Nurturing Social Networks Using Mobile Phones

Daniele Quercia, *MIT SENSEable City Laboratory, Cambridge, USA*
Jonathan Ellis, *Dept. of Computer Science, University College London, UK*
Licia Capra, *Dept. of Computer Science, University College London, UK*

Abstract

Youngsters maintain contact with each other through social-networking websites. We propose new ways of nurturing contacts by monitoring users' activity with mobile phones (i.e., by monitoring text messages, phone calls, and encounters captured by Bluetooth). We show that, based on user's activity, one is able to recommend new friends, track health of friendships (and alert users they may be neglecting their friendship), and make users aware of their mood (so that they take action and keep negative emotions under control).

I. Introduction

Adolescent depression is often a response to the normal process of maturing. It is consequently common, and it adversely affects school performance and impairs family relationships. Strong social support from friends has been found to be the most effective way of countering depression [1].

In the 1990s, social connections among youngsters have started to be effectively supported by social-networking websites and, nowadays, connections are made permanent and ubiquitous by mobile phones. Youngsters do not sit in front of their PCs to talk to their friends but constantly communicate with one another using mobile social-networking applications. This permanent online communication has been termed "connected presence" and, worryingly, some sociologists equated connected presence to social isolation as young people spent their time transfixed by mobile phone screens rather than other people. At that time, only few sociologists (Barry Wellman [2] being the first) questioned this viewpoint: they conducted small-scale studies and showed that people who are connected online often have higher levels of face-to-face interactions as well. This (controversial at the time) viewpoint is now widely accepted. Indeed, a large-scale sociological study found that owners of a mobile phone and social-networking users are more likely to belong to a local voluntary group such as a neighborhood association, sports league, youth group, church or social club [3].

Managing social-networking contacts is thus becoming a fundamental aspect of an adolescent's life. To help youngsters effectively grow and nurture their social relations, we have engineered a new technology for mobile phones that silently keeps track of people's colocation, as well as frequency of voice calls and text messages. These data is then processed by novel algorithms whose goal is:

- To effectively find social contacts based on encounters (Section II). Colocation data is collected and processed to detect friendship relations. Simulation studies conducted on real data demonstrate the ability of the inference engine to reveal these relations (Section II-C).
- To nurture online and off-line contacts (Section III). A simple inference engine detects patterns in both physical encounters (i.e., people's colocation) and social activity (i.e., phone calls and text messages), as well as deviations from such patters. If users suddenly become less sociable, the engine alerts them, perhaps suggesting them to get in touch with their friends. We are also studying the degree to which the engine can predict users' moods (e.g., happiness, sadness) simply based on their activity.

Both algorithms have been implemented and deployed on BlackBerry mobile phones (Section IV), causing negligible computational and communication overhead, thus confirming how this technology can silently run on modern mobile phones.

II. Finding Social Contacts: FriendSensing

FriendSensing is a framework that enables new members of social-networking websites to automatically discover their friends. It also help existing members to elicit new social relations, as they develop over time. In particular, *FriendSensing* automatically creates personalized recommendations of people a user may know in two steps:

Step 1 - Logging Encounters. Using short-range radio technologies ready available on almost all modern mobile phones (e.g., Bluetooth), each user transparently records encounters with colocated people. More precisely, each phone A keeps track of how many times it has met another phone B and how much time it has spent being colocated with B. We make here the assumption that a mobile phone is a *personal* device, and that it is not shared among people. Moreover, we assume it is possible to link devices (e.g., phone's Bluetooth ID number) to users' identities in social-networking websites (as pioneered by the Cityware project [4]).

Step 2 - Recommending Friends. Colocation records are processed to elicit relevant encounters and to arrange them into a weighted social network; this network is then traversed to compute personalized lists of people each user may know. FriendSensing does not prescribe *where* the processing of proximity records and the navigation of the inferred social network should occur: both can be performed either by the social-networking website (after these records have been uploaded) or by the mobile device itself (if such records are considered sensitive and should thus be maintained private).

We now present algorithms for proximity processing and for network navigation in general terms, and defer a discussion about the implications of different architectural deployments to our evaluation (Section II-C).

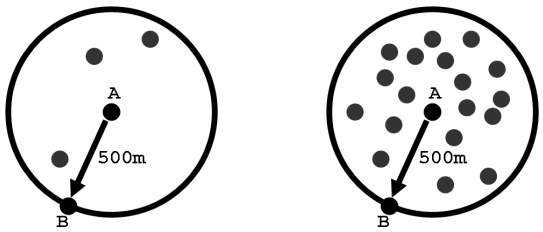
A. Processing Encounters

Once colocation logs have been collected, FriendSensing must filter out irrelevant encounters from *relevant* ones; that is, for each user A, it must identify which of A's encounters are likely to be A's friends. FriendSensing does so by computing the probabilities of A befriending other individuals (A's friendship probabilities) from proximity data.

Researchers have already suggested ways of computing these probabilities from *geographical proximity*, based on the intuition that friendship probability increases with geographic proximity - the closer two individuals are, the likelier they are to be friends. For example, they modeled the probability of A and B being friends as $p(A \rightarrow B) \propto dist(A,B)^{-r}$. That is, the probability of being friends with a person at a distance d decays as d^{-r} for some power of r (typically $r = 2$). As later demonstrated by Liben-Nowell *et al.* [5], the absolute value of geographic distance alone is insufficient to model friendship. To see how, consider that A and B live 500 meter apart: at the very same distance, A and B would likely be next-door neighbors in the countryside(Figure 1(a)), while complete strangers in central London (Figure 1(b)). This suggests that one also needs to consider population *density*. Libel-Nowell *et al.* [5] did so in a simple way - they replaced the absolute distance $dist(A,B)$ with a *ranked distance*: $p(A \rightarrow B) \propto 1/(rankDist_A(B)+1)$. The denominator is A's rank of B, which is the number of people who are closer to A than B is, and it is expressed as:

$$rankDist_A(B) = |\{C : dist(A,C) < dist(A,B)\}| + 1.$$

In other words, the probability of A befriending B depends on the number of people within distance $dist(A,B)$. The more dense the population between A and B, the lower B ranks. Consequently, at the same distance, B is more likely to befriend A in the countryside than in central London. This model was successfully evaluated on half a million profiles collected from the LiveJournal blogging website, suggesting that geography is a good predictor of friendship. However, geographical information is not widely available on mobile phones; should localization technology like GPS become a commodity, it would still fail to capture indoor encounters (e.g., at home, in the office, on the tube, in the pub). We thus need to reformulate the problem based on "mobile phone proximity".



(a) (b)

Fig. 1. Friendship Probability changes from **(a)** the countryside to **(b)** central London.

Using mobile phones, we keep track of: how many times a user A has met (e.g., it has been within Bluetooth range of) user B (frequency $freq(A, B)$), and how much time it has spent with B (duration $dur(A, B)$). We now need to express the friendship probability as a function of frequency and/or duration. One plausible way of doing so is to consider that the probability of A befriending B increases with $freq(A, B)$ and with $dur(A, B)$ respectively. However, as with geographical information, we cannot consider frequency or duration alone to compute friendship probabilities, because both of them are non-uniformly distributed. Indeed, individuals do have skewed mobility patterns; this has been shown not only for college students [6] (against whose movements we will run our evaluation), but also for conference attendees, and for hundreds of thousands of mobile users [7]. Rather than using absolute frequency and duration values, we have thus taken their rank. From *frequency*, the friendship probability becomes:

$$p(A \rightarrow B) \quad \propto \quad \frac{1}{rankFreq_A(B)} \ , \tag{1}$$

where $rankFreq_A(B) = |\{C : freq(A, C) > freq(A, B)\}| + 1$ (we add '+1' to avoid division by zero). Consequently, the probability of A befriending B depends on the number of people who have met A more frequently than B has done.

Similarly, by replacing frequency with *duration*, the friendship probability becomes:

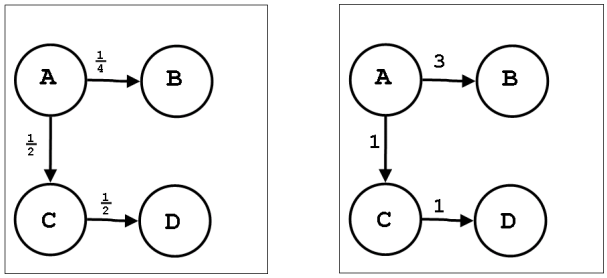
$$p(A \rightarrow B) \quad \propto \quad \frac{1}{rankDur_A(B)} \ , \tag{2}$$

where $rankDur_A(B) = |\{C : dur(A, C) > dur(A, B)\}| + 1$. Again, the probability of A befriending B depends not on $freq(A, B)$ itself but on the number of people who have met A for longer than B has done. The denominator of the friendship probability becomes very high for people with many friends - friendship probabilities tend to be low. However, this does not impact the way friends are ranked. Ranking does not depend on the friendship probabilities themselves but on their ordering. Of course, the ranking of users who are collocated in equal durations/frequencies are indistinguishable.

From the proximity logs, the above friendship probabilities can be computed and used to infer a *weighted social network of encounters*: each mobile device is represented as a node, and a link is added between any pair of individuals who have met at least twice (this is to remove encounters caused by chance). Each link $A \rightarrow B$ is then weighted using either *friendship probability* $p(A \rightarrow B)$ 2 (a)) or *friendship ranking* (i.e., A's ranking of B, which is computed from the friendship probability itself) - Figure 2 (b). We explain when to opt for probabilities and when for ranks next.

B. Computing Recommendation Lists

Once the network of encounters has been computed, FriendSensing processes it to compute personalized lists of people each user may know, that is, to predict which of A's encounters are likely to be A's friends.



(a) (b)

Fig. 2. Networks of Encounters. Link weights are **(a)** friendship probabilities or **(b)** ranks.

In the literature of social networks, this problem is called "link prediction" and different methods have been proposed to tackle it [8]. These methods assigns a $score(A, B)$ to a pair of nodes (A, B) following one of two possible strategies:

1. Shortest Path - The score between a pair of nodes A and B is the weighted length of the shortest path between them. The intuition behind it is that social networks are "small worlds" (individuals are connected by short chains) and, as such, if there are short paths between A and B, then A and B are likely to befriend each other. The shortest path algorithm accepts weights on the network that represent capacity constraints - in our case, weights that reflect how unlikely it is for two nodes to befriend each other. Since rankings reflect just that (the higher $rankDur_A(B)$ or $rankFreq_A(B)$, the less likely A befriends B), we adopt *rankings* as link weights in the social network of encounters. The path length is then weighted in the sense that it is the sum of the weights along the shortest path.

2. Markov Chain Algorithms - For this class of algorithms, the score between a pair of nodes A and B is computed as the fraction of time spent at B by a random walk in the network originating in A. In this type of network, weights should reflect connection strength between pairs of nodes. Therefore, we adopt friendship *probabilities* $prob(A \rightarrow B)$ as link weights instead. Then, to compute scores, algorithms in this class all convert the network in a first-order Markov chain (hence their common name). The idea is that, after starting at node A (which is called prior node), the walk may unfold in different ways depending on which of the following algorithms is deployed: (1) *PageRank with prior.* At each node, the walk either iteratively moves through one of the node's outgoing links (whose weights are transition probabilities) or jumps back to the prior node A. (2) *K-MarkovChain.* It is similar to "PageRank with prior". The difference is that the walk has now fixed length K. (3) *HITS with prior.* At each node, the walk either moves through one of the node's *incoming* or outgoing links or jumps back to the prior A.

Once scores for a walk originating in A have been computed, they are then used to build A's personalized recommendation list.

The *FriendSensing* framework thus offers eight strategies for recommending friends, derived from combining a strategy for processing proximity data into friendship probabilities (either *frequency* or *duration*), with one of the link-prediction algorithms (*shortest path*, *PageRank*, *HITS*, and *KMarkovChain*).

C. Evaluation

Simulation Setup. The goal of FriendSensing is to recommend to its users people they may know. To ascertain the effectiveness of FriendSensing at meeting this goal, we set up a simulation driven by real data collected as part of the Reality Mining project at MIT. The MIT traces contain colocation information from 96 subjects (staff and students) at the MIT campus over the course of the 2004-2005 academic year, to whom Bluetooth-enabled Nokia 6600 phones were given; colocation information (roughly 10 meters range) was collected via frequent (5 minute) Bluetooth device discoveries. Note that the users covered by the MIT dataset are young adults rather than 'youths'; in particular, thirty users were incoming freshmen, twenty were incoming masters students, and the remaining where older students and staff. However, we expect the results obtained to equally hold in mobility scenarios of youngsters; in fact, as existing analysis

demonstrates [9], [10], such traces share many unifying features (e.g., node inter-contact time, formation of cliques) with other mobility traces (e.g., Cambridge and Dartmouth traces[1]).

Beside providing mobility traces, the MIT dataset also implicitly includes information about the users' social network. In fact, it logs both the text messages sent, and the phone calls made by each phone in the study. Using this information, we have extracted a social network whereby a link between user A and user B is created if A sent a text message or made a phone call to B.

In our simulations, we used the MIT mobility traces to log encounters; using these logs, we ran FriendSensing and computed friends' recommendations. We then compared these recommendations with the MIT actual social network (largest connected component) and computed the fraction of the social network's ties correctly predicted by FriendSensing. We refer to this fraction as "*good* recommendations" g, and we study how g varies while we increase the percentage r of people recommended to each user from 0 to 100%.

Results. To study the effect of the colocation processing strategy separately from the link prediction strategy, we performed two sets of experiments.

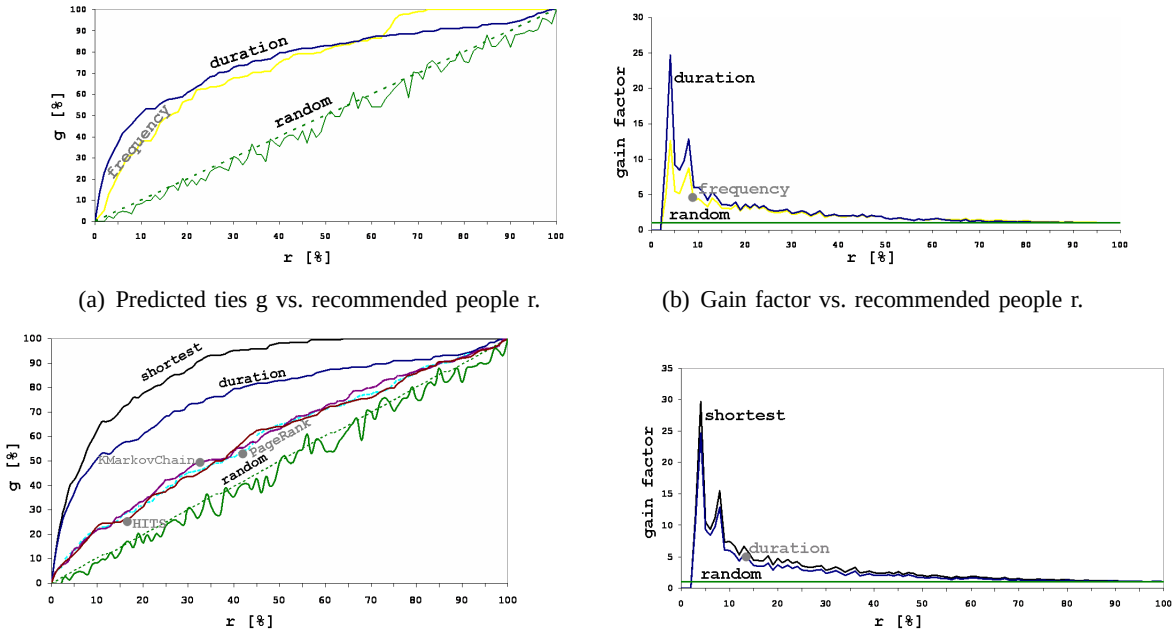
(1) Frequency vs. Duration. In the first set of experiments, we aimed to compare the effectiveness of *frequency* as a colocation processing strategy, as opposed to *duration*. We did so by disabling any link propagation strategy, and by using the ranking produced by the frequency / duration colocation processing strategies locally. This is equivalent to running FriendSensing on people's mobile devices, without reporting their proximity logs to the social-networking website (where the full FriendSensing approach, including link propagation, could be executed). Fig. 3(a) plots g (good recommendations) versus r (recommended people) for these strategies with respect to a *random* selection of people to recommend. For the random strategy, g increases linearly with r - the random strategy fluctuates around a straight line (dashed in the figure). That is because the more people are recommended, the likelier to get some of them right. At the extreme of $r = 100\%$ (all users have been recommended to each user), g reaches 100% (for all strategies). As for the two remaining strategies, they both perform significantly better than random. Note that *duration* discovers friends faster than *frequency*. To see now which strategy performs better over another, we compare *frequency* and *duration* against the random one. We do so by defining the *gain factor* over *random* as:

$$gain_{strategy} = \frac{g_{strategy}}{g_{random}}$$

where $g_{strategy}$ is the fraction of good recommendations for *strategy = duration | frequency*, and g_{random} is that for *random*. A gain factor of one means the strategy performs no better than random (no gain); a factor of two means that the strategy performs twice as better as random. Fig. 3(b) shows that *duration* gains more than *frequency* - especially so for the first 20% of people recommended. As one expects, frequency and duration die off up to a point where both of them flatten toward *random* (no gain). That is because, after recommending most friends, any strategy has left only few friends to recommend, and those are hard to predict.

(2) Duration and "Link Prediction". The second set of experiments compared the four different link prediction strategies presented in Section II-B. We did experiments whereby these strategies were executed on a social network of encounters built using *duration* information and *frequency* information. Since results obtained with *duration* were consistently better than those obtained with *frequency*, we report results for the former case only. Fig. 3(c) plots g versus r for all the four strategies. We also plot the results obtained with our baseline *random* strategy, as well as when using *duration* without propagation, to highlight what privacy-conscious users would miss by not sharing their colocation information for propagation processing. *PageRank*, *HITS*, and *KMarkovChain* perform equally and only show small differences due to confidence on the results. Those results are similar and come from the common use of Markov chains by the three algorithms. Also, one would be better off using only *duration* rather than combining it with those three

[1] http://crawdad.cs.dartmouth.edu/



(a) Predicted ties g vs. recommended people r.

(b) Gain factor vs. recommended people r.

(c) Predicted ties g vs. recommended people r.

(d) Gain factor vs. recommended people r.

Fig. 3. Evaluation Results.

algorithms. That is not necessarily bad news as it suggests that, by relying only on her own proximity information, a user both gets quality recommendations and, while doing so, she retains control of her own data. In line with the literature, *shortest path* performs best. Indeed, Fig. 3(d) shows that it gains more than *duration*, and it does so consistently. That is because, unlike *duration*, *shortest path* is able to suggest to a user A also those friends who belong to the A's social circle but have not been met by A yet.

III. Cultivating Social Contacts: SensingHappiness

Once youngsters find social contacts with FriendSensing, they then need to cultivate them. The idea we are currently exploring is whether, by logging not only encounters but also calls and text messages, mobile phones could help users cultivate their contacts in two ways:

1. Bringing awareness of friendship health. As research suggests [6], there is a misalignment between an individual's perception of how much time is being spent with their social relations, and how much time is actually spent in their proximity. In particular, perception seems to grossly overestimate reality. If users suddenly become notably less sociable with one of their friends, the application running on their phones could thus make them aware that they are perhaps neglecting their friendship. A friendship's wellbeing status can be represented by an avatar or a numeric code, in line with current work on affective computing (e.g., social-networking users of Healthii convey their well-being using avatars [11]). Studies confirm the importance of social interaction in the process of friendship formation and dissolution [12]; moreover, they highlight how people have a tendency to abandon asymmetric relationships (i.e., 'I call you but you never call me'). Of course, users may purposely neglect some relations; a fade-and-forget function can be applied to friends as one ceases interacting with them (i.e., the function archives people with whom one stops interacting). In so doing, one does not need to "unfriend" anyone; rather, it would suffice to just ignore undesirable relations for long enough. This would alleviate traumatic social rejections - currently, 'defriending' someone by dropping them from a friend list results, deliberately or accidentally, in upset feelings [13].

2. Predicting user mood from phone activity. To see how to predict user mood, consider recent findings from the Reality Mining project. This project tracked 94 subjects (students and faculty at MIT) using

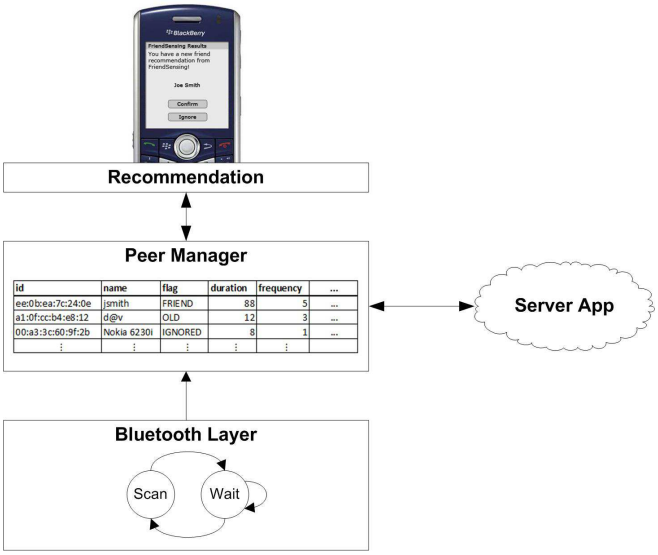


Fig. 4. Schematic representation of the implementation of FriendSensing

mobile phones over the course of nine months, and collected self-report survey data from each individual, where subjects were also asked about their satisfaction with their work group. In their analysis of the Reality Mining data, Eagle, Pentland, and Lazer [6] compared the behavioral data from mobile phones (mobility, calls, and text messages) with the self-report survey data. They found that job satisfaction can be predicted based solely on behavioral data. More specifically, they found that "having friends, especially ones to whom you were near at work, predicted satisfaction with the work group, and calling friends while at work was associated with lack of satisfaction with the work group." The researchers concluded that visible behavioral data (activity with mobile phones) offers an insight into invisible cognitive construct such as mood and job satisfaction. That is why we have started a new project called "SensingHappiness", in which we are analyzing data crawled from Twitter[2] to test whether visible level of activity on Twitter offers any insight into the inferred mood of its users. Twitter is a microblogging service that enables users to post messages ("tweets") of up to 140 characters and supports a variety of communicative practices, including public re-publishing of something other users have written ("re-tweets") in the attempt to spread the word.

Upon this data, we will test whether deviation from 'usual activity' of Twitter users can predict their mood. To do so, we will first detect what (re)tweets express sentiments (e.g., 'I feel …', 'I'm feeling …'); for each of these, we will use existing sentiment analysis algorithms [14] to determine their mood (i.e., happiness / sadness). We will then look at the (re)tweets of each user before and after expressing the sentiment, and we will study whether her level of activity (in terms of number of tweets and re-tweets) deviates from her 'normal' activity. By predicting mood, SensingHappiness makes users aware of their mood and, as a result, users can take action and pay particular attention to their behavior, if need be.

IV. Implementing FriendSensing and SensingHappiness on BlackBerry Phones

To bring our research out of the lab and to the end users, we have implemented FriendSensing and SensingHappiness for the BlackBerry platform (in particular, BlackBerry Pearl 8120). In selecting a target mobile platform, we had the following constraints: first, the platform must allow applications to seamlessly run in the background; second, it must have an API that is openly accessible through open-source languages; third, the API must support Bluetooth access. Although Google have now announced availability of a full Bluetooth API in the upcoming Android 2.0, at the time of development there was no support for it; the iPhone does not allow background applications to run, and any Windows

[2]http://twitter.com/

Mobile application would require development to be done in Microsoft's proprietary .NET languages. As a result, we developed our technology for BlackBerry, as further detailed below, even though a second implementation for Android 2.0-powered devices is underway.

After installing FriendSensing on her BlackBerry, a user is prompted to create a profile by typing her name and email address. The phone sends these two pieces of information, along with its Bluetooth ID and phone number, to a central server over GPRS. The server handles registration and distribution of user profiles. In its current version, the server simply consists of a PHP frontend and MySQL backend; however, these functionalities will be integrated into a Facebook application[3] in the next release. After the user registration step, three software blocks run on the phone (Figure 4):

- Every 10 minutes, the *Bluetooth Manager* initiates a scan of its proximity. The scan results in a list of Bluetooth IDs and device-friendly names of the phones in proximity.
- The *Peer Manager* then processes the list and keeps track of the number of time units each of the phones stays co-located. More specifically, for each phone, the *Peer Manager* collects number of colocations (frequency) and duration (expressed in time units), and also flags the phone to be either *new* (if the phone has not been in range before) or *seen-before* (if the phone has been in range in the past). This is done because it is necessary to filter out at an early stage those phones which have only been in range for short periods of time; only once a phone has been in range for more than 20 minutes do we consider it a potential person of interest and begin long-term tracking.
- Every 7 days, the *Recommendation Engine* produces a list of recommended social contacts, based on colocation frequency or duration; it then downloads the profiles of these people from the server, and shows the list on the phone's screen. The user either confirms or rejects each of the recommendations, and the Peer Manager accordingly switches the flags on the corresponding phones from *seen-before* to either *confirmed* or *rejected*, to avoid repeated recommendations. Social contacts who are confirmed will have SensingHappiness enabled (as discussed below), and will be added on the server's user profile.

Note that both the scan interval (10 minutes) and the aggregation interval (7 days) are tunable parameters. By setting the scan interval to 10 minutes, casual encounters which last only a few minutes are discarded, as not informative of actual social relations. By setting the aggregation interval to 7 days, we aim to capture people's routine, which typically revolves around weekly schedules.

Another service bundled with the FriendSensing application is SensingHappiness. Once a phone has been confirmed as a friend, we enable the SensingHappiness service for these devices. In this mode, the application also logs incoming and outgoing phone calls (duration) as well as the number of incoming text messages (frequency) from that person in order to gauge the strength of the friendship, not just based upon colocation, but also upon more direct forms of communication. After an observation period during which this data is logged (currently set to two weeks, to enable repetition of weekly users' behaviours), patterns of activity can be learned, and deviations from patterns detected. In the current implementation, such changes manifest themselves when the user visualises its social network: 'healthy' connections are represented as thick edges, while neglected relations are visualised as increasingly thinner lines.

The current implementation of the FriendSensing and SensingHappiness technology follows the thick-client model: all the logging and processing happens on the mobile phone. This choice is suitable for privacy-conscious users who prefer to retain full control over their social activity data. However, an alternative deployment could follow the 'thin client-thick server' model instead: the application on the mobile phone simply collects data, and then transfers aggregated information it to the server; all the processing and inferencing then happens server-side. The main advantage offered by this deployment is the possibility to transparently update (and improve) the reasoning engine on the server, without having to re-install a new version of the application on the phones. The next release of the FriendSensing technology will follow this model, with the reasoning taking place server side in a Facebook application, and the logging functionality being ported to a wider variety of mobile phones.

[3]http://developers.facebook.com/

A. Overhead Considerations

We have investigated the overhead incurred by the current implementation of the technology in terms of storage, battery, and communication.

Storage Overhead. The storage overhead a mobile phone would see in using FriendSensing / Sensing-Happiness depends on the number of managed profiles: for each profile, a phone only needs to store the Bluetooth ID and name of the profile along with four counters (frequency of counters, duration of encounters, number of text messages sent, duration of phone calls made), and one flag (set to either 'seen-before', 'new', 'confirmed', or 'rejected'). Even in metropolitan cities, where a phone could keep track of thousands of other devices (i.e., before they are being 'rejected'), the amount of storage used (a few Kilobytes) is negligible with respect to the size of modern mobiles phones (in order of a few Gigabytes).

Battery Consumption and Communication Overhead. Network communication is considered by far the most severe battery draining factor, while various studies have shown that computation causes negligible battery consumption [15]. In the current version of the technology, with all the processing happening client-side, communication is kept to a minimum (i.e., once a week the Bluetooth IDs of recommended friends are sent to the server, and their profiles are pushed back). In order to use the technology, users will be required to leave their Bluetooth enabled; as various studies demonstrate, many people already do so [16], [17], so we were primarily interested in determining the impact that the frequent Bluetooth scanning would have on battery life. To do so, we have run a simple comparative study, with two BlackBerry Pearl 8210 phones with their Bluetooth enabled, carried around by the same user for a week, with only one phone running the FriendSensing technology (i.e., performing Bluetooth scans every 10 minutes, and recording scan results) in both crowded and empty areas; all other functionalities (i.e., calling, texting) were disabled. The handset running FriendSensing had its battery depleted after 5 days, whilst the phone without FriendSensing (but still had Bluetooth turned on) still had around 40% of its battery capacity remaining at this point. Whilst this clearly shows that frequent Bluetooth scanning can have a fairly significant effect on battery life, the phone with FriendSensing did still manage to last for 5 days (not too bad for modern mobile devices, especially running a resource-intensive application) and it is important to remember that the scan interval is tunable by the user to achieve a desired trade-off between accuracy of friend predictions and battery life. When moving to the thin-client version of the technology, all the logged data will have to be transferred to the server for processing, with direct consequences on battery. To avoid drainage, aggregated data - computed using Kalman filters and the likes - could be transferred to the server instead of raw data; also, users could control when uploads occur (e.g., only when at home/when the device is being charged).

V. Related Work

Various approaches exist that aim to automatically discover social relations. They mainly differ in the information they process: social-networking profiles, emails, or data from portable devices.

Social-networking profiles. Chen *et al.* [18] proposed four algorithms for suggesting people on Beehive (IBM internal social-networking website). The algorithms are different combinations of two basic ideas. The first idea is to match people by common interests - to match, for example, those who blog on similar topics or share the same role within IBM. The second is to match people by social connections - to match those who are in "social proximity" of each other by, for example, connecting friends-of-friends. The two ideas match people by the content of their profiles, and the researchers conceded that their ways of matching people are preliminary and should be improved further.

Emails. Karagiannis and Vojnovic gathered the emails exchanged by more than 100,000 employees of their company's research labs [19]. They represented their data as a graph whose nodes are employees

and whose links are email exchanges. Then, to recommend new email addresses for contact lists, they connected "friends-of-friends" relationships.

Portable device data. Upon mobile phone data, Gupta *et al.* showed that it is possible to identify and recommend popular hangouts [20]. More recently, Wyatt *et al.* [21] built a framework with which collar devices capture audio readings and automatically suggest to their users who they may know. Using their audio sensors, collar devices record face-to-face conversations and, based on conversation length, they infer who is likely to befriend whom. The inference is made possible by knowing global properties (e.g., clustering coefficients) of the users' social network. Under this assumption, the promise is that one could *accurately* reconstruct the whole social network.

FriendSensing takes a different, more ubiquitous approach, whereby friends are being recommended starting from readily-available information (proximity data from mobile phones), requiring no a priori knowledge about a user and its social network. In the Reality Mining project, Eagle *et al.* demonstrated that social ties are likely to exist between individuals who behave in a similar way [6], [22]. In FriendSensing, we built upon that work and analyzed how different prediction strategies of social ties would perform.

SensingHappiness uses sentiment analysis techniques proposed by Dodds and Danforth [14]. To determine whether a small group of people is happy or sad, psychologists hand out questionnaires or conduct interviews. In that paper, Dodds and Danforth argued that people are more honest in personal writings than during formal psychological tests. For this reason, they crawled 2.4 million blogs, scanned the texts for more than 1000 emotionally charged words that a 1999 psychology study had ranked on a scale from 1 (miserable) to 9 (ecstatic), and calculated an average happiness score for each blog.

VI. Conclusion

We have explored the fact that behavioral data from mobile phones offers insights into people's friendships and mood, and we have proposed new algorithms for finding social-networking contacts (FriendSensing) and for nurturing online and off-line contacts by tracking their level of engagement with friends (SensingHappiness), which in turn is indicative of users' mood. We have implemented those algorithms on the BlackBerry platform. To bring the technology to a wider public, we are now re-engineering it so to port both FriendSensing and SensingHappiness to different platforms other than BlackBerry (i.e., Android, Symbian) using PhoneGap[4], that is, an open source development tool for building fast, easy mobile applications with JavaScript. The server side component is being replaced by a Facebook application, so to integrate this technology with a widely deployed social networking service. Once this cross-platform implementation is completed, we intend to run a user study that involves a group of year 12 students enrolled in the same school, so that they are expected to spend time together (e.g., while in class, during lunch breaks, studying in the library). The goal is to understand how the technology is perceived by youngsters (e.g., useful, unobtrusive, tedious, intrusive) and whether it ultimately helps nurturing their social contacts.

References

[1] S. Bhatia and S. Bhatia, "Adolescents' Social Environment and Depression: Social Networks, Extracurricular Activity, and Family Relationship Influences," *Journal of Clinical Psychology in Medical Settings*, vol. 16, pp. 346–354, 2009.

[2] K. Hampton and B. Wellman, "Neighboring in Netville: How the Internet Supports Community and Social Capital in a Wired Suburb," *City & Community*, pp. 277–311, 2003.

[3] K. Hampton, L. Sessions, E. J, and H. Rainie, "Social Isolation and New Technology," *Pew Internet Report*, November 2009.

[4] http://phonegap.com/

[4] V. Kostakosand and E. O'Neill, "Cityware: Urban computing to bridge online and real-world social networks," *Handbook of Research on Urban Informatics: The Practice and Promise of the Real-Time City*, pp. 195–204, 2008.

[5] D. Liben-Nowell, J. Novak, R. Kumar, P. Raghavan, and A. Tomkins, "Geographic Routing in Social Networks," *Journal of the National Academy of Sciences*, vol. 102, no. 33, pp. 11 623–11 628, 2005.

[6] N. Eagle, A. S. Pentland, and D. Lazer, "Inferring friendship network structure by using mobile phone data," *Proceedings of the National Academy of Sciences*, vol. 106, no. 36, pp. 15 274–15 278, August 2009.

[7] A. Chaintreau, P. Hui, C. Diot, R. Gass, and J. Scott, "Impact of Human Mobility on Opportunistic Forwarding Algorithms," *IEEE Transactions on Mobile Computing*, vol. 6, no. 6, pp. 606–620, 2007.

[8] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *Journal of American Society for Information Science and Technology*, vol. 58, no. 7, pp. 1019–1031, 2007.

[9] A. Chaintreau, P. Hui, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on opportunistic forwarding algorithms," *IEEE Transactions on Mobile Computing*, vol. 6, no. 6, pp. 606–620, 2007, fellow-Crowcroft,, Jon.

[10] P. Hui and J. Crowcroft, "Human mobility models and opportunistic communications system design," *Philosophical Transactions - Royal Society of London*, 2008.

[11] P. Andre, m. c. schraefel, A. Dix, and R. W. White, "Experience in social affective applications: Methodologies and case study," in *Proc. of* alt.chi, 2010.

[12] Gerhard G. Van De Bunt, Marijtje A.J. Van Duijn, and Tom A.B. Snijders, "Friendship Networks Through Time: An Actor-Oriented Dynamic Statistical Network Model," *Computational & Mathematical Organization Theory*, vol. 5, no. 2, 1999.

[13] danah boyd, "Friends, Friendsters, and MySpace Top 8: Writing community into being on social network sites," *First Monday*, December 2006.

[14] P. Dodds and C. Danforth, "Measuring the happiness of large-scale written expression: Songs, blogs, and presidents," *Journal of Happiness Studies*, July 2009.

[15] 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*. New York, NY, USA: ACM, 2008, pp. 337–350.

[16] L. McNamara, C. Mascolo, and L. Capra, "Media Sharing based on Colocation Prediction in Urban Transport," in *Proc. of the* 14^{th} *ACM International Conference on Mobile Computing and Networking (MobiCom)*, San Francisco, US, 2008.

[17] E. O. Neill, V. Kostakos, T. Kindberg, A. F. gen. Schiek, A. Penn, D. S. Fraser, and T. Jones, "Instrumenting the city: Developing methods for observing and understanding the digital cityscape," in *International Conference on Ubiquitous Computing*, 2006.

[18] Jilin Chen and Werner Geyer and Casey Dugan and Michael Muller and Ido Guy, " "Make New Friends, but Keep the Old": Recommending People on Social Networking Sites," in *Proc. of ACM Conference on Human Factors in Computing Systems (CHI)*, Boston, April 2009.

[19] Thomas Karagiannis and Milan Vojnovic, "Behavioral Profiles for Advanced Email Features," in *Proc. of* 18^{th} *International World Wide Web Conference (WWW)*, Madrid, April 2009.

[20] A. Gupta, S. Paul, Q. Jones, and C. Borcea, "Automatic identification of informal social groups and places for geo-social recommendations," *International Journal of Mobile Network Design and Innovation*, vol. 2, no. 3/4, pp. 159–171, 2007.

[21] Danny Wyatt and Tanzeem Choudhury and Jeff Bilmes, "Learning Hidden Curved Exponential Random Graph Models to Infer Face-to-Face Interaction Networks from Situated Speech Data," in *Proc. of the* 23^{rd} *Conference on Artificial Intelligence (AAAI)*, Chicago, July 2008.

[22] N. Eagle and A. Pentland, "Eigenbehaviors: Identifying structure in routine," *Behavioral Ecology and Sociobiology*, vol. 63, pp. 1057–1066, 2009.