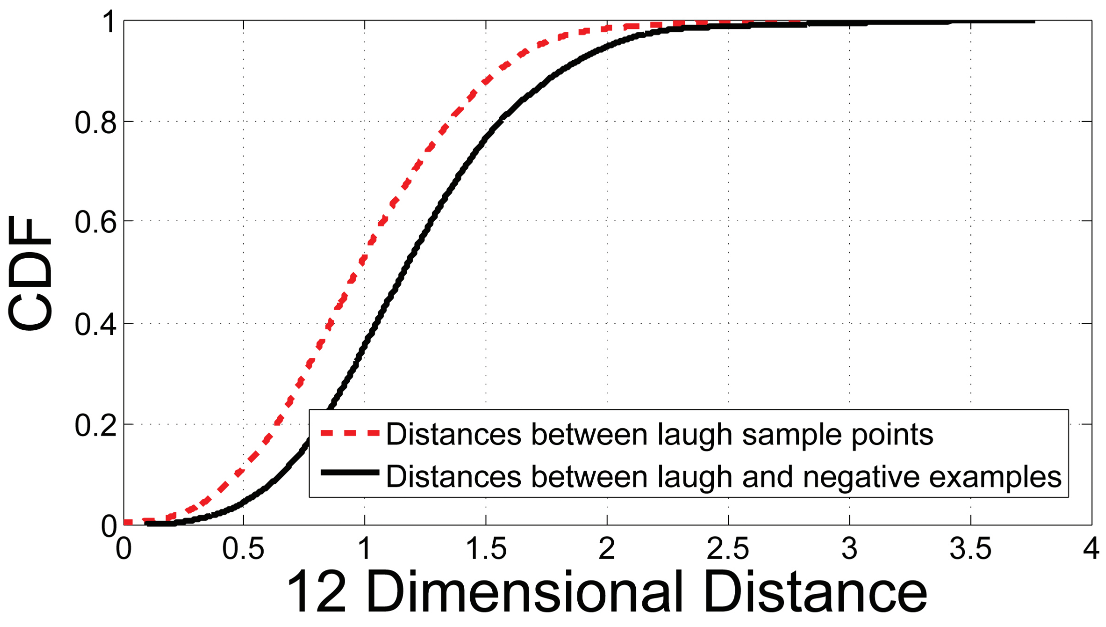


FIGURE 4.7: The CDFs show the distances between pairs of laugh samples, and distances between laugh and other sound samples.

(2) Group Behavior Pattern

The second event-trigger category exploits similarity in sensory fluctuations across users in a group. When we observe most members of a group behaving similarly, or experiencing similar variances in ambience, we infer that a potentially interesting event may be underway. Example triggers in this category are view similarity detection, group rotation, and acoustic-ambience fluctuation.

Unusual View Similarity. When phone cameras are found viewing the same object from different angles, it could be an event of interest (EoI). As mentioned earlier, some examples are people watching the birthday cake on a table, paying attention to a wedding toast, or everyone attracted by a celebrity's arrival. Recall that view-similarity was also used as a grouping mechanism. However, to qualify as a trigger, the view must remain similar for more than a threshold duration. Thus, augmenting the same notion of spatiogram with a minimum temporal correlation factor, we find good event-triggers. In Figure 4.8, each curve shows a pairwise similarity between two views in a group. The arrows show the two time-points at which three (2 pairs) out of four users are watching the same objects, that is i.e. their views show higher similarity than the threshold (empirically set as 0.75). Those three users are all triggered at the two time-points. Of course, such a trigger may be further correlated with other similarities in the acoustic and motion domains. Multi-sensor triggers is a part of our ongoing work.

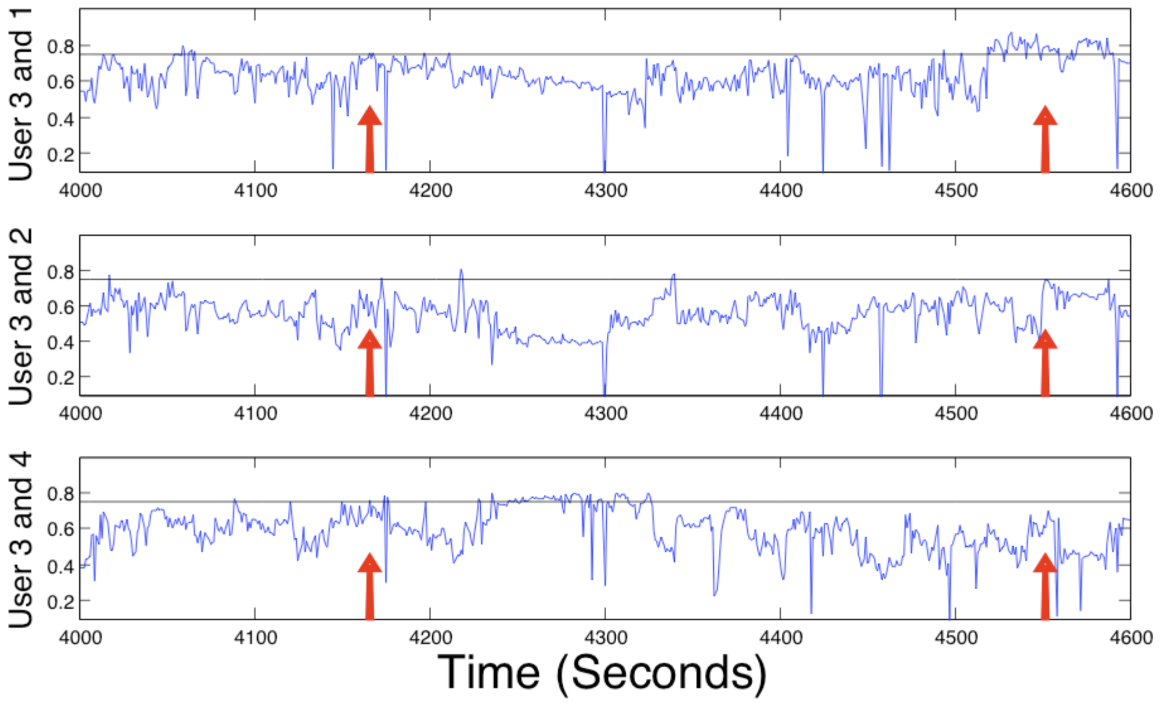


FIGURE 4.8: Pair-wise view similarity, at least among 3 phones, qualifies as a video trigger. Users 3, 1, and 4 are all above the threshold at around 4100 seconds; users 3, 1, and 2 see a trigger at around 4500 seconds.

Group Rotation. An interesting event may prompt a large number of people to rotate towards the event (a birthday cake arrives on the table). Such "group rotation" – captured through the compasses in several modern phones – can be used as a trigger. If more than a threshold fraction of the people turn within a reasonably small time window, MoVi considers this a trigger for an interesting event. For this, the compasses of the phones are always turned on (we measured that the battery consumption is negligible). The compass-based orientation triggers are further combined with accelerometer triggers, indicating that people have turned and moved together. The confidence in the trigger can then be higher. Such a situation often happens, e.g., when a break-out session ends in a conference, and everyone turns towards the next speaker/performer.

Ambience Fluctuation. The general ambience of a social group may fluctuate as a whole. Lights may be turned off on a dance floor, music may be turned on, or even the whole gathering may lapse into silence in anticipation of an event. If such fluctuations are detectable across multiple users, they may be interpreted as a good trigger. MoVi attempts to make use of such collaborative sensor information. Different thresholds on fluctuations are empirically set – with higher thresholds for individual sensors, and relatively lower for joint sensing. The current goal is to satisfy a specific trigger density, no more than two triggers for each five minutes. Of course, this parameter can also be tuned for specific needs. Whenever any of the sensor's reading (or combined) exceed the corresponding threshold, all the videos from the cameras become candidates for inclusion in the highlights. Figure 4.9 shows an example of the sound fluctuation in time domain, taken from the SmartHome visit. The dark lines specify the time-points when the average of one-second time windows exceed a threshold. These are accepted as triggers. The video-clips around these time-points are eventually "stitched" into the video highlights.

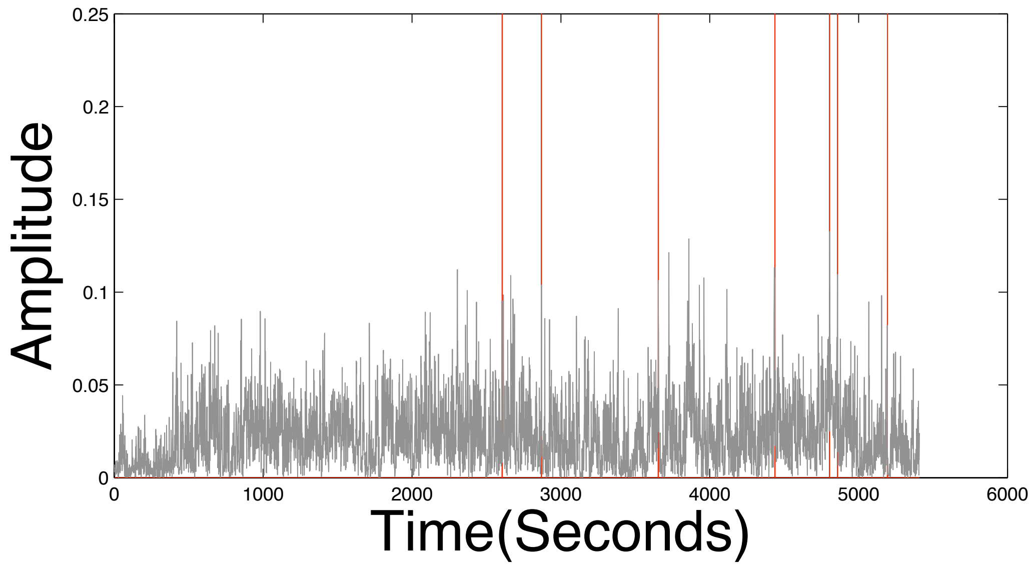


FIGURE 4.9: The fluctuations in the acoustic ambience are interpreted as triggers (time-points shown in black lines).

(3) Neighbor Assistance

The last category of event-trigger opportunistically uses human participation. Whenever a user explicitly takes a picture from the phone camera, the phone is programmed to send an acoustic signal, along with the phone's compass orientation. Other cam-

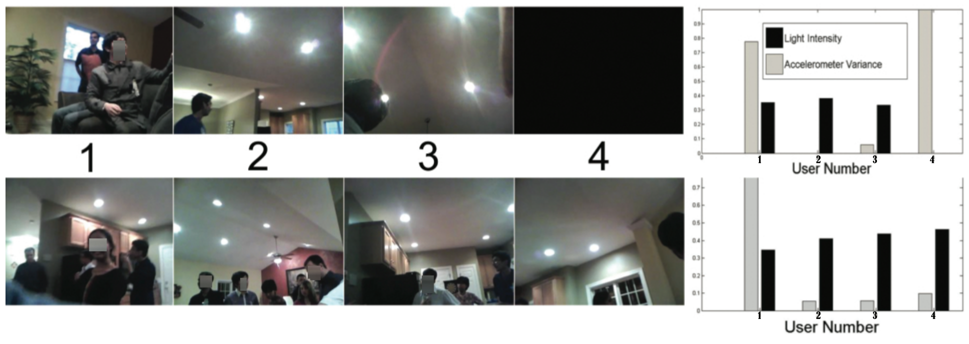


FIGURE 4.10: View selection based on a multiple sensing dimensions. The first view is chosen for inclusion in the highlights because of its better lighting quality, more number of distinct human faces, and less acceleration.

eras in the vicinity overhear this signal, and if they are also oriented in a similar direction, the videos from the camera are recruited as candidates for highlights. The intuition is that humans are likely to take a picture of an interesting event, and including that situation in the highlights may be worthwhile. In this sense, MoVi brings the human into the loop.

4.3.3 View Selection

The view selection module is tasked to select videos that have a good view. Given that cameras are wearable (taped on shirt pockets in our case), the views are also blocked by objects, or pointed towards uninteresting directions. Yet, many of the views are often interesting because they are more personal, and captures the perspectives of a person. For this, we again rely on multi-dimensional sensing.

Four heuristics are jointly considered to converge on the "best view" among all the iPods that recorded that event. (1) Face count: views with more human faces are given the highest priority. This is because human interests are often focused on people. Moreover, faces ensure that camera is facing a reasonable height, not to the ceiling or the floor. (2) Accelerometer reading ranking: to pick a stable view, the cameras with the least accelerometer variance are assigned proportionally higher points. More stable cameras are chosen to minimize the possibility of motion blurs in the video. (3) Light intensity: to ensure clarity and visibility, we ranked the views in the "regular" light class higher, and significantly de-prioritize the darker pictures. This is used only to rule out extremely dark pictures, which mostly are caused by blocking. (4) Human in the loop: finally, if a view is triggered by "neighbor assistance", the score for that view is increased.

Figure 4.10 shows two rows corresponding to two examples of view selection;

pictures were drawn from different iPod videos during the Thanksgiving party. The first view in each instance is selected and seems to be more interesting than the rest of views. Figure 4.11 illustrates the same over time. At each time-point, the blue circle tags the human selected view while the red cross tags the MoVi select one. When two symbols overlap, the view selection achieves right result. The most common reason that view selection fails is that all four views exhibit limited quality. Therefore, even for human selection, the chosen one is only marginally better.

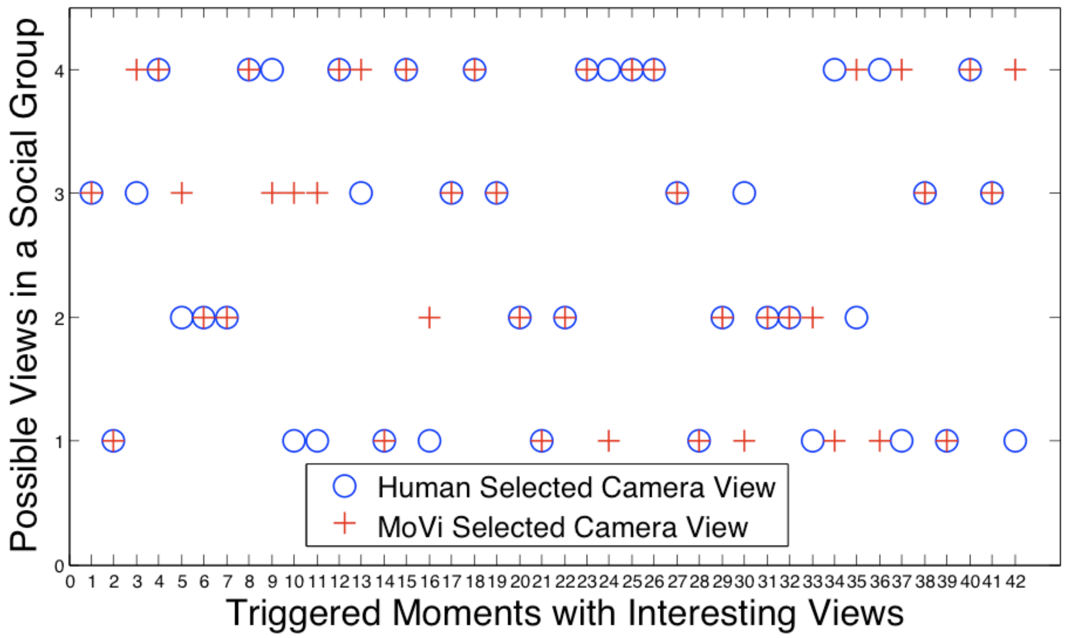


FIGURE 4.11: MoVi selects views that are similar to human selected ones.

4.3.4 Event Segmentation

The Event Segmentation module is designed to identify the logical start and end of an event. A clap after the "happy birthday" song could be the acoustic trigger for video inclusion. However, the event segmentation module should ideally include the song as well, as a part of the highlights. The same applies to a laughter trigger; MoVi should be able to capture the joke that perhaps prompted it. In general, the challenge is to scan through the sensor data received before and after the trigger, and detect the logical start and end that may associate with the trigger.

For event segmentation, we use the sound state-transition, computed during the sound classification/grouping phase, time as clues Lu et al. (2009). For example, when laughter is detected during conversation, we rewind on the video, and try to identify the start of a conversation. Gender based voice classification offers a finer ability to segment the video – if multiple people were talking, and a women's voice prompted the joke, MoVi may be able to identify that voice, and segment the video from where that voice started. Figure 4.12 shows our key idea for event segmentation.

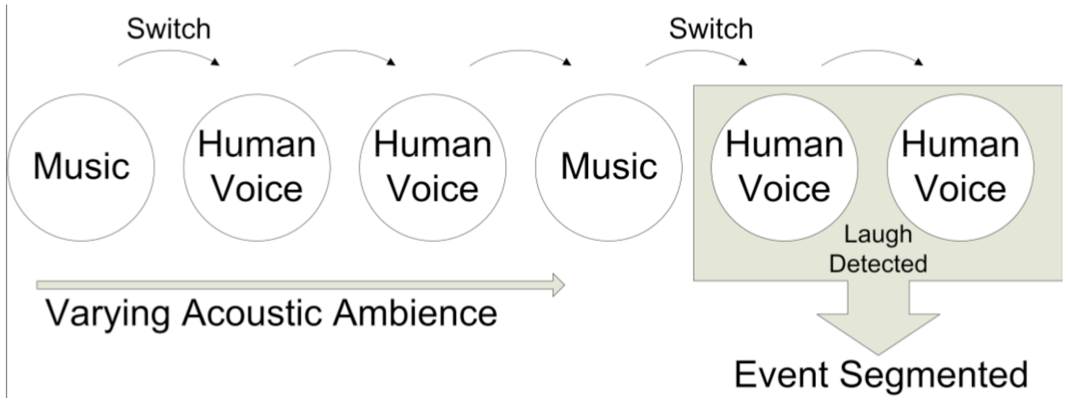


Figure 4.12: The scheme for segmenting events.

4.4 Evaluation

This section attempts to asses MoVi's overall efficacy in creating a video highlight. Due to the subjective/social nature of this work, we choose to evaluate our work by combining users' assessment with metrics from information retrieval research. We describe our experimental set-up and evaluation metrics next, followed by the actual results.

4.4.1 Experiment Set-up

Our experiments have been performed in one controlled setting and two natural social occasions. In each of these scenarios, 5 volunteers wore the iPod video cameras on their shirts, and clipped the Nokia N95 phones on their belts. Figure 4.13 shows an example of students taped with iPod Nanos near their shirt pockets. The iPods recorded continuous video for around 1.5 hours (5400 seconds), while the phones logged data from the accelerometer, compass, and microphone. In two of the three occasions, a few phone cameras were strategically positioned on a table or cabinet, to record the activities from a static perspective. All videos and sensor measurements were downloaded to the (MATLAB-based) MoVi server. Each video was organized into a sequence of 1 second clips. Together, the video clips from the volunteers form a 5×5400 matrix, with an unique iPod-device number for each row, and time (in seconds) indexed on each column. The sensor readings from the phones are similarly indexed into this matrix. MoVi's target may now be defined as the efficacy to pick the "socially interesting" elements from this large matrix.

The MoVi server analyzes the $<$ device, time $>$-indexed sensor readings to first form the social groups. During a particular time-window, matrix rows 1, 2, and 5 may be in the first group, and rows 3 and 4 in the second. Figure 4.14(2) shows an example grouping over time using two colors. Then, for every second (i.e., along each column of the matrix), MoVi scans through the readings of each phone to identify event triggers. Detecting a possible trigger in an element of the matrix, the server correlates it to other members of its group. If correlation results meet the desired threshold, MoVi performs view selection across members of that group. It is certainly possible that at time t_i, phone 2's sensor readings match the trigger, but phone 5's view is the best for recording this event(Figure 4.14(3)). MoVi selects

FIGURE 4.13: Users wearing iPods and Nokia phones.

this element <5, t_i >, and advances to perform event segmentation. For this, the system checks for the elements along the 5th row, and around column t_i. From these elements, the logical event segment is picked based on observed state-transitions. The segment could be the elements <5, t_{i-1} > to <5, t_{i+1} >, a 3 second video clip (Figure 4.14(4)). Many such video clips get generated after MoVi completes a scan over the entire matrix. These video clips are sorted in time, and "stitched" into a "movie". Temporal overlaps between clips are possible, and they are pruned by selecting the better view.

4.4.2 Evaluation Metrics

We use the metrics of *Precision*, *Recall*, and *Fall-out* for the two uncontrolled experiments. These are standard metrics in the area of information retrieval.

$$Precision = \frac{|\{\text{Human Selected} \cap \text{MoVi Selected}\}|}{|\{\text{MoVi Selected}\}|} \tag{4.1}$$

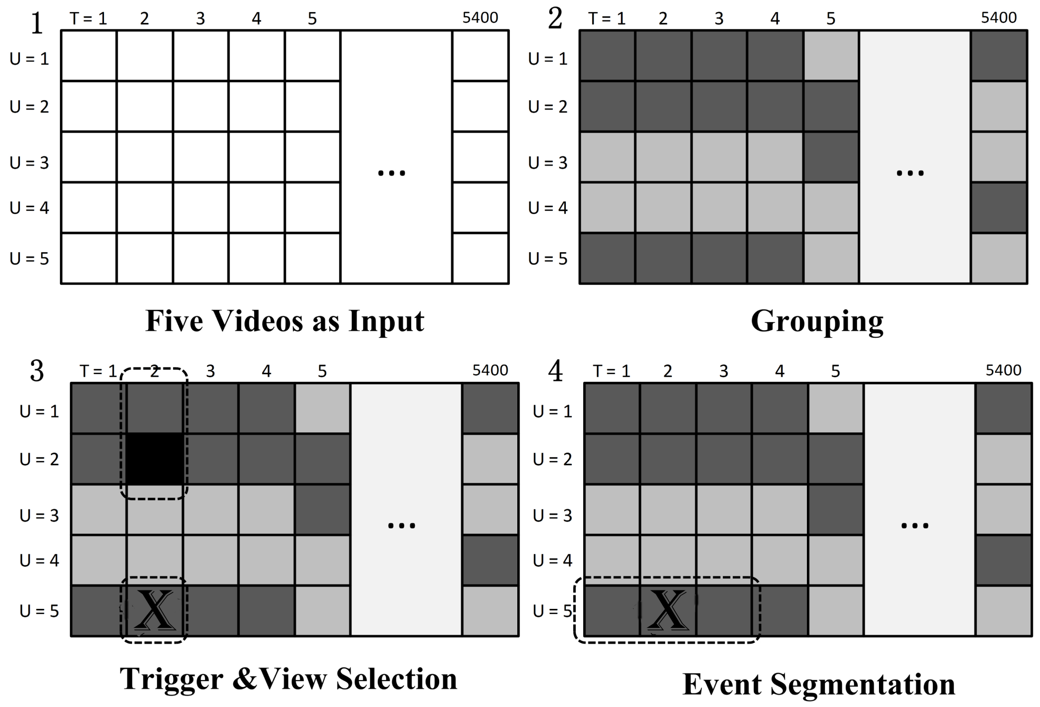


FIGURE 4.14: MoVi operations illustrated via a matrix.

$$Recall = \frac{|\{\text{Human Selected} \cap \text{MoVi Selected}\}|}{|\{\text{Human Selected}\}|} \tag{4.2}$$

$$Fall-out = \frac{|\{\text{Non-Relevant} \cap \text{MoVi Selected}\}|}{|\{\text{Non-Relevant}\}|} \tag{4.3}$$

The “Human Selected” parts are obtained by requesting a person to look through the videos and demarcate time windows that they believe are worth including in a highlights. To avoid bias from a specific person, we have obtained time-windows from multiple humans and also combined them (i.e., a union operation) into a single highlight[4]. We will report results for both cases. “Non-Relevant” moments refer to those not selected by humans. The “MoVi Selected” moments are self evident.

[4] For each experiment, one human reviewer has watched one full video from one camera, which lasts for more than an hour. All video sources from all cameras are covered.

4.4.3 Performance Results

(1) Controlled Experiment

The aim of the controlled experiment is to verify whether all the components of MoVi can operate in conjunction. To this end, a weekend gathering is planned with pre-planned activities, including watching a movie, playing video-games, chatting over newspaper articles, etc. This experiment is assessed rather qualitatively, ensuring that the expected known exciting events are captured well. Table 4.2 shows event-detection results. The first two columns show the designed events and their occurrence times; the next two columns show the type of triggers that detected them and the corresponding detection times. Evidently, at least one of the triggers were able to capture the events, suggesting that MoVi achieves a reasonably good event coverage. However, it also included a number of events that were not worthy of recording (false positives). We note that the human-selected portions of the video summed up to 1.5 minutes (while the original video was for 5 minutes). The MoVi highlights covered the full human-selected video with good accuracy (Table 4.3), and selected an additional one minute of false positives. Clearly, this is not a fair evaluation, and will be drastically different in real occasions. However, it is a sanity check that MoVi can achieve what it absolutely should.

(2) Field Experiment: Thanksgiving Party

The two field experiments were performed to understand MoVi's ability to create a highlights in real social occasions. This is significantly more challenging in view of a far larger event space, potentially shaking cameras from real excitement, greater mobility within the party, background music, other noise in the surroundings, etc. The first experiment was at a Thanksgiving party, attended by 14 people. Five attendants were instrumented with iPods and phones. After the party, videos from the five cameras were distributed to five different people for evaluation. Manually

Table 4.2: Per-Trigger results in single experiment (false positives not reported)

Event Truth	Time	Trigger	Det. Time
Ringtone	25:56	RT, SF	25:56
All watch a game	26:46	IMG	27:09
Game sound	26:58	SF	27:22
2 users see board	28:07	IMG	28:33
2 users see demo	28:58	SF	29:00
Demo ends	31:18	missed	
Laughing	34:53	LH, SF	34:55
Screaming	36:12	SF	36:17
Going outside	36:42	IMG, LI	37:18
RT:ringtone SF:sound fluctuation LI:light intensity			
IMG:image similarity LH:fingerprint			

Table 4.3: Average Trigger Accuracy and Event Detection latency (including false positives)

Triggers	Coverage	Latency	False Positive.
RT	100%	1 second	10%
IMG	80%	30 seconds	33%
LH	75%	3 seconds	33%
LI	80%	30 seconds	0%
SF	75%	5 second	20%

selecting the highlights from the full-length video was unanimously agreed to be a difficult and annoying task (often done as a professional service). However, with help from friends, we were able to obtain the *Human Selected* moments. The MoVi generated highlights were also generated, and compared against the manual version.

Figure 4.15 shows the comparative results *at the granularity of one second.* The X-axis models the passage of time, and the Y-axis counts the *cumulative* highlights duration selected until a given time. For instance, Y-axis = 100 (at X-axis = 1200) implies that 100 seconds of highlights were selected from the first 1200 seconds of the party. Figure 4.16 presents a zoom-in view for the time windows 2700-3500 and 1000-1400 seconds. We observe that the MoVi highlights reasonably tracks the Human

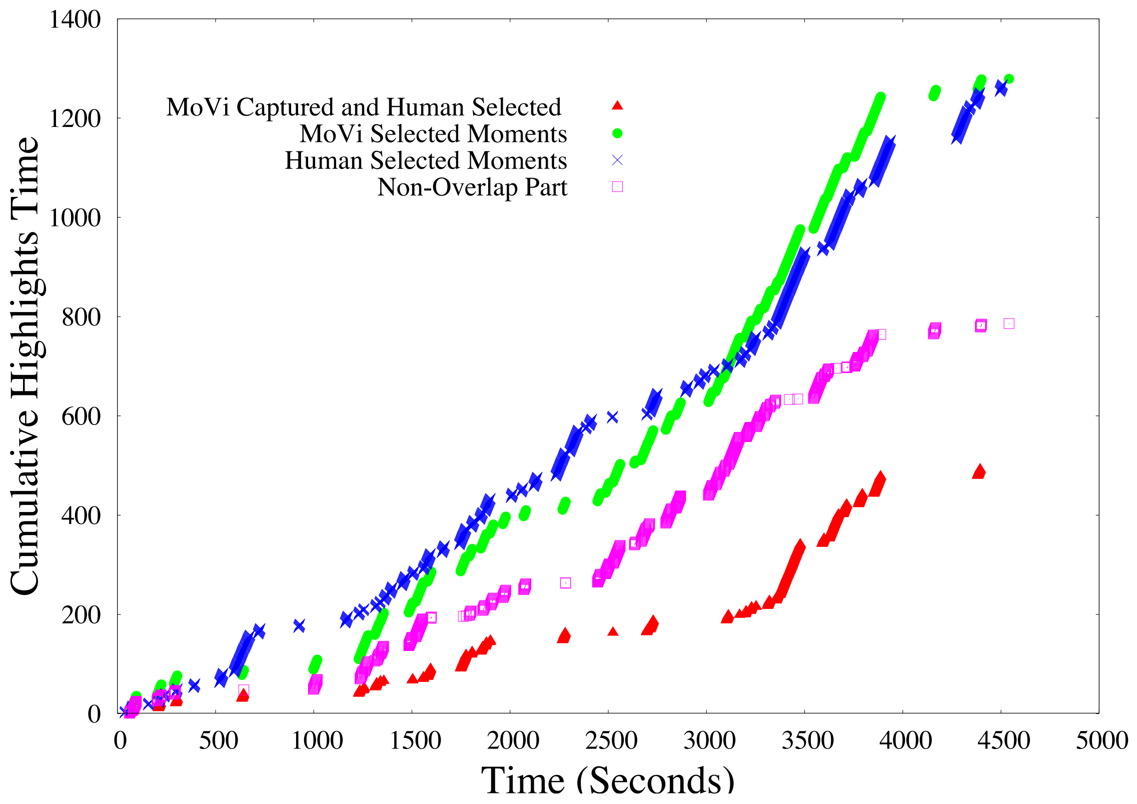


FIGURE 4.15: Comparison between MoVi and human identified event list (Thanksgiving)

Selected (HS) highlights. The curve (composed of triangles) shows the time-points that *both* MoVi and HS identified as interesting. The non-overlapping parts (i.e., MoVi selects that time, but HS does not) reflect the false positives (curve composed of squares).

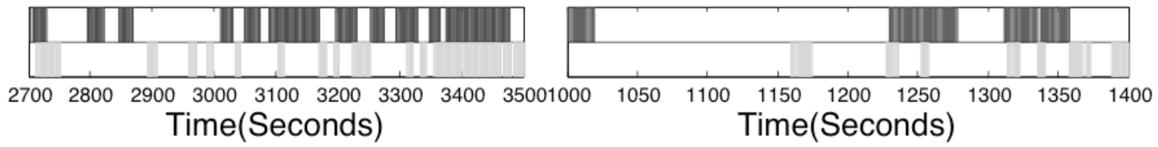


FIGURE 4.16: Zoom in view for two parts of Figure 4.15. Dark gray: MoVi, light gray: human selected

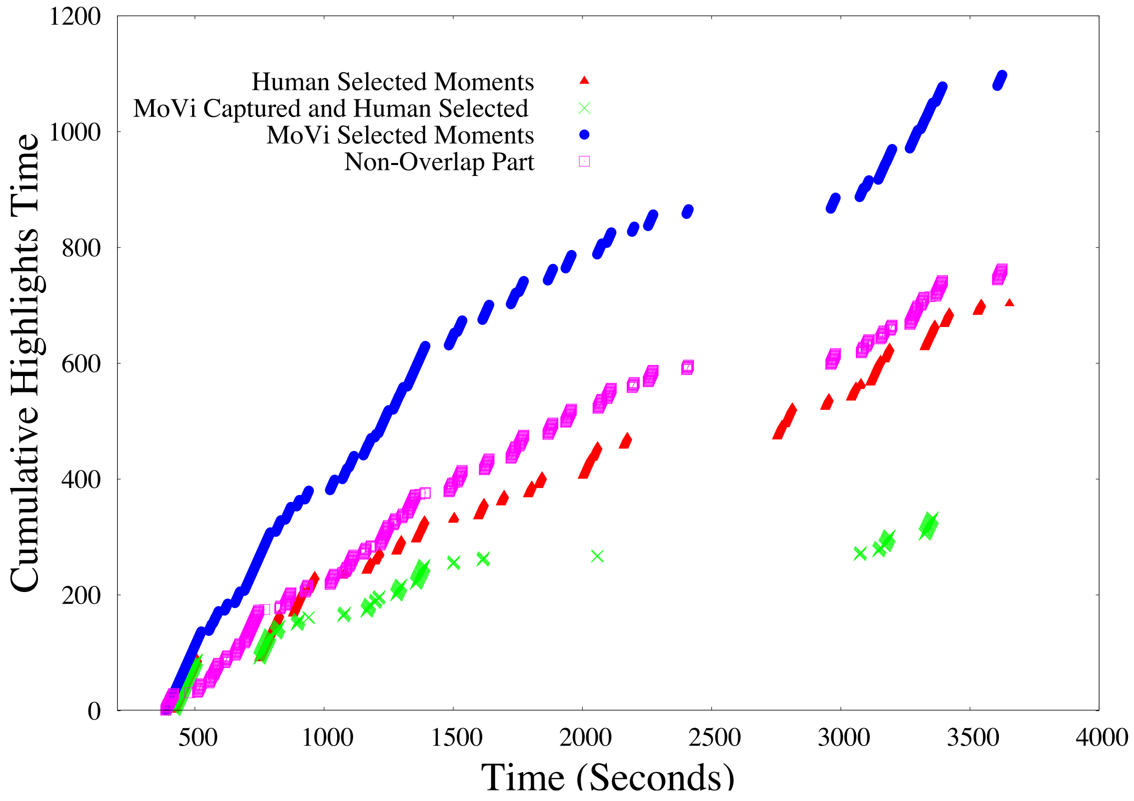


FIGURE 4.17: Comparison between MoVi and human identified event list (SmartHome)

Based on this evaluation, we computed the overall *Precision* to be 0.3852, *Recall* to be 0.3885, and *Fall-out* to be 0.2109. Notice that the overall precision is computed by using the union of all human selected video as the retrieval target. Therefore, if a moment is labeled as interesting by one user, it is considered interesting. We also compared the improvement over a random selection of clips (i.e., percentage of MoVi's overlap with human (MoH) minus percentage of Random's overlap with Human (RoH), divided by RoH). MoVi's improvement is 101% on average.

In general, the false positives mainly arise due to two reasons: (1) Falsely detected triggers: since the sensor-based event detection method cannot achieve 100%

accuracy, false positives can occur. Since we assign more weight to infrequently happening triggers such as laughter, we trade off some precision for better recall. (2) Subjective choice: the user reviewing the video may declare some of the events (even with triggers) as not interesting. Since this is a subjective judgment, false positive will occur.

Table 4.4 shows the per-user performance when the MoVi highlights is compared with individual user's selections. Since each user only selects a very small portion of the entire video, according to equation 4.1, the computed precision is expected to be low. As a result, Recall and performance gains over the Random scheme are more important metrics in this case. The average improvement proves to be 101%.

The results are clearly not perfect, however, we believe, are quite reasonable. To elaborate on this, we make three observations. (1) We chose a strict metric wherein MoVi-selected clips are not rewarded even if they are very close (in time) to the Human Selected clips. In reality, social events are not bounded by drastic separations, and are likely to "fade away" slowly over time. We observed that MoVi was often close to the human selected segments; but was not rewarded for it. (2) We believe that our human selected videos are partly biased – all users enthusiastically picked more clips towards the beginning, and became conservative/impatient over time. On the other hand, MoVi continued to automatically pick videos based on pure event-triggers. This partly reduced performance. (3) Finally, we emphasize that "human interest" is a sophisticated notion and may not always project into the sensing domains we are exploring. In particular, we observed that humans identified a lot of videos based on the topics of conversation, based on views that included food and decorative objects, etc. Automatically detecting such interests will perhaps require sophisticated speech recognition and image processing. In light of Google's recent launch of the Google Goggles, an image search technology, we are considering its application to MoVi. If MoVi searches its camera pictures through Google Goggles, and retrieves that the view is of a wedding dress, say, it could be a prospective trigger. Our current triggers are unable to achieve such levels of distinction. Yet, the MoVi-generated highlights was still interesting. Several viewers showed excitement at the prospect that it was generated without human intervention.

Table 4.4: Per-user performance (Thanksgiving party)

User	Precision	Recall	Fall-out	Over Random
1	21%	39%	23%	51%
2	5%	33%	12%	162%
3	9%	37%	25%	46%
4	18%	74%	20%	222%
5	4%	22%	17%	26%

Field Experiment: SmartHome Tour

The Duke SmartHome is a live-in laboratory dedicated to innovation and demonstration of future residential technology. Eleven members of our research group attended a guided tour into the SmartHome. Five users wore the iPods and carried the N95 phones. Figure 4.17 shows the results.

In this experiment, the human highlights creator did not find too many interesting events. This was due to the academic nature of the tour with mostly discussions and references to what is planned for future. The human selected moments proved to be very sparse, making it difficult to capture them precisely. MoVi's *Precision* still is 0.3048, *Recall* is 0.4759, and *Fall-out* is 0.2318. Put differently, MoVi captured most of the human selected moments but also selected many other moments (false positives). Compared to *Random* (discussed earlier), the performance gain is 102% on average. Table 4.5 shows the performance when manual highlights was created from the union of multiple user-selections.

Table 4.5: Per-user performance (SmartHome)

User	Precision	Recall	Fall-out	Over Random
1	21%	62%	23%	124%
2	19%	45%	25%	67%
3	6%	50%	22%	116%

In summary, we find that inferring human interest (especially semantically defined ones) is hard. Although this is a current limitation, MoVi's trigger mechanism can capture most events that have an explicit sensor clue. The highlighted video is of reasonably good quality in terms of camera-angle, lighting, and content. Although not a human-videographer replacement, we believe that MoVi can serve as an additional tool to complement today's methods of video-recording and manual editing.

4.5 Related Work

The ideas, algorithms, and the design of MoVi is drawn from a number of fields in computer science and electrical engineering. Due to limited space, it is difficult to discuss the entire body of related work in each of these areas. We discuss some of the relevant papers from each field, followed by works that synthesize them on the mobile computing platform.

Wearable Computing and SenseCam. Recent advances in wearable devices are beginning to influence mobile computing trends. A new genre of sensing devices is beginning to blend into the human clothing, jewelry, and in the social ambience. The Nokia Morph Virpioja et al. (2007), SixthSense camera-projectors Mistry. (2009), LifeWear, Kodak 1881 locket camera, and many more are beginning to enter the commercial market. A large number of projects, including MIT GroupMedia, Smart Clothes, AuraNet and Gesture Pendant Mann (1997); **?**); **?** have exploited these devices to build context-aware applications. Microsoft Research has recently developed SenseCam, a wearable camera equipped with multiple sensors. The camera takes a photo whenever the sensor readings meet a specified degree of fluctuations in the environment (e.g., change in light levels, above-average body heat). The photos are later used as a pictorial diary to refresh the user's memory, perhaps after a vacation Berry et al. (2007). MoVi draws from many of these projects to develop a collaborative sensing and event-coverage system on the mobile phone platform.

Computer Vision. Researchers in Computer Vision have studied the possibility of extracting semantic information from pictures and videos. Of particular interest are works that use audio-information to segment video into logical eventsZhang and Kuo (1998); Baillie and Jose (2004). Another body of work attempts scene understanding and reconstruction Li et al. (2008) by combining multiple views of the same scene/landmark to a iconic scene graph. On a different direction, authors in Ke et al. (2006) have investigated the reason for longer human-attention on certain pictures; the study helps in developing heuristics that are useful to shortlist "good" pictures. For instance, pictures that display greater symmetry, or have a moderate number of faces (identifiable through face recognition), are typically viewed longer Nilsson et al. (2007). Clearly, MoVi is aligned to take advantage of these findings. We are by no means experts in Computer Vision, and hence, will draw on the existing tools to infer social events and select viewing angles. Additional processing/algorithms will still be necessary over the other dimensions of sensing.

Information Retrieval. Information retrieval (IR) Baeza-Yates and Ribeiro-Neto (1999) deals with the representation, storage, and organization of (and access to) information items. Mature work in this area, in collaboration with Artificial Intelligence (AI) and Natural Language Processing (NLP), have attempted to interpret the semantics of a query, and answer it by drawing from disparate information sources Grossman and Frieder (2004). Some research on mobile information retrieval Carpineto et al. (2009) have focused on clustering retrieval results to accommodate small display devices. Our objective of extracting the "highlights" can be viewed as a query, and the mobile phone sensors as the disparate sources of information. MoVi is designed to utilize metrics and algorithms from information retrieval.

Sensor Network of Cameras. Recently, distributed camera networks have received significant research attention. Of interest are projects that observe and model sequences of human activity. For example, BehaviorScope Teixeira and Savvides (2007) builds a home sensor network to monitor and help elders that live home alone. Distributed views are used to infer networked cameras' locations. Smart cameras Bramberger et al. (2004) are deployed to track real time traffic load. These works provide us useful models to organize information from multiple sensors/mobile nodes in a manner that will provide good coverage and correlation.

People-Centric Sensing. In mobile computing, people-centric, participatory sensing through mobile devices are gaining rapid popularity. Example applications include CenseMe Miluzzo et al. (2008), which detects the user's activity status through sensor readings and shares this status over online social networks. SoundSense Lu et al. (2009) implements audio processing and learning algorithms on the phone to classify ambient sound types – the authors propose an audio journal as an application. While these systems are individual specific, others correlate information from multiple sources to generate a higher level view of the environment. PEIR, Micro-Blog, Urban Tomography Mun et al. (2009), are few examples in this area.

Our work may be considered a mash-up of diverse techniques that together realize a fuller system. Customizing the techniques to the target application often presents new types of research challenges that are imperceptible when viewed in isolation. As an example, deducing human collocation based on ambient acoustics have been a studied problem Eagle (2003). Yet, when applied to the social context, two physically nearby individuals may be participating in conversations in two adjacent dinner tables. Segregating them into distinct social groups is non-trivial. MoVi makes an attempt to assimilate the rich information feeds from mobile phones and process them using a combination of existing techniques drawn from vision, data-mining, and signal processing. In that sense, it is a new mash-up of existing ideas. Our novelty comes from the collaboration of devices and the automatic detection of interesting events. Our preliminary ideas have been published in Bao and Choudhury (2009).

4.6 Limitations and Ongoing Work

MoVi is a first step towards a longer term project on collaborative sensing in social settings. The reported work has limitations, several of which stem from the non-trivial nature of the problem. We discuss these limitations along with avenues to address some of them.

Retrieval accuracy. The overall precision of our system certainly has room for improvement. Since "human interest" is a semantically sophisticated notion, to achieve perfect accuracy is challenging. However, as an early step towards social event retrieval, the precision of around 43% can be considered encouraging Grossman and Frieder (2004); Lew et al. (2006); Baillie and Jose (2004).

Unsatisfying camera views. Though view selection is used, cameras in a group may all have unsatisfying views of a specific event. The video highlights for these events exhibit limited quality. This problem can be partly addressed by introducing some static cameras into the system to provide a degree of all-time performance guarantee. The ideas in this chapter can be extended to these static wall mounted/wearable cameras equipped with multiple sensors.

Energy consumption. Continuous video-recording on the iPod Nanos persists for less than 2 hours. The mobile phone sensors can last for around 4 hours. Thus, in parallel to improving our event detection algorithms, we are beginning to consider energy as a first class design primitive. One option is to explore peer to peer coordination among phones – few phones may monitor a social zone, allowing other phones to sleep. Lightweight duty cycling, perhaps with periodic help from the server, is a part of our future effort.

Privacy. User privacy is certainly a concern in a system like MoVi. For this chapter, we have assumed that attendants in a social party may share mutual trust, and hence, may agree to collaborative video-recording. This may not scale to other social occasions. Certain other applications, such as travel blogging or distributed surveillance may be amenable to MoVi. Even then, the privacy concerns need to be carefully considered.

Greater algorithmic sophistication. We have drawn from preliminary ideas, tools, and algorithms, in data mining, information retrieval, signal processing, and image processing. A problem such as this requires greater sophistication in these algorithms. Our ongoing work is focused towards this direction, with a specific goal of prioritizing among different event triggers. One advantage of prioritizing will permit relative ranking between event-triggers; this may in turn allow for creating MoVi highlights for a user-specified duration. At present, the MoVi highlights are of a fixed duration.

Dissimilar movement between phones and iPods. We often observed that the acceleration in the phone was not necessarily correlated to the vibration in the video-clip. This is a result of the phone being on the belt and the iPod taped to the chest. Sensors on different parts of the body may sense differently, leading to potential false positives. One possibility is to apply image stabilization algorithms on the video itself to gain better view quality.

4.7 Conclusion

this chapter explores a new notion of “social activity coverage”. Like spatial coverage in sensor networks (where any point in space needs to be within the sensing range of at least one sensor), *social activity coverage* pertains to covering moments of social interest. Moreover, the notion of social activity is subjective, and thus identifying triggers to cover them is challenging. We take a first step through a system called Mobile Phone based Video Highlights (MoVi). MoVi collaboratively senses the ambience through multiple mobile phones and captures social moments worth recording. The short video-clips from different times and viewing angles are stitched offline to form a video highlights of the social occasion. We believe that MoVi is one instantiation of *social activity coverage*; the future is likely to witness a variety of other applications built on this primitive of collaborative sensing and information distillation.

5

Conclusion and Future Work

We conclude this thesis with a summary of the overall research goals and a discussion about future work directions. My research goal is to explore the techniques that can support the next generation of mobile systems that can change the way people acquire, process and digest information. The leading companies in the mobile industry are already offering intelligent applications (e.g., googlenow, passbook) that aim to provide the exactly needed information at the correct time and location right at the user's finger tips. Future applications will continue this trend and even further blur the information boundary between the physical world and online spaces. Information in the physical world will be automatically perceived by sensors, instantaneously transmitted to the cloud, processed and presented to human users in the most informative yet meaningful manor.

For example, when a future tourist comes to NYC, her personal device should be able to plan a best tour route based on her interests and what shows or games are currently being held in the city. The system should also help her navigate through any part of the city without trouble, even inside the Metropolitan museum or at the platforms of NY subway. Whenever her attention is attracted by any scene, a beautifully structured building or a street performer with bizarre outfits, the system should capture that moment and tag it with meaningful descriptions. Around dinner time, the system may recommend the restaurant that best suits her taste, health constraints, and exhibits the right social dynamics (e.g., filled with people of similar tastes and engaging in interesting conversations). When she finally goes back to her hotel room, the system can optimize the room settings to suit her current status (e.g., dimmed lighting for better sleep). All the operations should be tailored to the user's taste to avoid being excessive or intrusive. In other words, human users should be liberated from both the struggle of seeking information and the burden of distilling it. The materialization of this vision entails many challenges and calls for innovations across different research domains in computing. The following sections elaborate on my visions in designing applications, integrating machine learning techniques and preserving user privacy.

Small Apps: Achieve Detailed Activity Recognition

In my future research, I would like to investigate how we can enable applications to further understand users' behaviors in deep details. Previous work has exploited context information for general activity recognition (e.g., movement modality, house activities). However, a gap still exists between the explored territory of general activities and the demand for detailed understanding of micro-activities (e.g., users' reactions during reading). One way to gradually bridge this gap is to pinpoint specific scenarios that can benefit the most from user behavioral analysis and tailor the machine learning methods for each scenario specifically.

My latest work on automatic content rating reflects my effort along this direction. The goal of this project is to identify how much a user likes a video through sensing her reactions while she is watching the video on mobile devices. Example reactions range from acoustic signatures of laughter to detect which scenes were funny, to the stillness of the tablet indicating intense drama. The final output of the system provides detailed ratings for each segment of the video, giving viewers informative hints about the quality of a video. This work is an example of how we can identify relations between sensing information and detailed activities and use them towards a specifically defined goal.

Big Data: Integrate Machine Learning with the Cloud

With mobile applications relying more on multi-dimensional sensing, it is time to adopt machine learning modules in the system design. One potential direction for such adoption is to integrate machine learning with cloud based platforms. This cross-application platform should be able to analyze sensing data collected from mobile devices in large-scale and allow applications to exchange data as well as the learned semantics. For example, a traffic analysis application can benefit from valuable insights of a driving detection application. Knowing how people are driving at a particular region can certainly facilitate deduce the traffic conditions. A cloud-based platform can help such two applications to collaborate by sharing data, classifiers and inference results, and eventually enable modularized cross-application development.

Privacy Concern: Preserve Privacy by Analyzing Inferences

With mobile devices capable of collecting massive sensing data, the privacy concern has caught unprecedented attention. Current practice of explicitly asking users' permission achieves little to alleviate the situation. The uncomprehensive descriptions of privacy permissions can rarely help the user to make their intended decisions. Moreover, the static settings have not taken into account the correlation between different sensing dimensions, leaving rooms of potential information leakage through cross inferences (e.g., a malware can infer sensitive locations from WiFi information, bypassing the related privacy settings).

In my opinion, the solution lies in in-depth analysis of sensing information sources. By closely analyzing the relations between sensing data and activities, we can not only understand how different sources correlate with each other, but also acquire detailed knowledge about what features in each information source are the most critical for activity inferences. With such understanding, we could provide users with comprehensive descriptions of permissions, preserve privacy by holding important features of sensing data, and automatically personalize each user's privacy setting that fits their true intention by knowing their preference towards different activities.

Bibliography

Agrawal, S., Constandache, I., Gaonkar, S., Roy Choudhury, R., Caves, K., and DeRuyter, F. (2011), "Using mobile phones to write in air," in *ACM MobiSys*.

Azizyan, M. et al. (2009a), "SurroundSense: Mobile Phone Localization Via Ambience Fingerprinting," in *ACM MOBICOM*.

Azizyan, M., Constandache, I., and Roy Choudhury, R. (2009b), "SurroundSense: mobile phone localization via ambience fingerprinting," in *ACM MobiCom*.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999), *Modern information retrieval*, Addison-Wesley Reading, MA.

Baillie, M. and Jose, J. M. (2004), "An audio-based sports video segmentation and event detection algorithm," in *CVPRW*.

Bao, L. et al. (2004), "Activity recognition from user-annotated acceleration data," *Pervasive Computing*.

Bao, X. and Choudhury, R. (2009), "VUPoints: collaborative sensing and video recording through mobile phones," in *ACM Mobiheld*.

Bay, H., Tuytelaars, T., and Van Gool, L. (2006), "Surf: Speeded up robust features," *Computer Vision–ECCV 2006*.

Berry, E., Kapur, N., Williams, L., Hodges, S., Watson, P., Smyth, G., Srinivasan, J., Smith, R., Wilson, B., and Wood, K. (2007), "The use of a wearable camera, SenseCam, as a pictorial diary to improve autobiographical memory in a patient with limbic encephalitis: A preliminary report," *Neuropsychological Rehabilitation*.

Birchfield, S. T. and Rangarajan, S. (2005), "Spatiograms versus histograms for region-based tracking," *IEEE CVPR*.

Bramberger, M., Brunner, J., Rinner, B., and Schwabach, H. (2004), "Real-time video analysis on an embedded smart camera for traffic surveillance," in *RTAS*.

Breiman, L. (2001), "Random Forests." in *Machine Learning*, vol. 45(1).

Butler, A., Izadi, S., and Hodges, S. (2008), "SideSight: multi-touch interaction around small devices," in *ACM UIST*.

Cantador, I., Fernández, M., Vallet, D., Castells, P., Picault, J., and Ribière, M. (2008), "A multi-purpose ontology-based approach for personalised content filtering and retrieval," *Advances in Semantic Media Adaptation and Personalization*, pp. 25–51.

Carpineto, C., Mizzaro, S., Romano, G., and Snidero, M. (2009), "Mobile information retrieval with search results clustering: Prototypes and evaluations," *Journal of the ASIST*.

Caruana, R., Niculescu-Mizil, A., Crew, G., and Ksikes, A. (2004), "Ensemble selection from libraries of models," ACM ICML '04.

Chang, C. C. and Lin, C. J. (2001), *LIBSVM: a library for support vector machines*, Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Cherubini, M., De Oliveira, R., Oliver, N., and Ferran, C. (2010), "Gaze and Gestures in Telepresence: multimodality, embodiment, and roles of collaboration," .

Cowan, L. and Li, K. (2011), "ShadowPuppets: supporting collocated interaction with mobile projector phones using hand shadows," in *ACM CHI*.

Cuervo, E., Balasubramanian, A., et al. (2010), "MAUI: Making Smartphones Last Longer with Code Offload," in *ACM MobiSys*.

Dada, A., Graf von Reischach, F., and Staake, T. (????), "Displaying dynamic carbon footprints of products on mobile phones," *Adjunct Proc. Pervasive 2008*.

Eagle, N. (2003), "Dealing with Distance: Capturing the Details of Collocation with Wearable Computers," in *ICIS*.

Ekman, P. and Friesen, W. (2003), *Unmasking the face: A guide to recognizing emotions from facial clues*.

Gaonkar, S., Li, J., Choudhury, R. R., Cox, L., and Schmidt, A. (2008), "Micro-Blog: Sharing and querying content through mobile phones and social participation," in *ACM MobiSys*.

Goel, M., Findlater, L., and Wobbrock, J. (2012), "WalkType: using accelerometer data to accomodate situational impairments in mobile touch screen text entry," in *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12.

Grossman, D. and Frieder, O. (2004), *Information retrieval: Algorithms and heuristics*, Kluwer Academic Pub.

Han, J. (2005), "Low-cost multi-touch sensing through frustrated total internal reflection," in *ACM UIST*.

Harris, F. J. (1978), "On the use of windows for harmonic analysis with the discrete Fourier transform," *Proceedings of the IEEE*.

Harrison, C. and Hudson, S. (2008), "Scratch input: creating large, inexpensive, unpowered and mobile finger input surfaces," in *ACM UIST*.

Harrison, C., Tan, D., and Morris, D. (2010), "Skinput: appropriating the body as an input surface," in *ACM CHI*.

Harrison, C., Schwarz, J., and Hudson, S. (2011), "TapSense: enhancing finger interaction on touch surfaces," in *ACM UIST*.

Homburg, H., Mierswa, I., Moller, B., Morik, K., and Wurst, M. (2005), "A benchmark dataset for audio classification and clustering," in *ISMIR*.

Honicky, R., Brewer, E., Paulos, E., and White, R. (2008), "N-smarts: networked suite of mobile atmospheric real-time sensors," in *ACM NSDR*.

Ishii, H., Wisneski, C., Orbanes, J., Chun, B., and Paradiso, J. (1999), "PingPongPlus: design of an athletic-tangible interface for computer-supported cooperative play," in *ACM CHI*.

Jacob, C. and Steglich, S. (2010), "ConBrowse-contextual content browsing," in *Consumer Communications and Networking Conference (CCNC), 2010 7th IEEE*, pp. 1–5, IEEE.

Jahrer, M., Töscher, A., and Legenstein, R. (2010), "Combining predictions for accurate recommender systems," ACM KDD.

Kane, S., Avrahami, D., Wobbrock, J., Harrison, B., Rea, A., Philipose, M., and LaMarca, A. (2009), "Bonfire: a nomadic system for hybrid laptop-tabletop interaction," in *ACM UIST*.

Ke, Y., Tang, X., and Jing, F. (2006), "The design of high-level features for photo quality assessment," in *IEEE CVPR*.

Kennedy, L. and Ellis, D. (2004), "Laughter detection in meetings," in *NIST Meeting Recognition Workshop*.

Lang, P., Greenwald, M., Bradley, M., and Hamm, A. (1993), "Looking at pictures: Affective, facial, visceral, and behavioral reactions," *Psychophysiology*.

Lew, M., Sebe, N., Djeraba, C., and Jain, R. (2006), "Content-based multimedia information retrieval: State of the art and challenges," *ACM TOMCCAP*.

Li, I., Nichols, J., Lau, T., Drews, C., and Cypher, A. (2010), "Here's what i did: sharing and reusing web activity with ActionShot," in *ACM CHI*.

Li, X., Wu, C., Zach, C., Lazebnik, S., and Frahm, J. M. (2008), "Modeling and recognition of landmark image collections using iconic scene graphs," in *Proc. ECCV*.

Logan, B. (2000), "Mel frequency cepstral coefficients for music modeling," in *ISMIR*.

Lu, H., Pan, W., et al. (2009), "SoundSense: scalable sound sensing for people-centric applications on mobile phones," in *ACM MobiSys*.

Luther, M., Fukazawa, Y., Wagner, M., and Kurakake, S. (2008), "Situational reasoning for task-oriented mobile service recommendation," *The Knowledge Engineering Review*, 23, 7–19.

Mann, S. (1997), "Smart clothing: The wearable computer and wearcam," *Personal and Ubiquitous Computing*.

McKinney, M. F. and Breebaart, J. (2003), "Features for audio and music classification," in *ISMIR*.

Miluzzo, E. et al. (2008), "Sensing Meets Mobile Social Networks: The Design, Implementation and Evaluation of CenceMe Application," in *ACM Sensys*.

Miluzzo, E., Varshavsky, A., Balakrishnan, S., and Choudhury, R. (2012), "Tapprints: your finger taps have fingerprints," in *ACM MobiSys*.

Mistry., P. (2009), "The thrilling potential of SixthSense technology," *TED India*.

Mistry, P. and Maes, P. (2009), "SixthSense: a wearable gestural interface," in *ACM SIGGRAPH ASIA 2009 Sketches*.

Mohan, P., Padmanabhan, V. N., and Ramjee, R. (2008), "Nericell: Rich monitoring of road and traffic conditions using mobile smartphones," in *ACM SenSys*.

Morris, T., Blenkhorn, P., and Zaidi, F. (2002), "Blink detection for real-time eye tracking," *JNCA*.

Mun, M., Reddy, S., Shilton, K., Yau, N., Burke, J., Estrin, D., Hansen, M., Howard, E., West, R., and Boda, P. (2009), "PEIR, the personal environmental impact report, as a platform for participatory sensing systems research," in *ACM Mobisys*.

Nakakura, T., Sumi, Y., and Nishida, T. (2009), "Neary: conversation field detection based on similarity of auditory situation," *ACM HotMobile*.

Nilsson, M., Nordberg, J., and Claesson, I. (2007), "Face detection using local SMQT features and split up snow classifier," in *IEEE ICASSP*.

Ouyang, T. and Li, Y. (2012), "Bootstrapping personal gesture shortcuts with the wisdom of the crowd and handwriting recognition," in *ACM CHI*.

Paolucci, M., Broll, G., Hamard, J., Rukzio, E., Wagner, M., and Schmidt, A. (2008), "Bringing semantic services to real-world objects," *International Journal on Semantic Web and Information Systems (IJSWIS)*.

Paradiso, J., Leo, C., Checka, N., and Hsiao, K. (2002), "Passive acoustic knock tracking for interactive windows," in *ACM CHI extended abstracts*.

Partridge, K., Chatterjee, S., Sazawal, V., Borriello, G., and Want, R. (2002), "TiltType: accelerometer-supported text entry for very small devices," in *Proceedings of the 15th annual ACM symposium on User interface software and technology*, UIST '02.

Priyantha, B., Lymberopoulos, D., and Liu, J. (2011), "LittleRock: Enabling Energy-Efficient Continuous Sensing on Mobile Phones," *IEEE Pervasive Computing*.

Qin, C. o. (2011), "TagSense: a smartphone-based approach to automatic image tagging," in *ACM MobiSys*.

Rabiner, L. R. and Juang, B. H. (1993), *Fundamentals of speech recognition*, Prentice hall.

Reddy, S., Parker, A., Hyman, J., Burke, J., Estrin, D., and Hansen, M. (2007), "Image browsing, processing, and clustering for participatory sensing: Lessons from a dietsense prototype," in *ACM EmNets*.

Rosenfeld, L. and Morville, P. (1998), *Information Architecture for the World Wide Web.*, O.

Sarwar, B. et al. (2001), "Item-based collaborative filtering recommendation algorithms," in *ACM WWW*.

Sato, M., Poupyrev, I., and Harrison, C. (2012), "Touche: enhancing touch interaction on humans, screens, liquids, and everyday objects," in *ACM CHI*.

Schoelkopf, B., Burges, C., and Smola, A. (1998), *Advances in Kernel Methods - Support Vector Learning*, MIT Press.

Sohn, J., Kim, N., and Sung, W. (1999), "A statistical model-based voice activity detection," *IEEE SPL*.

Teevan, J., Cutrell, E., Fisher, D., and Others (2009), "Visual snippets: summarizing web pages for search and revisitation," in *ACM CHI*.

Teixeira, T. and Savvides, A. (2007), "Lightweight people counting and localizing in indoor spaces using camera sensor nodes," in *ACM/IEEE ICDSC*.

Torniai, C., Battle, S., and Cayzer, S. (2007), “Sharing, discovering and browsing geotagged pictures on the web,” *Multimedia Integration & Communication Centre, University Firenze, Firenze, Italy, Hewlett-Packard Development Company, LP.*

Virpioja, S., Vayrynen, J., Creutz, M., and Sadeniemi, M. (2007), “Morphology-aware statistical machine translation based on morphs induced in an unsupervised manner,” *Machine Translation Summit XI.*

Wang, Y., Lin, J., Annavaram, M., et al. (2009), “A framework of energy efficient mobile sensing for automatic user state recognition,” in *ACM MobiSys.*

Wilson, A. (2004), “TouchLight: an imaging touch screen and display for gesture-based interaction,” in *ACM ICMI.*

Wilson, A. (2010), “Using a depth camera as a touch sensor,” in *ACM international conference on interactive tabletops and surfaces.*

Yu, C., Lakshmanan, L., and Amer-Yahia, S. (2009), “It takes variety to make a world: diversification in recommender systems,” in *ACM EDBT.*

Zhang, T. and Kuo, C. C. J. (1998), “Audio-guided audiovisual data segmentation, indexing, and retrieval,” in *SPIE.*

Biography

Xuan Bao was born on November 20, 1985 in Nanjing, Jiangsu Province, China. After four years of study in the capital city of Beijing, in 2008, he earned the Bachelor of Engineering degree in Information Engineering from Beijing University of Posts and Telecommunications. He obtained his Masters of Science degree from Electronical and Computer Engineering department in May 2010 from Duke University. He also earned his Doctor of Philosophy degree in Computer Science from Duke University in June 2013 with thesis entitled “Enabling Context-Awareness in Mobile Systems”, under the supervision of Prof. Romit Roy Choudhury. Xuan’s research has been published in top conferences like ACM MobiCom, MobiSys, UbiComp, IEEE InfoCom and HotMobile. His paper on event coverage via collaborative sensing was selected as the best paper for MobiHeld and 2nd prize at ACM MobiCom Student Research Competition. Some of his research projects have received broad media coverage (http://synrg.ee.duke.edu/media.htm). During the summers, he has been working as research interns at Microsoft Research, AT&T Labs, and Bell Labs. In Fall 2013, Xuan will be joining Samsung Research at San Jose, CA as a Researcher.