Understandable Learning of Privacy Preferences Through Default Personas and Suggestions

Jonathan Mugan, Tarun Sharma, Norman Sadeh

August 3, 2011

CMU-ISR-11-112

Institute for Software Research
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

This work has been supported by NSF grants CNS-0627513, CNS-0905562, CNS-1012763. This research was also supported by CyLab at Carnegie Mellon under grants DAAD19-02-1-0389 and W911NF-09-1-0273 from the Army Research Office. Additional support has been provided by Nokia, France Telecom, Google, and the CMU/Portugal Information and Communication Technologies Institute.

Key Words: User-controllable machine learning, location sharing

Abstract

As mobile and social networking applications continue to proliferate, they also increasingly rely on the collection of an ever wider range of contextual attributes, location being a prime example. Prior work has shown that people's privacy preferences when it comes to sharing this information are often complex and that expecting users to spend the time necessary to tediously specify these preferences is unrealistic. Yet research has also shown that users are concerned about their privacy and that adequately capturing their privacy will likely be necessary for some of these mobile and social networking applications to be more broadly adopted. The present article reports on research aimed at reducing user burden through the development of two types of user-oriented machine learning techniques: (1) techniques to automatically generate small numbers of user-understandable privacy profiles (or "personas") that users can chose from when configuring their privacy settings, (2) techniques to turn user feedback into suggestions for incrementally modifying a user's existing privacy settings. We study to what extent these techniques, by themselves and in combination, can help users rapidly converge towards their preferred privacy settings in the context of location sharing scenarios where the settings control when and where a user's location is shared with different groups of recipients (close friends and family, Facebook friends, members of the university community, and advertisers). Our results, which are based on data collected from 27 subjects over a period of 3 weeks, suggest that both types of techniques can significantly help users converge towards their desired privacy settings, with the biggest improvements typically resulting from the use of privacy personas.

1 Introduction

An increasing number of mobile and social networking applications rely on the collection of contextual information about their users. The emergence of tens of thousands of location-sensitive mobile phone applications is a prime example. While some of these applications have gained broad adoption, research has shown that the adoption of others has been significantly hampered by the lack of adequate privacy settings. Benisch et al. have shown that this is the case of location sharing applications such as Loopt and Google Latitude, where lack of adequate privacy settings limits users to sharing their location with a very small group of close friends and family members, which in turn significantly diminishes the value of these applications [2]. This work as well as that of others has shown that users often have rather complex privacy preferences when it comes to sharing their locations with others [17, 7, 6, 14]. Typical location sharing preferences may include rules such as "I am willing to let my colleagues see my location but only 9am to 5pm on weekdays and only when I am on company premises," or "I am willing to let my friends find me but only when I'm at bars on Friday night." Specifying such preferences can be a tedious exercise and not one that can realistically be expected from most users. Beyond the required time, users have been shown to often have difficulty articulating security and privacy policies [13, 17]. Even experienced users are known to often struggle with privacy settings such as those found on Facebook [11]. Often new users do not even know what their preferences really are. They need to first use the system to gain a better understanding of where their comfort level lies.

Current solutions to capturing people's privacy preferences have failed to address these challenges. Social networks such as Facebook have been criticized for the plethora of privacy settings they expose to users and the lack of support they provide when it comes to helping users configure them. Android manifests used by mobile apps to request users permission to access sensitive data and functionality request such permissions upfront, namely at the time when users download the app. They do not prompt users to possibly reconsider later on - let alone qualify the conditions when the sensitive data or functionality can be accessed.

One possible path to overcoming these challenges is to harness machine learning and see to what extent it can help capture people's complex privacy preferences while minimizing user burden. This is the approach taken in recommendation systems [1, 10]. In this article, we report on research aimed at using machine learning to identify small numbers of privacy personas that users can chose from to quickly configure potentially complex combinations of privacy settings. If users are to chose between such personas (or policies), the personas have to be easy to understand. While traditional clustering techniques have the potential to generate highly accurate combinations of privacy personas, these techniques cannot be used

in their simplest form, as they would generate policies that would be too complex for users to relate to. Instead, we look at ways of forcing such techniques to generate policies that are understandable. This includes abstracting away idiosyncrasies associated with different users through the introduction of canonical concepts such as "home," "work," or "work hours" that make it possible to reduce people's preferences to more manageable sets of understandable personas.

A second way in which machine learning can help reduce user burden is through the generation of suggestions aimed at helping users refine their privacy preferences over time. Systems such as Amazon, Netflix as well as location sharing applications such as Locaccino, have shown that users are often willing to provide feedback on recommendations or decisions made by a system. This feedback in turn can be used to generate more accurate recommendations, or, in the case of privacy preferences, generate more accurate models of a user's privacy preferences.

Specifically, we explore human-understandable learning of privacy policies in the domain of location sharing. With GPS-enabled smartphones now increasingly common, more people have the opportunity to benefit from location-based services. This makes salient the issue of location privacy [3, 15]. Currently, the most popular method for location sharing is through check-in applications such as Foursquare. But we believe that check-ins are popular in part because there currently is no good way for users to maintain privacy with continuous tracking. Continuous tracking allows usage scenarios that are not available with check-ins, for example meeting someone at a location. We believe that by making progress on privacy settings for continuous tracking, we can increase the usefulness of location-based services. Location sharing is also an example of a relatively new phenomenon, and this gives us a unique opportunity to study how preferences change as users get accustomed to using this new application.

In this paper, we examine the interplay between two complimentary techniques for learning a user's sharing preferences while minimizing user burden. The first approach uses the preferences from other users to learn default sharing personas. If the preferences of users cluster into a few default personas, then by finding which persona best fits the user, we can get the user a long way toward their desired policy with relative ease. Once a user is assigned a sharing persona, we can use the complementary technique of suggesting policy changes to take the user from that rough approximation to his or her desired policy.

These two techniques of default personas and automated suggestions have been examined independently in previous research. Default policies were evaluated in [16, 21], and they found that it was possible to find default personas for users for location sharing. We expand upon these efforts by expanding the use of canonical concepts. We define a canonical concept as a user-understandable abstraction. In [21], locations are represented as polygons on a map. In this work, we make

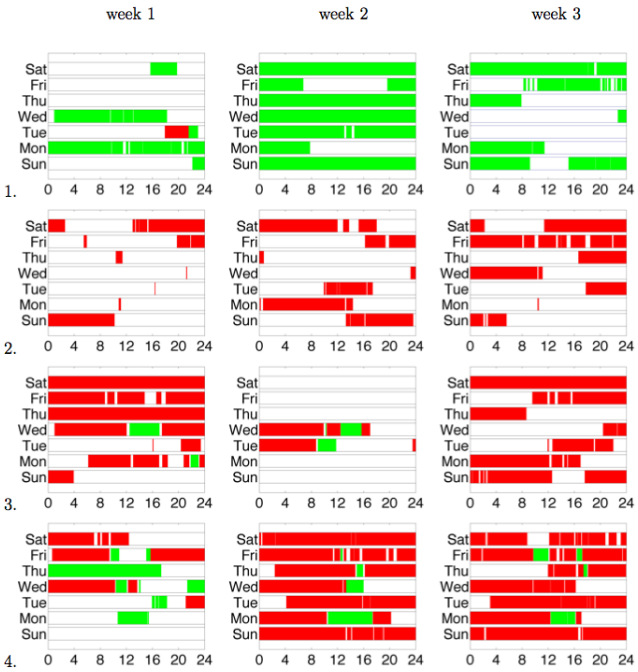


Figure 1: Sharing preferences of users 1 to 4 for the university requester group. The y axis is the day of week. The x axis is time of the day. The three columns represent weeks 1, 2 and 3, respectively. Red (dark gray in grayscale) represents deny, and green (light gray in grayscale) represents allow. White represents that we do not have the information for that time period.

location more general, understandable, and intuitive by instead using canonical locations such as "home" or "work." We expand upon [16] by abstracting time with the canonical concept of a "sharing window." We found that there was often a time during the day when users were willing to share their location and that this time widow varied by user. Our goal is to make the sharing policies more understandable while maintaining high accuracy.

While automated suggestions have been studied extensively within the area of recommendation systems for online shopping [1, 10], in a privacy context the user must understand the current policy. The user does not need to know what the next item the recommendation system will suggest, but the user does need to know if the system will share his or her location when the user is at home. Fang and LeFevre [8] learn a decision tree that specifies to which people a user would be comfortable

sharing his or her date of birth. There has also been work on automated suggestions that are understandable by the user [9]. They used simulated data, and in this work we use data from a study with real users. They also didn't consider location, and in this paper we find that location is particularly important for location sharing policies.

Our results show that by defining canonical concepts, we can learn default personas that take the user a long way toward his or her desired policy. We also show that using these canonical concepts that we can define accurate policies that consist only of a single feature. And we show that there are situations in which the combination of beginning with a default persona followed by neighborhood search is an effective way to bring the user to an accurate policy.

We first explain the study used to collect the data. We then define the canonical concepts used to learn the default personas. Following that description, we explain our method for learning default personas and we explain how the automated policy change suggestions are made via neighborhood search. We then evaluate our method, discuss the results and related work, and conclude.

2 Data Collection

The data for our experiments comes from a user study performed by Benisch et al. [2]. In the study, 27 subjects were tracked for a period of three weeks. At the end of every day, the subjects were asked if they would have been comfortable sharing each location they visited with individuals belonging to these four groups: i) close friends and family (fam), ii) Facebook friends (fb), iii) anyone associated with their university (univ), and iv) advertisers (ad). We refer to this collected data as *audit data*. Table 1 shows an example of the *audit data*. Each row is an observation of time spent by a user at a location. The last four columns specify the user's preference to share his or her location with the four requester groups. The value Y indicates that the user would allow that location be shared, and the value N indicates that the user would not allow that location to be not shared. Figure 1 graphically displays the sharing preferences of the first four users for sharing their location with members of the university based on time and day.

3 Mapping audit data to canonical concepts

To minimize user burden, we map the raw audit data to canonical concepts understandable by the user. Benisch et al.'s [2] analysis suggests that users location sharing preferences are complex and dependent on a number of factors, including

Table 1: Example of two audit points for a single user.

Uid	Arrival time	Departure time	Lat.	Long.	fam	fb	univ	ad
1	11/23/09 9:37	11/23/09 11:26	40.44	−79.95	Y	Y	Y	Y
1	11/22/09 22:23	11/23/09 9:35	40.45	−79.95	Y	N	N	N

time of day, day of week and location. We focus on three factors/variables: *time*, *day*, and *location*, and we define a canonical concept for each.

3.1 Canonical concept for *time*

On analyzing the audit data, we noticed that many of the users shared their location during one time window in a day. The width of this time window varied across users and across the four requester groups in the study. We use this observation to define the canonical concept of the *daytime share window*, or simply the *share window*.

3.1.1 Defining the Daytime Share Window

Each user has an individual daytime share window for each requester group. To find the times for each window, we find the largest time window when the user shares his or her location for at least 80% of the time. Such a time window can be found from the audit data (see Table 1). We know for each user, the arrival time and departure time of his or her every location observation and his or her sharing preference with each requester group for that observation. From all such observations of a user, we search the space of all possible lengths of time windows to find the longest time window when the user shares his or her location for at least 80% of the time he or she was located throughout the user study.

3.1.2 Mapping Times to the Share Window

We map the time spent by a user at a location to either in the *daytime share window* or not in the *daytime share window*. Since the width of the *share window* varies across requester groups, the mapping for the variable *time* will also vary across the requester groups. Hence, for each observation we get four mappings, one for each requester group. If the time spent at a location falls entirely within the *share*

window, we map that observation to in-window. If the time spent at a location falls completely outside the *share window*, we map that observation to not-in-window. If the time spent at a location falls partially in and partially out of the *share window*, we split the observation into fractions of time spent out of the *share window* and inside of the *share window*. Based on how many such fractions we get, each observation gets split into multiple fractions with each fraction either mapped to in-window or not-in-window.

3.2 Canonical concept for *day*

The variable *day* from the audit data can take on any value from the set of day names: {Monday to Sunday}. The domain of the values of day is implicitly canonical and understandable. Our investigation of the data suggested that the sharing preferences of the users were similar during weekdays and also similar for weekend. We therefore map the days of the week to the canonical values weekday and weekend.

3.3 Canonical concept for *location*

When collecting the audits, the user was asked if he or she was comfortable sharing his or her location with the four requester groups at each location visited. The locations were represented by their latitude and longitude and shown to the user on a map.

We analyzed the data and found that the users in our study spent on average 40.57% of their time at the most visited location, 18.74% of their time at the 2nd most visited location and 8.8% of time at the 3rd most visited location. So the top three accounted for almost 70% of the user's locations.

Based on these observations, we defined three canonical values for *location*: home, work and other. We map each latitude and longitude pair to one of these three canonical values. We mapped each latitude and longitude pair where the user spends most of his or her night-time to home. We mapped each latitude and longitude pair where the user spends most of his or her day-time to work. The remaining latitude and longitude pairs were mapped to *other*. We considered night time to be 8:00 pm to 8:00 am and day time to be 8:00 am to 8:00 pm. We checked the location mappings and found them to be reliable. The label *home* was mapped to some residential place and *work* was mapped to the university campus for all the users.

3.4 Result: A Small Number of Canonical States

Mapping audit data to canonical concepts enables us to reduce the values of the variables/features in our study. Table 2 shows the canonical features and the values that they can take. Based on our analysis, there are 12 possible states that matter,

Table 2: Canonical features and values

Feature	Values
time	{in-window, not-in-window}
day	{weekday, weekend}
location	{home, work, other}

specified by the various combinations of values the features in our study can take (ex. *time* = in-the-window, *day* = weekday and *location* = home). And, for each of these 12 states, a user has a sharing preference for each requester group. These 12 states should be meaningful to the user.

4 Learning Default Personas

One could imagine an application designer trying to guess, using his or her intuition, how many different "types" of users there are with respect to privacy and then guessing what the format of those privacy preferences would take. If we can learn the default personas from data, we can eliminate the effort and possible error associated with guessing. We learn default personas in a three-step process as is shown in Algorithm 1. First, a policy is learned for each user. Then the users are clustered based on sharing preferences. And finally, a default persona is created for each cluster. We experimented with k = 1, 2, 3 and 4 clusters (personas). Thus, for each requester group we get k groups of users such that the users within a group have similar sharing preferences.

Algorithm 1 Learning Default Personas

1: for each user, learn a policy for each requester group
2: cluster users with similar sharing preferences
3: learn a default persona for each cluster

4.1 Learning a policy for each user

Each observation in the canonical data represents the sharing preference of a user (Y for allow and N for deny) for a requester group given the features: *time*, *day* and *location*. A policy is a function that represents the sharing preferences of a user and maps the feature values to Y or N. Learning a mapping from the features to sharing preferences is a classification problem. There are many classifiers available such as Naive Bayes, support vector machines, neural networks, and decision trees. We choose to use decision trees because a decision tree can be represented as a set of human understandable rules, such as: "if location is home and day is weekend then classify the sharing preference as deny." For each user, we learn four decision trees, one for each requester group from the canonical audit data.

4.2 Clustering users with similar sharing preferences

For each user and each requester group, use the learned decision tree to create a vector of sharing preferences for all 12 canonical states. We call this vector a *policy vector*. We used the k-means clustering algorithm with hamming distance as the similarity metric to cluster the policy vectors. The hamming distance measures the number of states for which the sharing preference of two users is similar. Hence, two users are said to be similar if their sharing preferences are same for majority of the times. How much similar two users are, hence depends on how often their sharing preferences match. The clustering process produces groups of users with similar sharing behavior.

4.3 Learning a Default Persona for Each Cluster

We now know for each requester group which users have similar sharing preferences. But, what are these preferences? We learn a policy for each group in the same way we learned for individual users. That is, we approximate the sharing preferences of the group of users with a decision tree, with the only difference in the data used to learn the decision tree. Here, we use the data for all the individuals in a group to learn a policy (decision tree) for the group. We simply enlist the canonical data for all the users in a group and run the decision tree-learning algorithm on it. These policies of the groups are our *default personas*.

5 Incremental Suggestions

After the user has chosen a default persona, that user will have a policy that roughly matches his or her preferences. But that policy may not be perfect, so we need a

way to get the user closer to the desired policy. The transition to this better policy needs to be gradual because it is important that the user continue to understand the policy as it changes. To ensure that the policy undergoes a gradual transition, we use incremental suggestions. These incremental suggestions are found via neighborhood search. Neighborhood search is the process of iteratively considering all of the policies close to the current policy, and picking the one that gives the best improvement over the current policy.

After choosing a default persona, the user will have a policy that was generated using a decision tree. We consider each leaf that allows sharing, and we call each such leaf a *sharing rule*. So we have a policy represented as a set of rules.

The neighborhood search allows a single change to a single rule at each iteration. That change can take the form of:

1. adding a new feature and a value. Example: "allow at home" could be modified to be "allow at home only on weekend."
2. changing or adding a value of a feature. Example: "allow at home" to "allow at work," or we could change "allow at home" to be "allow at work or home."

Alternatively, a time window can be changed by:

1. expanding the share window by one hour.
2. contracting the share window by one hour.
3. splitting the share window into two with a one-hour distance between them.

6 Evaluation and Results

We wish to evaluate the procedure where the user begins to use a new application by first choosing a default sharing policy and then refining that policy using automated suggestions. We evaluate the effectiveness of this procedure using real data collected by Benisch et al. [2].

6.1 Evaluating Default Personas

We want to evaluate how well the default personas actually capture the user's sharing preferences. To do this, we varied the number of default policies from 1 to 5. The accuracy is calculated on the complete audit data of the user. For cases where we generate more than one default policy, we consider the most accurate default policy for each. Figure 2 shows the accuracy of the different default policies for each requester group averaged across all of the users.

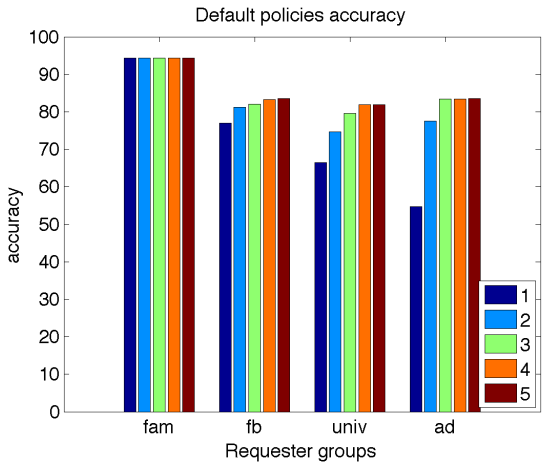


Figure 2: Accuracy of default policies for each requester group. The legend shows the number of default policies varying from 1 to 5

In Figure 2 we see that for the family requester group that varying the number of default policies does not lead to any change in accuracy. Having only one default policy can capture the sharing needs of the users with as high as 90% accuracy. We looked at the default policies suggestions for the family requester group and found that the default policy of always allowing the location to be shared was the most accurate. Since users appear to be comfortable sharing their location with their families, we do not include this group in further experiments.

For the remaining groups, a general trend seen is that accuracy increases if we generate more default policies. This clearly indicates that the underlying user population has diverse sharing needs for these groups as compared to the family requester group. We also note that by having 3–4 default policies can capture around 80% of the sharing needs of the users. Table 3 shows the default policies generated in the experiment. Notice that all of the default policies are intuitive and easily understandable with the exception of the third policy for advertisers. Only four users fell within this cluster, and they were usually willing to share while at a place other than work or home, and the time when they were at work or home tended to be outside of their sharing window. We will see in later experiments that we can learn even more intuitive default policies without significant loss of accuracy.

Table 3: Learned Default Personas

	Friends and Family	Facebook	University	Advertisers
Default 1	Always allow	Always allow	Always allow	Always allow
Default 2	Allow if at home	Always deny	Always deny	Always deny
Default 3	Allow if in-window **OR** allow if at home or other	Allow if at work or other	Allow if at work **OR** allow if at other on weekday	Allow if at work **OR** allow at other while not-in-window

6.2 Evaluating Default Personas + Incremental Suggestions

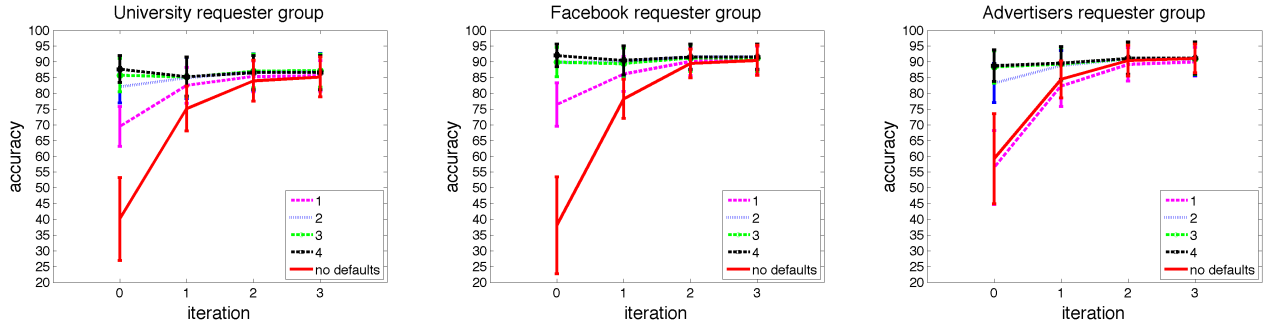
To evaluate the combination of choosing default policies followed by incremental suggestions, we first assume that the user chooses the best default policy for his or her needs. For incremental suggestions, we wish to model the real world scenario where audits arrive in chronological order and policy change suggestions are made as an ongoing process. Hence, the training data is split into three equal parts maintaining the order in which it was collected from the user. Each split contains an average of 21 audit points.

Algorithm 2 Evaluation Procedure for Each User

1: Randomly select 20% of the user's audit points as testing data
2: Split the 80% remaining training data into 3 parts maintaining temporal order
3: Learn k default policies using the audit points from the *other* users
4: Choose the default policy for the user that does best on all of the user's data
5: Determine the accuracy of the chosen default policy on the testing data (*iteration 0 in graphs*)
6: Generate an incremental suggestion using the chosen default policy and the first third of the training data
7: Determine the accuracy of the modified policy on the testing data (*iteration 1 in graphs*)
8: Generate an incremental suggestion using the modified policy and the first and second third of the training data
9: Determine the accuracy of the new modified policy on the testing data (*iteration 2 in graphs*)
10: Generate an incremental suggestion using the new modified policy and all of the training data
11: Determine the accuracy of the final policy on the testing data (*iteration 3 in graphs*)

We provide incremental suggestions in three iterations. The first iteration uses the first part of the training data, the second iteration uses the first and second part of the training data, and the third iteration uses all three parts of the training data. At each iteration, we use neighborhood search to find the policy one step away from the current policy that answers the maximum number of audit questions considered in that iteration correctly. This best policy is the incremental suggestion. If this policy is more accurate than the current policy of the user, it is suggested to the user and it becomes the current policy of the user. We assume that at each iteration the user accepts the incremental suggestion. This process is summarized in Algorithm 2.

We present the test set accuracy of the incremental suggestion at each iteration. Iteration 0 corresponds to the accuracy of the default policies on the test set. The accuracy graphs for each requester group averaged across the users in our study are shown in Figure 3. The graphs also show the error bars with 95% confidence intervals. We experimented with 1, 2, 3 and 4 default policies.



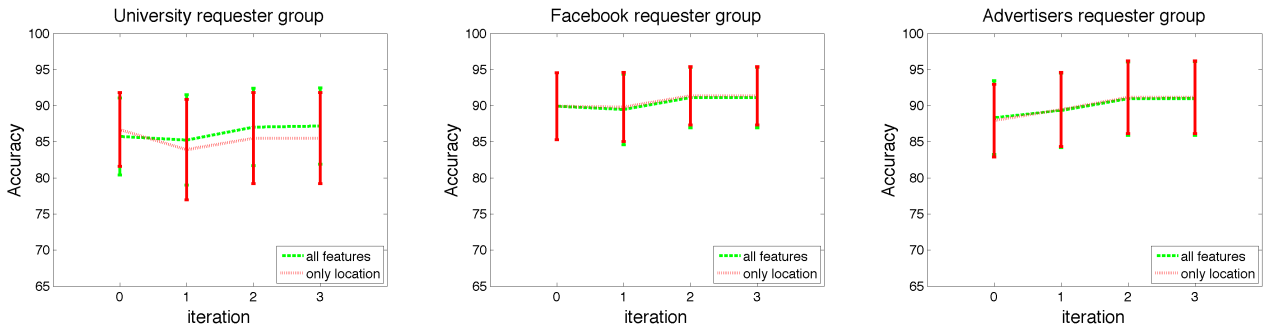
(a) (b) (c)

Figure 3: Test set accuracy of default policies followed by incremental suggestions for each requester group. The legend shows the number of default policies varying from 1, 2, 3, 4, and no defaults. Note that having no default is evaluated as a policy that always denies the location request. (Figure best viewed in color.)

6.3 Default Policies with One Feature

We want to explore if we can make the default policies even easier for a user to understand. Until now, we have generated the default policies using all of the features: *location*, *time* and *day*. Table 3 shows these default policies. We noted that some default policies are very simple and dependent on only one feature such as "allow if at work," while some default policies, though still understandable, might contain more multiple features. We further simplify these defaults by generating default policies that contain only one feature. We experimented with all the features in our study one at a time, and we generated one-feature default policies for each different feature.

Figures 4, 5, and 6 show the results for generating three, one-feature default policies fore each requester group. We see that single-feature default policies did well compared with multiple-feature default policies and that using the location feature worked particularly well. We see in Table 4 that these policies are simpler. Note that for the family requester group, since most people are willing to share with family, we did not learn 3 unique default policies.

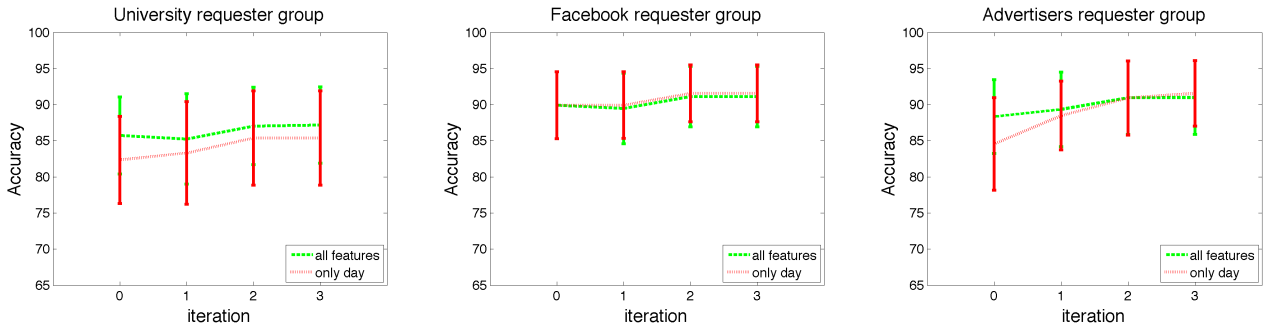


(a) location (b) location (c) location

Figure 4: Comparison of accuracy for each requester group when we generate 3 default policies each consisting only of location compared with default policies containing all features. We see that for location that the accuracy is nearly identical to using all of the features.

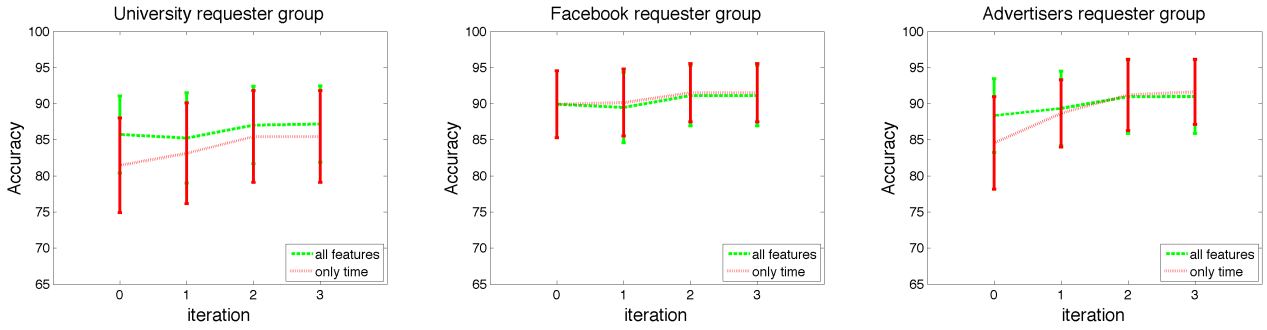
Table 4: Learned Default Personas Using Only Location

	Friends and Family	Facebook	University	Advertisers
Default 1	Always allow	Always allow	Always allow	Always allow
Default 2	Always allow	Always deny	Always deny	Always deny
Default 3	Allow if at home	Allow if at work or other	Allow if at work or other	Allow if at work or other



(a) day (b) day (c) day

Figure 5: Comparison of accuracy for each requester group when we generate 3 default policies each consisting only of day compared with default policies containing all features.



(a) time (b) time (c) time

Figure 6: Comparison of accuracy for each requester group when we generate 3 default policies each consisting only of time compared with default policies containing all features.

6.4 Valuing Privacy over Sharing

A user may be more unhappy if the system makes an error of disclosing the user's location when actually the user doesn't want to as compared to when the system conceals the location when the user want's to share. We performed the experiment again assuming that the user will be ten times more unhappy if the system discloses the location when the user doesn't want to as compared to the system concealing the location when the user wants to share.

We have been computing the accuracy as

$$\text{accuracy} = \frac{T_N + T_P}{T_N + T_P + F_P + F_N} \tag{1}$$

where T_N is the number of true negatives (not share when the user doesn't want to), T_P is the number of true positives (correctly share when the user desires it), F_P is the number of false positives (share when the user does not want the location shared), and F_N is the number of false negatives (not share when the users wishes the location to be shared).

To model the situation where having a user's location be given out when the user doesn't want it as being worse than the other way around, we let $c = 10$ and define accuracy as

$$\text{accuracy} = \frac{T_N + T_P}{T_N + T_P + cF_P + F_N} \tag{2}$$

Figure 7 shows the accuracy of the policies with a F_P error penalty of $c = 10$. The graphs for the case where we generate 3 default policies per requester group are shown in Figure 8. These graphs show us that with the penalty the accuracy is lower, but we see the same pattern.

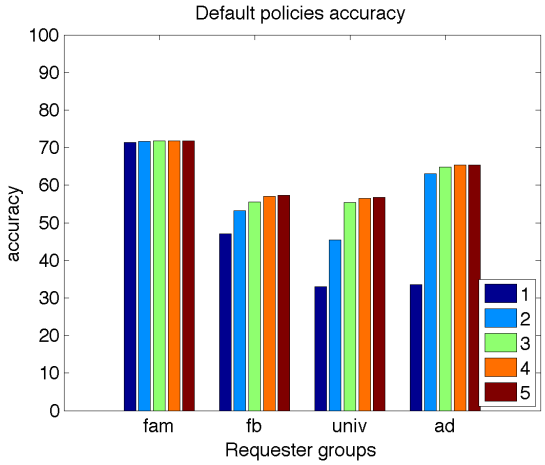
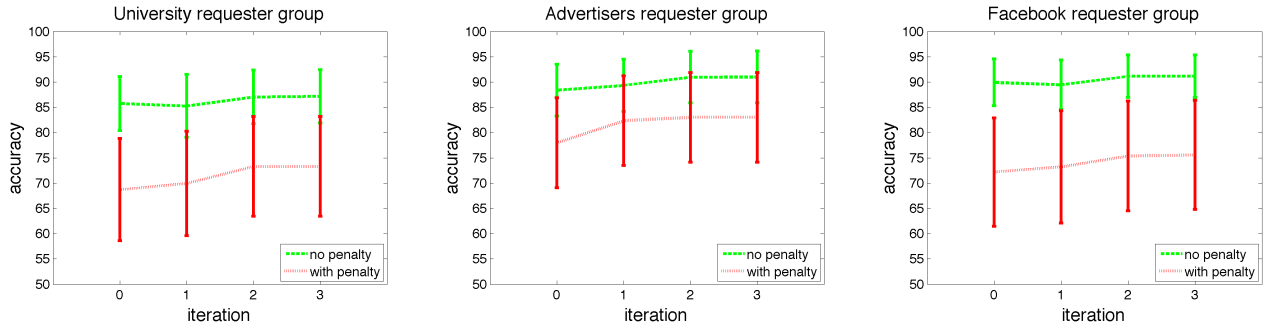


Figure 7: Default policy accuracy when an extra penalty is imposed for sharing when the user did not want the location shared.



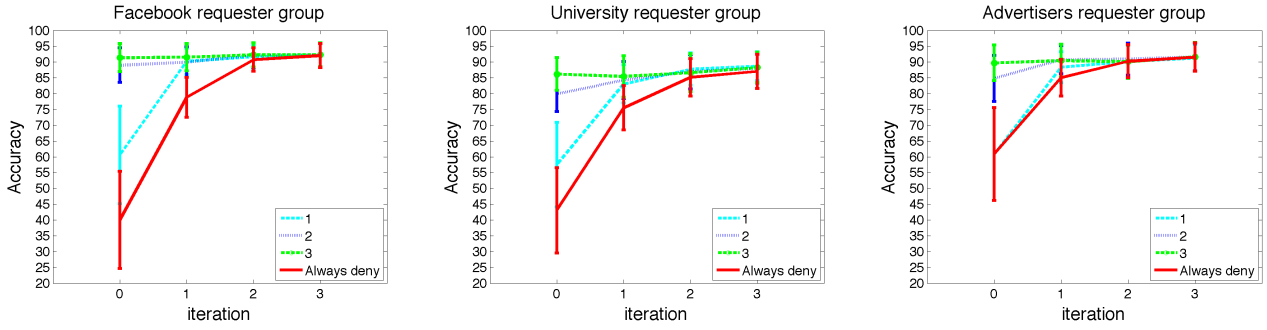
(a) (b) (c)

Figure 8: Comparison of accuracy for the university requester group when we incur an extra penalty for revealing the user's location when he or she wishes to be hidden.

6.5 Policies with one share window for all requester groups

Until now, we have worked with four different share windows, one for each requester group. If we build a wizard to assist users specify their default privacy policy, it will become cumbersome for users to specify a share window for each requester group. With an intent to further ease the process, we restricted our algorithm to have only one share window for all the requester groups. We find the longest time window when the user shares his or her location with all the requester groups for at least 80% of the time he or she was located throughout the user study.

We wanted to explore how accuracy of our policies is affected when we further introduce simplifications in our process. Figure 9 shows the accuracy plots for the same. As can be seen in Figure 9, the simplified policies perform well. Even if these simplified default policies are not fairly accurate, our incremental policy suggestion algorithm helps in increasing the accuracy.



(a) (b) (c)

Figure 9: Test set accuracy of default policies followed by incremental suggestions with one share window for all requester groups. The legend shows the number of default policies varying from 1, 2, 3 and *Always Deny* default policy.

6.6 F1 measure as an evaluation metric

We have been using accuracy as the evaluation metric, and we suggest to the user the most accurate policy based on the audit points. Accuracy, as a classification evaluation metric, works fine when we have evenly distributed data. By evenly distributed data we mean that the number of instances for each category to be predicted (allow or deny in our case) are almost equal. In our case the data for many users is skewed. This means that the number of instances of either one of the categories: allow or deny, are significantly less. So, for example, if a user denies to

share her location for 90% of the times, then an algorithm tuned to suggest the most accurate policy will suggest a policy: *always deny*, which will be 90% accurate. What is more interesting is to find under what conditions the user allows to share her location for the remaining 10% of the data.

The F1 measure is a well known evaluation metric used in the Information Retrieval community for skewed data. It is defined as the harmonic mean of precision and recall. Where precision is calculated as

$$\text{precision} = \frac{T_P}{T_P + F_P} \tag{3}$$

and recall is calculated as

$$\text{recall} = \frac{T_P}{T_P + F_N} \tag{4}$$

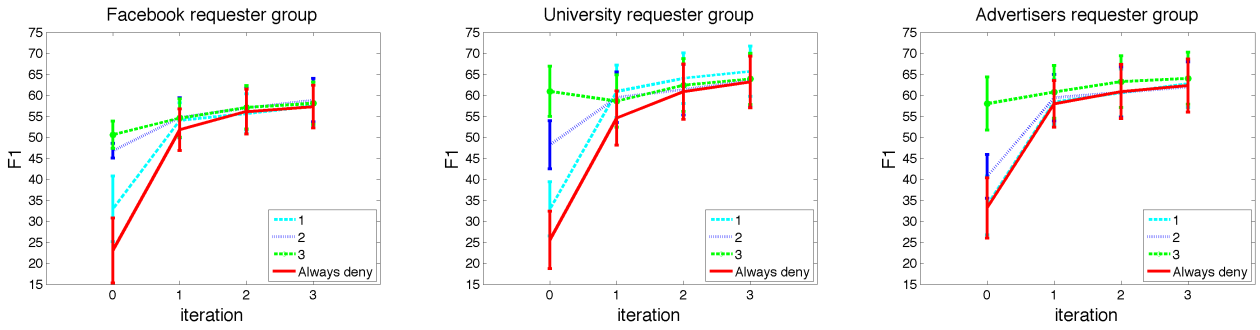
The F1 measure is calculated as

$$\text{F1} = \frac{2 * precision * recall}{precision + recall} \tag{5}$$

The F1 measure does not take into account the True Negatives for a category. In our case we have two categories or decisions to make: allow or deny. For each category we calculate the F1 measure and then take an average of the two F1 measures. Such an average is called macro F1 measure. The macro averaged F1 measure is sensitive to the category with fewer data points.

We use evaluation metrics in two ways. The first use is in the incremental suggestion algorithm as a measure of improvement of a neighboring policy over the current policy. Recall that we select the neighboring policy with the largest improvement. The second use of an evaluation metric is in communicating our results to the scientific community, such as in Figure 10. We have two different metrics, namely, accuracy and macro F1 measure, and either one can be used for each of the two purposes of the evaluation metric. We have already seen the results of using accuracy as a measure to make policy change suggestions and evaluating the system (Figure 3). We experimented with using the F1 measure to make policy change suggestions and to evaluate the system; Figure 10 shows the macro F1 values for each requester group averaged across the users in our study.

When we used macro averaged F1 measure for making suggestions and evaluation, we saw that the graphs (Figure 10) are very similar to the ones obtained when we use accuracy for evaluation (Figure 3). The difference is in the suggestions made by the neighborhood search for users with skewed data. For example, User 110 has very few data instances when she denies her location. If we use accuracy as an evaluation metric, then *always allow* policy is suggested. On the other hand, if macro F1 is used as an evaluation metric, then *allow if at work or other place*



(a) (b) (c)

Figure 10: Test set macro averaged F1 measure of default policies followed by incremental suggestions for each requester group. The legend shows the number of default policies varying from 1, 2, 3 and *Always Deny* default policy. The suggestions for policy change are chosen using macro averaged F1.

policy is suggested. The suggestions are however same for users that do not have skewed data. We also see a dip in the F1 measure for iteration 1 in Figure 10 (b). This is because in iteration 1 we have less amount of data and the suggestions are tuned to be in line with this data. When tested on randomly selected test data we might see a dip or increase in the macro F1 value for a user. It happens that in this case on average the macro F1 value dips in the 1st iteration.

We also experimented with using accuracy as a measure to make policy change suggestions and then evaluating the system using the macro F1 measure. Figure 11 shows these graphs. We wanted to see how the suggestions made by using accuracy as a metric fare when evaluated using F1. We obtained similar graphs as Figure 10. We saw that, on average, incremental suggestions made using accuracy had increasing F1 values. But, as we saw earlier, incremental suggestions using F1 measure made more sense for users with skewed data.

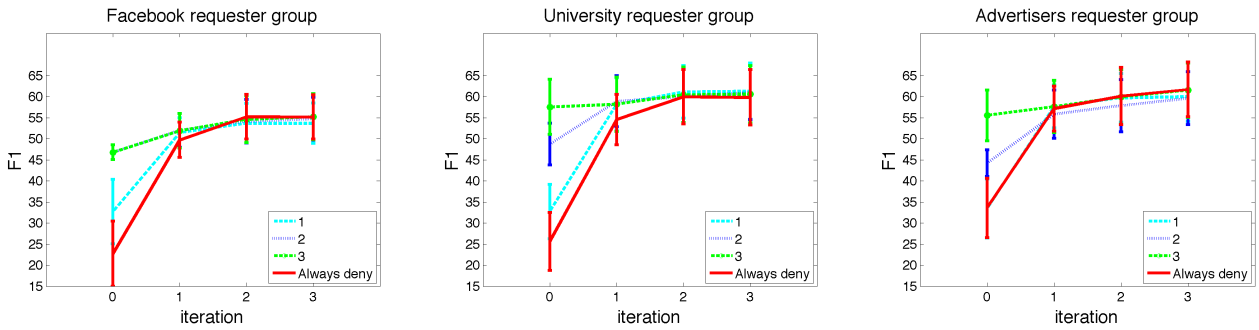
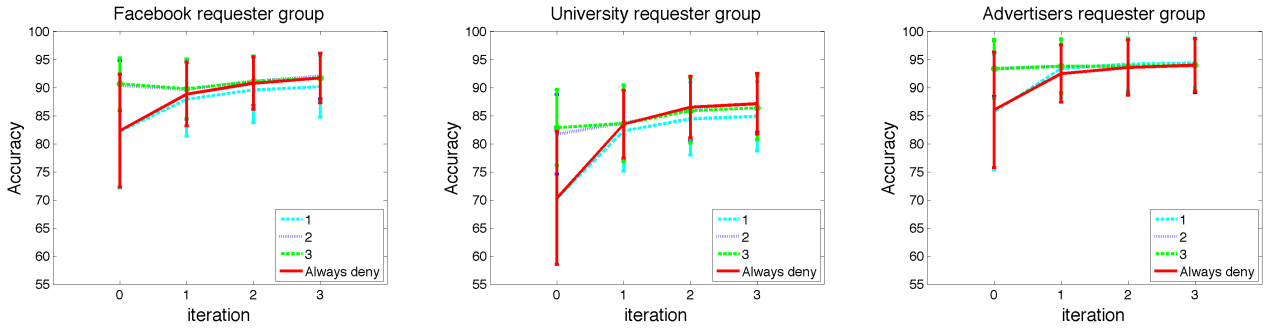


Figure 11: Test set macro averaged F1 measure of default policies followed by incremental suggestions for each requester group. The legend shows the number of default policies varying from 1, 2, 3 and *Always Deny* default policy. The suggestions for policy change are made using accuracy as an evaluation metric.

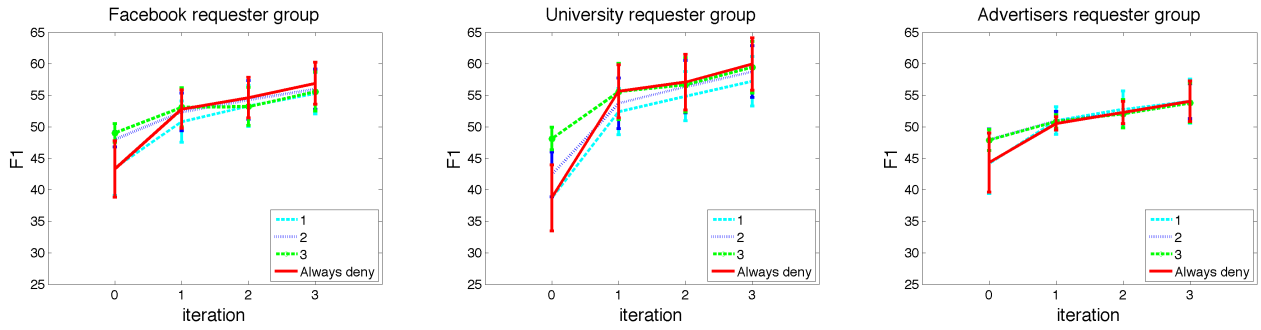
6.7 Validating our approach on new datasets

We also evaluated the performance (Accuracy and F1) of our approach of generating default policies and incremental suggestions on two completely new data sets. The new data comes from two different user studies: one conducted in China and another in the United States. These user studies are similar to the one conducted by Benisch et al. [2]. Figure 12 shows the accuracy graphs and Figure 13 shows the F1 graphs for the data collected from China user study. Figure 14 shows the accuracy graphs and Figure 15 shows the F1 graphs for the data collected from US user study. Evaluating our algorithm on dataset collected from two different user studies reinforced the utility of default policies and iterative suggestions in improving a user's policy. The accuracy and F1 measure graphs for the China and US user studies are similar to the graphs for the user study by Benisch et al. [2]. We also combined the data for users from United States from the new user study with the user study conducted by Benisch et al. [2]. Figure 16 shows the accuracy graphs for the combined data for users from United States. Benisch's et al. [2] user study was also conducted with users from United states and the users for both the studies were recruited from the university. Our observation was same even when we performed our experiment with more data. Two to three default policies were accurate enough for representing sharing needs of United States users.



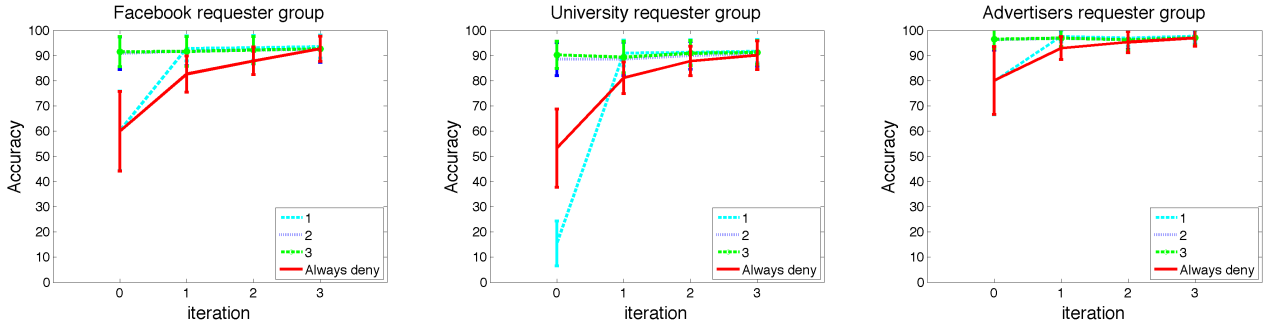
(a) (b) (c)

Figure 12: Test set accuracy of default policies followed by incremental suggestions for each requester group for the data collected from China user study. The legend shows the number of default policies varying from 1, 2, 3 and *Always Deny* default policy. The suggestions for policy change are made using accuracy as an evaluation metric.



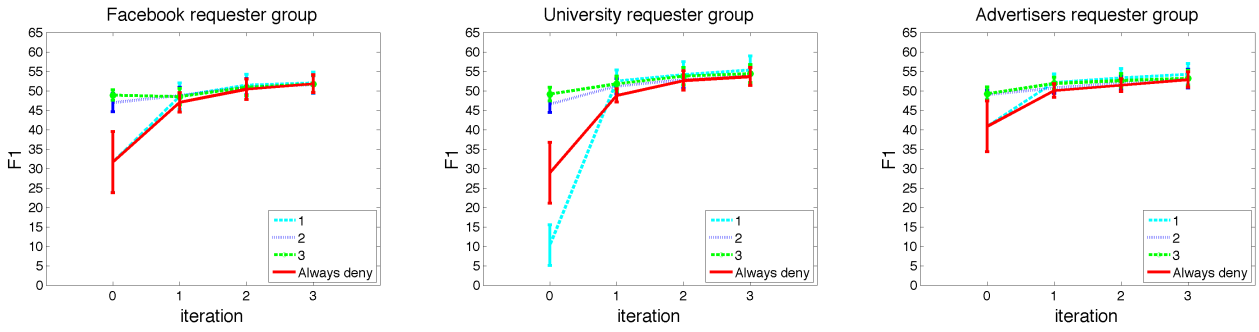
(a) (b) (c)

Figure 13: Test set macro averaged F1 measure of default policies followed by incremental suggestions for each requester group for the data collected from China user study. The legend shows the number of default policies varying from 1, 2, 3 and *Always Deny* default policy. The suggestions for policy change are chosen using macro averaged F1.



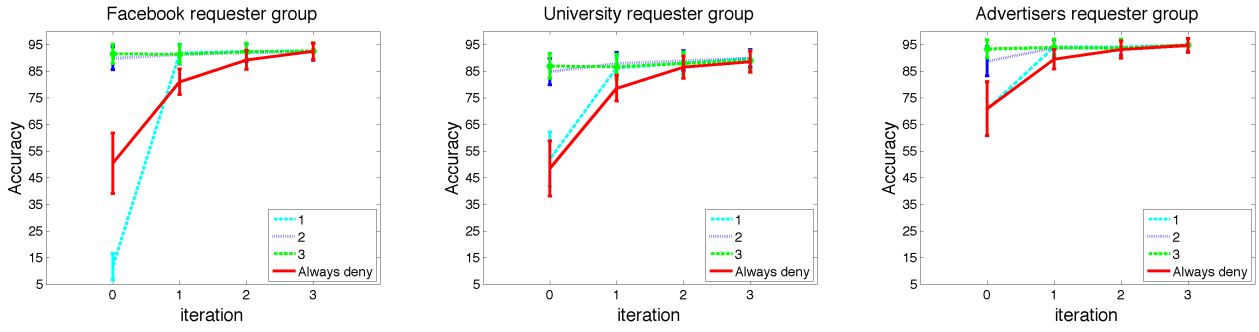
(a) (b) (c)

Figure 14: Test set accuracy of default policies followed by incremental suggestions for each requester group for the data collected from US user study. The legend shows the number of default policies varying from 1, 2, 3 and *Always Deny* default policy. The suggestions for policy change are made using accuracy as an evaluation metric.



(a) (b) (c)

Figure 15: Test set macro averaged F1 measure of default policies followed by incremental suggestions for each requester group for the data collected from US user study. The legend shows the number of default policies varying from 1, 2, 3 and *Always Deny* default policy. The suggestions for policy change are chosen using macro averaged F1.



(a) (b) (c)

Figure 16: Test set accuracy of default policies followed by incremental suggestions for each requester group. The data is the combined data collected from US user study and user study by Benisch. The legend shows the number of default policies varying from 1, 2, 3 and *Always Deny* default policy.

7 Discussion

We see in Figure 2 that by using canonical concepts we can learn a small number of default personas and still have relatively high accuracy. We can also get a sense of the performance of default policies by looking at Figure 17.[1]

Figure 17: Performance of default policies for users 1 to 4 throughout the study. The y axis is the day of week. The x axis is time of the day. The three columns represent week 1, 2 and 3 respectively. Cyan (light grey when viewed in black-and-white) represents correct predictions. Black represents the errors. White represents that we do not have the information for that time period.

A theme throughout the results is that the iterative suggestions to not improve the policy as much as one might expect. Users are not fully consistent in their responses, so we do not expect learning to reach 100% accuracy. However, iterative suggestions can improve a user's policy when the user starts out with only a single learned default or a default of "always deny" (no default) as is shown in Figure 3. We also found users for whom none of the default policies were able to capture their

[1] Need to say more here.

sharing needs with high accuracy and incremental suggestions were instrumental in helping to converge to a more accurate policy.

Table 5 shows three such users and the test set accuracy of the default policy followed by the test set accuracies of incremental suggestions at each iteration for the case when the requesting individual is a member of the university community. The results in Table 5 are for the case when we start with three default policies and assume that user chooses the most accurate one. We can see that in one iteration, that is after taking around 21 audit points, the incremental suggestion can improve the accuracy for the users under-represented by the default policies. We also include the case of user 4, where the default policy does a good job and the incremental suggestions complement it.

Table 5: Accuracy of default policies followed by incremental suggestions for specific users

Uid	Default policy	1st iteration	2nd iteration	3rd iteration
8	65.69%	72.82%	73.39%	73.39%
17	54.00%	77.77%	78.01%	78.01%
4	80.90%	91.79%	92.16%	92.16%

We can look more closely at user 8 in Table 5 to see an example of the kinds of suggestions that are made. The default policy for this user was "allow if at work OR allow if at other place on weekday." The first suggestion is to change this policy to "allow if at work or other place on weekday from 17:00 to 18:00 hrs OR allow if at work on weekend." As we can see in Figure 18, this suggestion makes sense.

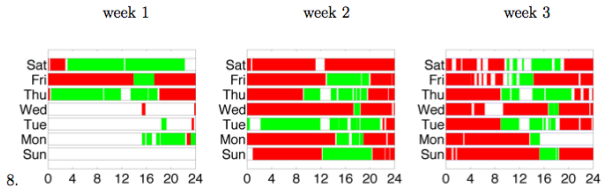
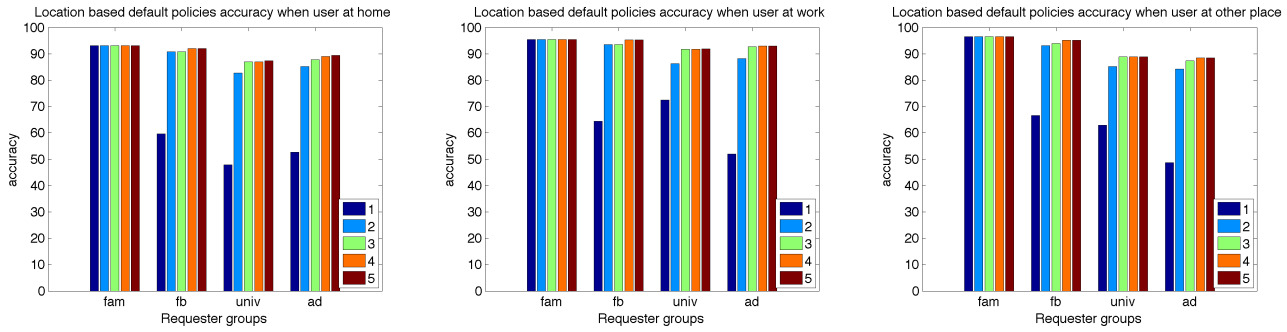


Figure 18: Sharing preferences of user 8. Notice the window at about 5:00 PM (17:00), this was captured by the incremental suggestion.

We found that we could make default policies simpler if we restricted our-

selves to the canonical concept of location as shown in 4, 5, and 6 and Table 4. We can look more deeply at this phenomenon by looking at the accuracy for default policies by type of location are shown in Figure 19. Across the three canonical locations of home, work, and other, we see the same pattern that we saw in Figure 2.



(a) (b) (c)

Figure 19: Accuracy of only location based default policies based on the location of the user.

8 Related Work

Shklovski et al. [19] looked at how tracking technology alters the power and responsibility dynamics in the relationship between a parole officer and the parolee. An interesting contrast looking at trust is provided by Boesen et al. [4]. They looked at the interplay between location tracking and trust among people in relationships. They argued that if you always can watch someone, then you might not have the opportunity to build up trust.

Within the domain of privacy, Scipioni and Langheinrich [18] outlined the privacy concerns associated with mobile location sharing technology. Mancini et al. [12] focus on location sharing privacy based on defining the situation relative to relationships and cultural situations, as opposed to simply the place on a map. And Toch et al. [20] explored how people are more willing to share their locations at places with high entropy (where a lot of different people go) as compared to places with low entropy. Brush et al. [5] explored how users felt about their location being traced at what level they would want their location obfuscated.

9 Conclusion

Mobile computing and social networking entail the collection of sensitive contextual information. Research has shown that many of these contextual attributes can give rise to complex privacy preferences, which cannot adequately be captured using current techniques. In this article, we reported on research to develop user-understable machine learning techniques capable of helping users specify their privacy preferences while alleviating user burden.

Our method focuses on the generation of user-understandable privacy personas and the generation of suggestions intended to help users incrementally refine their privacy preferences over time. Privacy personas are generated using clustering techniques that are constrained to the exploration of simple to understand privacy policies, enabling users to effectively identify the policy (or persona) that best captures his or her preferences. Incremental suggestions for policy refinement are generated using feedback obtained from users over time as they audit decisions made by their current privacy policy. By restricting suggestions to incremental modifications of the user's current policy, our technique aims to ensure that users can continue to relate to their privacy policy as it gradually evolves over time.

Specifically, we reported on experiments aimed to study the impact of these two approaches both in isolation and in combination. Our results, which are based on data collected from 27 mobile users over a period of 3 weeks, indicate that, when it comes to location sharing, a small set of two to four relatively simple privacy personas can go a long way towards capturing what would otherwise be characterized as a complex set of multi-faceted privacy preferences with both time and location restrictions. For some users however these privacy personas are not sufficient and incremental suggestions are necessary to help them converge towards acceptable levels of accuracy. In the absence of privacy personas, the role of incremental suggestions is even more significant. Future research will aim to further evaluate different parameters associated with our approach. This includes looking at how users respond to different numbers of privacy personas, personas of different levels of complexity, different numbers of suggestions, and other questions revealed by user studies.

References

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, pages 734–749, 2005.

[2] M. Benisch, P. Kelley, N. Sadeh, and L. Cranor. Capturing Location-Privacy Preferences: Quantifying Accuracy and User-Burden Tradeoffs. *Journal of Personal and Ubiquitous Computing*, 2011.

[3] A. Beresford and F. Stajano. Location privacy in pervasive computing. *Pervasive Computing, IEEE*, 2(1):46–55, 2005.

[4] J. Boesen, J. Rode, and C. Mancini. The domestic panopticon: location tracking in families. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pages 65–74. ACM, 2010.

[5] A. Brush, J. Krumm, and J. Scott. Exploring end user preferences for location obfuscation, location-based services, and the value of location. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pages 95–104. ACM, 2010.

[6] T. Burghardt, E. Buchmann, J. Müller, and K. Böhm. Understanding user preferences and awareness: Privacy mechanisms in location-based services. *On the Move to Meaningful Internet Systems: OTM 2009*, pages 304–321, 2009.

[7] S. Consolvo, I. Smith, T. Matthews, A. LaMarca, J. Tabert, and P. Powledge. Location disclosure to social relations: why, when, & what people want to share. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 81–90. ACM, 2005.

[8] L. Fang and K. LeFevre. Privacy wizards for social networking sites. In *Proceedings of the 19th international conference on world wide web*, pages 351–360. ACM, 2010.

[9] P. Kelley, P. Hankes Drielsma, N. Sadeh, and L. Cranor. User-controllable learning of security and privacy policies. In *Proceedings of the 1st ACM workshop on Workshop on AISec*, pages 11–18. ACM, 2008.

[10] G. Linden, B. Smith, and J. York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[11] M. Madejski, M. Johnson, and S. M. Bellovin. The failure of online social network privacy settings. Technical Report CUCS-010-11, Department of Computer Science, Columbia University, February 2011. In submission.

[12] C. Mancini, K. Thomas, Y. Rogers, B. Price, L. Jedrzejczyk, A. Bandara, A. Joinson, and B. Nuseibeh. From spaces to places: emerging contexts

in mobile privacy. In *Proceedings of the 11th international conference on Ubiquitous computing*, pages 1–10. ACM, 2009.

[13] R. Maxion and R. Reeder. Improving user-interface dependability through mitigation of human error. *International Journal of Human-Computer Studies*, 63(1-2):25–50, 2005.

[14] M. Mazurek, J. Arsenault, J. Bresee, N. Gupta, I. Ion, C. Johns, D. Lee, Y. Liang, J. Olsen, B. Salmon, et al. Access Control for Home Data Sharing: Attitudes, Needs and Practices. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI)*, 2010.

[15] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh. Location Privacy via Private Proximity Testing. In *18th Annual Network & Distributed System Security Symposium*, 2011.

[16] R. Ravichandran, M. Benisch, P. Kelley, and N. Sadeh. Capturing social networking privacy preferences. In *Privacy Enhancing Technologies*, pages 1–18. Springer, 2009.

[17] N. Sadeh, J. Hong, L. Cranor, I. Fette, P. Kelley, M. Prabaker, and J. Rao. Understanding and capturing peoples privacy policies in a mobile social networking application. *Personal and Ubiquitous Computing*, 13(6):401–412, 2009.

[18] M. Scipioni and M. Langheinrich. I'm Here! Privacy Challenges in Mobile Location Sharing. In *Second International Workshop on Security and Privacy in Spontaneous Interaction and Mobile Phone Use (IWSSI/SPMU)*, 2010.

[19] I. Shklovski, J. Vertesi, E. Troshynski, and P. Dourish. The commodification of location: dynamics of power in location-based systems. In *Proceedings of the 11th international conference on Ubiquitous computing*, pages 11–20. ACM, 2009.

[20] E. Toch, J. Cranshaw, P. Drielsma, J. Tsai, P. Kelley, J. Springfield, L. Cranor, J. Hong, and N. Sadeh. Empirical models of privacy in location sharing. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pages 129–138. ACM, 2010.

[21] E. Toch, N. Sadeh, and J. Hong. Generating default privacy policies for online social networks. In *Proceedings of the 28th of the international conference extended abstracts on Human factors in computing systems*, pages 4243–4248. ACM, 2010.