Privacy-aware Location Privacy Preference Recommendations

Yuchen Zhao
University of St Andrews
yz39@st-andrews.ac.uk

Juan Ye
University of St Andrews
jy31@st-andrews.ac.uk

Tristan Henderson
University of St Andrews
tnhh@st-andrews.ac.uk

ABSTRACT

Location-Based Services have become increasingly popular due to the prevalence of smart devices and location-sharing applications such as Facebook and Foursquare. The protection of people's sensitive location data in such applications is an important requirement. Conventional location privacy protection methods, however, such as manually defining privacy rules or asking users to make decisions each time they enter a new location may be overly complex, intrusive or unwieldy. An alternative is to use machine learning to predict people's privacy preferences and automatically configure settings. Model-based machine learning classifiers may be too computationally complex to be used in real-world applications, or suffer from poor performance when training data are insufficient. In this paper we propose a location-privacy recommender that can provide people with recommendations of appropriate location privacy settings through user-user collaborative filtering. Using a real-world location-sharing dataset, we show that the prediction accuracy of our scheme (73.08%) is similar to the best performance of model-based classifiers (75.30%), and at the same time causes fewer privacy leaks (11.75% vs 12.70%). Our scheme further outperforms model-based classifiers when there are insufficient training data. Since privacy preferences are innately private, we make our recommender privacy-aware by obfuscating people's preferences. Our results show that obfuscation leads to a minimal loss of prediction accuracy (0.76%).

Categories and Subject Descriptors

K.4.1 [**Computers and Society**]: Public Policy Issues—*Privacy*

General Terms

Security

Keywords

location-based services, privacy protection, recommender systems, prediction

1. INTRODUCTION

The popularity of mobile devices such as smartphones and tablets makes computing and services accessible anytime and anywhere. In this ubiquitous computing environment, the availability of getting users' location information prospers an increasing number of Location-Based Services (LBSs), e.g., MyTracks[1], Foursquare[2] and Facebook check-ins.[3] LBSs personalise users' service experience and enable users to share their location information and personal tracks to others; however, they introduce location privacy issues at the same time.

Overexposed location information can lead to many privacy problems. For instance, exposed users' location information could be collected and aggregated with other personal information to profile behaviour. By combining and analysing the semantic information of locations, durations of stay and frequencies, important places such as home and offices can be inferred from the integration of location information. Krumm et al. [15] show that by analysing users' trajectories, their home location can be distinguished within 60 metres. Anonymisation of such mobility patterns may be insufficient; Terrovitis and Mamoulis [22] show that even for anonymous users' trajectories, attackers who have partial knowledge about users can still identify these trajectories with their owners.

Since some places such as hospitals, political institutions and religious places have privacy implications on people who have revealed their location there, revealing these places will cause the disclosure of private facts such as health conditions, political views or religions. Disclosure of private facts is the main privacy issue caused by inappropriate location information dissemination.

Although it is possible to manually configure location sharing rules to prevent inappropriate sharing of data, this may be difficult for users. Users' location privacy preferences have been shown to be context-aware [1], which means they need fine-grained privacy rules to control their location information exposure. Configuring appropriate privacy rules, however, is not an easy task for them. In another study, Sadeh et al. [20] find that users have difficulties to configure their privacy rules, with settings being only 59% accurate at controlling exposure. The need for expressive rules to control location sharing is further demonstrated in a survey by Tsai et al. [24]. It is therefore necessary to help users to articulate and refine their privacy preferences.

To solve the usability problems, machine learning schemes have been proposed that can learn from users' previous privacy decisions and predict their future privacy preferences, thereby reducing the burden on users. But before such learning tools can train their models, it is necessary to collect individual users' privacy decisions

[1]http://www.google.com/mobile/mytracks
[2]http://foursquare.com/
[3]http://www.facebook.com/

in various contexts for a certain amount of time (e.g., several days). Such model-based machine learning classifiers may not perform well when the training data are insufficient (the "cold-start" period). When there are not enough training data about a new user on the predictor, or a user enters a new place where he or she has never been, the prediction accuracy may be low. Meanwhile, users have to reveal their privacy preference data to the prediction provider in order to build the models. These preference data contain not only users' privacy decisions, but also their presence in corresponding contexts (e.g., time, location, companion). Users may be unwilling to share such sensitive data.

Besides predictions based on individual models, predictions based on crowdsourcing, which uses opinions from a group of users, are also used to provide privacy preference recommendations. These crowdsourcing results could be made based on semantic analysis of locations [23], or activities associated with locations [13]. Since different users have different privacy preferences even for the same kind of location, a general recommendation from crowdsourcing based on the semantic analysis of locations may not be personalised enough.

Another way to address the prediction of users' location privacy preferences is *recommendation*, where we can use a recommender as a predictor to provide users with recommendations for location privacy preferences. Rather than predicting from personal decision histories, a recommender predicts a target user's privacy preferences by using the opinions from other users who are similar with the target user in terms of privacy preferences. User-user collaborative filtering (CF) [19], which is a technique of recommender systems, has been used successfully for a long time in many areas including electronic commerce such as Amazon. The assumption behind the user-user CF recommendation is that users have similar preferences on some items (e.g., products, movies, and news) may also have similar preferences on the other items. Thus when predicting the location privacy preferences for a target user, we can use the opinions from users who have similar location privacy preferences with the target user as the recommendation.

In this paper, we attempt to apply user-user CF to provide recommendations of location privacy preferences. Since these recommendations derive from crowdsourcing opinions, the system can also perform well even if there are insufficient data for new users during cold-start periods. Meanwhile, as the recommendation are from users who have similar location privacy preferences to the target user, the recommendations are personalised. Moreover, we attempt to minimise the privacy risks of sharing preferences with a recommender system by obfuscating users' real location privacy preferences with fake ratings without losing too much prediction accuracy. Our contributions are as follows:

- we introduce a recommender based on user-user CF for recommending location privacy preferences for users and evaluate it using real-world data sets;
- we modify the recommender to enhance privacy by obfuscating preferences, and show that this does not sacrifice performance while improving privacy;
- we show that our scheme outperforms the existing state-of-the-art during cold-start periods.

The rest of this paper is structured as follows. In Section 2 we present existing work on predicting users' location privacy preferences and privacy-aware recommender systems. We describe our scheme and how we evaluate its performance in Section 3 and 4. The results of our experiments are presented in Section 5 and we conclude our work in Section 6.

2. RELATED WORK

Users' location privacy has always been an important issue since the introduction of LBSs. Early work [2, 11, 6] focus on using anonymity to protected users' location information. Such methods generate an area which contains the target users' location and other users'. By this means, the LBS provider will be prevented from learning the exact locations of the target user. Apart from protecting users' location privacy from servers, users are also provided with access control mechanisms which enable them to manually choose to whom they want to share location information.

The usability issues, which cannot be solved by the conventional protection methods or manual control, still remain. Users' predefined location privacy rules have a low accuracy to match their decisions when they use LBSs and they also find it difficult to articulate a lot of privacy rules by themselves [20]. To alleviate users' burden of configuring complex privacy rules, machine learning techniques have been introduced by researchers to predict users' privacy preferences thereby helping them refine privacy rules or configuring them automatically. In Online Social Network (OSN) area, Fang and LeFevre propose a privacy wizard [9] to help users configure their privacy rules in online social network automatically. They use an active learning wizard to learn users' privacy preferences by asking them questions about different privacy decisions according to different groups of friends, thereby building a binary classifier based on the preference model. In the area of location privacy protection, Sadeh et al. [20] use machine learning methods (e.g., random forest) to refine users' location privacy rules. Results show that, compared with user-defined rules, the machine learning classifier can improve the accuracy of the rules.

Besides prediction accuracy, another important metric when using predictor to automatically configure users' privacy settings is how often the prediction overexpose of users' location information, which is the *privacy leaks* caused by the prediction to third parties. Bigwood et al. compare performances amongst different machine learning classifiers and predefined settings [3]. They also take into account the privacy leaks. Their results show that besides higher accuracy, predictions by machine learning classifiers can cause fewer privacy leaks than predefined rules. To make fine-grained predictions, Bilogrevic et al. propose a system called SPISM [4] which can semi-automatically predict decisions about location information sharing including the granularity of exposure. These model-based classifiers require a training process and are computationally expensive compared to the user-user CF recommenders that we investigate here.

Apart from using individuals' previous decisions to train models for prediction, another way to predict privacy preferences is to garner recommendations through crowdsourcing. Toch proposes a Super-Ego framework [23] that predicts privacy preferences based on crowdsourced results. Recommendations are made based on semantic categories of locations and the crowdsourced opinions for the same semantic category. They also use a user bias factor, which is a subtraction of the user's average score from the average score of the general population, and use it in a linear combination model to personalise prediction results. Jin et al. [13] propose a system that allows users to get privacy preference recommendations from a server to configure their own location privacy rules. The recommendation is made based on the activities happened in the locations. The results show that recommendations for location privacy preferences from such a system are helpful to users.

The latest study by Xie et al. [25] introduces recommender systems using collaborative filtering into the recommendation of location privacy preferences. They combine the recommendations from both user-user collaborative filtering and item-item collaborative filtering. When using recommender systems, however, the data used by recommenders are highly personalized, which poses privacy threats to users' personal information [21]. A number of research projects protect users' privacy when using recommender system by using homomorphic encryption [5] or obfuscating users' ratings [17, 18]. Polat and Du [18] use randomised response techniques to obfuscate users' binary ratings and the results show that by using this method the recommendations are still accurate.

Compared with these previous studies, we here test our recommender on real world data sets and test the performance of user-user CF during cold-start periods. Moreover, we investigate privacy-aware predictions and analyse the trade-off between the privacy level and performance.

3. APPROACH

3.1 Overview

We propose a location privacy recommender based on user-user CF, which provides users with recommendations for location privacy preferences. When recommending for a target user, the recommender uses opinions from other users who have high user similarities with the target user, thereby keeping the results personalised.

Our scheme contains three stages. First we transform users' privacy preferences to ratings and use them to formulate the privacy rating matrix. Second, when predicting privacy preferences to a target item for a target user u, we find the group of users who have high user similarities with u as the neighbours. Finally, we calculate the recommendation from the neighbours and if the result is higher than the threshold then we take a positive decision as the prediction, and vice versa. In addition, we further modify our scheme in a privacy-aware way.

3.2 Generating rating vectors

To make recommendations for location privacy preferences, first we need to transform users' location privacy preferences to ratings to different contexts. In recommender systems, users' opinions are described as a triple in the form of $(user, item, rating)$, which means that the *user* has given a $rating$ to the $item$. The item could be anything such as movies or books that we want to recommend to users. The ratings could be continuous (e.g., a number between 0 and 1) or discrete (e.g., like and dislike, or star ratings from 1 to 5). In our scheme, in order to apply user-user CF to make location privacy preference recommendations, we use *contexts* (i.e., the combination of location and time) as items and users' *decision*s as ratings. A *decision* can be positive (sharing) or negative (not sharing), and we use 1 as a negative rating and 5 as a positive rating.

To formulate the $item$ in the rating triple, we use the combination of location attributes and the time attributes to describe contexts. We denote the set of location attributes by L and the set of time attributes by T. We can then use the Cartesian product of L and T to represent all the possible combinations of location and time, which are all possible contexts. We use I to represent the set of $item$s by:

$$I = L \times T.$$

A user may have many preferences for the same time and location context, and the *decision* in these preferences may be different. In user-user CF, each user can have only one rating to an item. For each user and each item, we therefore use the most frequent decision for the corresponding context as the rating.

After calculating users' ratings for items which they have given decisions, for each user we have a vector to describe his or her ratings to every item, in the form of $(r_{u,1}, r_{u,2}, \dots, r_{u,|I|})$, where $r_{u,i}$ means the rating of user u to item i. All these rating vectors can be represented as a matrix formed with individual users as rows and specific items as columns, where a value denotes the rating of a user to a specific item.

In Table 1, we give a simple example which has 5 users (from u_1 to u_5), 2 location categories (*home* and $leisure$) and 2 time slots (*morning* and *evening*). Therefore we have 4 contexts as the items for users to rate. Each cell represents a user's location sharing decision, either positive (P) or negative (N), in the corresponding context. Ratings could be blank if users have not been present in the contexts, such as the NULL rating of u_1 for $(leisure, morning)$.

3.3 Calculating user similarity

Before recommending location privacy preferences of item i for user u, we need to find the group of users with similar location privacy preferences with u and aggregate their opinions as the recommendation. This group of users must have rated some items which u has rated and also have rated item i. To calculate the similarity between two users, say u_1 and u_2, we calculate $Sim(\vec{u}_1, \vec{u}_2)$, where $Sim()$ is the similarity function and $\vec{u}_1$ and $\vec{u}_2$ are the rating vectors of u_1 and u_2 respectively. The value range of $Sim(\vec{u}_1, \vec{u}_2)$ depends on the function. In our scheme we use the cosine similarity, which is calculating the cosine of the angle between $\vec{u}_1$ and $\vec{u}_2$ in a $|I|$-dimensional vector space, as the user similarity function and the higher the value is, the more similar u_1 and u_2 are.

Users may have different standards to rate items. For instance, if a user who usually gives low ratings to items gives a high rating to a new item, this rating means more significant than a high rating from another user who always gives high ratings. Normalisation is thus used to convert rating vectors from different users to vectors in the same scale when calculating user similarities. In our scheme, we use the mean centring normalisation [7], which is commonly used, to normalise the user vectors by subtracting the mean rating of a user from each rating in the rating vector.

After calculating similarities, we take the n users with the highest user similarities with u. In recommender systems, the value of n is application specific. In our scheme we use the $n = 8$ which causes the least overexposure in order to take users' location privacy as the most important metric. It also has good performance in terms of prediction accuracy, as we show in Section 5.

Returning to the example described by Table 1, when making recommendation for u_2 about the decision in $(leisure, evening)$, we get the neighbours which contain u_3, u_4 and u_5, because they give the same ratings as u_2 for the first three items. u_1 will not be selected since it has opposite ratings to u_2.

3.4 Predicting privacy preferences

As explained above, the neighbours must have enough similarity with user u and also have ratings for the predicted item. In practice, the calculation of neighbours might fail; for example, u can be a new user without any previous decisions or none of the neighbours having rated item i. Therefore a baseline predictor is necessary when the recommendation from neighbours fails. We use u's mean item rating, which is the mean value $\bar{r_u}$ of u's known ratings, as the user baseline predictor. If u is a new user without any previous ratings, then we use the mean rating of predicting item i, which is the mean value $\bar{r_i}$ of item i's all known ratings from other users, as the item baseline predictor. Under the extreme circumstance, when the item baseline predictor fails, which means no one has rated item i, we use the global mean rating which is the mean value of all other known ratings as the baseline predictor. The prediction of baseline predictors would be decimal. Since we use 5 to represent

User	(home, morning)	(home, evening)	(leisure, morning)	(leisure, evening)
u_1	P	N	NULL	N
u_2	N	P	P	?
u_3	N	P	P	P
u_4	N	P	P	P
u_5	N	P	P	N

Table 1: An example location privacy rating matrix for 5 users and 4 items. Ratings are positive (P), negative (N) or unrated (NULL). The question mark denotes the context in which a prediction needs to be made: (leisure, evening) for u_2.

positive and 1 to represent negative, if the decimal prediction result is greater than 3, we take the prediction as positive and vice versa.

If the neighbours can be generated successfully, we calculate the prediction rating for the item i from the neighbours of u. We draw on the user-based normalised prediction method [7] to make the recommendation. We denote the prediction for user u for item i by:

$$\hat{r}_{u,i} = \bar{r}_u + \frac{\sum_{v \in N_i(u)} w_{u,v}(r_{v,i} - \bar{r}_v)}{\sum_{v \in N_i(u)} |w_{u,v}|}.$$

$N_i(u)$ is the set of user u's neighbours who have rated item i and v indicates any user in these neighbours. $r_{v,i}$ is user v's rating for item i and $(r_{v,i} - \bar{r}_v)$ is the normalised rating by subtracting user v's mean rating $\bar{r}_v$. $w_{u,v}$ is the similarity between user u's and user v's normalised rating vectors.

We use the median value between the negative and positive as the decision threshold θ ($\theta = 3$ in our experiments) and then the recommendation for item i for user u can be represented as:

$$R_{u,i} = \begin{cases} negative & \text{if } \hat{r}_{u,i} \leq \theta \\ positive & \text{if } \hat{r}_{u,i} > \theta \end{cases}$$

In our scheme, when $\hat{r}_{u,i} = \theta$, we take the recommendation as negative. The reason is that this equality means a low confidence of the recommendation. In our scheme we want to decrease the overexposure as much as possible thus we take the negative decision in this instance. In practice, users may have different thresholds for decisions, which means a prediction result might be acceptable to some users to share their location data. But for some privacy-sensitive users, they may have higher θ values. We leave this question about users' acceptance of recommendations for our future research. In this paper we assume all the users have the same threshold.

In the Table 1 example, for user u_2 we have the positive subgroup of neighbours $\{u_3, u_4\}$, whose recommendation for $(leisure, evening)$ is positive, and the negative subgroup of neighbours $\{u_5\}$, whose recommendation is negative. Since all these three users give the same ratings with u_2 for the first three items, their similarities with u_2 are the same. Thus the prediction should be greater than the median value between negative and positive and we take the positive recommendation as the decision.

3.5 Obfuscating rating vectors

When using the location privacy preference recommender, users have to reveal their rating vectors to the recommender. The recommender could be provided by a third party which is independent with LBS service providers thereby avoiding the recommendations being influenced by LBS service providers for their own benefits. We assume that the third party recommender is semi-honest [10], which means it follows the procedures of generating recommendations but also tries to learn more information about users. Since rating vectors not only describe their ratings to different contexts, but also mean they have been to the location at a specific time, the recommender can learn users' location information from their rating vectors. When making recommendations based on fine-grained contexts, some contexts may contain sensitive information. For instance, when users require recommendations for location release decisions for a new context, they have to send their rating vectors which may contain previous ratings for sensitive contexts such as (club, working time) or (friend's home, night), which should not be known by the recommender. Since all users' rating vectors will be stored for generating recommendation and it is reasonable to assume that users are matched with their rating vectors, the recommender can record a specific user's location history. Furthermore the recommender will become a target for attackers who intend to steal users' location information.

In recommender systems, the sensitivity of items is a privacy issue. Ideally we hope a recommender should provide users with recommendations without knowing what the items are. Homomorphic encryption [5] has been introduced to achieve this but it is too computationally expensive for large data sets in the real world due to proof of user data validity and the need of a trusted third party. An alternative approach is to add noise information which follow a certain distribution into users' original data. Thus the privacy of individuals will be protected and the recommendation accuracy can also be guaranteed because of the certain distribution of the noise.

To enable users to use the recommender in a privacy-aware way, which means the recommender cannot easily know whether a rating is a real rating of a user, we obfuscate users' rating vectors by introducing fake ratings. The method is based on the work of Polat and Du [18]. A recommendation with full privacy means the recommender cannot learn if a rating for an item is real. To achieve this, for each user, we denote the number of rated items by m_t and a noise factor by α. This provides an upper limit of the number of fake ratings denoted by $m_{max} = \alpha m_t$. Then we randomly create an integer m_f between 0 and m_{max} and $m_f \sim U[0, m_{max}]$. Finally we randomly add m_f fake ratings in blank positions of the rating vector and $m_f/2$ of them are positive ratings and the rest of them are negative ratings. Since the fake ratings are randomly generated, their influence on users' similarity when calculating on plenty of users should not be significant.

4. EVALUATION

We evaluate our scheme to answer three questions:

1. can our recommender perform as well as model-based machine learning classifiers with sufficient training data?
2. can the privacy-aware recommender increase the privacy level of recommendations without too much loss of performance?
3. can our recommender improve the prediction performance compared with the other two schemes during the cold-start period?

4.1 Datasets

		Predicted Decision	
		positive	**negative**
Actual Decision	**positive**	True Positive (TP)	False Negative (FN)
	negative	False Positive (FP)	True Negative (TN)

Figure 1: Confusion matrix of actual *decision* and predicted *decision*.

We use the LocShare dataset from the CRAWDAD wireless network data archive [16] , which contains real-world user location privacy preference data. These data were collected both in London and St Andrews and have been sanitised to protect the privacy of participants.

Because of the sparsity of samples in London, we only use the data from St Andrews. We use users' identifications ($N = 40$), categories of location, time when they made location privacy decisions and the decisions. For the set of location categories, we have:

$$L = \{Food\ and\ Drink, Leisure, Retail, Residential, Academic, Library\}$$

and for the set of time slots, we have:

$$T = \{Morning, Noon, Afternoon, Evening, Night\}.$$

Thus the *item* set I should be in the form of:

$$I = L \times T = \{(Food, Morning), (Food, Noon), \ldots, (Library, Night)\}$$

and cardinality of I (the number of *items*) is:

$$|I| = |L| \cdot |T| = 30.$$

4.2 Metrics

When recommending location privacy preferences, we divide the data set into training sets and testing sets. A training set is used to generate the rating matrix and a testing set is used to be compared with the recommendations. For each location preference in the test set, we assume all the *decision*s have not been given. Then we use the recommendation model generated from the training set to make recommendations for the *user* and *item*.

To compare the actual *decision* in test sets and the predicted *decision* (recommendation) made from the recommender, we use a confusion matrix as shown in Figure 1 to represent the possible results.

The first metric is *accuracy* – how many predictions among all predictions are successful; that is, the recommendations match users' actual decisions. According to the confusion matrix, we define the prediction accuracy as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Another metric is the number of privacy *leak*s – how many predictions cause the overexposure of users' location information. When using recommendations as predictors, there are two types of failures (i.e., FN and FP). The FN will conceal users' location information when they actually want to share and the FP will reveal users' location information when they actually want to conceal. In terms of privacy, the FP is more harmful than FN because the former will accidentally cause location privacy leaks. Therefore when comparing different schemes which have similar prediction accuracies, the one with fewer privacy leaks is more reliable. We define privacy leaks as:

$$Leak = \frac{FP}{TP + TN + FP + FN}$$

We implement the recommender in a privacy-aware way by adding fake ratings, so the performance of recommendations (both *Accuracy* and *Leak*) will be affected. The aim is to provide an acceptable trade-off between the utility and the privacy level of the recommender. Thus we take the loss of *Accuracy* and the increase of *Leak* caused by fake ratings into account. We also consider the privacy level of the recommender, which is the probability of preventing the recommender from observing users' real ratings.

Note the *Leak* is different from the privacy level of the recommender. The *Leak* is the failure caused by the recommender when it overexposes location information to third parties. The privacy level of a recommender is the probability of hiding real ratings (either the truth that a user has rated an item, or the real value of a rating) from the recommender.

4.3 Methodology

We compare our scheme with existing model-based machine learning classifiers and crowdsourcing semantic predictions. The former builds individual models from users' decision histories and predicts based on the models it learns. The latter uses the crowdsourcing opinions from the same semantic category of locations.

First, we test the performance of the recommender with sufficient training data. We assume that the system has run for a period of time and every user has provided enough privacy preferences. We use the classifiers (Naïve Bayes, J48 and Rotation Forest) from [3] and the crowdsourcing semantic predictors from [23] in our test as the benchmark. We set the neighbourhood size ($n = 8$) that can provide the fewest *Leak* in our experiment. In the comparison among our scheme and other methods, we consider both *Accuracy* and *Leak*. We run 100 rounds of experiments and in each round we use different random seeds to generate 10 subsets. Then we use 10-fold cross-validation to evaluate the performance.

Second, we test the performance of the privacy-aware recommender by adding fake ratings into the training sets. We test the influence of different noise factors α on the performance of our recommender and for each α value we also run 100 rounds of 10-fold cross-validation tests. We use the performance of the recommender without noise as the benchmark and with this we can measure the loss of *Accuracy* and the increase of *Leak*.

Finally, we test the recommender with insufficient training data, to evaluate performance in the cold-start period. We assume the system has run for a period of time, which means users have already given enough ratings, and then a new user with little previous information uses the recommender. In this scenario, the prediction may be inaccurate for this new user because of the lack of individual information. We compare the performance of our recommender with the best performance of model-based classifiers and the crowdsourcing semantic prediction. For the training sets, every time we select one user as the new user and remove all the data of this user from the training sets. Then we increasingly add the data of the new user by small fractions into the training sets to train the recommender and classifiers. We use the rest of the data of the new user as the testing sets. For each user we run 100 rounds and in each round we use different random seeds to generate the small fractions of data added into training sets. To simulate the cold-start period, we start with 1% of the new user's data and increase it with 1% until 10%. Parameters of our experiments are shown in Table 2.

We use the model-based machine learning classifiers from the Weka package [12] to implement and the Lenskit recommender toolkit API [8] to implement the recommenders in our experiments.

5. RESULTS

The results of our experiments meet our expectations. First, user-user CF recommenders can perform as well as model-based machine learning classifiers do and better than crowdsourcing semantic prediction does. Second, the privacy-aware recommender will not sacrifice too much performance to increase the privacy level. Third, the user-user CF recommender can improve the performance during cold-start period.

We also tested a recommender in our experiments based on Matrix Factorization [14] (MF), which is known as a more advanced scheme. It uses gradient descent to learn a matrix factorization and we use 200 counts of iterations as the stopping condition. We test the number of latent features from 1 to 20 and our results show that the number of latent features which can provide the best performance is 1. The overall performance and the performance during cold-start period of MF scheme, however, are worse than the user-user CF in our experiments.

5.1 Overall performance

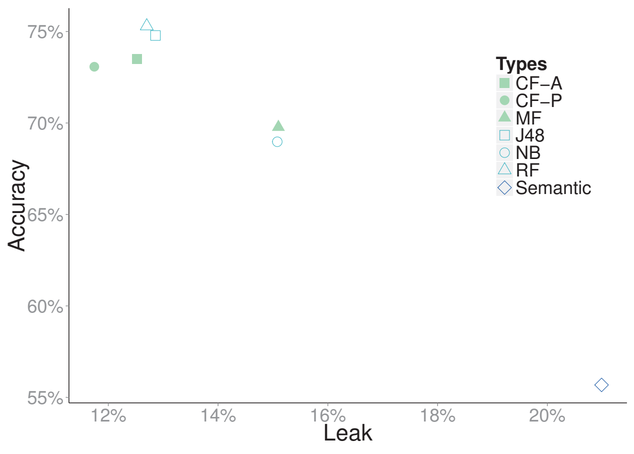

Figure 2: ***Accuracy*** **and** ***Leak*** **of CF (CF-A has the highest** ***Accuracy*** **and CF-P has the lowest** ***Leak*** **), MF, model-based machine learning classifiers (J48, Naïve Bayes, Rotation Forest) and crowdsourcing semantic predictions. The CF recommender outperforms crowdsourcing semantic prediction and MF in terms of both** ***Accuracy*** **and** ***Leak*****. The** ***Accuracy*** **of using CF is close to the best performance of model-based machine learning classifiers and it causes fewer** ***Leak*****.**

When comparing the performances of the user-user CF recommender, MF, model-based classifiers and crowdsourcing semantic prediction, we consider both *Accuracy* and *Leak*. As shown in Figure 2, the x-axis represents the *Leak* and the y-axis represents the *Accuracy*. We hope the predictor can be as accurate as possible and meanwhile cause less overexposure, which means a good predictor is supposed to be on the left-top of the plot.

The performance of crowdsourcing semantic prediction (Semantic) is on the right-bottom. Its *Accuracy* is 55.68%, which approximates the performance of random guess. It has the most *Leak* among all the schemes. The reason of the poor performance of crowdsourcing semantic prediction may be the fact that different users have different privacy preferences even for the same context. A general prediction from crowdsourcing for a semantic category is not personalised.

Among the machine learning classifiers in our experiments, the Rotation Forest (RF) performs the best both in terms of *Accuracy* and *Leak*. The performance of J48 decision tree is close to that of RF. The performance of Naïve Bayes is worse than RF and J48 but is still better than Semantic. The good performance of the model-based machine learning technique is expected since in our 10-fold cross-validation experiment the classifiers can use 90% of the user data to train models. These models could describe users' location privacy preferences comprehensively and are sufficient for the testing data.

Our privacy-aware user-user CF recommender (CF-P, $n = 8$) has the fewest *Leak* (11.75%) compared with all its counterparts, which means it causes the least overexposure of users' location information. And the *Accuracy* of our recommender (73.08%) is close to the RF (75.30%), which is the best performance of the model-based classifiers in our experiments. We also show the highest *Accuracy* (CF-A, $n = 22$) that our recommender can achieve when using another size of neighbourhood. The increase of the *Accuracy* is not significant (0.43%) however it causes more *Leak* (increased by 0.78%). The MF has a lower *Accuracy* (69.77%) and causes more *Leak* (15.10%) than CF does.We use two-sample paired t-test as the significance test for RF and CF, RF and MF, Semantic and CF, Semantic and MF, and CF and MF. The results show that $p < 0.001$ for all pairs.

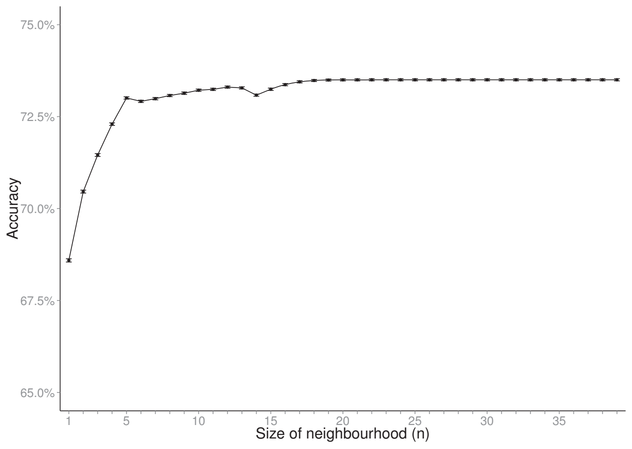

Figure 3: The change of ***Accuracy*** **with the increase of neighbourhood size (*n*).** ***Accuracy*** **increases until** ***n*** **reaches 5 and changes slightly until** ***n*** **reaches around 18. Then it keeps stable.**

As shown in Figure 3 and Figure 4, the performance of the recommender becomes better with the increase of the neighbourhood size until it reaches a certain value. After that point, the performance keeps stable without significant change. The reason is that when the neighbourhood size is small at the beginning, the difference between individual users will influence the recommendation. As the neighbourhood size increases and more users are selected, the influence of individual neighbour users will be reduced because of averaging. If we keep increasing the neighbourhood size, users with low similarities will be introduced but their contributions to the group weight (group similarity) are not significant. Thus the performance of the recommender will not change too much.

Overall performance test	Rounds of experiments: 100
	Per round: 10-fold cross-validation
Privacy-aware recommender	Rounds of experiments: 100
	Per round: 10-fold cross-validation
	α from 1.0 to 20.0 increases by 0.5
Cold-start test	Rounds of experiments: 100
	Per round: Iterate each user as the new user
	Per user: Add personal data from 1% to 10% increase by 1% to training sets

Table 2: Parameters of three experiments

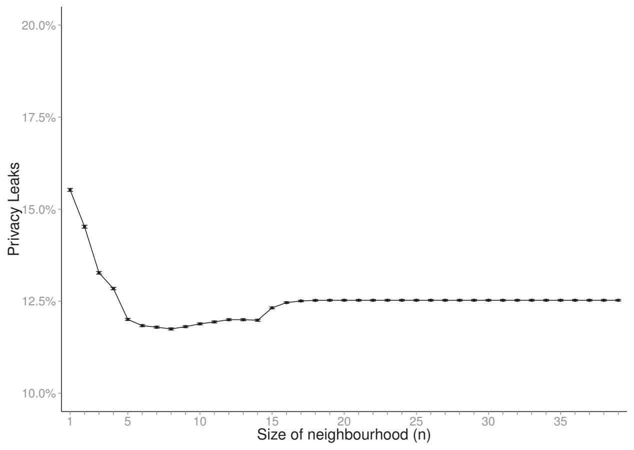

Figure 4: The change of *Leak* with the increase of neighbourhood size (*n*). *Leak* decreases until *n* reaches 5 and changes slightly until *n* reaches around 16. Then it keeps stable.

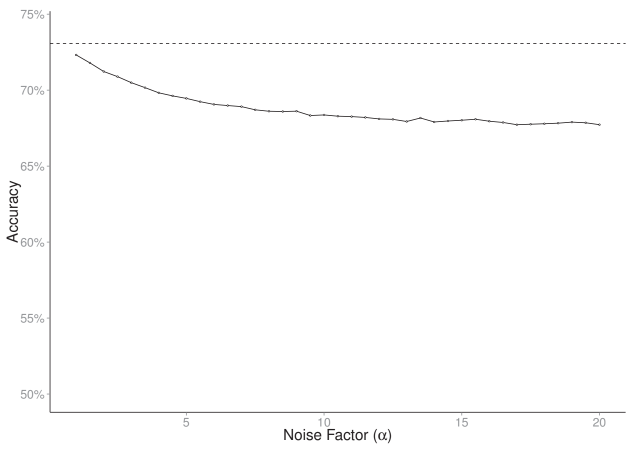

Figure 5: *Accuracy* of the privacy-aware recommender with different noise factor (α). The dashed line represents the *Accuracy* of the recommender without fake ratings. The loss of *Accuracy* is minimal (0.76%, $\alpha = 1$) when α is small. It increases with the growth of α and reaches 5.35% when $\alpha = 20$.

The results suggest that our proposed recommender has a close overall performance to the best performance of model-based machine learning classifiers and significantly outperforms crowdsourcing semantic predictions. Meanwhile our recommender causes less overexposure of location information, which makes a more reliable predictor. The MF scheme, as a more advanced scheme, did not achieve better performance than user-user CF in our experiments. Since the number of latent features with the best performance on our data set is 1, we think that our data set is too small for MF to find latent features, which leads to its poor performance.

5.2 Privacy-aware recommendations

By adding fake ratings into users' rating vectors, we prevent the recommender from learning users' real ratings. The recommender can only guess whether an item is really rated by a user with a certain probability. The more fake ratings we add, the lower the probability of the exposure of a user's real rating will be. Meanwhile the performance of the recommender will be influenced by the added noise. We hope the influence could be controlled within a certain range.

As shown in Figure 5, the x-axis represents the noise factor α and the y-axis represents the *Accuracy* of the privacy-aware recommender. The horizontal dashed line represents the *Accuracy* of the recommender without fake ratings. We take the value of α from 1.0 to 20.0. As the α increases, the loss of *Accuracy* increases (from 0.76% to 5.35%). To evaluate the privacy level of the privacy-aware recommender, we use the percentage of expected value of fake ratings among all the ratings ($\frac{E(m_f)}{E(m_f)+m_t}$). Since $m_f \sim U[0, m_{max}]$, $E(m_f) = \frac{m_{max}}{2} = \frac{\alpha m_t}{2}$. Therefore we have the privacy level (PL) $\frac{\alpha}{\alpha+2}$. When $\alpha = 0$, which is the recommender without fake ratings, the PL is 0, which means the recommender knows that all the ratings in the vector are real and the user has been in the corresponding contexts. When $\alpha = 1.0$, the PL is 33.33%, which is at the expense of only 0.76% loss of *Accuracy*. Figure 6 shows the increase of *Leak* caused by adding fake ratings (from 0.86% to 2.21%). When $\alpha = 20$, the theoretical expected fake ratings should be ten times as true ratings. The length of a rating vector, however, is fixed ($|I|$=30) and in our data set the average number of rated items per user is 7. Therefore the actual number of fake ratings sometimes could not be its theoretical expected value and that is the reason why the influence of fake ratings tends to be stable as α increases. It also means that the increase of attack expense is more significant when α is small, due to the fixed length of a vector. Here we only analyse attacker's difficulty of detecting fake ratings in a single vector. Integrating individual vectors and detecting which items contribute more than others in the recommendation might be a more powerful attack method to detect fake ratings and apparently it needs more computational expenses. We leave this topic for future research.

The results show that by randomly adding fake ratings which follows uniform distribution, the privacy-aware recommender can provide higher privacy level at the expense of minimal loss of performance. When the number of users is large enough, the contribution of fake ratings to users' similarities will become very small due to their randomness. By this means, individuals' rating vectors are obfuscated and their privacy can be protected.

5.3 Cold-start test

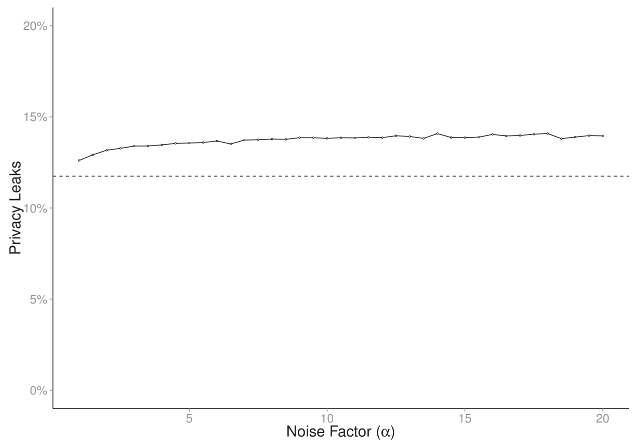

Figure 6: ***Leak*** **of the privacy-aware recommender with different noise factor (α). The dashed line represents the** ***Leak*** **of the recommender without fake ratings. The increase of** ***Leak*** **is minimal (0.86%, α = 1) when α is small and reaches 2.21% when α = 20. Its change with the growth of α is not significant.**

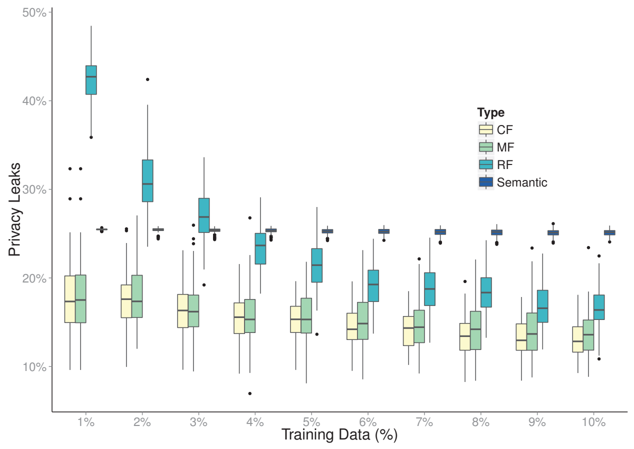

Figure 8: ***Leak*** **of CF, MF, RF and crowdsourcing semantic prediction during the cold-start period. The CF recommender causes fewer** ***Leak*** **than RF and crowdsourcing semantic prediction during the cold-start period. MF performs slightly worse than CF does.**

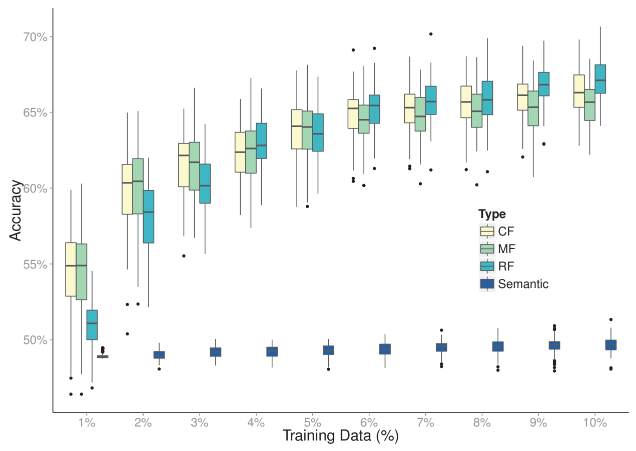

Figure 7: ***Accuracy*** **of CF, MF, RF and crowdsourcing semantic prediction during the cold-start period. CF recommender can provide higher** ***Accuracy*** **than RF (except 4%) until using 6% of personal data for training. The** ***Accuracy*** **of using CF is higher than using crowdsourcing semantic prediction. MF performs slightly worse than CF does.**

We compare the performance of CF, MF, RF and Semantic schemes during the cold-start period. As shown in Figure 7, the x-axis represents the percentage of the new user's data added into the training sets and the y-axis represents *Accuracy*. From 1% to 5%, CF has higher *Accuracy*s than RF does, except 4%. After 5%, the RF has enough data of the new user to train the personal model and the *Accuracy* surpasses CF's. The Semantic scheme has the lowest *Accuracy* during the whole period. The addition of the new user's data has little impact on the crowdsourcing opinions. MF performs slightly worse than CF does.

Figure 8 shows the comparison among the *Leak* of four schemes. The CF has the fewest leaks during the whole period. Compared with RF, the *Leak* of CF is much fewer than the RF's at the beginning of this period. Due to the low influence of the new user's data on crowdsourcing opinions, the performance of the Semantic scheme does not change too much and its *Leak* are more than the CF's. MF performs slightly worse than CF does.

The results show that our proposed scheme can improve the performance of prediction during the cold-start period. This means when a new comer uses the recommender, the prediction quality can reach an acceptable level in a shorter period of time. It also suggests that our scheme can significantly decrease the over-exposure of location information during the cold-start period, especially at the beginning. This means that applying user-user CF recommender to predicting users' location privacy preferences is more reliable than using model-based machine learning classifiers or crowdsourcing semantic prediction. In future work we will consider the design of a hybrid predictor which has the benefits of both user-user CF and model-based machine learning classifiers.

6. CONCLUSIONS

In this paper, we have proposed a location privacy recommender by using user-user CF. We test this technique on real-world data sets and compare its performance with other schemes including, matirx factorization, model-based machine learning classifiers and crowdsourcing semantic prediction. The results suggest that our scheme outperforms crowdsourcing semantic prediction, both in prediction accuracy and privacy leaks. With providing fewer privacy leaks, the prediction accuracy of user-user CF is close to the RF method, which is the best performance of model-based machine learning classifiers. During the cold-start period, our scheme shows better performance both in prediction accuracy and privacy leaks than the RF scheme does.

We also implement a privacy-aware recommender which enables users to obfuscate their rating vectors by adding fake ratings. The results show that the privacy-aware recommender can increase the privacy level of recommendations at minimal expense of loss of performance. Due to the randomness of the added noise and their distribution, this expense will become negligible when the number of users is high enough.

The results of our experiments suggest that besides using model-based machine learning classifiers to predict users' privacy preferences based on their previous decisions, recommendations from people who have similar privacy decisions can also match users' future decisions and can keep a low possibility of privacy leaks. Since recommender systems based on CF have been widely used in practice and it is less computationally expensive than model-based systems, it will be more practical to be used to predict privacy preferences in real-world applications. In addition, the privacy-aware feature makes it reliable.

Our future work is to better understand common users' acceptance of the recommender, we plan to conduct user studies to investigate under what circumstances they trust the recommendations from social choices. We are also interested in the question of whether the form of recommendations (e.g., recommendations only, recommendations with confidence or reasons) has an impact on users' decisions. The expected results will help us make the recommender more informative for users.

Another direction of future work is to investigate fine-grained ratings of location privacy preferences. In this paper, due to the limitation of data sets, we only use two kinds of decisions for sharing or not sharing. Finer-grained scales (e.g., ratings from 1 to 5 with 1 as the interval) might describe users' preferences more precisely.

7. REFERENCES

[1] D. Anthony, T. Henderson, and D. Kotz. Privacy in Location-Aware Computing Environments. *IEEE Pervasive Computing*, 6(4):64–72, Oct. 2007. doi:10.1109/MPRV.2007.83.

[2] A. Beresford and F. Stajano. Location privacy in pervasive computing. *IEEE Pervasive Computing*, 2(1):46–55, Jan. 2003. doi:10.1109/MPRV.2003.1186725.

[3] G. Bigwood, F. Ben Abdesslem, and T. Henderson. Predicting location-sharing privacy preferences in social network applications. In *Proc. of AwareCast*, Newcastle, UK, 2012. Online at http://www.ibr.cs.tu-bs.de/dus/Awarecast/awarecast2012_submission_1.pdf.

[4] I. Bilogrevic, K. Huguenin, B. Agir, M. Jadliwala, and J.-P. Hubaux. Adaptive information-sharing for privacy-aware mobile social networks. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing - UbiComp '13*, pages 657–666, New York, NY, USA, 2013. ACM Press. doi:10.1145/2493432.2493510.

[5] J. Canny. Collaborative filtering with privacy. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pages 45–57. IEEE Comput. Soc, 2002. doi:10.1109/SECPRI.2002.1004361.

[6] C.-Y. Chow, M. F. Mokbel, and X. Liu. A peer-to-peer spatial cloaking algorithm for anonymous location-based service. In *Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems - GIS '06*, pages 171–178, New York, NY, USA, 2006. ACM Press. doi:10.1145/1183471.1183500.

[7] C. Desrosiers and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook*, pages 107–144. Springer, 2011. doi:10.1007/978-0-387-85820-3_4.

[8] M. D. Ekstrand, M. Ludwig, J. A. Konstan, and J. T. Riedl. Rethinking the recommender research ecosystem. In *Proceedings of the fifth ACM conference on Recommender systems - RecSys '11*, pages 133–140, New York, NY, USA, 2011. ACM Press. doi:10.1145/2043932.2043958.

[9] L. Fang and K. LeFevre. Privacy wizards for social networking sites. In *Proceedings of the 19th international conference on World wide web - WWW '10*, pages 351–360, New York, NY, USA, 2010. ACM Press. doi:10.1145/1772690.1772727.

[10] O. Goldreich. *Foundations of Cryptography: Volume 2, Basic Applications*, volume 2. Cambridge University Press, 2009.

[11] M. Gruteser and D. Grunwald. Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In *Proceedings of the 1st international conference on Mobile systems, applications and services - MobiSys '03*, pages 31–42, New York, NY, USA, 2003. ACM Press. doi:10.1145/1066116.1189037.

[12] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009. doi:10.1145/1656274.1656278.

[13] H. Jin, G. Saldamli, R. Chow, and B. P. Knijnenburg. Recommendations-based location privacy control. In *2013 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, pages 401–404. IEEE, Mar. 2013. doi:10.1109/PerComW.2013.6529526.

[14] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009. doi:10.1109/MC.2009.263.

[15] J. Krumm. Inference attacks on location tracks. *Pervasive Computing*, 62:127–143, 2007. doi:10.1007/978-3-540-72037-9_8.

[16] I. Parris and F. Ben Abdesslem. CRAWDAD data set st_andrews/locshare (v. 2011-10-12). Downloaded from http://crawdad.org/st_andrews/locshare/, Oct. 2011.

[17] H. Polat and W. Du. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *ICDM '03 Proceedings of the Third IEEE International Conference on Data Mining*, pages 625–628, 2003. doi:10.1109/ICDM.2003.1250993.

[18] H. Polat and W. Du. Achieving private recommendations using randomized response techniques. *Advances in Knowledge Discovery and Data Mining*, 3918:637–646, 2006. doi:10.1007/11731139_73.

[19] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: an open architecture for collaborative filtering of netnews. In *Proc. of CSCW*, pages 175–186, 1994. doi:10.1145/192844.192905.

[20] N. Sadeh, J. Hong, L. Cranor, I. Fette, P. Kelley, M. Prabaker, and J. Rao. Understanding and capturing people's privacy policies in a mobile social networking application. *Personal and Ubiquitous Computing*, 13(6):401–412, Oct. 2008. doi:10.1007/s00779-008-0214-3.

[21] K. Shyong, D. Frankowski, J. Riedl, et al. Do you trust your recommendations? an exploration of security and privacy issues in recommender systems. In *Emerging Trends in Information and Communication Security*, pages 14–29. Springer, 2006.

[22] M. Terrovitis and N. Mamoulis. Privacy Preservation in the Publication of Trajectories. In *The Ninth International Conference on Mobile Data Management (mdm 2008)*, pages 65–72. IEEE, Apr. 2008. doi:10.1109/MDM.2008.29.

[23] E. Toch. Crowdsourcing privacy preferences in context-aware applications. *Personal and Ubiquitous Computing*, 18(1):129–141, 2014. doi:10.1007/s00779-012-0632-0.

[24] J. Tsai, P. Kelley, L. Cranor, and N. Sadeh. Location-sharing technologies: Privacy risks and controls. In *Research conference on communication, information and internet policy (TPRC)*, 2009. Online at http://ssrn.com/abstract=1997782.

[25] J. Xie, B. P. Knijnenburg, and H. Jin. Location sharing privacy preference: analysis and personalized recommendation. In *Proc. of IUI*, pages 189–198, 2014. doi:10.1145/2557500.2557504.