Irina Diaconita, Andreas Reinhardt, Delphine Christin, Christoph Rensing: *Bleep Bleep! Determining Smartphone Locations by Opportunistically Recording Notification Sounds*. In: Proceedings of the International Conference on Mobile and Ubiqituous Systems: Computing, Networking and Services (MobiQuitous), December 2014.

Bleep Bleep! Determining Smartphone Locations by Opportunistically Recording Notification Sounds

Irina Diaconita
KOM
TU Darmstadt
Darmstadt, Germany
diaconita@kom.tu-darmstadt.de

Andreas Reinhardt
School of CSE
UNSW
Sydney, Australia
andreasr@cse.unsw.edu.au

Delphine Christin
University of Bonn,
Fraunhofer FKIE
Bonn, Germany
christin@cs.uni-bonn.de

Christoph Rensing
KOM
TU Darmstadt
Darmstadt, Germany
rensing@kom.tu-darmstadt.de

ABSTRACT

Every day, we carry our mobile phone in our pocket or bag. When arriving at work or to a meeting, we may display it on the table. Most of the time, we however do not change the ringtone volume based on the new phone location. This may result in embarrassing situations when the volume is too loud or missed calls and alarms when it is too low. In order to prevent such situations, we propose a non-intrusive opportunistic approach to determine the phone location and later adapt the ringtone accordingly. In our approach, we analyze the attenuation of the played ringtones to determine the nature of the surrounding elements. We evaluate our approach based on a prototypical implementation using different mobile phones and show that we are able to recognize the sole phone location with a precision of more than 94%. In a second step, we consider different surrounding environments and reach a precision of 89% for the phone position and 86% for the combination of the phone position and noise level of the environment, respectively.

Keywords

audio-based context detection, phone position, active environment probing

1. INTRODUCTION

Mobile phones are everywhere. Users carry them in their bags or pockets and leave them on tables during meetings or lunches. While the phones' location varies during the users' daily activities, the ringtone volume however remains the same across the day. Hence, users may miss phone calls if the volume is too low and their phone is stored in their bag. On the other side, a loud ringtone may attract the attention to the user in inappropriate situations, such as conferences or business dinners. In the current state-of-the-art, users can set and select different ringtone profiles, such as outdoor environment or do-not-disturb mode. They however need to manually activate them when their context changes. As shown by our daily experience, most users simply forget to do it. To avoid missing important calls or embarrassing situations, we therefore propose to assist the users in automatically adapting the ringtone volume to both users' and phones' contexts.

Related work on context detection often makes use of external sensors [21] or imposes constraints on the user regarding the phone position and the type of environment they are in [1, 10]. Hence, the unobtrusive nature of these application is not given. Our approach does not impose any constraints regarding the way the user carries his phone or the environment he's in. Furthermore, our approach is based on opportunistic recordings of the user's phone notifications. Thus, we avoid the battery drainage that duty cycling incurs. Furthermore, we can probe the environment at no extra cost and update the phone volume exactly when it is needed. The active probing of the environment allows us determine the current phone position based on the sound attenuation. In order to interpret the readings and infer the current situation, we apply feature engineering and use MFCC and Delta MFCC coefficients, coupled with machine learning techniques.

Our contributions can be summarized as follows. We first identify the current phone's location by analyzing the attenuation of the ringtone emitted by the phone in real-time.

By doing so, no additional audio signals need to be generated, which potentially could disturb the users. In our prototype implementation using Nexus 5, Samsung Galaxy Nexus and Galaxy S3 smartphones, we are able to distinguish between the following locations: (1) in a bag, (2) on a desk, (3) in users' hand, and (4) in a pocket. In addition to the phone's location, we also determine the noise level in its surrounding in order to adapt the volume of the ringtone accordingly. To demonstrate the performance of our scheme, we have collected 5,474 samples for ten users in different environments including offices, homes, outdoor, shopping malls, transportation means, and restaurants. We have further tested our solutions for ringtones, alarms and notification sounds. The results show that we can determine the phone position with more than 94% precision on average and the phone position together with the environment noise level with 86% precision.

The remainder of this paper is structured as follows. We first compare our solution to existing work in Section 2 and introduce our concept in Section 3 We give details about our prototype implementation in Section 4. We present our evaluation settings and evaluation results in Section 4.3, before addressing future work and concluding this paper in Section 6.

2. RELATED WORK

Usage of smartphone-integrated sensors for context-aware applications has become more and more popular, as their number and quality steadily increased. Existing approaches include medical applications using accelerometers [3] and gyroscopes [11] or cameras and microphones [32] for detecting emergencies or monitoring and guiding the overall behavior of the user [9]. GPS and WiFi traces are used for determining user mobility patterns and predicting their movements [35, 25] and cameras are used for indoor localization [6]. Another class of applications uses combinations of multiple sensors. Miluzzo et al [17] use GPS, accelerometer, camera and microphone readings to determine the user's activity, while Azizyan et al [1] use WiFi, microphone, camera, accelerometer and light sensor readings to establish the type and ambient of a given location.

We will focus in what follows on applications that use a smartphone's built-in microphone, since our approach relies on that sensor. This area encompasses two categories of approaches, one focusing on human-produced sounds and the other on environment sounds. The first class of solutions targets tasks such as speech recognition [28, 26], speaker recognition [13] and stress detection [14]. The category of smartphone applications focusing on environment sounds is the one that includes our approach as well. Extensive work has been put in medical applications which include cough detection and monitoring [10], physiological anomaly detection [8] and sleep monitoring [7]. Another class of applications focuses on detecting user activity and the events taking place in the proximity of the user [24, 16]. Musical genre recognition [36, 33] and music modelling [12] represent a popular research direction. Rana et al [23] propose a participatory urban noise mapping system. There are also general purpose frameworks like Auditeur [20], where the user can decide himself on the relevant sound samples to be used and types of events to be detected.

Schmidt et al [27] developed an early system to distinguish between five phone modes, two referring to the phone position (hand and table), two to the phone state (silent and general) and one to the environment (outside). For this purpose, they used a combination of external sensors including two accelerometers, a photodiode, temperature and pressure sensors, a CO gas sensor and a passive IR sensor.

Similarly to our solution, Miluzzo et al [18] propose a solution that relies on microphone, camera, accelerometer, gyroscope and compass readings to determine phone position. Unlike our solution, they only distinguish between two states of the phone: inside (a bag, a pocket, etc.) and outside (in the hand, on a desk, etc.) Furthermore, we do not only record environment sounds, but actively probe the environment by piggybacking the phone's notification sounds.

Siewiorek et al [30] use calendar information, as well as accelerometer, microphone and light sensor readings to distinguish between four states of the user: uninterruptible, idle, active, and default/normal. Although our goal is not to determine the user's interest on receiving a phone call, the distinction between the default and active states of the user, and thus the corresponding adaption of the phone ringtone volume is to be noted. However, our solution takes into account not only the environment noise level but also the phone position in order to adapt the volume of all notification sounds generated by the phone.

3. CONCEPT

As one previous study has shown [4], demographics play an important role regarding the position people carry their phones. Nevertheless, bag, hand, trouser pocket or belt clip are the most common positions, accounting for the preferences of roughly 90% of the respondents of the aforementioned study. Most commonly, women carry their phones in their bags, men in their pockets or belt clips, but there are also specific geographic locations where everyone favors carrying their phones in their hands. While all these positions imply the fact that the user is carrying his phone along, there are also situations when the user leaves it behind.

Phone position is essential for many context-aware applications that rely on specific constraints for gathering the data, like the phone being in a silent environment, exposing the camera or the phone being worn on the user's body. Most important though is to determine if the user left his phone behind. Thus, knowing the phone's position can help avoid inaccurate results and pointless battery drainage.

Furthermore, user comfort is affected by the volume of the phone's notification sounds and setting it manually is an overhead that is easily overseen. For instance, the ringtone should be loud during the train commute when the phone is in the bag, or phone calls will likely be missed. The same volume will be disturbing after arriving in the office, especially if the phone is on a desk or in the user's hand. The study of Böhmer et al [2] on mobile application usage behavior shows that the most used apps throughout the day are the communication ones. Furthermore, 49.6% of the chains of app usage are started by a communication app. This underlines the importance of properly adjusting the notification sounds even for this category of applications alone,

but the same stands true for all other notification sounds like message alerts and even more so for alarms: not hearing the wake up alarm because the phone is still in the bag could cause serious discontent.

To overcome these problems, we have exploited in the past the different propagation patterns of sound in the different phone positions [5]. Thus, we have played different pilot sequences, like Gaussian noise, and recorded them at the same time. Though the classification results were accurate and the required sound window was just 10 ms, the periodic probing of the environment was quite power hungry. Furthermore, in the previous application, if a change in phone position or user location had occurred in the middle of a monitoring interval and a notification came in during the same interval, there was no way to adapt the phone volume specifically for it.

With this solution, we opportunistically use the phone's notification sounds as pilot sequences. Thus, only when a phone call or a message is received, or when an alarm is triggered, the notification sounds are recorded and then classified with regard to phone position and environment noise level. Therefore, the volume of the phone is adapted for the specific incoming notifications. We rely on the assumption that the sound is attenuated differently for the most common phone positions. While this stands true to a certain extent of recordings of environment sounds alone, the attenuation patterns are much more obvious when using pilot sequences, as we will also show in Section 4.3. We only need a recording of a very short chunk of the ringtone to determine the phone position and afterwards the volume can be changed accordingly for the rest of the ringtone. The current approach is also energy efficient, as the phone's position is determined only when needed and by piggybacking a sound that was played anyways.

We distinguish between four different phone positions: pocket, hand, bag, and desk. These were decided based on [4], considering that pockets and belt clips are quite similar both regarding signal propagation and desired phone behavior. Furthermore, the case of a phone the user left behind is most likely included in the case of a phone lying on a desk or a table. The other alternative is that the phone is placed in a bag that was left behind.

Besides phone position, there is a second aspect that determines the optimal volume of the notification sounds, and that is the noisiness of the environment. Thus, we differentiate between silent environments (home, office) and noisy environments (outdoors, shops, cafés, public transportation means) without determining the specific situations, as they would not bring any extra information for determining the ideal ringtone volume.

We have considered and implemented two different architectures for our system. For both approaches we use a smartphone to record a chunk of the ringtone, alarm or notification sound, the smallest being 100 ms. Then, the first system, **PCD-P** (Phone Context Detection – Phone), preprocesses the sample and classifies it on the phone. Based on the classification result, the phone volume would be adapted accordingly.

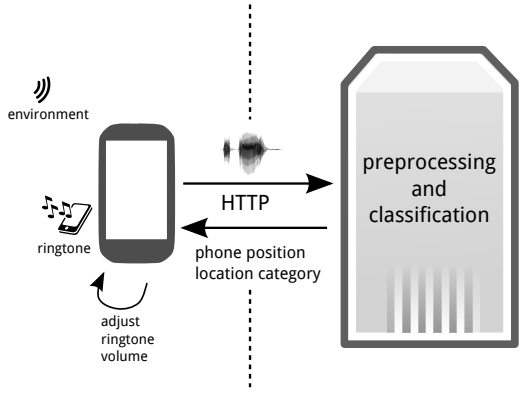

Figure 1: The PCD-S architecture

For the second system, **PCD-S** (Phone Context Detection – Server), the smartphone sends the recording to a server, where it is further preprocessed and classified. The server sends back to the phone the determined phone position and noise level, based on which the phone could adapt its volume accordingly. Figure 1 shows the architecture of PCD-S.

During our experiments, the delay caused by sending the data to the server and receiving back the classification result was, on average, 97 ms for a 100 ms recording. Thus, this second solution is feasible for adapting the volume for the very notification sound we collected the recording and carried out the classification. In Section 4.3 we present a comparison of the processing times and battery consumption for the two approaches.

On the server, the samples are first preprocessed, then we leverage machine learning technologies. First, we reduce the dimension space by extracting the feature array by using well-established features from the domain of audio signal analysis. Then, we classify the samples using tree and cluster-based approaches.

4. IMPLEMENTATION DETAILS

In this section we will first offer a brief overview of the differences between PCD-P and PCD-S, then we will go on and present the applications we used for collecting the samples and classifying the samples.

4.1 Differences Between PCD-P and PCD-S

What differentiates the two approaches is that one processes the data locally, while the other sends the data to a server, has the data processed on that server and receives back the result. PCD-P was done in Android and we have used Weka for Android [15] for the classification. PCD-S consists of an Android app for recording and sending the audio files to the server, and a classifier running on the server that uses Python and the scikit-learn library [22].

However, these are all the differences between the two systems. The steps followed and the methods used for the data collection itself and for the classification were the same for both approaches. Therefore, in what follows, we will present only one description for the implementation of the two components.

4.2 Sample Collection

For collecting the samples, we used an Android app that plays ringtones, alarm and notification sounds, and records them at the same time. The sampling rate was 44.1 kHz. Given that the optimal volume is to be decided after the classification, for recording the samples the app sets volume for each audio sample that is played in order to avoid clipping. Our processing pipeline on the server side includes silence removal, so we do not need to take measures against the few tens of milliseconds between the beginning of the recording and the beginning of the sound playback. We considered the ringtones, alarms and notification sounds of Samsung Galaxy Nexus, and ringtones of Samsung Galaxy S3. Despite some common audio tracks, most of them differ between phones.

4.3 Sample Classification

Our processing pipeline, as shown in Figure 2 includes windowing, silence removal, Fourier transformation, feature extraction and classification.

Thus, we first apply a rectangular windowing function with a fixed length window size to the recordings. Since notification sounds often contain silence between the repetitions of the same audio sample, it is preferrable to record more than just one window. After comparing the results for different window sizes, we decided to use 4096 sample windows, which corresponds to about 100 ms. Afterwards, we remove the windows that contain only silence. This is essential for the classification process, since, as we have already mentioned, ringtones and alarm sounds almost always include some silence intervals. Therefore, we calculate the signal energy for every window and remove all windows where it falls below a certain threshold. Next, we convert the signal from the time domain to the frequency space by using a Fourier transform.

Then, to reduce the dimensions of the data, we go on to extract the feature array. We experimented with four different types of features: Mel Frequency Cepstral Coefficients (MFCC), Delta Mel Frequency Cepstral Coefficients (Delta MFCC), the Band Energy (BE) and the Powerpectrum (PS). As expected MFCC and Delta MFCC obtained the best results. MFCC are "perceptually motivated" [33], emulating the behavior of the human auditory system [31]. This type of features has been successfully used for speech [34, 19] and speaker recognition [31], but also for music modeling and general audio classification tasks [20, 24]. Similarly to [20] and [31], we used the first N=13 MFCC coefficients. MFCC features are calculated for individual windows, so there is no temporal aspect taken into account. As a solution to this, Delta MFCC coefficients are calculated as the first order derivative of the MFCC coefficients with respect to time.

The final step was the classification of the features. We used the following classifiers: Decision Trees (DT), Random Forest (RF), Gaussian Naive Bayes (GNB), K-Nearest Neighbors (KNN), Support Vector Machine Classifier (SVC), and Gaussian Mixture Model (GMM). Given the way our data is clustered, tree-based and distance-based algorithms offered the best results as all feature vectors form clusters in the N-dimensional feature space. This happens due to the fact that the feature arrays are made up of numerical values

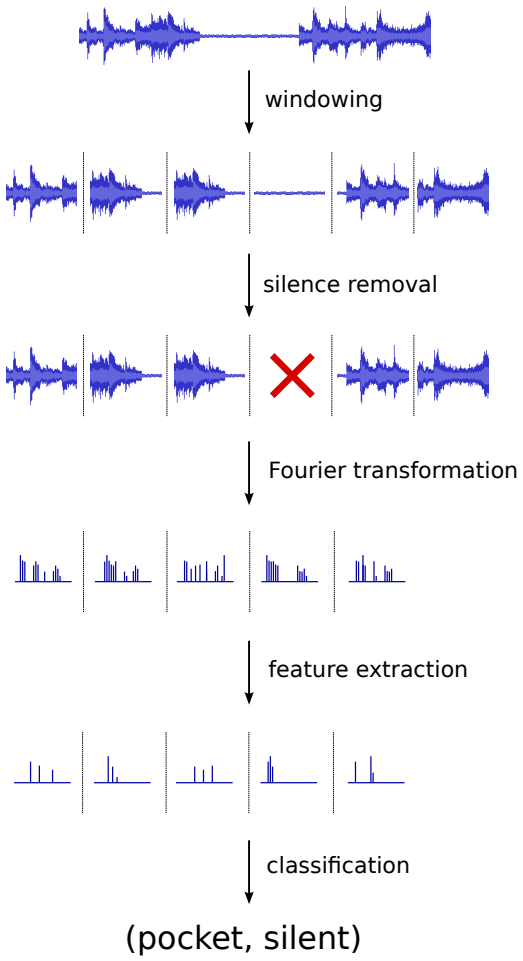

Figure 2: Processing pipeline of the audio samples

expressing distribution (spectral shape) of the signal's frequency components. These values are almost constant for each kind of environment, while differing between the different kinds of environment, which gives us the basis for our work.

5. EVALUATION

In this section we evaluate our system from multiple points of view. We first describe our evaluation setup and compare the energy and time efficiency of PCD-P and PCD-S in Subsection 5.1. Afterwards, we present an overview of the types of recordings we use. In the following subsections we proceed to evaluate and compare the accuracies of the different types of features and classifiers in determining the phone position based on ringtones, alarms and notification sounds respectively.

5.1 Evaluation Setup

We tested our approach on a Samsung Galaxy Nexus, a Samsung Galaxy S3 and a Nexus 5, which could all support a 44.1 kHz sampling rate required for an optimal classification.

Both PCD-P and PCD-S use the same methods and, thus, yield the same results, so we measured their battery consumption and processing times in order to decide which sys-

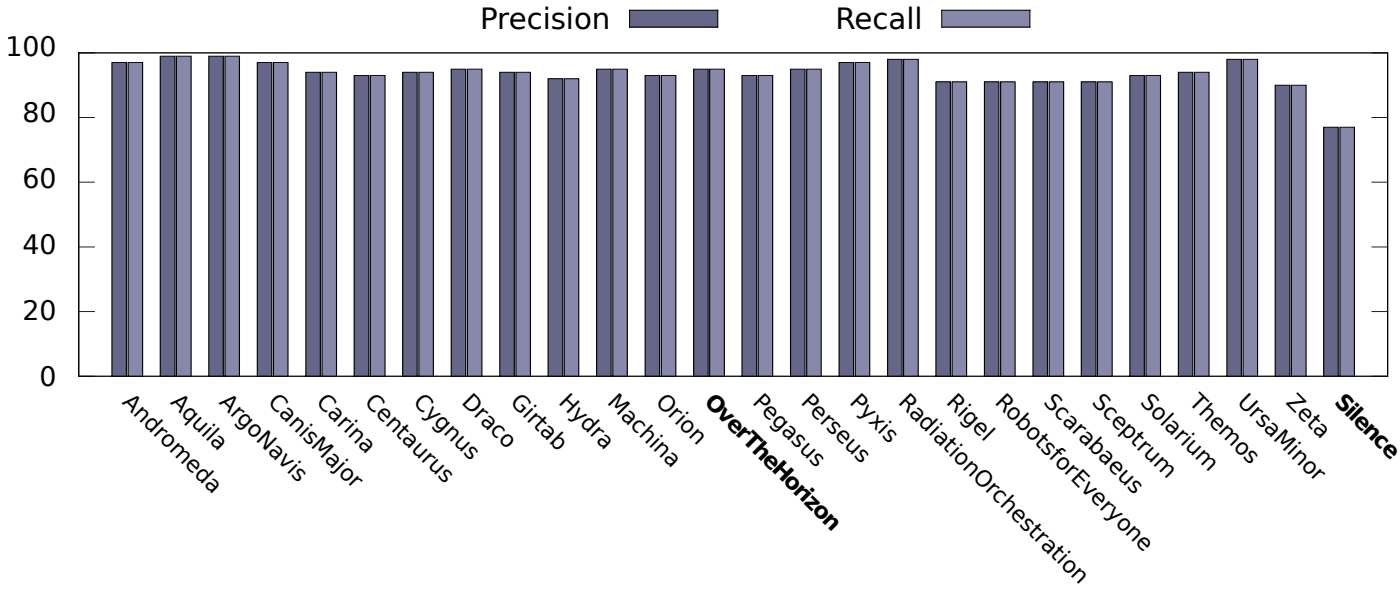

Figure 3: Classification results for all Samsung Galaxy Nexus ringtones when using MFCC and Random Forest

tem to use for the evaluation.

As far as the energy consumption is concerned, the sample collection step is similar for both approaches. The difference stems from the fact that one app sends the files to the server and one classifies it locally. Therefore, we measured and compared the drop in battery percentage caused by processing one hour's worth of 100 ms samples for the two cases. This means, for a 10-fold cross validation approach, that the system was trained with nine hours' worth of recordings. We picked the time interval of one hour since it is quite unlikely for a user to receive enough phone calls and notifications to need to classify more than 36000 samples over one day.

PCD-S has caused a 4% drop in battery percentage, whereas PCD-P has lead to a 36% drop in battery percentage. The first approach took 20 minutes to send the data to the server and receive the answers, plus 0.416 minutes to classify the data on the server. PCD-P needed 238 minutes to finish the task.

As far as processing times are concerned, classifying one 100 ms sample in the afore-mentioned setting took on average 0.692 ms on the server and 397 ms on the phone.

5.2 Types of Recordings

We considered all the pre-installed ringtones of the Galaxy Nexus and the Galaxy S3, as well as the pre-installed alarms and notifications for the Galaxy Nexus. With very few exceptions, the original audio files that came with the phone were different for each model. We had two users collect the ringtone samples in noisy environments. Furthermore, we compare the results of the ringtone recordings with those of the environment sounds alone, in order to show the improvement brought by our system.

The alarm sounds are most commonly used as wake-up alarms, so the user is most likely in a silent environment. Therefore, the alarm samples were only recorded in a silent environment. The notification sounds, which have a wide range of usage, from receiving text messages to system notifications, were recorded in noisy settings.

We studied in more detail the case of the ringtone "Over the Horizon", one of the most popular Samsung ringtones. For this purpose, we had ten users gather samples in both silent and noisy environments. While we do not differentiate between the specific situations, the settings in which the recordings were gathered include offices and homes as silent environments and outdoors, public transportation means, shops, cafés and parties as noisy environments. Furthermore, we studied in more detail the case of "Proxima", one of the most common Samsung notification sounds, including both silent and noisy settings in the evaluation.

Based on the reasons presented in Section 3, we considered four main phone positions: pocket, bag, hand, and desk. We had 5,474 samples in total, which make up more than 880 minutes of recordings. For the training and evaluation of the classifiers we applied 10-fold cross-validation, thus using 90% of the data for training and 10% for the evaluation.

5.3 Classification of Ringtone Recordings

In what follows, we will analyze the performance of the various types of features and classifiers for the ringtones that come with Samsung Galaxy Nexus and Galaxy S3. Samsung Galaxy Nexus has 25 pre-installed ringtones, with an average duration of 8.5 seconds, while Galaxy S3 has 34 pre-installed ringtones, with an average duration of 25.2 seconds.

Figure 3 presents the classification results for the 25 pre-installed Samsung Galaxy Nexus ringtones in noisy environments, taking into account the four positions we mentioned in the beginning – bag, hand, pocket, and desk – while for

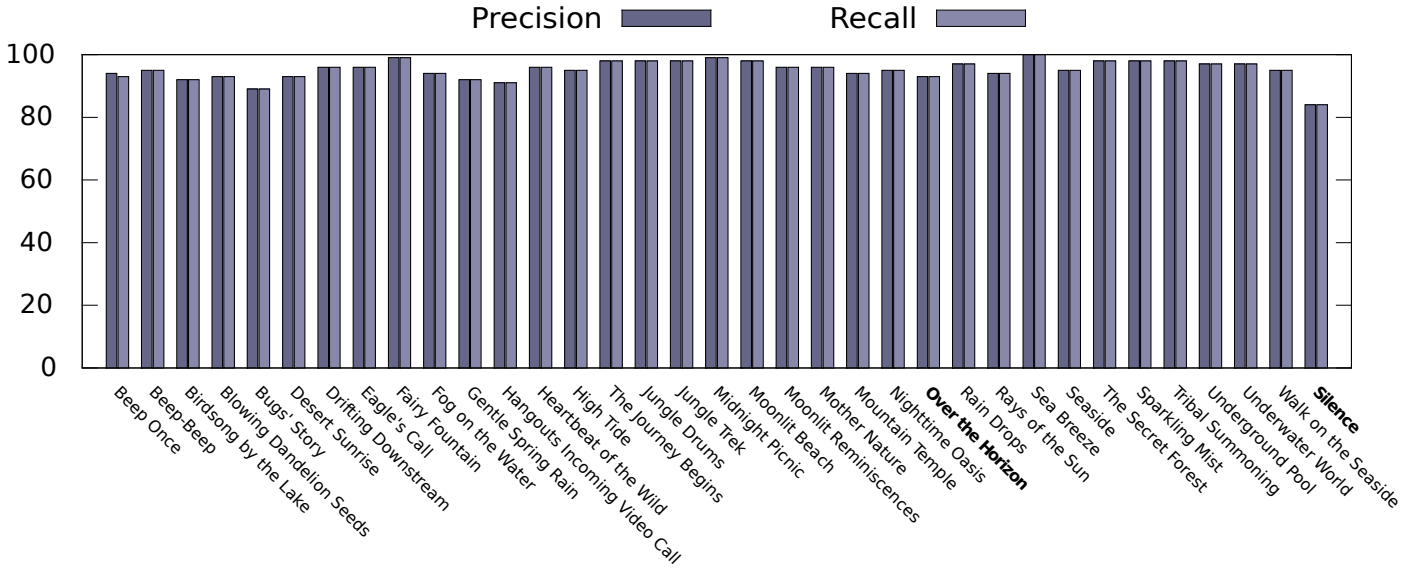

Figure 4: Classification results for all Samsung Galaxy S3 ringtones using Delta MFCC and Random Forest

the desk also distinguishing between the two possible ways of placing the phone: facing the ceiling or facing the desk. We used 1,600 samples, totaling 200 minutes.

In this case, we have used the MFCC coefficients as features and Random Forest as the classifier. Our choice was motivated by the fact that MFCC was proven to offer very good results in audio classification problems [24, 20], while Random Forest is particularly suited for the way our data was clustered.

For comparison, we also included the classification results of the recordings of the environment sounds alone, which have noticeably worse results. Thus, the ringtone recordings have an average precision of 94.4%, while the recordings of the environment sounds alone reach only 77%.

Figure 4 shows the precision and recall for the 34 Galaxy S3 ringtones. We used 1,329 samples, with a total duration of 560 minutes. In this case, we decided to pick Delta MFCC as feature. MFCC is calculated over the individual windows, thus not taking into account the temporal aspect, while Delta MFCC is calculated as the first derivative of the MFCC with respect to time. Thus, Delta MFCC is expected to produce better results then MFCC.

The classification results are similar to those of the Galaxy Nexus, although, on average slightly better. Again, one can notice the clear improvement brought by the ringtone recordings compared to the environment sounds recordings, from 84% precision to 95.6% on average.

5.4 Ringtone Case Study: "Over the Horizon"

In what follows, we will analyze the classification results of one of the most common Samsung ringtones, "Over the Horizon". Figure 5 shows the spectrum of recordings of the ringtone taken in the four main phone positions, as well as the spectrum of the ground truth. Thus, the differences in the spectral shape are quite obvious, especially for positions

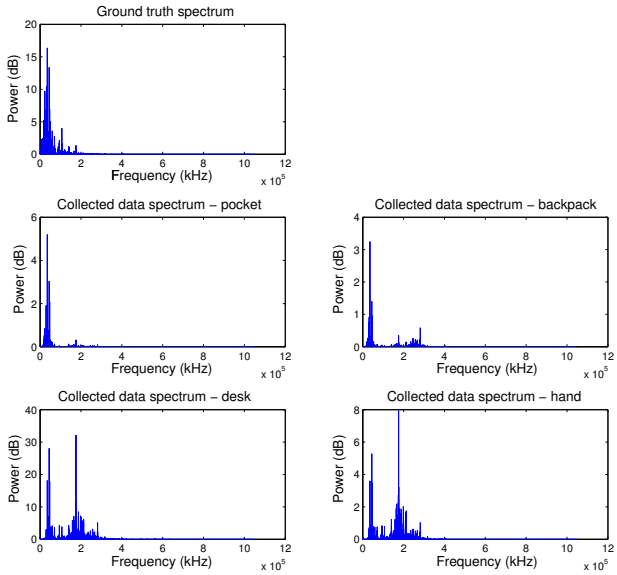

Figure 5: The spectrum of the "Over the Horizon" ringtone recordings for various phone positions

where we expected the sound to be muffled, like in a pocket or, to a lesser extent, in a bag. While differences in spectral shape are less pronounced when comparing the desk and the hand, one can notice the clear distinctions in signal power. As expected, the power values for the desk have a significantly greater range then those for the other phone positions.

To test our concept, we gathered recordings from ten users in various kinds of silent and noisy environments (home, office, outdoors, public transportation means, shops, restaurants, cafés) and aimed to distinguish between the four most common phone positions. We used three types of phones, Samsung Galaxy Nexus, Samsung Galaxy S3 and Nexus 5.

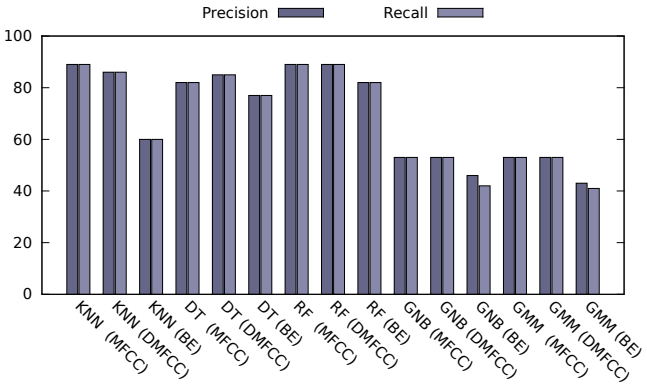

Figure 6: Comparison of classifier results for "Over the Horizon"

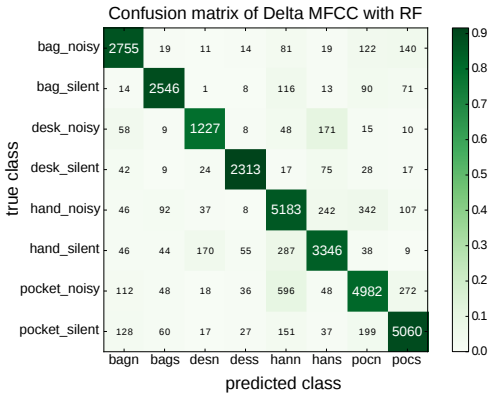

Figure 7: Classification results for position and environment noise level when using "Over the Horizon" as pilot sequence

Figure 6 shows the precision and recall of the different combinations of feature arrays and classifiers that we experimented with. It can be clearly noticed that MFCC and Delta MFCC yield the best results regardless of the overall performance of the classifier and its suitability for the current problem. Tree-based approaches are less affected by the type of feature then other classifiers, having good and very good results in all cases. Out of these, Random Forest has the best overall results. In contrast to Random Forest, the second best classifier, K-Nearest Neighbors is much more sensitive to the kind of feature array that is used. Thus, it offers very good results for MFCC and Delta MFCC, but is quite inaccurate when using BE.

Next, we will look closer at the classification results of Delta MFCC and Random Forest,when trying to determine the phone position and the noise level of the environment. Given that the data is clustered on two levels, Random Forest is one of the most appropriate classifiers for our problem. Figure 7 presents the corresponding confusion matrix. As expected, the precision and recall dropped slightly, reaching 86%, given the increased complexity of the classification problem. Thus, it can be noticed that in some of the noisy situations, the noise itself leads to a mislabeling of the phone position, such confusions happening for instance between the desk and the hand. Despite these, the overall performance of our classifier in determining main phone positions and the noise levels of the environment is very good.

5.5 Classification of Alarm Sound Recordings

In what follows, we will look into the classification results of the six Samsung Galaxy Nexus alarm sounds. Figure 8 compares the F-score results of the classification using two different types of features, MFCC and Delta MFCC, and the same classifier, Random Forest. We picked Random Forest since it was the classifier that yielded the best results on average, and want to compare the precisions of MFCC and Delta MFCC. Given that as Delta MFCC takes into account also the variation of the MFCC coefficients in time, we expect it to lead to better results. We consider a balanced F-score, calculated according to [29], as shown in equation (1).

$$F\text{-}score = \frac{2}{\frac{1}{recall} + \frac{1}{precision}} \tag{1}$$

One can notice that for "Cesium" and "Hassium" Delta MFCC has slightly worse results than MFCC. This happens due to the structure of these audio files, as 50% or more of their duration is made up of silence, which is removed during the preprocessing. Thus, less than half of the training data was used and this led to poorer classification results than those of the other audio files, given that we collected and used an equal number of samples for all the alarm sounds.

However, given that "Cesium" is the default Galaxy Nexus alarm sound, we will have a closer look at its classification results. Delta MFCC and Random Forest yield a 96% F-score. As shown in Figure 9, most confusions between the situations are negligible, the only one standing out being the one between the desk and the user's hand. This is not unexpected, given the similarities in the sound propagation patterns between the two situations and the lower number of windows that were left for training after the silence removal.

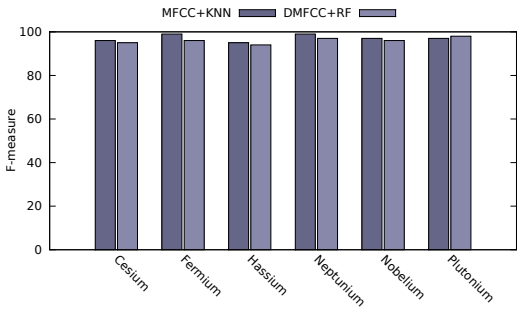

Figure 8: Phone position classification results for all Samsung Galaxy Nexus alarm sounds

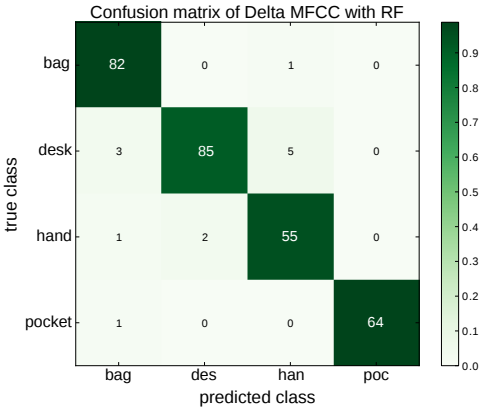

Figure 9: Classification results for "Cesium" alarm sound

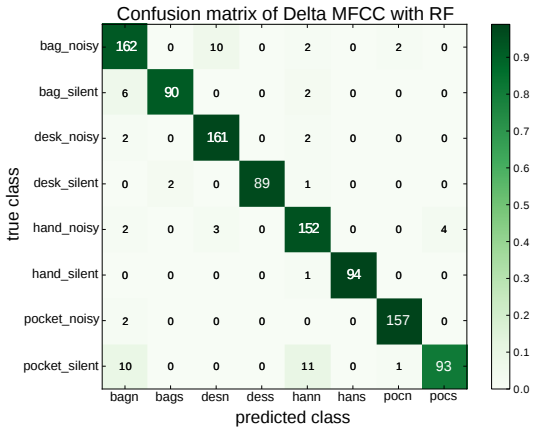

Figure 11: Classification results for "Proxima" notification sound

5.6 Classification of Notification Sound Recordings

Next, we evaluated our solution on the 17 notification sounds that are included by default in the Samsung Galaxy Nexus, gathering over 2,000 sound samples. The clips have lengths between 0.6 and 4.4 seconds and are made up of up to 58% silence.

Figure 10 compares the F-scores of the classification results for all the notification sounds for two combinations of types of features and classifiers that provide the best results. Here as well it can be that the sound files that contain significantly longer periods of silence percentage-wise, like "Hojus" and "Lalande" have also lower F-scores.

When comparing the performances of the two classifiers, Random Forest and Delta MFCC have obviously better results. This highlights not only the better performances of DMFCC compared to MFCC, but also that Random Forest is being more suited for this case. Thus, given the structure of the data, building multiple trees is more efficient and accurate than clustering the data on N=13 dimensions, like K-Nearest Neighbors does.

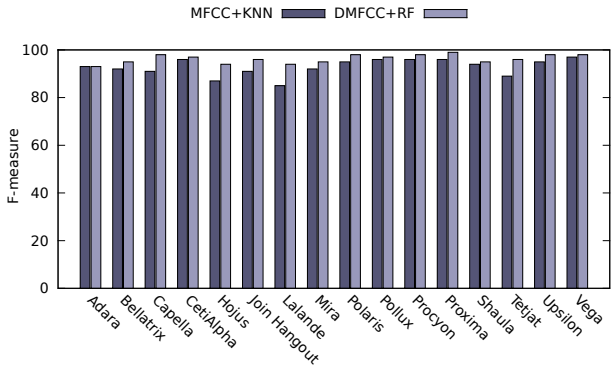

Figure 10: Phone position classification for all Samsung Galaxy Nexus notification sounds

Next, we shall focus on the default notification sound for Galaxy Nexus, which is "Proxima". We used Delta MFCC and Random Forest to classify the phone the position and the noise level of the environment, obtaining a 94% precision, recall and F-score. As it can be noticed in Figure 11, there are very few false positives and false negatives, the overall results of the classification being very good.

6. CONCLUSIONS

Smartphones have seen numerous exciting new functionalities added in recent years. Still, telephony and messaging remain at their core and tones for incoming calls and notifications are played back frequently. Their volume and the additional usage of the vibration motor, however, need to be manually set by the user according to the environment. With many context changes throughout the day, forgetfulness may lead to inappropriate settings and awkward moments. We have hence presented a system that allows for the opportunistic classification of the environment based on recording the tones whilst they are being played back. With only 100 ms of sampling required, the environment of the phone can be quickly determined and the notification tone volume adapted. Our evaluation has shown that precision and recall values in excess of 90% could be achieved for the location detection based all ringtones deployed on the Galaxy Nexus phone. Furthermore, "Over the Horizon", one of the most popular Samsung ringtones, has shown very good performance to also classify the ambient noise level, with precision values of 86% regardless of the phone position. Our evaluations have been based on more than 880 minutes of ringtone recordings collected from three different smartphone models. They unambiguously show that our approach is viable and can be easily implemented in order to automatically adjust the notification tone volume based on the user's current environment.

6.1 Next Steps

As a future extension, we want to monitor the user's preferred phone locations and habits. The ringtones and alarms sounds would definitely be useful for this endeavor, but the other notification sounds could play a bigger role. Thus messages, emails, Skype or other messenger app notifications, as

well as notifications coming from various apps or the system are much more common and numerous during the day than phone calls and can offer in an opportunistic fashion a much better image of the user's habits. Furthermore, it would be interesting to explore a potential correlation between the phone position and the user's current activity or type of activity. For instance, it might be the case that the user always leaves the phone on his desk when in a meeting or working at his PC, or that he always carries the phone in a bag when walking or taking public transportation means.

Acknowledgment

This work is supported by funds from the German Federal Ministry of Education and Research under the mark 01PF10005B and from the European Social Fund of the European Union (ESF). The responsibility for the contents of this publication lies with the authors.

7. REFERENCES

[1] M. Azizyan, I. Constandache, and R. Roy Choudhury. SurroundSense: Mobile Phone Localization via Ambience Fingerprinting. In *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking*, 2009.

[2] M. Böhmer, B. Hecht, J. Schöning, A. Krüger, and G. Bauer. Falling Asleep with Angry Birds, Facebook and Kindle: A Large Scale Study on Mobile Application Usage. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, 2011.

[3] A. K. Bourke, J. O'Brien, and G. Lyons. Evaluation of a Threshold-based Tri-axial Accelerometer Fall Detection Algorithm. *Gait & Posture*, 2007.

[4] Y. Cui, C. Jan, and I. Fumiko. A Cross Culture Study on Phone Carrying and Physical Prsonalization. In *Usability and Internationalization. HCI and Culture*, 2007.

[5] I. Diaconita, A. Reinhardt, F. Englert, D. Christin, and R. Steinmetz. Do You Hear What I Hear? Using Acoustic Probing to Detect Smartphone Locations. In *Proceedings of the 1st Symposium on Activity and Context Modelling and Recognition*, 2014.

[6] R. Elias and A. Elnahas. An Accurate Indoor Localization Technique Using Image Matching. In *Proceedings of the International Conference on Intelligent Environments*, 2007.

[7] T. Hao, G. Xing, and G. Zhou. iSleep: Unobtrusive Sleep Quality Monitoring Using Smartphones. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, 2013.

[8] D. Hong, S. Nirjon, J. A. Stankovic, D. J. Stone, and G. Shen. Poster Abstract: a Mobile-Cloud Service for Physiological Anomaly Detection on Smartphones. In *Proceedings of the 12th International Conference on Information Processing in Sensor Networks*, 2013.

[9] M. Kranz, A. Möller, N. Hammerla, S. Diewald, T. Plötz, P. Olivier, and L. Roalter. The Mobile Fitness Coach: Towards Individualized Skill Assessment Using Personalized Mobile Devices. *Pervasive and Mobile Computing*, 2013.

[10] E. C. Larson, T. Lee, S. Liu, M. Rosenfeld, and S. N. Patel. Accurate and Privacy Preserving Cough Sensing Using a Low-cost Microphone. In *Proceedings of the 13th International Conference on Ubiquitous Computing*, 2011.

[11] Q. Li, J. A. Stankovic, M. A. Hanson, A. T. Barth, J. Lach, and G. Zhou. Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information. In *Proceedings of the 6th International Workshop on Wearable and Implantable Body Sensor Networks*, 2009.

[12] B. Logan et al. Mel Frequency Cepstral Coefficients for Music Modeling. In *Proceedings of the 11th International Society for Music Information Retrieval Conference*, 2000.

[13] H. Lu, A. J. Bernheim Brush, B. Priyantha, A. K. Karlson, and J. Liu. SpeakerSense: Energy Efficient Unobtrusive Speaker Identification on Mobile Phones. In *Pervasive Computing*. 2011.

[14] H. Lu, D. Frauendorfer, M. Rabbi, M. S. Mast, G. T. Chittaranjan, A. T. Campbell, D. Gatica-Perez, and T. Choudhury. StressSense: Detecting Stress in Uconstrained Acoustic Environments Using Smartphones. In *Proceedings of the ACM Conference on Ubiquitous Computing*, 2012.

[15] R. J. Marsan. Weka for Android. *GitHub Repository*, 2011.

[16] A. Mesaros, T. Heittola, A. Eronen, and T. Virtanen. Acoustic Event Detection in Real Life Recordings. In *18th European Signal Processing Conference*, 2010.

[17] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell. Sensing Meets Mobile Social Networks: the Design, Implementation and Evaluation of the CenceMe Application. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, 2008.

[18] E. Miluzzo, M. Papandrea, N. D. Lane, H. Lu, and A. T. Campbell. Pocket, Bag, Hand, etc.-Automatically Detecting Phone Context Through Discovery. In *Proceedings of the ACM International Workshop on Sensing Applications on Mobile Phones*, 2010.

[19] S. Molau, M. Pitz, R. Schluter, and H. Ney. Computing Mel-Frequency Cepstral Coefficients on the Power Spectrum. In *Proceedings the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2001.

[20] S. Nirjon, R. F. Dickerson, P. Asare, Q. Li, D. Hong, J. A. Stankovic, P. Hu, G. Shen, and X. Jiang. Auditeur: A Mobile-Cloud Service Platform for Acoustic Event Detection on Smartphones. In *Proceedings of the The 11th International Conference on Mobile Systems, Applications, and Services*, 2013.

[21] S. Nirjon, R. F. Dickerson, Q. Li, P. Asare, J. A. Stankovic, D. Hong, B. Zhang, X. Jiang, G. Shen, and F. Zhao. MusicalHeart: a *Hearty* Way of Listening to Music. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, 2012.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in

Python. *Journal of Machine Learning Research*, 2011.

[23] R. K. Rana, C. T. Chou, S. S. Kanhere, N. Bulusu, and W. Hu. Ear-phone: an End-to-end Participatory Urban Noise Mapping System. In *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks*, 2010.

[24] M. Rossi, S. Feese, O. Amft, N. Braune, S. Martis, and G. Troster. AmbientSense: A Real-Time Ambient Sound Recognition System for Smartphones. In *Proceedings of the International Conference on Pervasive Computing and Communications*, 2013.

[25] S. Scellato, M. Musolesi, C. Mascolo, V. Latora, and A. T. Campbell. Nextplace: A spatio-temporal prediction framework for pervasive systems. In *Pervasive Computing*. Springer Berlin Heidelberg, 2011.

[26] R. Schlegel, K. Zhang, X.-Y. Zhou, M. Intwala, A. Kapadia, and X. F. Wang. Soundcomber: A Stealthy and Context-Aware Sound Trojan for Smartphones. 2011.

[27] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. Van Laerhoven, and W. Van de Velde. Advanced Interaction in Context. In *Handheld and Ubiquitous Computing*, 1999.

[28] R. V. Shannon, F.-G. Zeng, V. Kamath, J. Wygonski, and M. Ekelid. Speech Recognition With Primarily Temporal Cues. *Science*, 1995.

[29] W. M. Shaw Jr, R. Burgin, and P. Howell. Performance Standards and Evaluations in IR Test Collections: Cluster-based Retrieval Models. *Information Processing & Management*, 1997.

[30] D. Siewiorek, A. Krause, N. Moraveji, A. Smailagic, J. Furukawa, K. Reiger, F. L. Wong, and J. Shaffer. SenSay: A Context-Aware Mobile Phone. In *Proceedings of the 16th International Symposium on Wearable Computers*, 2012.

[31] S. Srivastava, S. Bhardwaj, A. Bhandari, K. Gupta, H. Bahl, and J. Gupta. Wavelet Packet Based Mel Frequency Cepstral Features for Text Independent Speaker Identification. In *Intelligent Informatics*. 2013.

[32] B. U. Töreyin, Y. Dedeoğlu, and A. E. Çetin. HMM Based Falling Person Detection Using Both Audio and Video. In *Computer Vision in Human-Computer Interaction*. 2005.

[33] G. Tzanetakis and P. Cook. Musical Genre Classification of Audio Signals. *IEEE Transactions on Speech and Audio Processing*, 2002.

[34] R. Vergin, D. O'shaughnessy, and A. Farhat. Generalized Mel Frequency Cepstral Coefficients for Large-Vocabulary Speaker-Independent Continuous-Speech Recognition. *IEEE Transactions on Speech and Audio Processing*, 1999.

[35] L. Vu, Q. Do, and K. Nahrstedt. Jyotish: Constructive Approach for Context Predictions of People Movement from Joint Wifi/Bluetooth Trace. *Pervasive and Mobile Computing*, 2011.

[36] A. Wang. An Industrial Strength Audio Search Algorithm. In *Proceedings of the 4th Symposium Conference on Music Information Retrieval*, 2003.