# Capturing Social Networking Privacy Preferences:

## Can Default Policies Help Alleviate Tradeoffs between Expressiveness and User Burden?

Ramprasad Ravichandran, Michael Benisch,
Patrick Gage Kelley, and Norman M. Sadeh

School of Computer Science,
Carnegie Mellon University,
Pittsburgh PA 15217, USA
{rravicha,mbenisch,pkelley, sadeh}@cs.cmu.edu

**Abstract.** Social networking sites such as Facebook and MySpace thrive on the exchange of personal content such as pictures and activities. These sites are discovering that people's privacy preferences are very rich and diverse. In theory, providing users with more expressive settings to specify their privacy policies would not only enable them to better articulate their preferences, but could also lead to greater user burden. In this article, we evaluate to what extent providing users with default policies can help alleviate some of this burden. Our research is conducted in the context of location-sharing applications, where users are expected to specify conditions under which they are willing to let others see their locations. We define canonical policies that attempt to abstract away user-specific elements such as a user's default schedule, or canonical places, such as "work" and "home." We learn a set of default policies from this data using decision-tree and clustering algorithms. We examine tradeoffs between the complexity / understandability of default policies made available to users, and the accuracy with which they capture the ground truth preferences of our user population. Specifically, we present results obtained using data collected from 30 users of location-enabled phones over a period of one week. They suggest that providing users with a small number of canonical default policies to choose from can help reduce user burden when it comes to customizing the rich privacy settings they seem to require.

**Key words:** User modeling, Privacy, Mining default policies

## 1 Introduction

Social networking sites such as Facebook and MySpace thrive on the exchange of personal content such as pictures and activities. These sites are discovering that people's privacy preferences are very rich and diverse. While in theory, providing users with more expressive settings to specify their privacy policies

gives them the ability to accurately specify their preferences [7], it can also lead to significant increases in user burden. In this paper, we investigate the extent to which generated default policies can alleviate user burden. The use of default policies has proven to be practical in other domains such as the configuration of compact P3P policies in web browsers (e.g. Internet Explorer, Firefox). Here we explore an extension of this approach by introducing the concept of canonical default policies. These policies abstract away idiosyncratic elements of a user context, making it possible to expose and discover common elements across otherwise seemingly disparate user policies.

Specifically, we explore how we can learn default policies using machine learning techniques, and evaluate their effect in alleviating some of the user's burden in defining their privacy policies. Our objective is to minimize the number of edits the user has to make to the default policy to arrive at a policy which she is willing to use. In this work, we use accuracy as a metric to approximate the user's burden to reach an acceptable policy starting from some initial policy (e.g. a blank policy or a default policy). We assume that a user is more likely to be comfortable using default policies of higher accuracy, and requires less editing to arrive at an acceptable final policy. As reported in previous studies (e.g. [22]), users generally do not require policies that are 100% accurate to start using an application. An example is a user with a policy that allows for less sharing than she ideally would like to have.

In this study, we consider the scenario of helping a new user identify suitable default privacy settings. We present her with a choice of default privacy settings that have been learned from our current set of users' privacy settings that she can easily understand and modify to specify her desired initial policy. This work is complementary to research efforts that make it easier to edit policies to converge to desirable final policies (e.g. user controllable learning [16], example-critiquing [10]).

We conduct this study in the context of location-sharing applications, where users are expected to specify conditions under which they are willing to let others (e.g. friends, colleagues, family members) see their locations based on different contexts (e.g. based on day of the week, time of the day, or where they are).

Prior work [7, 22] has shown that users' privacy policies can be very rich. For example, a user may have a policy that only allows her colleagues to access her location information when she is at work, and during regular business hours. Trying to derive default policies through direct application of machine learning techniques does not yield intuitive or usable polices. We show that one can abstract away individual elements of a user's schedule and the set of locations she visits, to arrive at what we call canonical policies, such as 'allow access while at work', or 'deny access while at home in the evening'. We further show that canonical policies lend themselves to identification of more meaningful and intuitive default policies that users are more likely to be able to customize.

We evaluate the accuracy with which a combination of canonical default policies can cover the final privacy policies for a population of 30 users whose privacy preferences were collected during the course of a week-long study. We

learn users' individual policies using a decision tree algorithm, and cluster the individual policies into a set of more general default policies. We further discuss tradeoffs between intuitiveness of the default canonical policies and the number of such policies. The main contribution of the work is to show that canonical default polices seem to offer a practical solution where traditional application of machine learning techniques yield unintuitive and unusable policies. Our results further suggest that in the case of location-sharing preferences considered in this study, aiming for about 3 default policies is the "sweet-spot", though additional studies may be needed to further validate this result.

## 2 Related Work

To the best of our knowledge, there hasn't been any prior work on learning default policies in location-sharing services. However, we briefly summarize some of the work from the location-sharing services and user preference learning literature in general.

### 2.1 Preference Learning

There has been much prior work in applying machine learning techniques to learn users' preferences. In the recommendation systems literature ([8] is an excellent survey), many successful systems have been proposed and built in both—industry (e.g. Amazon.com, Netflix.com), and academia (e.g. [20],[1]). Recommendation systems can usually be classified into content-based (where the recommendation is based on past behavior), collaborative-filtering (where recommendations are based on preferences of other people with similar taste), or a mixture of both. There are basically two categories of collaborative-filtering: nearest neighbor methods (e.g. [11],[24]), and latent factor modeling (e.g. [12],[23]). Although latent-variable models are closest to our line of work, contrary to their approach, our focus is not in arriving at more efficient representations, but in reducing the amount of user burden in further customizing their policies.

In [16], the authors look at user-controllable learning, which is most related to the field of example-critiquing. Unlike their approach where the user and system tweak a common policy model, our work is geared towards helping users bootstrap their privacy policy.

### 2.2 Location-Sharing Services

There are many commercial location-sharing services such as Loopt [2], Mobimii [4] and Mobikade [3]. Significant academic work has been done in this area (e.g. [25],[21]), where the focus has been on deployment, accurate location detection and implementation of user's privacy policies. A fair amount of work has been done in looking at the types of information that people are willing to share, and the conditions under which they are willing to share, in the form of diary studies [6], interviews [13–15], surveys [18], and experience sampling

techniques [9, 17, 22]. Lederer et. al. suggest that the nature of the requester is the primary factor in choosing whether to disclose information or not [18], while Consolvo et al. determined that along with requester information, the reason for the request and the level of detail were important factors as well [9]. In [7], the authors determined that along with the requester type, time of the request and location of the person during the request were important factors in determining whether to disclose their location information.

## 3 Experimental Setting

Our experimental data comes from a study conducted over the course of two weeks in October 2008. We supplied 30 human subjects with Nokia N95 cell phones for one week at a time (15 subjects per week).

Subjects were recruited through flyers posted around the university campus. Our 30 subjects were all students from our university. The sample was composed of 74% males and 26% females, with an average age of about 21 years old. Undergraduates made up 44% and graduate students made up 56% of the sample.

Our only requirement for entry into the study, was that they must already have an AT&T or T-mobile phone plan, allowing them to transfer their SIM card, into the phone we provided. We required that for the duration of the study the participants use the phone as their primary phone. This requirement ensured that the subjects kept their phones on their person and charged as much as possible. Each of the phones was equipped with our location tracking program, which ran at all times in the background, recording the phone's location using a combination of GPS and Wi-Fi-based positioning to an easily locatable text file on the phone.

Each day, subjects were required to visit our web site and upload this file, from their phone, containing their location information. We processed this file immediately and then presented the participants with a series of questions based on the locations they had been since they last uploaded (allowing flexibility in case they had missed a day).

Every question pertained to a specific location that the participant had been during that day, with a map showing the location and the duration they remained there. For example, a question may have asked "Would you have been comfortable sharing your location between Tuesday October 28th, 8:48pm and Wednesday October 29th, 10:39am with:"

The users were then presented with four groups, to assess whether or not they would have been comfortable sharing their location with each of these groups. The four different groups of individuals were: i) close friends, ii) immediate family, iii) anyone associated with Carnegie Mellon, and iv) the general population, or anyone.

In each of these assessments, or audits, the participants tagged the information gathered with one of the following four classes

1. *Allow* - The participant would allow the requester to share her location.

2. *Deny* - The participant would prefer to not share her location with the requester.
3. *Part of the time* - The participants would like more granularity so that she can allow requester to access her location part of the time.
4. *Part of the group* - The participant would like more granularity so that she can allow only a part (one or several members) of the requester group to access her location information. The main purpose of this designation was to ensure that we did not over-simplify the types of groups we made available in the study.

An additional option was provided for participants to designate the displayed location as inaccurate. When this designation was made the participant was not required to audit for each group, however in practice this was used very infrequently (less than 1%).

The data collection process, specifics of the software implementation, and survey details from pre- and post- questionnaires are explained in more detail in the study [7], and are not repeated here. Although we use the same physical location and auditing data, our focus is different from that study. [7] examines the impact of different levels of expressiveness for privacy mechanisms. On the contrary, we are interested in learning a few default policies that captures the users' privacy policies as accurately as possible while ensuring the policies are both simple and intuitive.

## 4 Data Exploration

In all, we collected a little more than 3800 hours of location information. Less than 1% of the audits were marked as inaccurate. We would like to determine which attributes are useful in specifying default policies. We first only focus on two of the features: time of day, and day of the week. Figure 1 shows a typical sample of a user's access policies – we can see a particular user's audit, with time of day (in hours) on the Y-axis, and day of the week on the X-axis. We have used different colors to indicate the different audit statuses as follows: allow (*green*), deny (*red*), inaccurate location information (*white*), and finally missing information (*dark blue*). An audit is marked missing if the user did not provide information for that time period, or if the user had tagged that audit as 'part of the group'.

### 4.1 Data Cleanup

To learn users' default policies from these data, we first handle the audits with missing information and inaccurate location information; and classify them as 'allow' or 'deny'. The main reason for classifying all data is that we would like to consider duration as a feature in our classifier, and we want to make sure that missing information for a few minutes doesn't affect our classifier. We could inadvertently introduce two types of errors: we classify a deny as allow (false

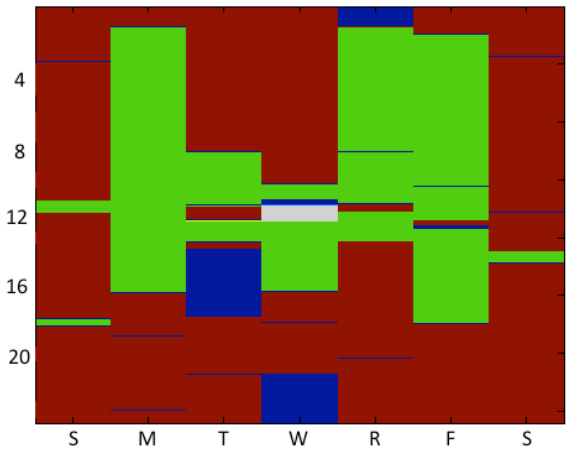


**Fig. 1.** A typical un-sanitized audit. The requester type is 'university community'. The x-axis in each figure is the day of the week, while the y-axis is time of the day (in hours)

positive), or we classify an allow as a deny (false negative). Different users may face different levels of unhappiness depending on the type of classification error. We try to account for this uneven level of happiness by considering the *conservativeness* of a user, as explained below. The conservativeness ratio, $\kappa$ is defined as the ratio of the user's unhappiness when the system wrongly allows access, to the user's unhappiness when the system wrongly denies access, i.e.

$$\kappa = \frac{Unhappiness\ of\ instance\ wrongly\ classified\ as\ allow}{Unhappiness\ of\ instance\ wrongly\ classified\ as\ deny} \tag{1}$$

Our definition of the conservativeness ratio assumes that this ratio is fixed for a user across all mistakes that the system makes. For instance, in a real life situation, a user may be more lenient when a university associate is wrongly denied access to her location, than when the requester is a family member, but we do not consider such scenarios here.

We would like to point out that unlike in [7], we assume that the cost of the system making a mistake is equal to the reward when the system gets a correct prediction. In other words, we assume that for every correct prediction, the user's unhappiness reduces by a unit, while for every wrong prediction, the user's unhappiness increases by a unit.

We may have missing information in our data due to many reasons - the data/location may not have been recorded because the cell-phone was turned off; the GPS/Wi-Fi may not have reported an accurate position; or the specifications (with respect to the number and type of groups, and resolution of time during events) may not have had enough expressiveness. The final classification

of audits for these missing period depends on a number of factors – source of the error, duration of the error, classification of the encapsulating period, and the conservativeness factor, $\kappa$. We found that there is very little correlation between the classification for the same time frame across days, and so we could not use that information to classify missing periods.

If the information is missing for just a couple of minutes (we used a threshold of 5 minutes in this work), or if the source of the missing information is in the data collection process, we classify the audit based on a linear combination of the permissions of the previous and next periods weighted by the conservativeness of the user. In other words, if the missing period is in between two periods with same access types (e.g. both allow), then we classify the period with a similar access type (e.g. allow). The trickier case is when the missing period is in between periods of different access types. For example, if a user allowed access in the period leading to the missing period, and denied access in the period following, and has a conservativeness ratio greater than 1, we deny access during the missing period. Specifically,

$$\delta(period) = \begin{cases} 0 \text{ if } period \text{ is } allow; \\ \kappa \text{ if } period \text{ is } deny. \end{cases} \tag{2}$$

$$class(missing) = \begin{cases} allow \text{ if } \delta(prev.\ period) + \delta(next\ period) < 0.5; \\ deny \ \text{ otherwise.} \end{cases} \tag{3}$$

### 4.2 Some Observations

The sanitized data (time of day, and day of the week dimensions) for the 30 users when the requester is a member of the University is shown in Figure 2. It has only two possible states: allow (*green*) and deny (*red*). The missing audits and inaccurate audits have also been classified albeit with zero penalty weight, which is not shown in this figure. We assign zero penalty weight to make sure we are not penalized for classifications for which we did not have ground truth.

Preferences collected on weekdays were similar, as were those collected on weekends. We can see this trend in Figure 2, where in each of the 30 cells, the access pattern during most of the day on Sunday and Saturday are correlated while access patterns during nights on Friday and Saturday are similar. Thus, we combined the days of the week into a feature (that we use in our algorithms) with just two values: *weekdays*, and *weekends*. We would also like to note that this grouping would lead to a bias in our learning samples since we have more weekday training samples than weekend training samples. However, assuming that the user cares about the average performance of the system, this should not be a problem since this reflects the actual proportion of weekdays to weekend days in a week.

Another trend that we noticed in our data was that, taking all users' audits into account, variance within a requester type was much lower than between requester types. For instance, when the requester type is 'family', there is a higher tendency for the access to be 'allow', than when the requester type is

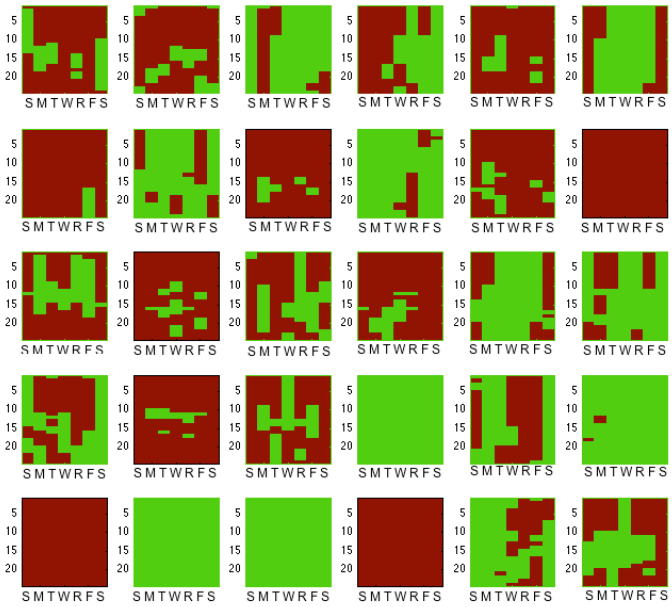


**Fig. 2.** Sanitized audits of the 30 users (Location shared with University members). The x-axis in each figure is the day of the week, while the y-axis is time of the day (0 - 24 hours)

'anyone'. This prompted us to learn rules conditioned on the requester type. Thus, much of the analysis in the rest of the paper will look at each requester type separately, and we will explicitly mention when we do not do so.

We also noticed (as may be expected), that there is a tendency for access permissions to switch from 'allow' to 'deny' as we move from family / close friend requester types to the 'anyone' requester type. For instance, if someone denied their close friends from seeing their location during a particular period, it is highly likely that requesters in the 'university members', and 'anyone' categories would also be denied access. We do not take advantage of this observation in our algorithms currently, but we may use this in the future.

## 5 Methodology

As mentioned earlier, we would like to examine the extent to which we can identify groups of users with similar privacy policies, and ascertain good default policies for them. In addition to the policies being accurate, our aim is to ensure that these policies are both intuitive and understandable, so that our users can customize them to fit their specific needs. While direct application of machine learning algorithms may achieve high accuracy, a naive application of learning may not guarantee understandable and customizable policies. To illustrate this point, in Figure 3, we show the result of applying K-means directly on the input

data. Here we cluster the privacy profiles of our users when the requester is a family member, based on the time of day, and day of the week. We set the number of clusters to 4. Each cluster corresponds to a characteristic privacy profile.

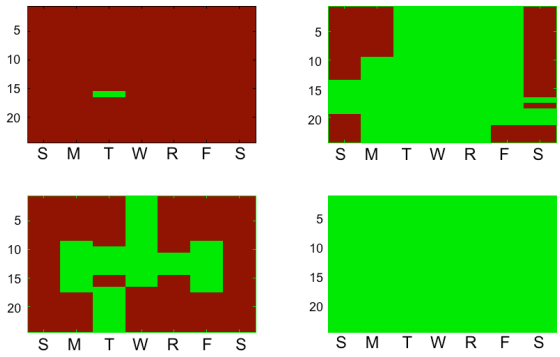


**Fig. 3.** Base case: Non-canonical and unintuitive policy clusters for Family obtained from direct application of clustering (red = *deny*, green = *allow*).

As we can see from Figure 3, using actual times to determine privacy policies yielded rules that were not generalizable and unintuitive. The main reason could be that our user's demographics (students) have schedules that are more event-based (e.g. after work), rather than time based (e.g. after 5:00 PM). In order to capture this, we incorporate a coarser notion of time by dividing a day into coarser intervals of different granularity. We experimented with different intervals of 1, 2, 3, 4, 6, 8, and 12 hours. Every 'activity' is then tagged with the periods during which it occurred. For example, let us consider the case of 3 hour intervals. If a user allowed access from 2:00 PM - 11:30 PM, then we tag that event with three time tags - periods 6, 7 and 8, corresponding to the intervals (3:00 PM - 6:00PM, 6:00 - 9:00 PM, and 9:00 PM to 12:00 AM). We include all the periods where the event occured at least $\gamma$% of the time. We determine $\gamma$ as follows

$$\gamma = \begin{cases} 80 \text{ if } \textit{period is allow}; \\ \frac{80}{\kappa} \text{ if } \textit{period is deny}. \end{cases} \tag{4}$$

where, $\kappa$ is the conservativeness ratio defined earlier. The reasoning behind this is that if a user was very conservative ( say $\kappa = 5$), and denied access during a portion (say around 40% of the time) of a particular period, we would make sure that the classification of the period in the mined rule lowers the expected unhappiness of the user (and classifies it as deny). We chose 80% as the cutoff (Equation 4) after experimenting with different values for the cutoff. Also, as

noted in our data exploration step, we incorporated a coarser notion of day of the week (as weekday or weekend).

Next, we incorporated 'canonical' location information in our study. As mentioned earlier, our location measurement devices (GPS and Wi-fi positioning on the phone) returned location in terms of latitude and longitude observations. We wanted to get a tag for this location, so that we can group locations that have similar context. We used Wikimapia [5] to get this tag information. Wikimapia allows users to tag locations on a google map. Users of Wikimapia mark an area on the map provided, and give a tag for that area such as 'Kiva Han Cafe'. We used this previously tagged information to reverse-geocode the users' latitudes and longitudes into a named location (e.g. Carnegie Mellon, Starbucks etc.). We then tagged these locations as *off-campus, school, on-campus residence, restaurant, mall,* and, *unclassified* depending on the named location. We also noticed different access patterns between people who lived on-campus and those who lived off-campus. Hence, we created different location categories for them. Our University is spread out, and there are a few restaurants very near campus buildings - closer than the resolution of our positioning estimates. In cases where we don't have a clear classification for a particular location, we have two entries in our data - one for campus, and the other for restaurants.

The last attribute we incorporated was the duration of the event. We wanted to see whether the duration of the event would have an impact on the privacy profiles. For instance, a user might want to allow her family to know her location if she was working longer than usual.

As mentioned in the Introduction, the aim of this paper is to present the new user with a choice of default policies learned from existing users. In the first part of this section, we deal with helping users learn a suitable policy using their own audits using a supervised learning approach. We identify a few features, and show how these features can be used to learn an individual privacy policy for the user. In the latter part, we describe how to mine suitable default policies from these individual policies.

### 5.1 Learning Individual User's Policies: An empirical approach to approximating accuracy limits

Along with the fact that we needed to learn policies for new users from their first set of audits, the main reason for this step is also operational — in the data that we collected, we did not collect policies from users in the form of rules. Hence, we need to use this step to determine rules from the audits to determine default policies for our users too.

In order to learn rules from the user's audits, we use a decision tree to generate them based on the attributes that were dicussed earlier. We used the C4.5 decision tree algorithm [19] to mine rules automatically from each of the users' audits separately. We used 10-fold cross validation to ensure that there was no overfitting. We trained various combinations of features (requester type, time in 4 hour intervals, day of the week, tagged location and duration of the event) to understand which of the features were instrumental in reducing the test error. In

addition to lower test error, we prefer simpler rules trading off a slightly higher test error. We achieve this by ensuring that the depth of the tree was small. In all, save 1 of the 30 users, the type of requester was the first feature based on which the tree was branched - thus confirming our previous notion that requester type is the most important feature.

We studied the accuracy of our classifier by considering an exhaustive combination of all the features. We observed that using 'duration' as a feature increases both the error and the complexity of the policies (with $p < 0.05$). So, in the rest of the paper, we only use 'canonical' location, day of the week (weekday / weekend) and a coarse representation of time as the relevant features for our classification.

In Figure 9, we give a sample individual policy learned by the decision tree for a specific requester group. In this case, the learned simple policy specifies that the user allows access to her location to anyone from the university only when she is not at home [1]. Hence, this policy does not depend on the day of the week (w.r.t. weekday / weekend) or the time of the day.

In the next set of results, we represent the error in estimation as accuracy loss. Accuracy loss is the error rate in classification weighted by the conservativeness ratio (Equation 5). For instance, if the policy generated has 10% of the periods wrongly classified as allow, and 5% of the periods wrongly classified as deny, and the conservativeness ratio ($\kappa$) of the user is 2, the accuracy loss is calculated to be 0.25.

$$\text{accuracy loss} = \frac{Period\ wrongly\ classified\ as\ allow}{Total\ period\ classified} \cdot \kappa + \frac{Period\ wrongly\ classified\ as\ deny}{Total\ period\ classified} \tag{5}$$

We first compare the effect of the granularity of the time information. In Figure 5, we plot a histogram of the test error for various intervals of time, arranged by requester type. We have only included a few of the time intervals in this plot. As we see in Figure 5, we got the best accuracy when we used time intervals of 1 hour. This is useful when learning individualized policies, but often leads to very fractured and possibly unintuitive policies when trying to identify default policies for an entire population of users (Figure 4).

Next, we wanted to see the effect of conservative policies. As mentioned before, the conservativeness ratio ($\kappa$) is the ratio between a misclassified allow (i.e. the system decides to deny when the user actually wanted to allow) and a misclassified deny. We plot a histogram of the test error in Figure 6, for various ratios. The ratios are represented as $1{:}\kappa$ in the figure. We see that as a user gets more conservative, the mean error increases (i.e. the distribution spreads out).

[1] Since, the requester cannot differentiate between 'offline' mode and 'deny' mode, having a deny policy for a specific canonical place may not necessarily reveal one's location.

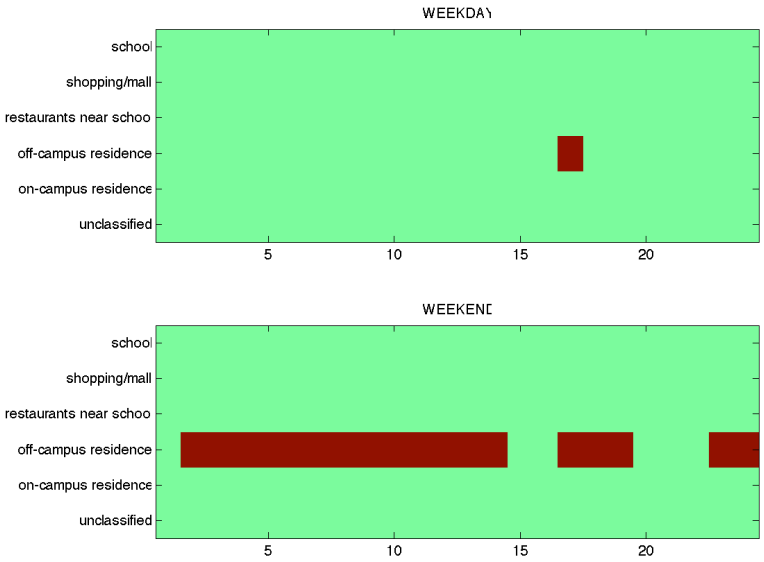


**Fig. 4.** An example of a less intuitive policy that was learned with time period of 1. Area shaded in green is allow, while area shaded in red is deny.

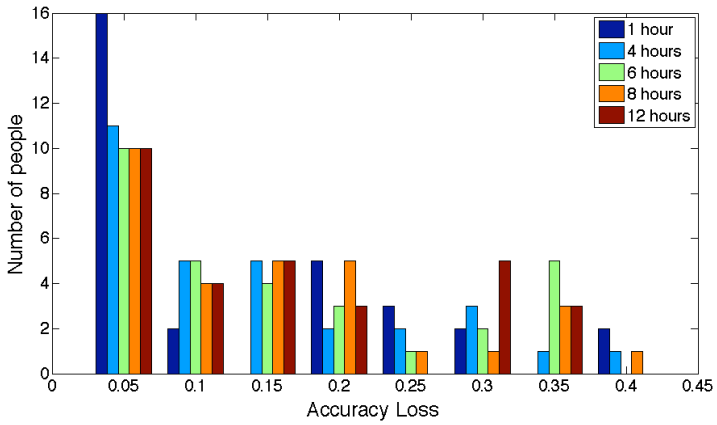


**Fig. 5.** Histogram of accuracy loss for different time granularity for the University group. Here, each instance is a user, and the accuracy loss is calculated across all audits by that user. We see that as granularity increases, error increases too.

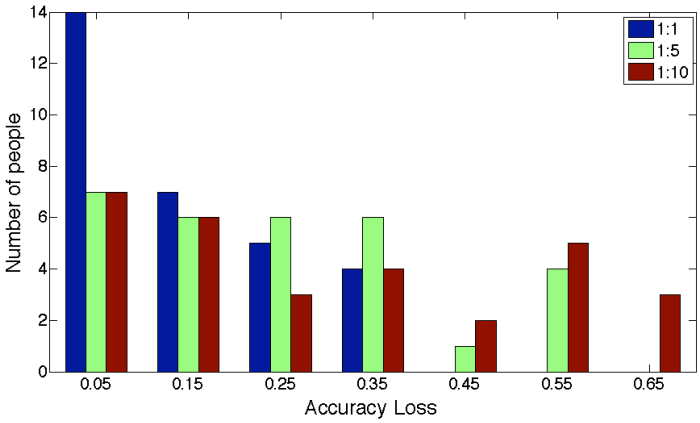


**Fig. 6.** Histogram of accuracy loss for different conservative ratios for people from the University group. Here, each instance is a user, and the accuracy loss is calculated across all audits by that user. We see that error increases for more conservative users.

### 5.2 Identifying Promising Default Policies

At the end of the previous step, we have a set of individual policies. Each policy has a set of rules mined for each user based on their audits. Now, we intend to cluster people into groups such that the ideal policies of the people in each cluster have very little variance, and we can thus ensure that a default policy that is very similar to one person's ideal policy is likely to be very similar to others in the group.

We use the K-means clustering algorithm to cluster users into groups. As input to the clustering algorithm, we use the individual policies - either, learned from the users' decision trees (as shown in the previous section), or taken as input directly from an existing user of the system. More specifically, each of the feature vectors is an individual user's decision tree applied to the entire state-space weighted by the frequency of occurence of the various rules in the users training set. This step is required since the learned tree uniformly weighs all rules as equally likely. Once the users are grouped into various clusters, the final rules can be learned again by concatenating the users' rules into a single list of rules, and using a decision tree on this concatenated list. It may seem intuitive to learn directly from the individual policies, instead of using the users' rules. But, learning directly from the individual policies does not take into account the fact that decision tree outputs uniformly weigh all instances – including instances where the tree has never seen a single training point.

We experimented with different values for $k$ (the number of clusters). The number of clusters corresponds to the number of default policies that the new user gets to choose from. We show a comparison of accuracies achieved by different values of $k$, varying them from 1 to 4 in Figure 7. We see that as the

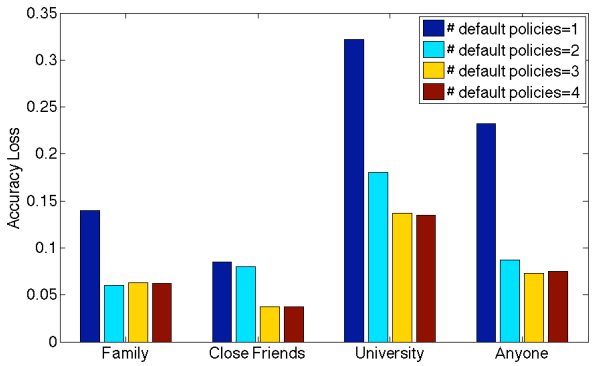


**Fig. 7.** Accuracy loss for different requester groups and for different number of default policies per group when the day is divided into 6 intervals of 4 hours each.

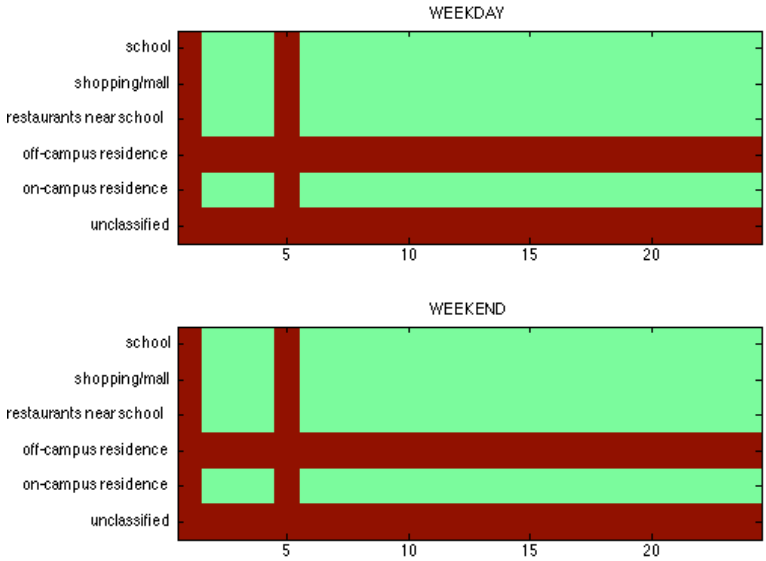


**Fig. 8.** An example of a less intuitive default policy that was learned (One of the four default policies learned for the friends requester type). It's classified as less intuitive since the policy generated is very fractured.

number of default policies increase, the overall classfication error reduces. There is a marked decrease in error when going from 1 default policy to 2 default policies. In contrast, [7] suggests that the users had 6 time rules, 19 location based rules and 24 location / time rules to describe their privacy policy completely, without restricting the rules to be intuitive. While our policies are specified in a slightly different fashion, we see that we are able to get within 90% accuracy with as little as 3 intuitive rules for each requester group type.

Using more clusters leads to some policies that are highly specific to a small set of users, and are less intuitive for a general new user (Figure 8). In constrast, Figure 9 shows an example of an intuitive default policy when the requester is a university friend, and we have three default policies. We use the duration of each rule of the policy (i.e. the size of the contiguous block in the figure) as a simple measure of 'intuitiveness' of the policy. Rules with very short durations are classified as less intuitive. We also see a marginal increase in error, as we increase the time resolution from 1 hour to 6 hours in most cases, although as we saw in Figure 4, the intuitiveness of the resulting policy decreases with 1 hour resolution.

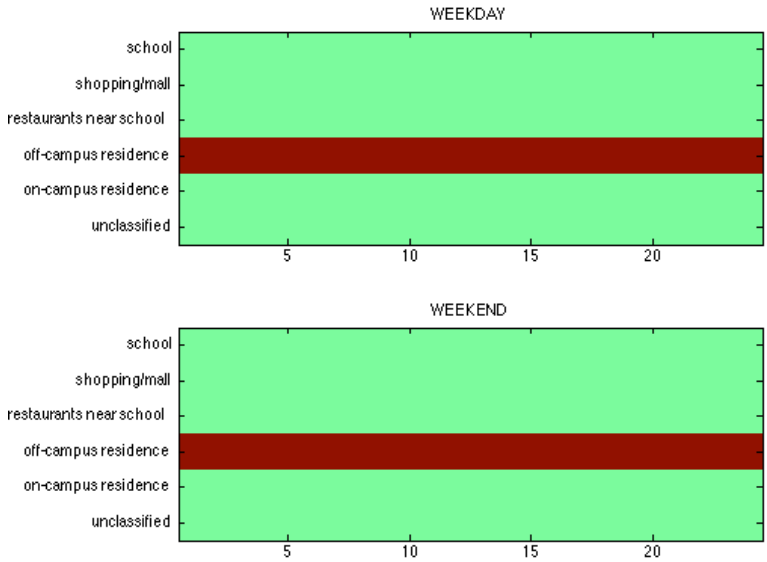


**Fig. 9.** An example of an intuitive default policy that was learned (One of the three default policies learned for the University member requester type).

## 6 Conclusions and Future Work

In this work, we considered the scenario of helping a new user identify suitable default privacy settings. We presented her with a choice of default privacy settings that have been learned from our current set of users' privacy settings

that she can easily understand and modify to specify her desired initial policy. We conducted this study in the context of location-sharing applications, where users specified conditions under which they were willing to let others see their locations based on different contexts.

We demonstrated that deriving default policies through direct application of machine learning techniques did not yield intuitive or usable polices. We then showed that one could abstract away individual elements of a user's schedule and the set of locations they visit, to arrive at what we call canonical policies, that lent themselves to identification of more meaningful and intuitive default policies that users were more likely to be able to customize.

We evaluated the accuracy of default canonical policies for a population of 30 users whose privacy preferences were collected during the course of a week-long study. We learned the users' individual policies using a decision tree algorithm, and clustered the individual policies into a set of more general default policies. We demonstrated the relationship between conservativeness and accuracy, and between time granularity and accuracy. We further discussed tradeoffs between intuitiveness of the default canonical policies and the number of such policies. The main contribution of the work was to show that canonical default polices seem to offer a practical solution where traditional application of machine learning techniques yield unintuitive and unusable policies. Our results further suggest that in the case of location-sharing preferences considered in this study, there was a sweet-spot of around 3 default canonical policies per requester group. We have included the final set of default policies obtained in the Appendix.

We are extending this experiment in Locaccino (`http://www.locaccino.org`), where we are doing a much larger study with many more Facebook users spanning over multiple weeks. In Locaccino, users actively share their locations based on their specified privacy policies. We hope to utilize the more diverse set of users to study the impact of the suggested canonical default policies.

## 7 Acknowledgements

This work is supported by NSF Cyber Trust grant CNS-0627513 and ARO research grant DAAD19-02-1-0389 to Carnegie Mellon Universitys CyLab. Additional support has been provided by Microsoft through the Carnegie Mellon Center for Computational Thinking, FCT through the CMU/Portugal Information and Communication Technologies Institute, and through grants from FranceTelecom and Nokia. Our WiFi-based location tracking functionality runs on top of technology developed by Skyhook Wireless. We would also like to thank the members of the User-Controllable Security and Privacy for Pervasive Computing project at Carnegie Mellon University for their advice and assistance with our study.

## References

1. Duine project `http://www.telin.nl/project/Home.cfm?id=387&language=en`.

2. Loopt `http://www.loopt.com`.
3. Mobikade `http://www.mkade.com`.
4. Mobimii `http://www.mobimii.com`.
5. Wikimapia `http://www.wikimapia.com`.
6. Louise Barkhuus and Anind K. Dey. Location-based services for mobile telephony: a study of users' privacy concerns. In Matthias Rauterberg, Marino Menozzi, and Janet Wesson, editors, *INTERACT*. IOS Press, 2003.
7. Michael Benisch, Patrick Gage Kelley, Norman Sadeh, Tuomas Sandholm, Lorrie Faith Cranor, Paul Hankes Drielsma, and Janice Tsai. The impact of expressiveness on the effectiveness of privacy mechanisms for location sharing. Technical Report CMU-ISR-08-141, Carnegie Mellon University, 2008.
8. JS. Breese, D Heckerman, and C Kadie. In *Empirical Analysis of Predictive Algorithms for Collaborative Filtering*, 1998.
9. Sunny Consolvo, Ian E. Smith, Tara Matthews, Anthony LaMarca, Jason Tabert, and Pauline Powledge. Location disclosure to social relations: why, when, & what people want to share. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 81–90, New York, NY, USA, 2005. ACM.
10. Boi Faltings, Pearl Pu, Marc Torrens, and Paolo Viappiani. Designing example-critiquing interaction. In Jean Vanderdonckt, Nuno Jardim Nunes, and Charles Rich, editors, *Intelligent User Interfaces*, pages 22–29. ACM, 2004.
11. Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 230–237, New York, NY, USA, 1999. ACM.
12. Thomas Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):89–115, 2004.
13. Jason I. Hong and James A. Landay. An architecture for privacy-sensitive ubiquitous computing. In *MobiSys '04: Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pages 177–189, New York, NY, USA, 2004. ACM.
14. Eija Kaasinen. User needs for location-aware mobile services. *Personal Ubiquitous Comput.*, 7(1):70–79, 2003.
15. Fahim Kawsar and Tatsuo Nakajima. Persona: a portable tool for augmenting proactive applications with multimodal personalization support. In *MUM '07: Proceedings of the 6th international conference on Mobile and ubiquitous multimedia*, pages 160–168, New York, NY, USA, 2007. ACM.
16. Patrick Gage Kelley, Paul Hankes Drielsma, Norman M. Sadeh, and Lorrie Faith Cranor. User-controllable learning of security and privacy policies. In Dirk Balfanz and Jessica Staddon, editors, *AISec*, pages 11–18. ACM, 2008.
17. Ashraf Khalil and Kay Connelly. Context-aware telephony: privacy preferences and sharing patterns. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 469–478, New York, NY, USA, 2006. ACM.
18. Scott Lederer, Jennifer Mankoff, and Anind K. Dey. Who wants to know what when? privacy preference determinants in ubiquitous computing. In *CHI '03: CHI '03 extended abstracts on Human factors in computing systems*, pages 724–725, New York, NY, USA, 2003. ACM.
19. Ross J. Quinlan. *C4.5: Programs for Machine Learning (Morgan Kaufmann Series in Machine Learning)*. Morgan Kaufmann, January 1993.

20. Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the ACM 1994 Conference on Computer Supported Cooperative Work*, pages 175–186. ACM Press, 1994.
21. Norman Sadeh, Fabien Gandon, and Oh Buyng Kwon. Ambient intelligence: The mycampus experience. Technical Report CMU-ISRI-05-123, Carnegie Mellon University, 2005.
22. Norman Sadeh, Jason Hong, Lorrie Cranor, Ian Fette, Patrick Kelley, Madhu Prabaker, and Jinghai Rao. Understanding and capturing peoples privacy policies in a mobile social networking application. *Personal and Ubiquitous Computing.*
23. Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted boltzmann machines for collaborative filtering. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 791–798, New York, NY, USA, 2007. ACM.
24. Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms, 2001.
25. Roy Want, Andy Hopper, Veronica Falcao, and Jonathan Gibbons. The active badge location system. *ACM Trans. Inf. Syst.*, 10(1):91–102, 1992.

## Appendix

In Table 1, we present the set of default canonical policies that were mined from our experiment. The policies are presented by type of requester. These results are based on a conservativeness ratio of 1, and the days were divided into 6 time periods — morning, mid-day, afternoon, evening, night and late-night.

**Table 1.** Default policy options arranged by requester types. A user selects a default policy by choosing one default rule for each category of requesters.

| Default Canonical Policies Learned | |
|---|---|
| **Requester Type** | **Default Rules** |
| Family Members | 1. Allow always<br>2. Deny if in an unlabeled location |
| Close Friends | 1. Allow Always<br>2. Deny if in off-campus residence in the mornings<br>3. Deny if at school during late-nights |
| University Colleagues | 1. Allow Always<br>2. Deny on weekends and weeknights<br>3. Deny if at off-campus residence |
| Anyone | 1. Deny Always<br>2. Allow if in school during morning-to-afternoon on weekdays<br>3. Allow if in school |