Smartphone interruptibility using density-weighted uncertainty sampling with reinforcement learning

Robert Fisher
Machine Learning Department
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3815
Email: rwfisher@cs.cmu.edu

Reid Simmons
Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3815
Email: reids@cs.cmu.edu

***Abstract*—We present the *In-Context* application for smartphones, which combines signal processing, active learning, and reinforcement learning to autonomously create a personalized model of interruptibility for incoming phone calls. We empirically evaluate the system, and show that we can obtain an average of 96.12% classification accuracy when predicting interruptibility after a week of training. In contrast to previous work, we leverage *density-weighted uncertainty sampling* combined with a reinforcement learning framework applied to passively collected data to achieve comparable or superior classification accuracy using many fewer queries issued to the user.**

I. Introduction

With the proliferation of mobile devices, interruptibility has become a defining problem. Users often forget to change the settings on their mobile devices throughout the day, which results in inappropriate interruptions or important notifications being missed [12]. However, modern mobile devices are being outfitted with broad sensing suites and relatively powerful computational capabilities, giving those devices the ability to monitor and adapt to changing social contexts. We introduce the *In-Context* smartphone application, which uses a combination of signal processing, active learning, and supervised machine learning to create a personalized policy for changing a user's ringtone autonomously. This application leverages a smartphone's GPS, accelerometer, microphone, proximity sensor, and computing power to identify similar contexts and act according to the user's observed historical preferences. The techniques being used in this application could be applied in any setting in which we wish to personalize an instrumented system—for instance on a sensor-equipped power-wheelchair, we may wish to generate customized reminders for the user to conduct pressure relief exercises.

There are several practical and theoretical challenges involved in building such a system. On a practical level, the system should be able to operate using only those sensors found in a standard touch-screen smartphone, without requiring the user to wear additional instrumentation. Furthermore, the power consumption of the system must be such that the user can continue to use his or her phone throughout the day. On a theoretical level, a variety of latent variables, which the onboard sensors cannot observe, may factor into users' preferences in different contexts. Also, because the system is being designed to reduce the intrusiveness of the device, unnecessary or inappropriate queries of the user should be avoided.

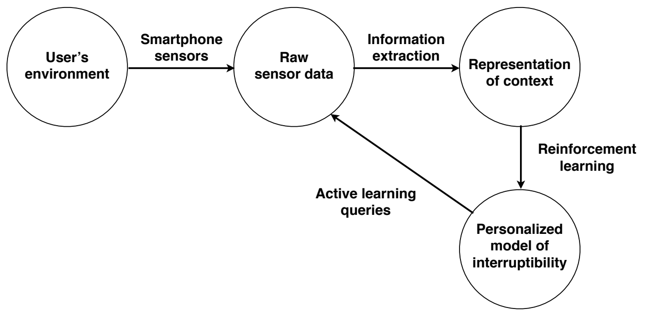

Fig. 1. The In-Context framework

An overview of the In-Context system is shown in Figure 1. We use information extraction algorithms on the phone's sensor data to build a representation of the user's current context. In particular, we use a voice activity detection algorithm on audio data and a phone posture recognition algorithm on accelerometer and proximity sensor data. Given a representation of the current context, we passively monitor the user's changes to their hardware ringer setting, as well as their responses to incoming calls. We treat this behavior as a noisy reinforcement signal, because users may have forgotten to change their ringer setting, or they may be basing their decision on latent variables. For users who consistently change their ringer setting according to their preferences, the model trained on the passively collected data is sufficient. However, we also allow the system to use an active learning framework to select contexts in which to query the user about their true preferences in cases where the system has significant uncertainty about the correct setting in the current context.

Most previous attempts to determine user interruptibility, in mobile as well as desktop applications, have relied on active user input to determine their preferences [1], [7], [19]. Some of this work has explicitly considered the user's current interruptibility when deciding whether to issue prompts for input [14], but all of these systems perform poorly with users who are often unwilling or unable to respond to queries.

Our system expands on previous work in two central ways.

First, we allow the system to learn about users by passively observing their day-to-day behavior with their phone, such as when they change the ringer setting or respond to incoming calls. This allows us to learn an effective model using either a small number of questions or no questions at all. Secondly, we leverage a new metric for actively learning, one not previously used in the application of interruptibility. While most existing systems have issued questions to the user based on *uncertainty sampling* [11], we propose the use of *density-weighted uncertainty sampling* [17], which considers how representative the current context is of other contexts in the user's data-set, in addition to the system's uncertainty about the current user preference. We discovered that this approach allows us to attain an aggregate classifications accuracy of 96%, while requiring fewer queries of the user than previous approaches such as uncertainty sampling.

II. Related Work

Researchers have been studying interruptibility in the field of human-computer interaction for many years. This research was initially in the domain of desktop computers, when multi-tasking applications began introducing irritating interruptions to users while they worked. Early research focused on using only information about the state of the user's software to determine interruptibility [15], but more recent work has instead been using sensors to perceive the user's environment in order to achieve context-aware interruptibility [3], [5].

The growing popularity of mobile devices has reinvigorated interest in determining interruptibility. Many users have a consistent internal model of when and why they want their mobile device turned on [20], but many of us often forget to change the device's settings at the appropriate times. For many users, autonomously learning their preferences requires the ability to sense factors in the user's current environment. To achieve this, many previous context-aware systems have required users to place wearable sensors around their body [4], [18]. However, modern smartphones, like the Apple iPhone or Google's Android platform, feature an array of useful sensors that can allow us to circumvent the need for specialized hardware. Only a small number of systems have been designed to leverage the power of these new hardware platforms when predicting interruptibility [14].

Nearly all previous work in the field of interruptibility has relied on actively asking the user questions about their preferences in different contexts [6], [8]. Unfortunately, this approach leads to scenarios where the system interrupts the user at inopportune moments in order to issue a query about their preferences in the current context. Some systems have alleviated this drawback somewhat by introducing a decision-theoretic component that allows the system to ask questions only when the cost of interruption is low [14]. However, when there is a high cost of asking questions, or when the user ignores the queries, many of these system fail to perform better than random guessing.

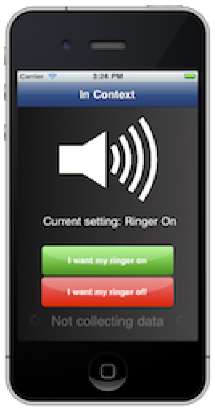

Fig. 2. User interface for data collection

III. Dataset

Data was collected over a seven-day period from five volunteers using iPhone brand smartphones. The data collected included readings from the phone's 3-axis accelerometer, GPS unit, microphone, proximity sensor, as well as user activity and responses to incoming phone calls. The state of the user's hardware ringer switch (on or off) was also collected with every sample. Data was read from the sensors for only a ten-second period once every ten minutes, in order to preserve the phone's battery life. Under these conditions, we estimated that our system is able to run for 23 hours continuously on an iPhone 4 handset, or 19 hours continuously on an iPhone 3GS. For purposes of system evaluation, each user was also queried approximately once every two hours to provide their true preference for the ringer setting in the current context. In addition, each user was also permitted to provide their current preference to the system at any time, which would postpone the next prompt for user input by two hours. The graphical user interface is shown in Figure 2.

After the raw data was collected, it was passed through information extraction algorithms on board the smartphone, and the output of these algorithms was stored. The details of the information extraction are provided in section IV. In particular, we represented a user's context using seven core pieces of information, described in Table I. We have done our best to minimize the invasiveness of the system on the user's privacy by encrypting or deleting data as much as possible. Although some private information is collected, previous work suggests that most users are willing to divulge some private information in return for services with high utility [19].

There are other modes of data which could be collected on a smartphone but were not used in this study. For instance, only one of our users reported keeping their smartphone calendar up-to-date, so calendar events were not collected in our dataset. Additionally, we did not record the identity of incoming callers at the request of several of our study participants.

IV. System Overview

This section describes the primary components of the In-Context system. The first subsections present the information extraction algorithms for phone posture recognition and voice activity detection, as well as the smoothing routine applied

Context feature	Details
Phone posture	A number in the set $\{0, 1, 2\}$ indicating if the phone is 0: Resting on table 1: In user's hand 2: In pocket or bag
Voice Activity	A bit indicated the presence of human speech
Sound level	A number in the set $\{0, 1, 2\}$ indicating if the sound level is quiet, average, or high.
Hour	An integer in 0-23 indicating the hour of the day
Weekday	An integer in 1-7 indicating the day of the week
Location	The latitude and longitude of the current location. These numbers are hashed using a secret key before being recorded to preserve privacy.
Ringer switch	The current setting of the hardware ringer switch.

TABLE I
THE REPRESENTATION OF A USER'S CONTEXT

#	Feature	Description
1	Fourier mean	The sample mean of the magnitudes of all Fourier coefficients in the sample.
2	Fourier variance	The sample variance of the magnitudes of all the Fourier coefficients.
3	Total signal power	The sum of the squared magnitudes of all the Fourier coefficients
4	Mid-range power	The sum of the squared magnitudes of the Fourier coefficients in the 250-600Hz range of the spectrum.
5	Ratio	The ratio of the mid-range power over the total signal power.
6	Zero crossings	The number of zero crossings in the Linear PCM encoding of the audio signal
7-16	Band power	9 features representing the signal power in 100Hz bands from 1 to 1000Hz. The bands are 1-100Hz, 101-200Hz...901-1000Hz.

TABLE II
VOICE ACTIVITY DETECTION FEATURES

	GMM	SVM
In pocket	86.7%	91.3%
Out of pocket	90.5%	95.1%

TABLE III
VOICE ACTIVITY DETECTION ACCURACY

to the output of both. Next, we describe the techniques we evaluated for predicting a user's preferences using only the passively collected reinforcement signal (changes to ringer settings and responses to phone calls). Finally, we describe our use of density-weighted uncertainty sampling to select the contexts in which we wish to issue active queries for the user's preferences.

A. Phone Posture Recognition

Previous work has shown that having knowledge of the user's physical activities can be used to help determine interruptability [4]. However, accurately classifying a user's activity generally requires one or more accelerometers placed at known locations around a user's body. With a mobile phone, a user may carry the phone in their pocket, purse, or on their belt, so we do not have a known reference point from which to conduct activity recognition. Instead, we simplify the problem to trying to estimate the current physical posture of the device itself. In this task, we wish to determine if the phone is resting on a flat surface, if it is being actively held by the user, or if it is placed in a pocket, purse, backpack, etc. To address this problem, we collected labeled data from these three classes, using the 3-axis accelerometer and the proximity sensor of the phone. The data was divided into overlapping half-second frames, with the sample mean and variance of the accelerometer axes recorded for each frame. The number of times that the proximity sensor was triggered over the half-second was also recorded. A linear support vector machine was then trained to differentiate these classes, attaining 91.4% accuracy over 89 test samples.

B. Voice Activity Detection

Audio data was collected from the smartphone devices at a sample rate of 8192Hz. Ten seconds of audio was recorded, and this signal was broken into 20 half-second samples. For each of these samples, a Fast Fourier Transform is used to extract 16 features, presented in Table II. We empirically compared multiple classifiers for use in the voice activity detection task. A support vector machine with a linear kernel and a Gaussian mixture model were both trained on labeled audio samples to differentiate audio samples containing human speech from samples that do not contain speech. Although previous work has shown this approach to be effective at the voice activity detection task [10], [13], there is one complication that arises in a mobile devices application: the device may be in a user's pocket or handbag when the sample is collected, resulting in a significantly dampened signal and many false-negative predictions by the classifier. Because we are able to detect when the phone is in a pocket using the accelerometers and proximity sensor, we train a second speech detection classifier for this scenario. A linear support vector machine and Gaussian mixture model were also trained in this instance, with a new set of trained weights to account for the dampened signal.

The performance of the classifiers with the phone in and out of a pocket is shown in Table III. The testing set included many noisy audio samples without voice activity, such as music and sounds of car traffic. Based on these results, the linear support vector machine was selected for deployment in the In-Context application.

C. Smoothing of Information Extraction Output

Because several seconds of data is collected whenever we wish to sense the current context, both of the information extraction algorithms actually produce a sequence of predictions as their output. Rather than simply using the mode label of the sequence, we instead consider the certainty of the classifiers over the time interval. For phone posture recognition, we select the label with the most total certainty over the sequence. However, for the voice activity detection, we recognize that pauses in a conversation may result in many elements of the label sequence receiving a negative label, even though there is actually a conversation present in the environment. Therefore when smoothing the predictions of the voice activity detection algorithm, rather than selecting the most confident label, we instead require that the total confidence surpass a certain threshold. This is formalized in equation 1, in which the smoothed label Y_{VAD} is based on a sequence of data-points $\mathbf{x_1}...\mathbf{x_T}$. For each element in the sequence, the distance to the decision boundary of the SVM is given by $\mathbf{w} \cdot \mathbf{x_t}$. The threshold, C, was selected to maximize classification accuracy on a labeled parameter validation set.

$$Y_{VAD} = I\left(\left[\sum_{t=1}^{T} \mathbf{w} \cdot \mathbf{x_t}\right] \geq C\right) \tag{1}$$

D. Preference Classification

This section addresses the problem of trying to predict a user's preferences in a given context, given their preferences in previous contexts. We employ a variant of the nearest-neighbors algorithm for the task of selecting ringer preferences in different contexts. We compared this algorithm to a variety of other classification algorithms, including a support vector machine, decision tree, and naive Bayes algorithms, and found that the nearest-neighbor algorithms outperformed these alternatives.

For the purposes of classification, we use the first six variables presented in table I as features, and we used the hardware ringer switch as the classification label. However, it has been noted in previous work that many users forget to set their ringer switch to their preferred setting [12], so the setting of the hardware ringer switch will not always reflect the user's true preference. However, we believe that at the moment a user changes their ringer setting, this setting is most likely correct for their current context (or near future contexts). Therefore, rather than treat the ringer switch as a strict binary label, we think of it more as a reinforcement signal, which degrades over time. If it has been several hours since the user set the ringer preference, we are less confident in this signal, whereas if the switch has just been set, we are much more confident. To capture this behavior, equation 2 shows the exponential decay function we use to weight the samples. In this equation, the weight W_i of datapoint i is based on the hardware ringer switch, Y_i, of this sample, which adopts value 1 if the switch is on and -1 if the switch is off. The exponential decay parameter λ was selected using the validation set. It was empirically determined that small changes to these parameters do not have a significant impact on classifier performance, so it is not necessary to learn them for each user. The variable h denotes the number of hours since this setting was selected, rounded to the nearest whole number. The weight function has an additional benefit as well. When this system is deployed on a user's phone, if the preference classifier is working correctly, we envision the users no longer needing to change their ringer setting. By ignoring the ringer setting if it has been a long time since the user set it, we allow the system to take over completely when the user is satisfied with the system.

$$W_i = \begin{cases} Y_i e^{-h/\lambda} & : h \leq 12 \\ 0 & : h > 12 \end{cases} \tag{2}$$

The distance function for the nearest neighbor classifier is given in equation 3. This function describes the distance between two recorded context C^i and C^j. For each feature d in contexts i and j, we consider the difference $|f_d^i - f_d^j|$. For the phone posture, voice activity, weekday, and sound level features, this is simply the Hamming distance. For the hour feature, the difference is $max(|h_i - h_j|, 4)$. For the location feature, the difference is the indicator function, denoting whether these two locations are within 150 meters of one another. For each feature k, we have a distance parameter d_k, which was selected on a validation set taken from a single user's data.

$$D(C^i, C^j) = \sum_{k=1}^{6} d_k |f_d^i - f_d^j| \tag{3}$$

Using the distance function given by equation 3, we have the decision policy given in equation 4. If we wish to predict the ringer setting for a context C, we take a weighted summation over the k contexts in the user's history with the smallest distance to the current context. If this summation is non-negative, the algorithm predicts that the user would like the ringer turned on. Otherwise the ringer is turned off. It is worth noting that this prediction function gives us an obvious confidence measure, namely the weighted summation of distances to the nearest contexts. The larger the magnitude of this summation, the more confident the algorithm is of the prediction. Empirical results for this algorithm are given in section V.

$$Pred(C) = I\left(\left[\sum_{i=1}^{k} \frac{W_i}{D(C, C^i)^2}\right] \geq 0\right) \tag{4}$$

E. Active Learning

Active learning is a framework in which a learning algorithm is able to query an oracle for the label of specific data-points. In the context of interruptibility, the user acts as the oracle, and these queries are presented *in situ* so as to benefit from the increased accuracy of experience sampling [2]. In scenarios in which a labeling oracle is available, active learning has been shown to greatly increase classification accuracy [16].

Uncertainty sampling is a popular metric for selecting which points to query the oracle about. With uncertainty sampling, the data-points that the classification algorithm is most uncertain about are selected for labeling by the active learning oracle [11]. Entropy is a common measure of algorithm uncertainty, as given in equation 5. A high entropy value for a data-point, X, indicates high uncertainty.

$$H(X) = -\sum_{l=0}^{1} P(Y(X) = l|X) \log[P(Y(X) = l|X)] \tag{5}$$

While uncertainty sampling often works well, in our mobile phone application we are likely to see many samples densely packed around a small number of contexts (e.g. the user is at work or at home), plus a small number of previously unknown contexts (such as when the user tries a new restaurant). Although the algorithm may be highly uncertain about an unusual context, such as the restaurant, this context is not representative of much of the user's activity, so labeling it will provide limited benefit.

Therefore, we propose the use of *density-weighted uncertainty sampling* [17]. In this framework, the algorithm favors asking the user to label data-points that the system is uncertain about, but which are also representative of a large number of other samples in the data. The metric function for density-weighted uncertainty (DW) sampling is given in equation 6. In this equation, $sim(X, Y)$, is a function representing how similar two points X and Y are. For our similarity function, we used the squared reciprocal of the distance function from equation 3.

$$DW(X) = H(X) \sum_{i=1}^{n} sim(X, X^i) \tag{6}$$

With density-weighted uncertainty sampling, we wish to query the label of the sample, X, that maximizes $DW(X)$. This will be a sample that the algorithm is uncertain about labeling, but which is also representative of several other data-points in our dataset. We also use an additional heuristic, in which the algorithm will not request the label of a point if a similar data-point has already been labeled.

We collected 50-100 labeled data-points for each volunteer using the In-Context system. These data-points were collected using experience sampling, according to a uniform query schedule. Of these labeled data-points, 50% were set aside for testing for each user. The remaining labeled data-points were used to evaluate the benefit of allowing the system to actively request labels. In the next section, we compare density-weighted uncertainty sampling to standard uncertainty sampling, which is the technique used in most previous interruptibility prediction systems [4], [9].

V. Results

Figure 3 shows a comparison of four different classifiers for predicting ringer preferences. We evaluated the nearest-neighbors algorithm described in section IV-D, a support vector machine with an RBF kernel, Naive Bayes, and a decision tree using the information-gain metric. Additionally, the support vector machine was evaluated on a single user when given the raw features used for information extraction, rather than the output of the information extraction algorithms (voice activity detection and phone posture recognition). From the summary of experimental results, shown in table IV, we see that that these two information extraction algorithms have a significant positive impact on classification accuracy.

The effects of active learning queries on classification accuracy are shown in figure 4. In this experiment, we compared

	Features	**RBF SVM accuracy**
With Information Extraction	6	95.3%
Without Information Extraction	33	59.3%

TABLE IV
Effects of voice activity detection and phone posture recognition (Information extraction)

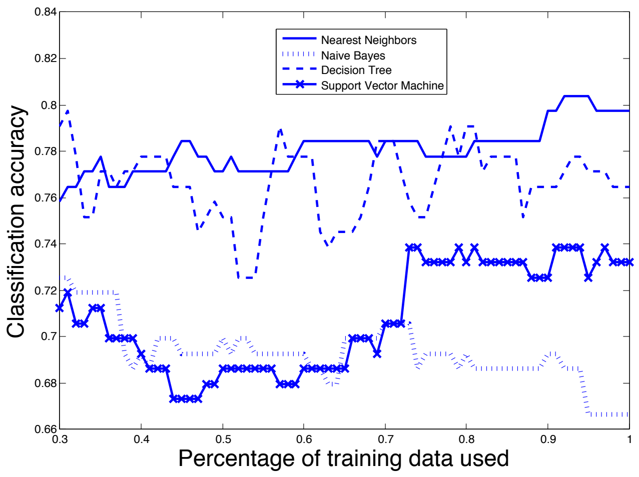

Fig. 3. Comparison of preference classifiers

the performance of standard uncertainty sampling against density-weighted uncertainty sampling. The classification accuracy is averaged across all five users. We see that density-weighted uncertainty sampling consistently outperforms uncertainty sampling. When given the maximum number of active labels (15), the classifier with density-weighted uncertainty sampling attained an average classification accuracy of $96.12\% \pm 3.37\%$. This equates to approximately two queries per day. When the number of queries is dropped to one per day, the classification accuracy is $87.86\% \pm 5.68\%$. With no active queries at all, the classification accuracy is $81.46\% \pm 6.20\%$. However, we note that there was one user who did not set his hardware ringer switch consistently, so the classifier with no active labels attains only 52.0% accuracy for this user. The other four users attain an average passive classifier accuracy of 87.82%.

Hardware ringer switches have not previously been used to train interruptibility classifiers, presumably because the noise was thought to be too great. We see that, in general, this is not the case, with four of our five users attaining a classification accuracy above 80% with no active labels. One user attained 96.32% accuracy using no active labels. By starting with a much higher baseline, we need fewer queries to users to push classification accuracy above 95%. Compared to a system that relies only on user queries [14], we are able to produce comparable accuracy with much fewer queries, and much greater accuracy when the user is willing to answer only a small number of queries. We additionally see that the density-weighted uncertainty sampling provides increased accuracy compared to regular uncertainty sampling. Furthermore, we conjecture that the density term will prevent the system from issuing queries every time the user travels to a new context.

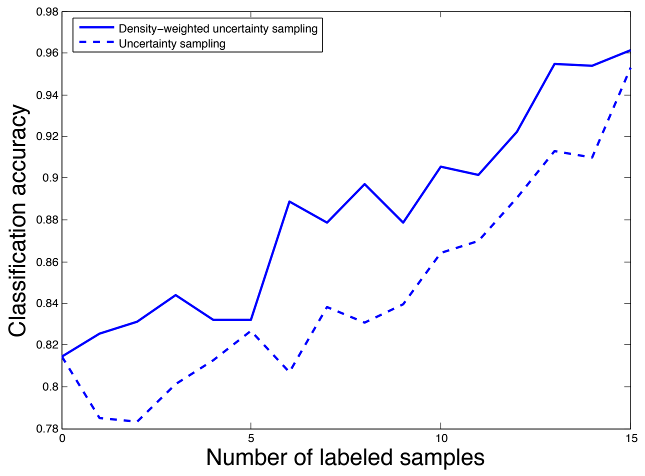

Fig. 4. Active learning curve

VI. Conclusions and Future Work

We see that both density-weighted uncertainty sampling and reinforcement learning on passively collected data are beneficial for classification accuracy. Future work should include further evaluation on the effects of using density-weighted uncertainty sampling in an on-line setting. We conjecture that density-weighted uncertainty sampling could also be combined with a decision theoretic framework to improve satisfaction as well as classification accuracy.

The algorithms that allow us to build a context-aware system are very general, and other applications should be explored in the future. The authors are currently working to bring this work into a smart-wheelchair application, in which the user of the chair is reminded to conduct pressure relief exercises according to a personalized model of interruptibility. Smart homes and automobiles would be other avenues for future research. The representation of context described in this paper could also be transmitted off the phone to allow for use in other devices, for instance a context-aware smartphone could notify the user's house that the user is on their way home, allowing the lights and heat to be activated in preparation for their arrival. As more of the devices in our environment become instrumented with sensors, we believe the importance of context-aware computing will continue to grow, as will the pressure to minimize the annoyance of being interrupted by those self-same devices.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Cooperative Agreement EEC-0540865 as well as by a National Science Foundation Graduate Research Fellowship. We also acknowledge the Pittsburgh chapter of the American Rewards for College Scientists (ARCS) program for their generous support.

References

[1] G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical report, Citeseer, 2000.

[2] M. Csikszentmihalyi and R. Larson. Validity and reliability of the experience sampling method. *The experience of psychopathology: Investigating mental disorders in their natural settings*, pages 43–57, 1992.

[3] J. Fogarty, S.E. Hudson, C.G. Atkeson, D. Avrahami, J. Forlizzi, S. Kiesler, J.C. Lee, and J. Yang. Predicting human interruptibility with sensors. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 12(1):119–146, 2005.

[4] J. Ho and S.S. Intille. Using context-aware computing to reduce the perceived burden of interruptions from mobile devices. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 909–918. ACM, 2005.

[5] E. Horvitz and J. Apacible. Learning and reasoning about interruption. In *Proceedings of the 5th international conference on Multimodal interfaces*, pages 20–27. ACM, 2003.

[6] E. Horvitz, P. Koch, and J. Apacible. Busybody: creating and fielding personalized models of the cost of interruption. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 507–510. ACM, 2004.

[7] E. Horvitz, P. Koch, R. Sarin, J. Apacible, and M. Subramani. Bayesphone: Precomputation of context-sensitive policies for inquiry and action in mobile devices. *User Modeling 2005*, pages 251–260, 2005.

[8] S.S. Intille, J. Rondoni, C. Kukla, I. Ancona, and L. Bao. A context-aware experience sampling tool. In *CHI'03 extended abstracts on Human factors in computing systems*, pages 972–973. ACM, 2003.

[9] A. Kapoor and E. Horvitz. Experience sampling for building predictive user models: a comparative study. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 657–666. ACM, 2008.

[10] T. Kinnunen, E. Chernenko, M. Tuononen, P. Fr
"anti, and H. Li. Voice activity detection using mfcc features and support vector machine. In *Int. Conf. on Speech and Computer (SPECOM07), Moscow, Russia*, volume 2, pages 556–561. Citeseer, 2007.

[11] D.D. Lewis and W.A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12. Springer-Verlag New York, Inc., 1994.

[12] A.E. Milewski and T.M. Smith. Providing presence cues to telephone users. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 89–96. ACM, 2000.

[13] J. Ramirez, P. Yélamos, JM Górriz, and JC Segura. Svm-based speech endpoint detection using contextual speech features. *Electronics letters*, 42(7):426–428, 2006.

[14] S. Rosenthal, A. Dey, and M. Veloso. Using decision-theoretic experience sampling to build personalized mobile phone interruption models. *Pervasive Computing*, pages 170–187, 2011.

[15] A. Sasse, C. Johnson, et al. Coordinating the interruption of people in human-computer interaction. In *Human-computer interaction, INTERACT'99: IFIP TC. 13 International Conference on Human-Computer Interaction, 30th August-3rd September 1999, Edinburgh, UK*, volume 1, page 295. IOS Press, 1999.

[16] B. Settles. Active learning literature survey. *Machine Learning*, 15(2):201–221, 1994.

[17] B. Settles and M. Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079. Association for Computational Linguistics, 2008.

[18] D. Siewiorek, A. Smailagic, J. Furukawa, A. Krause, N. Moraveji, K. Reiger, J. Shaffer, and F.L. Wong. Sensay: A context-aware mobile phone. In *Proceedings of the 7th IEEE International Symposium on Wearable Computers*, volume 248. Citeseer, 2003.

[19] L. Terveen, J. McMackin, B. Amento, and W. Hill. Specifying preferences based on user history. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves*, pages 315–322. ACM, 2002.

[20] A. Toninelli, D. Khushraj, O. Lassila, and R. Montanari. Towards socially aware mobile phones. In *First Workshop on Social Data on the Web (SDoW)*. Citeseer, 2008.