Collecting Big Datasets of Human Activity One Checkin at a Time

Theus Hossmann, Christos Efstratiou, Cecilia Mascolo
Computer Laboratory, University of Cambridge, UK
{th420, ce291, cm542}@cl.cam.ac.uk

ABSTRACT

A variety of cutting edge applications for mobile phones exploit the availability of phone sensors to accurately infer the user activity and location to offer more effective services. To validate and evaluate these new applications, appropriate and extensive datasets are needed: in particular, large sets of traces of sensor data (accelerometer, GPS, micro- phone, etc.), labelled with corresponding user activities. So far, such traces have only been collected in short-lived, small-scale setups. The primary reason for this is the difficulty in establishing accurate ground truth information outside the laboratory. Here, we present our vision of a system for large-scale sensor data capturing, leveraging all sensors of todays smart phones, with the aim of generating a large dataset that is augmented with appropriate ground-truth information. The primary challenges that we address consider the energy cost on the mobile device and the incentives for users to keep running the system on their device for longer. We argue for leveraging the concept of the checkin – as successfully introduced in online social networks (e.g. Foursquare) – for collecting activity and context related datasets. With a checkin, a user deliberately provides a small piece of data about their behaviour while enabling the system to adjust sensing and data collection around important activities.

In this work we present up2, a mobile app letting users check in to their current activity (e.g., "waiting for the bus", "riding a bicycle", "having dinner"). After a checkin, we use the phone's sensors (GPS, accelerometer, microphone, etc.) to gather data about the user's activity and surrounding. This makes up2 a valuable tool for research in sensor based activity detection.

Categories and Subject Descriptors

C.5.3 [**Computer System Implementation**]: Microcomputers

HotPlanet'12, June 25, 2012, Low Wood Bay, Lake District, UK.

General Terms

Measurement, Experimentation, Human Factors

Keywords

Smart Phone, Sensing, Activity Detection, Checkin, Big Dataset

1. INTRODUCTION

The proliferation of smart-phone devices has enabled the wide deployment of mobile phone applications exploiting the availability of phone sensors to accurately infer the user activity and location to offer more effective services. To this end, a number of experimental systems attempt to detect user context and activity through the fusion of sensing modalities that are available on the users' mobile devices: accelerometer, RF traces (bluetooth, WiFi), audio signals captured through the device's microphone [1, 2, 3, 4].

Yet, aside location based services (e.g., Foursquare, Google Latitude), we have not yet witnessed the large scale deployment of more elaborate activity and context detection systems able to infer more information about the user than just their current location. The limited success of such systems in the real world lies primarily in the practical difficulty in designing accurate activity detection algorithms and evaluating their performance in the wild. Existing activity detection systems are developed and evaluated through experiments in the lab or in controlled environments. This practice is considered necessary in order to allow researchers to capture the ground truth during data collection, which then is used both for training the machine learning components of the system, and evaluating their performance. The end result, however does not give accurate indications on how a system would actually perform in a real environment, where sensed information can vary significantly from what was captured during the experiments.

One of the key reasons for this trend is the lack of appropriate data sets collected in a global scale, that can be used to both design accurate activity and context detection algorithms, and evaluate their performance in the real world. Such envisioned datasets should include sensor data traces, from a wide range of environments and activities, involving a large number of users — measured in thousands versus the typical tens of users involved in common activity detection studies — and covering a large time span — measured in months. More importantly, the dataset should include accurate metadata about the ground truth, identifying the activity and context of the user when the data is collected.

Despite the availability of different sensor reading platforms for mobile phones [5, 6], collecting such a large scale dataset bears many challenges. The most critical ones are: 1. The users must have an incentive to provide data, 2. providing the data must be easy and unobtrusive , 3. privacy must be respected, 4. battery power of the user's device must be conserved.

In this paper we present our attempt to generate such a dataset, through the deployment of the mobile phone application *up*2, which allows the capturing of sensing traces, and appropriate ground truth meta data, in a global scale.

The rest of this paper is structured as follows. In Section 2 we describe our vision of augmenting checkins to collect activity data, followed by a discussion of the challenges of such an approach in Section 3. We then describe the design and function of our prototype application in Section 4. In Section 5 we discuss some limitations of the checkin-based approach and conclude in Section 6.

2. CHECKIN-BASED DATA COLLECTION

Our aim with the deployment of the *up*2 application is to generate a dataset of captured mobile phone sensor data, that is tagged with appropriate ground truth information about user activity. The ground truth information should include a range of human activities and contexts. For example, generating data that is tagged with activities such as "watching TV", "riding the bus", "preparing dinner", "having a drink at the pub". This dataset can then be used for the development and evaluation of activity detection applications using real-world traces. By attempting to capture such data in a global scale, we can enable the development and evaluation of activity detection systems across a wide range of environments. For example, allowing a researcher to compare how "riding a train" activity can be detected for a user riding a train in India versus a user riding a MagLev in Japan.

As our aim is to enable the development of a wide range of activity and context detection applications, ground truth can cover a versatile set of meta information. In most domains, the concepts of activity and context are both quite loosely defined. This means that capturing such information would require user involvement, where the user is asked to specify the ground truth. In our system we intend to leverage the well established concept of *checkin* that is widely used by common location based services. In typical location based applications, users are allowed to checkin to a particular location. This is used as a trigger for the system to capture the user's location and associate the location with the ground-truth: typically a venue or a building identified by accessing a map at the back-end. In the *up*2 case there is no concept of a *back-end map* that can be used to associate sensed information with the ground truth. A checkin in *up*2 enables the user to specify their activity and context manually which is used for tagging sensed data. Moreover, by adopting the checkin approach we can avoid the problems associated with continuous sensing on mobile systems. The checkin can act as a trigger to initiate sensing only when metadata is offered by the user.

For the successful deployment of *up*2 however it is necessary to address a number of challenges that are primarily related to the acceptability of this app as a common mobile phone application by a large number of users.

3. CHALLENGES

The key challenge in the success of *up*2 is to ensure the wide adoption of the application by a large number of user, i.e. to ensure the high acceptability of the application. Acceptability is a function of a number of variables. In this context acceptability can be considered as the ratio of the valuable/positive effects that an application offers over the sum of the negative effects caused by the application. Usability and ease of use can ensure that using the application will not lead to high levels of frustration for the users; Energy consumption caused by an application is a key aspect that could lead a number of users to remove a useful application from their mobile device; Privacy concerns can act as counter incentives for most mobile sensing applications. The application should aim to reduce any negative effects caused by these three aspects or counter balance them with strong incentives and added-value features, that would make users consider the possible cost as reasonable, in order to receive the benefits of the service. The following sections describe our approach in addressing these issues.

Ground truth vs Usability.

The first major challenge concerns the input of the ground truth by the user. Ideally, we would like to ask for very precise and detailed information about their current activity: What are you doing? Since when? Until when? With whom? Etc. While such detailed questionaries are possible in small scale experiments, we have to balance the level of detail we expect with the potential negative impact on the application's usability which could then affect its wide scale adoption. The goal in achieving ease of use in our application is to reduce the number of clicks required for a checkin to the bare minimum, and reduce the number of options that the user needs to select from at any stage. Reaching this goal comes at the cost of loosing certain pieces of information. In our design, we decided to constrain the collected information by asking the user to only checkin to their current activity. This allows us to capture a small piece of information about the ground truth requiring very little effort from the side of the user. On the downside, we miss information about the duration of a particular activity.

Capturing more detailed ground truth information can be achieved by following an iterative deployment strategy. We envision a deployment scheme where the application can be improved through the data that is collected, by incorporating activity detection mechanisms into future versions of the application. In such manner the checkin process can be further simplified by making the application "smart" enough to detect the most probable activity during checking. This would allow us to incorporate more options about the ground truth, while maintaining the same number of clicks per checkin. In the current implementation of *up*2, a checkin requires 5 clicks, and in average 10 seconds for a typical user to perform. This result is inline with typical location based applications such as Foursquare where a checkin requires a minimum of 4 clicks.

Energy consumption.

One of the major challenges in mobile phone sensing is the impact on the phone's battery life. While a notable reduction of battery lifetime may be acceptable in an experimental setting, it would however have severe consequences

in the wide scale adoption of a typical mobile phone application. Ideally, an application should not have any noticeable impact on battery lifetime. Different approaches have been proposed to reduce energy consumption. In [7], an approach for continuous sensing, energy consumption is limited by efficiently processing measured data. A different approach is to introduce duty cycling and adapt the sampling rate according to the predicted outcome of the sampling (i.e., increase the sampling rate during interesting events) [8]. Such approaches are useful for services where the sensors have to be permanently active. In our case, however, the requirements are different. Since we only know the activity at the time of a checkin, we use the checkin to trigger the sensors for a short period of time.

Choosing that window of sampling at the right time and of the right length is a challenge. First, we need to account for some time the user takes to stow away the phone and start or resume the reported activity. In order to make sure to capture the right moment, the sampling can be repeated (e.g., for a certain number of times with an exponentially growing sleep interval in-between). Second, the sampling period should be short enough to limit energy and storage resource usage, while keeping it long enough for the data sample to be useful. The ideal choice of these parameters may further depend on the activity. Short term activities (e.g., "riding an elevator") may need different settings than longer term activities (e.g., "sleeping").

Considering again our iterative deployment strategy our aim is to "learn" the right sampling schemes for each activity through post-hoc analysis of the collected data. This would lead to future versions of the applications with higher energy efficiency without compromising the accuracy of the collected sensor data.

Privacy.

Sensor data is highly sensitive. The benefit of using self reported checkin data (as opposed to continuous sensing) is that the user can decide whether to report an activity or not. Of course, the usage of the data must be properly declared to the user to allow an informed decision. Further, the user should be given the option to keep data private, either in general or on a per checkin or even per sensor basis. In order to publish the dataset for scientific use, we need to further explore the tradeoff between anonymization and usefulness of the data.

Incentives & Gamification.

The last but perhaps the most important challenge in the design of *up*2 is the engagement and retention of the user base of the application through appropriate incentives. A starting point to find incentives is to look at location checkin services. A survey on Foursquare users [9] describes a variety of motivations ranging from personal tracking to discovering new friends. We can broadly classify motivations into the three categories *social*, *game* and *personal*. For many users, initially the game mechanisms (e.g., collecting points and badges) are an important factor, but on the long term the social factors are more important. In the following we summarise thoughts about potential incentive mechanisms for each of the three categories.

Game incentives: Collecting points (and making leader boards) or badges (e.g., for a certain number of checkins) has proven effective for location based services. Further, the service could issue specific challenges (e.g., to collect a badge for a certain number of "walking" checkins). Yet, such incentives may motivate cheating by checking in to arbitrary activities. Careful verification or sanity checks are required to maintain a high quality dataset (see also Section 5).

Social incentives: Signalling to friends is another well established incentive in similar services. Matching activity patterns of different users could further be used to suggest new friends. However, such schemes have the possible problem of self selection bias where users would avoid checking in to certain activities that are considered unfavourable or uninteresting behaviour by their friends.

Personal incentives: Recently, *life-logging* and the *quantified self* attract more and more attention. Supporting people who like to log all aspects of their life could be a good application scenario for activity checkins. As an additional personal incentive, the application could support goal setting (i.e., setting a personal goal to run at least three times a week and monitor progress through checkins). However, goal setting may bias the collected data towards positive and desirable activities.

Having discussed challenges of large scale activity data collection, we now proceed to describe our prototype system.

4. THE UP2 APPLICATION

To collect sensor data labeled with the respective activity of the user, we have implemented a prototype mobile phone service called *up*2. The service consists of two parts: an Android application and an API to store and query data, based on the Google App Engine framework[1]. In this section, we describe the design and function of both, client App and back-end API.

4.1 Client App

User account.

In our prototype, we use the Google Account of the mobile phone user for authentication[2]. Upon first starting the client application (or after logging out), the user can choose one from the list of all accounts registered on the phone. After confirming access to the selected account, the application receives an authentication token and cookie which is included in all requests to the *up*2 API. The user can then choose a username, uniqueness of which is ensured by the server.

Checkin.

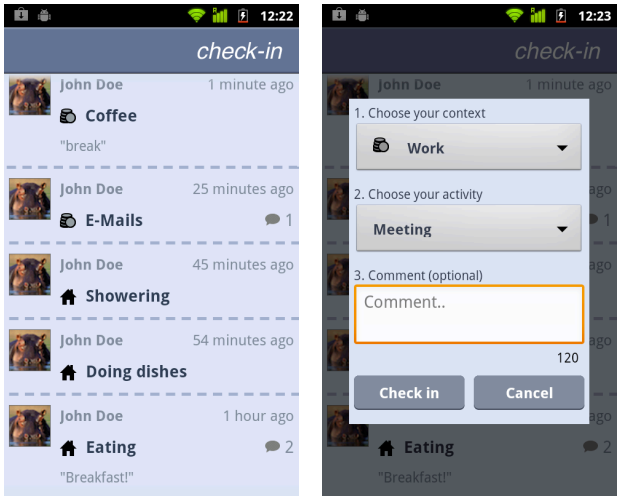
After starting the application on the mobile phone, the user is presented with a summary of their own and friends' activities (Figure 1a). From this main screen, a button in the header of the application allows to create a new checkin. In the checkin screen (Figure 1b), the user chooses the current activity in a two step process. First, a *context* has to be

[1] `https://appengine.google.com/`

[2] In future versions of the application, we plan to decouple the *up*2 user account from the Google Account, so that users can register and access the service without their Google Account.

Going out	Home/Living	Work	Mobility/Transport	Leisure/Vacation
Eating	Shopping	Meeting	Walking	Strolling
Drinking	Cooking	E-Mails	Running	Hiking
Dancing	Eating	Reading	Elevator	Climbing
Concert	Housework	Writing	Bicycle	Football
Date	Showering	Telephone	Motorbike	Tennis
	Sleeping	Chatting	Car	Beach
	TV	Coffee	Bus	Swimming
	Computer		Tram	Dancing
	Gaming		Train	
			Plane	

Table 1: List of pre-defined activities.



(a) The main screen. (b) The checkin screen.

Figure 1: up2 screenshots.

selected from a drop down list of 5 pre-defined contexts: 1. *Going out* 2. *Home/Living* 3. *Work* 4. *Mobility/Transport* and 5. *Leisure/Vacation*. Depending on the selection of the context, a second drop down list then contains the respective activities. Table 1 contains a list of all pre-defined activities for all contexts. We choose this two step process to offer a large number activities, while at the same time having lists of manageable size. Additionally, the checkin screen offers a text field to write a short comment describing the checkin. If the user wishes to check in to an activity which is not in the pre-defined list, custom activities can be defined (and assigned to one of the contexts) from an option in the menu. *By allowing self-defined activities, we hope to get a comprehensive, crowd sourced list of daily activities.*

Sensing.

A checkin triggers the application to sample the phone's sensors. The sensors are managed by the *SensorService*, a subsystem of the *up*2 client app. The SensorService currently uses three sensors. 1. The *accelerometer* for measuring motions to which the phone is exposed to. The accelerometer is often used in activity detection systems [4, 7]. 2. The *microphone* to and record a sample of the sound of the environment. Such sound samples can be used to classify the environment (e.g., speech, music, silence) [2, 7]. 3. The *location service* (using GPS, as well as access point and cell tower locations). Since many activities are bound to places, location (and change of location) is an important clue for activity detection [7]. In the future we plan to include additional sensors. WiFi and Bluetooth scans could be added to get a more detailed picture of the networks and devices surrounding a user during an activity. Further, the Gyroscope could be an additional source for precise detection of motion by measuring changes in the orientation of the phone.

Since we do not know exactly when the user starts and ends the reported activity, one critical challenge is to ensure the collected sensor data captures normal behaviour of the user during an activity. For example, a user may check in to "running" just before she actually starts running – hence, sampling for a short duration immediately after the checkin would not capture the actual activity. Thus, we have to find a good solution to the tradeoff of duration of sensing (consuming energy and storage) and usefulness of the data (sampling for too short time or at the wrong time). There are three parameters to tune to find a good heuristic of when to sample: *When* to start, for *how long* and *how often* to repeat the sampling. For the current prototype we empirically found that waiting for 30 seconds, and then sensing for a period of 60 seconds is a good initial strategy. Currently we sample only one time interval. In the future, we plan to further investigate the best strategy. For example, the sensing may be repeated after exponentially increasing sleeping periods.

To make sure the user is aware of the sensing going on in the background, an icon shows in the notification bar during the sensing.

Social Network and Privacy.

For the social aspect of sharing activity checkins, the user can search for usernames of friends. Upon finding a friend, a friendship request can be sent, which leads to a notification on the friend's phone. Such friendship requests can be accepted or ignored. Thus, for privacy reasons, we require social ties to be mutual (as opposed to uni-directional "follower" relationships in services such as Twitter). Checkins of friends automatically appear in the main screen of the application, and users can post comments on the checkins of their friends. However, a user can choose in the settings to keep checkins private – in this case they will not be visible to anyone.

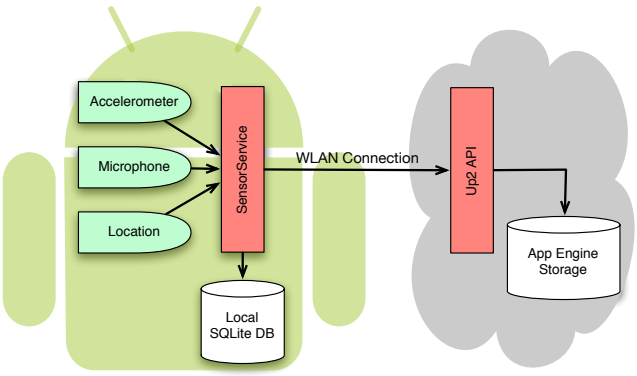


Figure 2: 3G offloading of sensor data.

3G Offloading of Data Transmission.

The sensor data of a checkin is a substantial amount of data. On a HTC Nexus One, a sampling period of 60 seconds, with default audio and accelerometer sampling rates records on average about 72 KB of data, on a Samsung Galaxy SII the same settings result in an average of about 173 KB of data[3]. Thus, for economic use of the user's data plan, we offload the transmission of sensor data from 3G to WLAN: Since the sensor data is delay tolerant, we can cache it locally and transmit only when connected to a WLAN. The flow of the sensor data is shown in Figure 2. The sensor data is cached in the local SQLite database on the phone by the SensorService. After sensing has completed, the data is flagged as dirty. Further, *up*2 uses a broadcast receiver to listen to notifications about connectivity changes sent by the operating system. Once we detect WLAN connectivity, all the "dirty" data is transmitted and deleted from the cache. To prevent the cache from growing infinitely in case a user never has WLAN connection, we further implement a garbage collector which is invoked once a day. The garbage collector deletes all data that is older than one day.

Note that we use the offloading only for sensor data. Checkins are less tolerant to delay and are published immediately, so that friends receive the latest checkins when polling the server.

4.2 Server API

The client application communicates with the backend by calling (over HTTPS) the *up*2 REST API. Resources are represented in JSON notation. Adhering to a REST architecture and using JSON formatting, allows us to keep the interface simple and generic such that other client applications (e.g., for other operating systems) could be easily implemented. Table 2 lists the supported API calls. Note that collections include only the resources a user has access to. For example GET /activities lists only the user-defined activities of the authenticated user.

To keep the transmitted sensor data small, the JSON representation of sensor data contains Base64 encoded binary data.

[3]Note that the default sampling rates depend on the hardware of the device, hence the large difference between the two devices.

5. DISCUSSION

After describing the vision, challenges and prototype of our approach, we now proceed to discuss some limitations and our ideas to mitigate these.

One problem with the user reported data is that we do not know how reliable the labels are. Mislabelling can happen if the user knowingly reports wrong data (e.g., if the incentives set to promote cheating by checking in to wrong activities), if the system is used in an unforeseen way (e.g., users do not check in at the actual time of the activity), or in case of system failure (e.g., broken sensors). Thus, we need a way to detect erroneous data. In location based services the GPS can be used to check that the user actually is where they pretend to be. While this mechanism can be circumvented (e.g., by using mock locations) by "malicious" users, it at least introduces a small barrier to cheating. Similarly, we intend to perform sanity checks based on the sensor data in the backend. For example, we can verify that the accelerometer actually shows high values in case of physical activities like running. While such tests can easily be fooled, they allow us to at least detect systematic errors in the system or its usage. Checks can be either made on a *per checkin* or *per user* basis – in case of the latter, a user could be assigned some sort of reputation value based on the ratio of passed sanity checks. Further, our hope is that we can collect enough data to allow conservatively filtering to obtain high quality data.

6. CONCLUSION

In this paper, we have argued for the need of big datasets of mobile phone sensor data, labelled with human activities. The availability of such datasets could advance the field of sensor-based activity detection and lead to exciting new applications and services. The problem we address here is to motivate a large number of users to participate in such measurements. Our approach is to augment the concept of the checkin: In an activity checkin application with social and/or gaming incentives, users can at the same time provide data and enjoy a fun and useful service. To this end, we have presented our prototype Android application called *up*2, which lets users log and report their current activity to their social network. Following a checkin, *up*2 uses the location, accelerometer and microphone sensors on the smart phone to collect data during a brief time window and uploads the data to the backend using the *up*2 API. This allows us to collect samples of labelled sensor data in an energy efficient way. By offloading the data transmission from 3G to WLAN, we can further conserve precious bandwidth of the user's data plan.

In the future, we plan to deploy the application to a larger audience. Since we need more experimenting with making the interface user-friendly (i.e., reducing the hassle of checking in) and providing the right incentives to make the service attractive for long term usage, we plan to first deploy the application in a controlled environment, based on invitations. To this end, we hope to motivate some of the workshop participants for testing and using our application. Further, we intend to use the data to create algorithms detecting and predicting the user's activity before the checkin happens. Incorporating such algorithms into the application, we have a useful setting to test and evaluate various activity detection algorithms.

Resource	GET	POST	DELETE
/users	List of friends		
/users/[id]	Retrieve one user		
/activities	List user-defined activities	New user-defined activity	
/activities/[id]			Delete user-defined activity
/checkins	List 50 checkins (param: since)	Create new checkin	
/checkins/[id]	Retrieve one checkin		Delete checkin
/comments	List comments (param: since)	Create new comment	
/comments/[id]			Delete comment
/sensordata		Upload data (param: checkin id)	
/sensordata/[id]	Retrieve sensor data		Delete sensor data
/friendships		Friendship request (param: user id)	
/friendships/[id]			Delete friendship

Table 2: Summary of *up*2 API calls.

7. REFERENCES

[1] K. K. Rachuri, M. Musolesi, C. Mascolo, P. J. Rentfrow, C. Longworth, and A. Aucinas, "Emotionsense: a mobile phones based adaptive platform for experimental social psychology research," in *Proceedings of the 12th ACM international conference on Ubiquitous computing*, ser. Ubicomp '10. ACM, 2010.

[2] H. Lu, W. Pan, N. D. Lane, T. Choudhury, and A. T. Campbell, "Soundsense: scalable sound sensing for people-centric applications on mobile phones," in *Proceedings of the 7th international conference on Mobile systems, applications, and services*, ser. MobiSys '09. ACM, 2009.

[3] 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*, ser. SenSys '08. ACM, 2008.

[4] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, "Activity recognition from accelerometer data," in *Proceedings of the 17th conference on Innovative applications of artificial intelligence*, ser. IAAI'05. AAAI Press, 2005.

[5] J. Kukkonen, E. Lagerspetz, P. Nurmi, and M. Andersson, "Betelgeuse: A platform for gathering and processing situational data," *IEEE Pervasive Computing*, 2009.

[6] (2012) funf, open sensing framework. [Online]. Available: http://funf.media.mit.edu/

[7] H. Lu, J. Yang, Z. Liu, N. D. Lane, T. Choudhury, and A. T. Campbell, "The jigsaw continuous sensing engine for mobile phone applications," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '10. ACM, 2010.

[8] K. K. Rachuri, C. Mascolo, M. Musolesi, and P. J. Rentfrow, "Sociablesense: exploring the trade-offs of adaptive sampling and computation offloading for social sensing," in *Proceedings of the 17th annual international conference on Mobile computing and networking*, ser. MobiCom '11. ACM, 2011.

[9] J. Lindqvist, J. Cranshaw, J. Wiese, J. Hong, and J. Zimmerman, "I'm the mayor of my house: examining why people use foursquare - a social-driven location sharing application," in *Proceedings of the 2011 annual conference on Human factors in computing systems*, ser. CHI '11. ACM, 2011.