Towards Publishing Recommendation Data With Predictive Anonymization

Chih-Cheng Chang
Rutgers University
Department of Computer Science
Piscataway, NJ, USA
geniusc@cs.rutgers.edu

Brian Thompson
Rutgers University
Department of Computer Science
Piscataway, NJ, USA
bthom@cs.rutgers.edu

Hui (Wendy) Wang
Stevens Institute of Technology
Department of Computer Science
Hoboken, NJ, USA
hwang@cs.stevens.edu

Danfeng Yao
Rutgers University
Department of Computer Science
Piscataway, NJ, USA
danfeng@cs.rutgers.edu

ABSTRACT

Recommender systems are used to predict user preferences for products or services. In order to seek better prediction techniques, data owners of recommender systems such as Netflix sometimes make their customers' reviews available to the public, which raises serious privacy concerns. With only a small amount of knowledge about individuals and their ratings to some items in a recommender system, an adversary may easily identify the users and breach their privacy. Unfortunately, most of the existing privacy models (e.g., k-anonymity) cannot be directly applied to recommender systems.

In this paper, we study the problem of privacy-preserving publishing of recommendation datasets. We represent recommendation data as a bipartite graph, and identify several attacks that can re-identify users and determine their item ratings. To deal with these attacks, we first give formal privacy definitions for recommendation data, and then develop a robust and efficient anonymization algorithm, Predictive Anonymization, to achieve our privacy goals. Our experimental results show that Predictive Anonymization can prevent the attacks with very little impact to prediction accuracy.

Categories and Subject Descriptors

G.1.3 [**Mathematics of Computing**]: Numerical Linear Algebra—*Sparse, structured, and very large systems*; H.2.8 [**Database Management**]: Database Applications—*Data mining*; K.4.1 [**Computers and Society**]: Public Policy Issues—*Privacy*

General Terms

Algorithms, Experimentation, Performance, Security

Keywords

Anonymization, Sparsity, Prediction, Clustering, Privacy

1. INTRODUCTION

To help consumers make intelligent buying decisions, many websites provide *recommender systems* [29] that give users suggestions of items of potential interest to them. The recommender systems collect users' input (e.g., reviews, ratings, etc.), compare the collected data to others, and calculate a personalized list of recommended items for the user. It has been proven to be effective at delivering users more intelligent and proactive recommendations [32].

To support advanced data mining and prediction algorithms, data owners sometimes publicly release their recommendation datasets. The released datasets may include information that is legally protected, or otherwise private or sensitive data, such as buying records and movie-viewing histories. For example, in 2006, Netflix, the world's largest online DVD rental service, announced a one million-dollar *Netflix Prize* for improving their movie recommendation algorithm.[1] To aid contestants, Netflix released a Netflix Prize dataset containing more than 100 million movie ratings, created by around 500 thousand Netflix subscribers between December 1999 and December 2005. As the dataset contains users' private preferences to the movies, Netflix removed customers' names to protect their privacy. However, this naively anonymized data suffers from re-identification attacks as recently demonstrated [26].

1.1 Motivation Examples

With some additional knowledge about a user's review history, an adversary may be able to uniquely identify and consequently learn additional information about the user. For example, suppose the adversary knows her co-worker Alice watched *Pretty in Pink*, a movie from the eighties which has not been reviewed by many people. By matching with

[1]The prize was claimed in September 2009. http://www.netflixprize.com

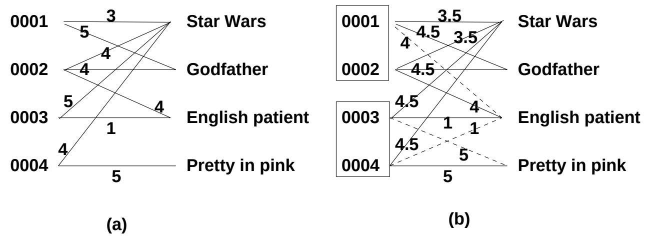


Figure 1: An example: (a) The original graph with user names replaced with IDs, (b) The k-anonymized graph where $k = 2$. The rectangle boxes represent anonymization groups. Dotted lines represent fake edges added for the purpose of anonymization. New ratings are computed as the average of existing ratings in the anonymization group.

the released dataset in Figure 1 (a), the adversary can identify that ID 0004 corresponds to Alice. Consequently the adversary learns all movies that Alice has reviewed and her preferences to these movies.

Online shopping is very popular on the Web today, and costumers of E-commerce sites such as Amazon.com or E-bay.com are under the same privacy risk. For example, an adversary knows his neighbor Bob recently purchased a not-so-popular cell phone from Amazon. By searching the phone model, the adversary can learn other items that Bob has bought through the collaborative filtering features provided by Amazon (e.g., what do customers ultimately buy after viewing this item, customers who bought this item also bought, etc.). Furthermore, if Amazon decides to publish their recommendation data for research or other purposes, insufficient anonymization could leave users' personal information vulnerable to attack. Similar issues arise with content sharing website Digg.com, which recently implemented collaborative filtering to recommend new articles or web content based on users' viewing and rating histories. Knowledge of a user's history could reveal sensitive information, e.g. a user's sexual orientation or political affiliation.

Such privacy risks have been studied by Narayanan et. al. who show that very little knowledge about an individual subscriber is necessary to identify her record if it is present in the Netflix dataset [26]. For instance, 84% of subscribers can be uniquely identified if the adversary knows 6 out of 8 movies outside the top 500 most popular movies. The ease of attack is largely due to the *sparsity* of the Netflix Prize data. The intuition is that unpopular movies are rarely rated by users; thus, by rating unpopular movies the user distinguishes herself from the crowd. These examples show that removing user names is not sufficient to protect users from attack; with certain auxiliary knowledge of users' reviews, the adversary is still able to violate their privacy and obtain sensitive information. Such background knowledge can be easily obtained from personal blogs, public bulletin board systems (BBS), and other related recommender systems (e.g., the IMDB website[2]).

Releasing anonymized data to the public for research is an inevitable trend[3] and has the potential to provide society with substantial benefits in many fields, including healthcare, medical sciences, and social sciences. However, there is a tradeoff between the utility and privacy of anonymized data. In recommender systems, as the anonymized data deviates from its original in attempts to preserve privacy, predictions based on the anonymized data may become less accurate. Sparsity in recommendation data significantly increases the difficulty of such anonymization tasks in real-world datasets, a challenge which we aim to address. In this paper, we take on the task of developing an efficient, utility-preserving approach for anonymizing large-scale real-world recommendation datasets.

1.2 Challenges

Although privacy preservation in data publishing has been studied extensively under several privacy models (e.g., k-anonymity [33] and l-diversity [22]), most algorithms proposed are only designed for relational datasets. There have also been several studies on privacy-preserving publishing of graphs [9, 17, 34, 36]. In this work, we model recommendation databases as labeled bipartite graphs. The additional structure and labels make users more susceptible to privacy attacks and introduce new challenges to anonymization that have not yet been addressed in the literature.

Most importantly, none of existing anonymization work has studied the effect that sparsity has on privacy. Real-world recommendation data is quite sparse; that is, each individual's rating profile only contains values for a small fraction of items in the database [26]. The so-called *sparsity problem* has an adverse affect on anonymization: it increases the difficulty of designing anonymization schemes that provide acceptable predication accuracy. Unfortunately, the existing anonymization algorithms are not effective when applied to sparse recommendation datasets.

In addition, most of the existing privacy methods are built around the assumption that there are two *non-overlapping* value sets: sensitive values, which need to be kept private, and quasi-identifier values, which can be used by the adversary to identify individuals. In recommender systems, these two sets are not disjoint; *all information in a recommendation dataset could be sensitive, and can also potentially be used as quasi-identifiers.* This additional challenge requires new privacy models that are applicable to recommender systems.

1.3 Contributions

In this paper, we study how to publish sparse recommender data in a privacy- and utility-preserving manner. Our high-level approach is to group and average similar user profiles together. However, in a sparse dataset, finding similar users is challenging – even users with similar tastes have a relatively small overlap in items rated. Our main idea is that before anonymization, we *reduce sparsity by padding the data with predicted values.* This pad-then-anonymize

[2]The Internet Movie Database, http://www.imdb.com
[3]Netflix has already announced plans for a second Netflix Prize with a new dataset.

approach is able to uncover and leverage the latent interests of users that would otherwise be lost without the preprocessing step. Our solution, *Predictive Anonymization*, is a powerful and general approach for publishing all types of recommender data. We summarize our contributions as follows.

- We formalize privacy and attack models for recommendation databases. We model the review data as a labeled bipartite graph, where two disjoint sets of nodes represent users and items, respectively. We formally define two types of adversary attacks, namely a *structure-based attack* and a *label-based attack*. We also give definitions of *k-anonymity* and *l-diversity* in the context of recommendation databases.
- We develop *Predictive Anonymization*, a novel technique to pad, cluster, and anonymize recommendation data. We propose several variations of the algorithm and analyze their privacy guarantees.
- We perform a set of experiments to test the effectiveness and efficiency of our approach. Our experiments are carried out on the entire Netflix Prize dataset, which contains 480,189 users and 17,770 movies. Finally, we use the results to study the privacy-utility trade-off when anonymizing recommendation data.

Organization of the paper The rest of paper is organized as follows. Section 2 provides a brief background of recommender systems. Our privacy model is defined in Section 3. In Section 4 we present our Predictive Anonymization method. Complexity and security analysis are given in Section 5. Extensions to the anonymization algorithm, based on the idea of l-diversity, are given in Section 6. The experimental results are described and analyzed in Section 7. Section 8 presents related work. Section 9 summarizes the paper.

2. RECOMMENDER SYSTEMS

Recommender systems produce automatic predictions about the interests of users by collecting preference information from many users. A recommender system consists of: (1) A set of users $U = \{u_1, \ldots, u_m\}$, (2) A set of items $O = \{o_1, \ldots, o_n\}$, (3) An ordered set of possible rating values S, and (4) A set of user ratings $\{(u, o, r)\}$ where $u \in U$, $o \in O$, and $r \in S$ is the rating value assigned by the user u to an item o (only if u has rated o).

Given a recommender system, the ratings can be represented as an $m \times n$ matrix R. Each cell $r_{i,j}$ is either a real number r, which corresponds to the triplet (u_i, o_j, r), or 0 if user u_i has not rated item o_j. This leads to a natural representation of the system as a bipartite graph. We refer to the vertices that represent users, denoted by V_U, as *user nodes*. Vertices representing the items, denoted by V_O, are called *item nodes*.

Definition 1. **Bipartite Review Graph** A recommender system (U, O, R) corresponds to a *bipartite review graph* $G = (V_U \cup V_O, E, L)$, where each user $u_i \in U$ corresponds to a node $v_{u_i} \in V_U$, each item $o_j \in O$ corresponds to a node $v_{o_j} \in V_O$, and each non-zero entry $r_{i,j}$ in the rating matrix corresponds to the edge $(v_{u_i}, v_{o_j}) \in E$. $L : E \rightarrow S$ is the *label function*, which assigns to each edge $(v_{u_i}, v_{o_j}) \in E$ the label $r_{i,j} \in S$, one of the possible rating values. Thus the rating (u_i, o_j, r) corresponds to the edge $e = (v_{u_i}, v_{o_j}) \in E$ being labeled with $L(e) = r$.

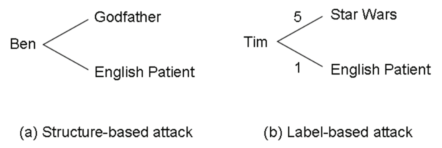


Figure 2: Models of adversary knowledge

Furthermore, the *review graph of user u* is the subgraph $G_u = (\{v_u\} \cup N(v_u), E_u, L_u)$, where $N(v_u) \subseteq V_O$ is the neighborhood of v_u, i.e. $N(v_u) = \{v_o \mid (v_u, v_o) \in E\}$, and $E_u \subseteq E$ and $L_u \subseteq L$ are the corresponding edges and labels in the subgraph induced by $\{v_u\} \cup N(v_u)$.

An example of a bipartite review graph is shown in Figure 1 (a). In this graph, the review graph of user 0001 consists of his/her user node, two movie nodes, *Star Wars* and *Godfather*, and two labeled edges connecting the user node with the two movie nodes.

The task of recommender systems is to predict the rating $r \in S$ that a user $u \in U$ would assign to an item $o \in O$. Much research has been done to improve the quality of prediction by combining information from many users, a task known as *collaborative filtering*. Most existing approaches to this task are variations of k-nearest neighbors (e.g., [2], [14]) or singular value decomposition (SVD) (e.g., [15]). We refer the readers to [5] for a good survey of collaborative filtering algorithms.

3. PRIVACY AND UTILITY MODELS

In released recommendation data, e.g. Netflix, usernames are replaced with unique integer IDs, while item names and ratings are made public for data analysis purposes. In this paper, we identify two privacy goals in anonymizing recommendation data.
(1) Node identification privacy: re-identification of individuals in a released database is considered a privacy breach.
(2) Link existence privacy: the knowledge of which items are reviewed by a specific user is considered private information of that user.

3.1 Attacks Models

By gathering information from external data sources, an adversary may know a subset of items that a specific user has reviewed. The adversary can try to uniquely identify the user by matching this background knowledge with the released dataset. Based on this, we define the *structure-based attack*.

Definition 2. **Structure-based Attack** Given a released bipartite review graph $G^* = (V_U \cup V_O, E^*, L^*)$, let $G_u^A = (\{v_u\} \cup N^A(v_u), E_u^A, \emptyset)$ be the subgraph representing the adversary knowledge of user u. If there are k user nodes, each of which $v_{u'} \in V_U$ has $G_u^A \subseteq G_{u'}^*$, we say that user u is identified by the *structure-based attack* with probability $1/k$.

Figure 2 (a) shows an example of the adversary knowledge for a structure-based attack. The adversary knows that Ben has watched the movies *Godfather* and *English Patient*. By matching this knowledge to the released data in Figure 1 (a), the attacker uniquely identifies Ben as user 0002.

In addition to the structure-based attack that is based on knowledge of which items have been reviewed by a user, the attacker may also know the ratings of these items. Such additional adversary knowledge enables the *label-based attack*.

Definition 3. **Label-based Attack** Given a released bipartite review graph $G^* = (V_U \cup V_O, E^*, L^*)$, let $G^A_u = (\{v_u\} \cup N^A(v_u), E^A_u, L^A_u)$ be the subgraph representing the adversary knowledge of user u. If there are k nodes, each of which $v_{u'} \in V_U$ has $G^A_u \subseteq G^*_{u'}$, we say that user u is identified by the *label-based attack* with probability $1/k$.

Figure 2 (b) shows an example of adversary knowledge for the rating-based attack. The adversary knows that Tim has given a low rating to the movie *English Patient*. By matching this knowledge to the released data (Figure 1 (a)), the adversary uniquely identifies Tim as user 0003, since the other user who reviewed *English Patient* gave a high rating.

The label-based attack is a stronger model than the structure-based attack. Thus by giving privacy guarantees against the label-based attack, we are protecting against the structure-based attack as well. In the following, we mainly focus on the label-based attack.

3.2 Privacy Model

In practice, it is hard to predict the amount of background knowledge that an adversary has gained. Therefore, we aim to provide protection against the strongest adversary that we have considered, the label-based attack. To achieve this goal, we adapt the definition of *k-anonymity* [30, 33] to our model. The conventional k-anonymity model defines quasi-identifier attributes (publicly available information that may be used to identify individuals) and sensitive attributes (private information known only by the individual). These two sets of values are assumed to not overlap. In our problem, the nodes, edges, and labels in the released review graph are both quasi-identifiers and sensitive values. To address this problem, we define k-anonymity in recommender systems as follows:

Definition 4. **k-anonymity** Given a bipartite review graph $G = (V_U \cup V_O, E, L)$, let $G^* = (V_U \cup V_O, E^*, L^*)$ be the review graph of the released dataset. We say G^* satisfies *k-anonymity* if for every user $u \in U$, there are at least $k-1$ other users $\{u_i\}_{i \in I}$ such that G^*_u is isomorphic to $G^*_{u_i}$. We say that $\{u\} \cup \{u_i\}_{i \in I}$ is the *anonymization group* of u.

Intuitively, Definition 4 requires that for each user node u, there are at least $k-1$ other user nodes that have identical review graphs to u in terms of both structure and labels. Thus the k-anonymity model is effective for defending against both the structure-based and label-based attacks.

3.3 Utility Measure

Since data owners publish their recommendation data to seek improved recommendation algorithms, it is desirable that the released data preserves prediction accuracy as much as possible. Unfortunately, existing utility measures for relational databases (e.g., the ratio of nodes in the generalization taxonomy trees [19, 35], the distance between the distributions of the original and anonymized datasets [20], and the estimation error of aggregate query answers [28]) cannot be applied to recommender systems to achieve this utility goal.

We propose using the *root mean squared error* (RMSE) to measure the accuracy of prediction results. More specifically, given the original recommendation dataset D and its anonymized version D^*, let $RMSE_D$ and $RMSE_{D^*}$ be the average RMSEs of the prediction results on D and D^*, respectively. Intuitively, the closer these two RMSEs are, the better utility that the anonymization preserves. Our RMSE-based method allows us to easily measure the prediction accuracy of both original data values and their anonymized ones, which is a general approach for computing the amount of information change during anonymization.

4. PREDICTIVE ANONYMIZATION

To protect sensitive information about users in a recommender system, the data owner should anonymize recommendation data before publishing it. We delineate several goals for an effective anonymization algorithm:

1. **Privacy Goal**: the anonymized dataset must satisfy k-anonymity
2. **Utility Goal**: the published data should preserve prediction accuracy
3. **Performance Goal**: the algorithm must be efficient for large datasets

We design a general method, *Predictive Anonymization*, that achieves these goals. The key idea in our method is a *predictive padding* step that aims to amplify the original data features by strategically replacing null entries with meaningful values. However, if not done carefully, padding may destroy original data patterns and cause information loss.

In this section, we outline the general Predictive Anonymization procedure and provide details of our implementation. The procedure consists of three major steps: (1) strategically pad the data to reduce sparsity, (2) construct anonymization groups from the pre-processed data, and (3) homogenize the ratings within each group.

4.1 Step 1: Predictive Padding

Recommender systems are typically sparse [16, 32, 31]; that is, the percentage of items reviewed by an average user is small. Sparsity can be detrimental to privacy because it decreases the amount of auxiliary information needed for re-identification [26]. Furthermore, it hampers the effectiveness of anonymization that is based on grouping users with similar ratings. Due to data sparsity, even users with very similar preferences may have very small overlap in the items they have rated. Thus, anonymization on the *original* data may not effectively group similar users, which impairs the accuracy of prediction.

We take a novel approach to anonymization by padding the data with predicted values before constructing anonymization groups. An important insight of our technique is that similar users are not necessarily those who have rated the same items; rather, it is users who have *similar item preferences*. Since the end goal is to preserve prediction accuracy in the anonymized dataset, it is essential to form anonymization groups of users with similar item preferences to minimize information loss. We use a method called *Regularized SVD (Singular Value Decomposition)* to achieve that goal.

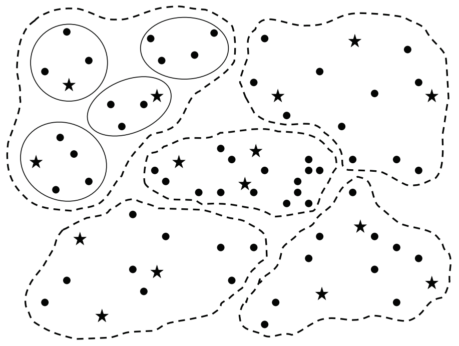


Figure 3: Our Sample and Cluster Procedure. The randomly chosen sample points are designated by stars. The dashed lines identify the bins, and solid ovals represent the final anonymization groups.

Regularized SVD is a matrix approximation method that has been effectively used in the domain of natural language processing, and has more recently been proposed for collaborative filtering [12]. Given an input matrix A and an accuracy parameter r, SVD outputs the matrix A' of rank r with the minimum approximation error in terms of least squared distance from A. Regularization imposes additional restrictions that prevent overfitting, which enables us to extract pertinent information from the existing ratings and use it to inpute the missing values. Note that although we chose to use Regularized SVD for our implementation, predictive padding is a general approach that supports any method of imputation.

At the end of the padding phase, all null ratings have been replaced with predicted values, effectively eliminating the sparsity problem. This predictive padding does not affect the original data, which may still be used to construct the final released anonymized dataset. Yet, the padded data significantly improves the accuracy of clustering that is based on user similarities, which is explained next.

4.2 Step 2: Forming Anonymization Groups

Our high-level anonymization strategy is to partition users into anonymization groups of size at least k so that an adversary can not distinguish between the ratings of individuals within a group. This is accomplished using a clustering algorithm that guarantees a minimum cluster size. At the end of the procedure, each cluster has at least k users with similar item preferences.

Since the most accurate clustering algorithms for sparse recommendation data run in super-linear time, we employ a sampling technique to improve the efficiency of our algorithm. Our clustering procedure differs from existing clustering-based anonymization work (e.g., [1]) by computing and utilizing pair-wise similarity values for clustering weighted bipartite graphs. Also, our method based on sampling and bin-assigning is scalable to large datasets. Our clustering procedure (shown in Figure 3) works as follows:

1. Randomly select a set of sample points from the dataset.

2. Cluster the sample points into equal-sized "bins".

3. Partition the entire dataset, placing each point into the nearest bin.

4. Determine anonymization groups by clustering points within each bin.

Clustering Algorithm The task of clustering data has been studied in great depth. One classic clustering technique is known as the *k-Means algorithm* [21, 23]. To avoid confusion with the k in k-anonymity, we use the term *t-Means* throughout the paper. The t-Means algorithm takes an input parameter, t, and partitions a set of objects into t clusters so that the resulting intra-cluster similarity is high.

In practice, however, it has been observed that classic clustering algorithms frequently produce clusters whose sizes are of skewed distribution (i.e., some clusters are very large while some are small or even empty), especially when clustering high-dimensional datasets [4]. In contrast, a balanced distribution of cluster sizes is desired in Step 2 to ease computation for subsequent steps, as well as in Step 4 to minimize information loss during anonymization while guaranteeing the k-anonymity privacy requirement.

To achieve this, we use the *Bounded t-Means* algorithm from [34], which guarantees that the sizes of all t clusters are no smaller than a pre-defined lower bound. We first apply the Bounded t-Means algorithm to the sample points to find balanced bins for partitioning the dataset. To reduce the complexity of the entire clustering procedure, we choose the value $t = \sqrt{n}$. More details of why we pick this value are explained in Section 5.1.

Finally, we apply the Bounded t-Means algorithm to cluster the users in each bin. For bin B_i we set the parameter $t = |B_i|/k$, so that every cluster is guaranteed to contain at least k users. By grouping similar users into the same bin, we incur little information loss by first using our sampling procedure, compared to if we had clustered all the users at once. However, our procedure yields a significant speed-up in run-time for the clustering step, especially when performed on large datasets. See Section 5.1 for details.

Similarity Metric To perform Bounded t-Means clustering, we must first define our user similarity metric. Let Δ be the difference between the highest and the lowest possible ratings in the dataset, i.e. $\Delta = max(S) - min(S)$. Then given two users u and u' and their corresponding rating vectors $(r_1, \ldots, r_n)$ and $(r'_1, \ldots, r'_n)$, we define the similarity between u and u' to be

$$sim(u, u') = 1 - \frac{\sum_{1 \le i \le n} (1 - d_i)^2}{n}$$

where $d_i = |r_i - r'_i|/\Delta$ for $1 \le i \le n$.

It is straightforward to verify that $sim(u, u') = 0$ only when u and u' match exactly on every common entry, and $sim(u, u') = 1$ only when u and u' have opposite minimum and maximum ratings for each item. This method essentially gives a squared penalty for large differences in item preferences between two users. We studied the effectiveness of this metric against other similarity metrics. More details can be found in Section 7.4.

Cluster Centers An operation used frequently in the Bounded t-Means algorithm is computing the center of a cluster. We adapt the idea of *virtual centers* from [34] to our problem.

Consider a cluster C of users $\{u_1, \ldots, u_k\}$, each u_i assigned with a rating vector $\langle r_{i,1}, \ldots, r_{i,n} \rangle$ to the items $o_1, \ldots, o_n$. Then the virtual center c of cluster C is a vector of ratings $\langle \hat{r}_1, \ldots, \hat{r}_n \rangle$ such that $\hat{r}_j = \sum_{i=1}^{k} \overline{r}_{i,j}/n$, where $\overline{r}_{i,j}$

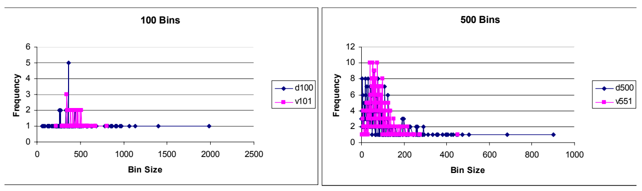


Figure 4: Real User vs. Virtual Center Points

is the padded rating of $r_{i,j}$. Note that since we are using the padded user rating vectors, there are no null ratings.

Our experimental results demonstrate the effectiveness of using virtual centers as opposed to real users as center points. Figure 4(a) shows that using real user centers for clustering the sample points results in bin sizes varying from 50 to 2000, whereas virtual centers yield bin sizes within the range 200 to 900. Since balanced clusters facilitate easy data handling and faster running time in later anonymization steps, we use virtual centers in our implementation.

4.3 Step 3: Homogenization

To defend against both the structure-based attack and label-based attack, our final step is to homogenize the users in each anonymization group so they have identical review graphs. A straightforward way to do this is to apply the popularly used generalization and suppression techniques [30, 33]. However, as pointed out by [26], generalization and suppression may completely destroy the utility of the data for collaborative filtering. We take a different approach, *homogenization*, which consists of adding fake edges and labels so that all users within an anonymization group are connected to the same set of item nodes with the same ratings.

Formally, the homogenization of the anonymization group corresponding to cluster C of users $\{u_1, \ldots, u_k\}$ has the following operations: *union*, *complete*, and *average*. First, we construct the *union of the review graphs* of all users in C. Let the union result be $G_C = (C \cup N(C), E_C, L_C)$. Second, we add fake edges between users and items to create G^*_C, a *complete bipartite subgraph*; i.e., $E^*_C = C \times N(C)$. For instance, in Figure 1 (b), user 0001 and 0002 are in the same anonymization group. A fake edge is added between user 0001 and movie *English Patient*, so that both user 0001 and 0002 review the same set of movies in the review graph. Third, we re-label all the edges in G^*_C, including the fake ones, with the appropriate average ratings. When calculating the average ratings, we can use either the ratings in the original dataset or the ones in the padded dataset. Following this, we design two homogenization schemes: *padded anonymization* and *pure anonymization*.

Padded anonymization uses the ratings in the padded dataset. That is, for each anonymization group, we calculate the homogenized rating of each item as the average padded rating over all users in the group. Formally,

$$(\forall u_i \in C, o_j \in O) \quad L^*_C(u_i, o_j) = \hat{r}_j = \sum_{u_i \in C, r_{i,j} \neq 0} \overline{r}_{i,j}/m,$$

where m is the number of users in the database, and $\overline{r}_{i,j}$ is the padded rating of user i for item j. Note that an average rating is computed for every item in the dataset, since in padded anonymization there is an edge between every user and every item in the anonymized review graph.

Pure anonymization uses the ratings in the original dataset. That is, for each anonymization group, we refer back to the unpadded data and calculate the homogenized rating of each item as the average rating over only users who rated that item in the original dataset. Formally,

$$(\forall u_i \in C, o_j \in N(C)) \quad L^*_C(u_i, o_j) = \hat{r}_j = \sum_{u_i \in C, r_{i,j} \neq 0} r_{i,j}/k',$$

where k' is the number of users in C who rated item o_j.

Note that in all variants of Predictive Anonymization, all users in cluster C are assigned the same homogenized rating $\hat{r}_j$ for each item o_j. The difference is that in padded anonymization, the released dataset contains padded values, which effectively obscure real data and thus provide stronger privacy protection against homogeneity attacks (more details are in Section 6). However, the padded-anonymized data deviates significantly from the original dataset, as it is strongly influenced by padded values. If the predictions used in the padding step were not accurate, this may adversely affect the utility of the released data. Furthermore, much of the structure of the original data is lost – including sparsity, one of the most defining characteristics of recommendation data. This casts doubt on the ability of padded anonymization to preserve the *integrity* of the data. On the other hand, a released dataset that has undergone pure anonymization does not contain any padded values, and thus better preserves the integrity and utility of the data, although it may provide weaker privacy guarantees. A more detailed comparison of the two methods is described in Section 7.

5. ANALYSIS

In this section, we analyze the complexity and security of our predictive anonymization algorithms.

5.1 Complexity Analysis

Let m be the number of users, and let n be the number of items ($m = 480,189$ and $n = 17,700$ in the Netflix dataset). Using recent techniques for optimization of SVD, Step 1 (Section 4.1) can be performed in $O(mn)$ time. Step 2.1 (Section 4.2) takes $O(s)$ time, where s is the size of the sample. The bounded t-means algorithm in Step 2.2 (Section 4.2) has complexity $O(st_1n)$. For Step 2.3 (Section 4.2), using the center points from the sample to partition the dataset into bins runs in $O(mt_1n)$ time; clustering on each bin takes time $O(|B_i|tn) = O((|B_i| * |B_i|/k)n)$, where $|B_i|$ is linear in m/t_1. Thus the complexity is $O(m^2n/(t_1^2k))$. There are t_1 bins overall, so the total complexity is $O(m^2n/(t_1k))$. To reduce the quadratic complexity in m, we set $t_1 = \sqrt{m}$, which results in the complexity of this step being $O(m^{3/2}n/k)$. For Step 3 (Section 4.3), homogenization of each cluster C takes complexity $O(|C|n) = O(kn)$. There are m/k clusters, thus the total complexity is $O(mn)$. Based on the above, the complexity of the entire anonymization approach is $O(mn + s\sqrt{mn} + m^{3/2}n + m^{3/2}n/k + mn)$. Since $s < m$, the complexity is $O(m^{3/2}n)$.

5.2 Privacy Analysis

In this section, we analyze the guarantee that our *Predictive Anonymization* algorithm provides both *node re-identification privacy* and *link existence privacy* defined in Section 3.1. The following theorem is analogous to the correctness of the k-anonymity model on relational databases.

THEOREM 5.1. ***Node Re-identification Privacy*** Let G be the bipartite review graph for a recommender dataset, and let G^* be the corresponding released review graph. If G^* is k-anonymous, then a user cannot be re-identified in G^* with confidence greater than $\frac{1}{k}$.

As shown in Section 4.3, fake edges are added between user vertices and item vertices during the homogenization step, which prevents the adversary from explicitly determining which edges exist in the original dataset (or furthermore their labels). Assume that all (user, item) ratings are independent, both the existence and the values of the ratings. Furthermore, assume the adversary has no prior knowledge about the likelihoods that users have rated items. Then the confidence with which the adversary can learn the existence of a link is at most $1/k$. This claim is stated concisely as follows.

THEOREM 5.2. ***Link Existence Privacy*** Assume that all ratings are independent. Then an adversary with no prior knowledge employing a label-based attack cannot predict the existence of an edge (v_u, v_o) with confidence greater than $\frac{1}{k}$.

Suppose an adversary has prior knowledge that the probability a user has rated item o is p. Obtaining this knowledge is often feasible in practice by learning aggregate information about the database. For example, in the Internet Movie Database (IMDB), the most frequently rated movie is "The Shawshank Redemption", which has been rated by 2.4% of registered users. Assume that $p \leq 1/k$ (a reasonable assumption due to the sparsity of recommender datasets). We claim that even with this additional prior knowledge, the adversary cannot significantly improve his confidence that a user has rated item o.

THEOREM 5.3. ***Link Existence Privacy With Prior Knowledge*** Assume that all ratings are independent. Then an adversary with prior knowledge $p \leq \frac{1}{k}$ for item o employing a label-based attack cannot predict the existence of an edge (v_u, v_o) with confidence greater than $\frac{1}{k} + \frac{p}{2-kp}$.

Proofs can be found in the Appendix. Note that the greatest confidence gain occurs when $p = 1/k$, at which point the adversary has a $2/k$ confidence probability. Therefore, assuming that $p \leq 1/k$, the maximum confidence in predicting the existence of a link is bounded by $2/k$. Furthermore, recommendation data is typically very sparse, so it is of note that the adversary confidence tends to $1/k$ as $p \to 0$.

While it may be difficult to learn the existence of a link, learning the non-existence of links is easy with a label-based attack. However, we claim that due to the sparsity of the data and the practical significance of a link, it is reasonable to assume that only positive link existence should be considered sensitive.

6. ACHIEVING L-DIVERSITY

The l-diversity model provides complementary privacy protection to k-anonymity. In relational data, l-diversity requires that sensitive attributes should have *diversity* by having at least l distinct values in each k-anonymous class [22]. However, the definition and security implications of l-diversity in recommender databases are unclear. Therefore, we give the first formal definition of l-diversity for labeled bipartite review graphs, and an algorithm to realize both k-anonymity and l-diversity in recommender systems.

To appreciate the need for l-diversity in recommendation data, we need to first understand a subtle attack against link privacy. Once the anonymization group of a target victim is identified via a structure-based or label-based attack, it becomes easier to target that user for more sophisticated attacks. Although the adversary cannot explicitly identify any user, more can be deduced from the anonymized data than what we want to allow. For example, suppose an adversary only has the background knowledge to perform a structure-based attack. After identifying the correct anonymization group, since the k users in that group have identical review profiles, the adversary can easily learn new information about the ratings that the target user gave to those items.

This problem is further exacerbated by the fact that some items are rarely reviewed. Exploring rare items to re-identify users in Netflix data was recently studied [26], and facilitates the easy identification of a user's anonymization group, leaving the target susceptible to the above attacks. We refer to these attacks as *homogeneity attacks* following [22]. The threat of these homogeneity attacks motivates the need for l-diversity.

Definition 5. **Homogeneity Attack in Bipartite Graphs** Given a released bipartite review graph $G^* = (V_U \cup V_O, E^*, L^*)$, let $G_u^A = (\{v_u\} \cup N^A(v_u), E_u^A, L_u^A)$ be the subgraph representing the adversary knowledge for a user u. Let $\{v_{u'}\}$ denote the set of nodes, including v_u, each of which $v_{u'} \in V_U$ has $G_u^A \subseteq G_{u'}^*$. If all the nodes in $\{u'\}$ are *identical*, then we say the *review profile* of user u is uniquely identified by the homogeneity attack.

Unlike relational data, our k-anonymization algorithm alone provides some degree of privacy even in the face of a homogeneity attack (See Theorem 5.3). Because we add fake edges during anonymization to average the k users, a rating in the anonymized data does not necessarily mean that the target user has rated that item, or if so, that the anonymized rating accurately reflects the true rating of that user. However, as explained above, homogeneity attacks can be effective in some scenarios.

First, we formally define (b, l)-diversity. We present an extension to the *Predictive Anonymization* algorithm that realizes both k-anonymity and (b, l)-diversity. The b indicates that the adversary's prior knowledge includes at most b items that have been reviewed by the user. Intuitively, (b, l)-diversity requires that every subset of b items must be included in at least l different anonymization groups.

Definition 6. (b, l)**-diversity** Given a bipartite review graph $G = (V_U \cup V_O, E, L)$, let $G^* = (V_U \cup V_O, E^*, L^*)$ be the corresponding k-anonymized review graph with anonymization groups $\dot{\bigcup} C_i = U$. We say G^* satisfies (b, l)-*diversity* if for every set of b items $B = \{o_1, \ldots, o_b\} \subset O$ that have been rated by a user, there are at least l distinct anonymization groups C_i such that $B \subset N(C_i)$.

To achieve $(1, l)$-diversity, we modify our algorithm as follows: After homogenization is performed, we check whether each item o has been covered by l groups. If it has not, we randomly select anonymization groups C such that $o \notin C$ and add fake edges between item o and all user nodes in C, so that every item o is connected to at least l groups. The labels on these fake edges are computed using the padded values for the corresponding users. The above method can be easily generalized to realize (b, l)-diversity, the details of which are omitted here.

7. EXPERIMENTS

We have done a set of experiments to evaluate both the effectiveness and efficiency of our k-anonymization algorithm (without l-diversity). Specifically, we want to evaluate the impacts of padding and anonymization on the utility and structure of the anonymized data, and the amount of information change introduced by the anonymization. In this section, we describe our experiment design and results.

7.1 Setup

We ran our experiments parallelized on 6 different machines. Four of the machines are equipped with eight Intel(R) Xeon(R) CPU 3.00GHz, 16GB memory and CentOS 5.2 Linux, and the other two machines are equipped with two Intel(R) Core(TM)2 Duo CPU at 3.00GHz, 3GB memory and Fedora 8 Linux. We implemented our algorithm in C++, Java, and Perl.

We use the entire Netflix dataset for our experiment. The original data contains a total of 480,189 users' ratings on 17,770 movies. The ratings range from 1 to 5, with 0 meaning a rating does not exist. The Netflix challenge set (a.k.a. probe set) is used to evaluate the performance of a prediction algorithm. It contains 459,178 users and 1,425,333 user-movie pairs to be predicted. The users are a subset of the original Netflix dataset. In our experiments, we use the challenge set to evaluate the utility and information loss in the anonymized data, by comparing to a fixed prediction algorithm, namely SVD. (We do not aim to develop a new collaborative filtering mechanism.) Briefly, the challenge set is used as follows. For each user-movie pair in the challenge set, one needs to predict the corresponding ratings based on the dataset $\hat{D}$, where $\hat{D}$ is the Netflix Prize set excluding the challenged entries.

We use the open-source SVD implementation in the Netflix Recommender Framework [25] for padding, and also for prediction in some experiments, which is described in more detail later. After running SVD padding, the size of the (padded) dataset is about 36GB. All data is written into hard disk in binary format, and accessed using the *mmap* system call. Due to the file size limit for mmap and in Linux, we split the padded dataset into 40 binary files.

To measure the utility of the anonymized dataset, we modify the conventional error computation by calculating the RMSE for users *before* and *after* their anonymization. This new error quantification approach for anonymized recommender data is called *target deviation* and is defined as follows: For user u in the challenge set, we (the data owner) identify u', the anonymized version of u, among other anonymized users, and then output the predicted ratings of u' as our predictions for u. The advantage of target deviation is the direct and simple quantification of differences before and after the anonymization by leveraging the background knowledge of the data owner[4].

[4]The data owner is able to uniquely identify u' from u by keeping track of the anonymization process, whereas the public cannot.

Experiment Series	RMSE*
Original Data	0.951849
Padded Anonymization ($k = 5$)	0.95970
Padded Anonymization ($k = 50$)	0.95871
Pure Anonymization ($k = 5$)	2.36947
Pure Anonymization ($k = 50$)	2.3771

Table 1: The target deviation RMSEs

Rating Range	No. of Ratings in Original Dataset	No. of Ratings After Padding
[0]	98.84%*	0
[1]	0.053%	0.79%
[2]	0.117%	14.12%
[3]	0.334%	46.71%
[4]	0.390%	33.49%
[5]	0.267%	4.89%

Table 2: The rating histograms before and after SVD padding. *This value is the number of zero entries.

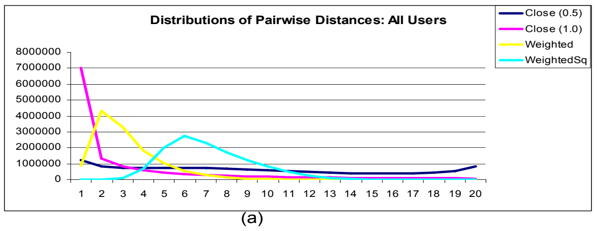
7.2 Evaluation of the Utility of Anonymized Data

First, we measure the utility of both *pure anonymization* and *padded anonymization* schemes (details in Section 4.3). The RMSE results of both experiments under different k values are shown in Table 1. The high RMSE values (2.36947 and 2.3771) for the pure anonymization is due to both the limited data size and the sparsity of the anonymized data. With $k = 50$, there are only 9,294 anonymized users (i.e., groups) in the public released data. Among them, 80% of the ratings are null if the averaging is done on the original data (as in pure anonymization), as opposed to no null ratings in the padded data (used for padded anonymization).

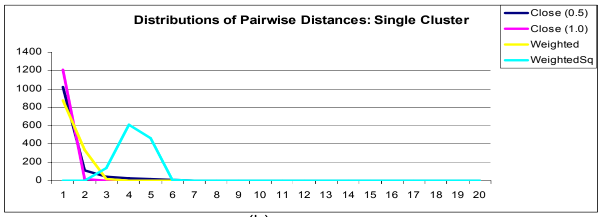
7.3 Data Characterization and Clustering Evaluation

We characterize the data sparsity and compare the sparsity before and after our padding procedure. We simply count the number of ratings that fall within a range. The results are shown in Table 2. It is clear that padding significantly changes the distribution of ratings in the dataset, in particular, null ratings. Padding data with SVD tremendously reduces the data sparsity, and provides a rich context for identifying similar users, as the pair-wise user similarity computation is more accurate and meaningful.

We characterize the various user similarity metrics that may be used in the clustering algorithm. We evaluate four metrics: closeness-0.5, closeness-1.0, weighted similarity, and our weighted-squared similarity measure Closeness-a is a simple similarity measure on two vectors V_1 and V_2 by counting two corresponding vector entries similar if their difference is within threshold a. Weighted similarity assigns a weight to various ranges to penalize discrepancies. All similarity values are normalized to within [0,1], and are categorized into 20 disjoint ranges, namely: $[0, 0.05)$, $[0.05, 0.1)$, $\ldots$, $[0.95, 1.0)$. Figure 5 (a) shows the distribution of pairwise similarities for 5000 users under the four measures before clustering (after padding), and (b) shows the distribution of similarities between users within one single cluster.



(a)



(b)

Figure 5: Comparison of four distance metrics in user-user similarity computation. WeightedSq denotes our similarity measure used in this work.

Experiments With Similar-User Deviation	RMSE
Padded Anonymization ($k = 50$)	1.00563
Pure Anonymization ($k = 50$)	1.17525

Table 3: RMSEs for padded and pure anonymization experiments with k = 50.

The shift in distribution to the left indicates that the clustering algorithm is able to group similar users. However, if a similarity measure is too relaxed (e.g., closeness-1.0), then the similarity values are artificially inflated, which does not provide a good indicator in clustering. In comparison, a more strict similarity measure such as ours has a more fine-grained ability to distinguish similarities in user profiles.

7.4 RMSE By Similar-User Deviation

Target deviation is more accurate in reflecting the information loss incurred during anonymization. It implicitly assumes that the anonymized user is the most similar to herself before the anonymization, as the RMSE is computed by directly comparing the ratings of a user *before* and *after* anonymization. To eliminate the assumption in target deviation, we define a *similar-user deviation*, which is a more realistic method for computing the utility of anonymized data. It is based on user-based collaborative filtering. For a user u in the challenge set, we find the anonymized user v (in the anonymized dataset) is most similar to u according to a similarity measure. We apply v's ratings as our prediction for u, and then compute the RMSE for the entire challenge set. Table 3 shows the experimental results computed based on our similar-user deviation definition for both padded anonymization and pure anonymization experiments with k = 50.

To evaluate the differences between the two deviation metrics, we define *self-rank* of a user u in the challenge set as the rank of her anonymized version u' among other anonymized users in terms of similarity to u. Specifically, compute and sort the similarities between u and all the anonymized users; identify the rank of u'. (Note that the data owner performing the anonymization is able to uniquely identify u' from

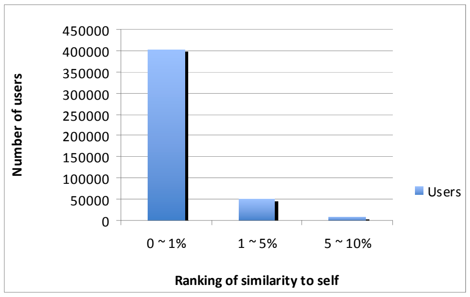


Figure 6: Number of users whose self-rank (See Section 7.4) is within a certain percentage.

u, whereas the public cannot.)

In target deviation, the self-ranks are assumed to be one for all users in the challenge set. With similar-user deviation, most self-ranks values are quite low, indicating that target deviation is a good approximation in the error computation. We categorize the self-rank values in Figure 6.

Experiment summary We quantify and compare the information loss in different experiment setups, in particular when the original data is released as opposed to the padded data. *Padded anonymization* is effective in preserving the data quality with low prediction errors. Our privacy analysis also shows that the padded data improves user privacy as a positive side effect. The value of k does not have a significant impact on prediction accuracy. This proves that strategically replacing null entries with padded values has positive impacts on the utility of anonymized data.

Although preserving prediction accuracy, the padded anonymization method loses authentic data properties and the released data cannot support statistical queries. For example, one is unable to find out what percentage of users have rated a movie. This underlines an intrinsic tradeoff between user privacy and data utility.

In comparison, *Pure Anonymization* mitigates these issues to some degree, as it homogenizes on the original dataset (as opposed to padded data). For example, it can accurately answer queries such as, *What is the average rating on a movie?* However, homogenization on the original data as in the Pure Anonymization method gives much higher RMSE than Padded Anonymization. Averaging incurs high information loss and affects data patterns even with a small k value. This undesirable result also validates earlier predictions by others [26].

8. RELATED WORK

Our anonymization problem is related to privacy-preserving publishing of relational databases, anonymization of network graphs, publishing of unlabeled bipartite graphs, and privacy-preserving collaborative filtering. We describe some of the related work in these areas in the following.

Privacy-preserving publishing of relational databases There has been a great deal of work on privacy-preserving publishing of relational databases. However, due to differences between relational and recommendation datasets (see Section 1), many of the techniques for achieving anonymity in relational databases are ineffective when applied to recommender systems.

Byun et. al. introduce a suppression-based algorithm for anonymizing relational databases [6]. Suppression is ineffective for recommendation data because of the sparsity problem, and may therefore result in a high degree of information loss. Furthermore, suppression is undesirable because the anonymization process changes the format of the original database, and thus may render it invalid for some prediction algorithms.

Another active research area in privacy-preserving databases is *differential privacy* (e.g., [10, 11, 24]). While very promising for contexts in which aggregate query results are sufficient, differential privacy is not applicable to our model, where the desired utility is prediction accuracy of collaborative filtering on anonymized data.

Anonymization of network graphs Anonymization of social network graphs has attracted much attention recently [17, 3, 36]. Since we model recommendation databases as bipartite graphs, any method proposed for general graphs could be directly applied. However, due to the specific bipartite structure of recommendation data, the techniques suggested in the graph anonymization literature are not effective and do not adequately address the additional challenges posed by recommender systems.

Publishing of unlabeled bipartite graphs Cormode et. al. studied the problem of privacy-preserving anonymization of bipartite graphs [9]. Similar to our work, their privacy goal is to protect the association between the nodes in two partitions in the graph (in our case, users and items). However, their work concerns unlabeled graphs where nodes may have attribute values; on the other hand, we consider labeled bipartite graphs, with the possibility of edge labels being used as part of the adversary knowledge. As they state, their solution is only "applicable in situations where it is considered safe to publish the unlabeled graph." Our privacy model does not make this assumption. In addition, we have different target utility goals: they aim to preserve the accuracy of SQL aggregate queries on the released dataset, whereas we measure success by prediction accuracy.

An elegant anonymization approach was proposed by Ghinita et. al. to handle sparse unlabeled bipartite graphs by capturing underlying data correlations [13]. However, their model requires a universal set of sensitive items, and allows only the other items to be used as quasi-identifiers. In comparison, our model allows that any item may be sensitive, and furthermore, any item may be used to re-identify users. The existence of edge labels in recommendation datasets introduces additional challenges, and thus demands different privacy and attack models and new anonymization approaches. Also, their utility measure is based on aggregate query results, rather than prediction accuracy.

Privacy-preserving collaborative filtering Canny proposes two schemes for privacy-preserving collaborative filtering [7, 8] in which a community of users compute a public aggregate of their ratings without exposing any individual users' ratings. Their solutions involve a homomorphic encryption mechanism. In a similar fashion, Hsieh et. al. [18] also take an encryption-based approach. Polat et. al. consider the same problem under a centralized framework [27], in which users send their data to a central server that will conduct the collaborative filtering. Instead of encryption, they propose for users to randomize their private ratings such that the center server cannot derive the truthful ratings, but will still be able to compute the collaborative filtering result from the perturbed data.

It is important to note that we do not claim to offer a new collaborative filtering algorithm, but rather to provide an anonymization technique that produces anonymized datasets on which any collaborative filtering algorithm could be performed. Each of the above solutions apply only to a particular collaborative filtering method, and thus could not be used to achieve the desired goals of this paper.

9. CONCLUSIONS AND FUTURE WORK

In this paper, we report our efforts towards anonymizing Netflix Prize dataset, the difficulties of which are well publicized due to the Narayanan-Shmatikov attack [26]. Our techniques and results have general applications in anonymizing other sparse bipartite recommendation data. We focus on *padding* as a pre-processing step to reduce data sparsity during anonymization. Our proposed approach is called predictive anonymization. We gave SVD as a concrete padding technique before anonymization, but our methodology can be applied with other padding algorithms. We formally defined the model and developed a practical and efficient anonymization algorithm called *Predictive Anonymization.*

Our studies using the Netflix Prize dataset to evaluate the effectiveness of our algorithm in preserving utility of the anonymized data. Padded-anonymized data gives excellent privacy and low prediction errors, however, the data authenticity is low due to the padded values in the released dataset. In comparison, pure-anonymized data has improved data authenticity, but yields high prediction errors. Our study experimentally illustrates the tradeoffs between data utility/authenticity and privacy in anonymization. Padding is a useful pre-processing step for eliminating data sparsity during data anonymization, in particular for finding and grouping similar users as we demonstrated.

For future work, we are planning to take a different approach to improve the utility of the pure anonymization method. Instead of averaging values in the homogenization step, we could permute the rating values for each item within each anonymization group. This may better preserve some characteristics of the data, but it remains to be seen how this approach will affect prediction accuracy and other utility measures. We will conduct extensive experiments with various parameters to investigate the effectiveness of this approach in achieving our privacy and utility goals.

10. REFERENCES

[1] G. Aggarwal, T. Feder, K. Kenthapadi, S. Khuller, R. Panigrahy, D. Thomas, and A. Zhu. Achieving anonymity via clustering. In *Proceedings of the Symposium on Principles of Database Systems (PODS)*, pages 153–162, 2006.

[2] K. Ali and W. van Stam. Tivo: making show recommendations using a distributed collaborative filtering architecture. In *Proceedings of the ACM international conference on Knowledge Discovery and Data Mining (KDD)*, 2004.

[3] L. Backstrom, C. Dwork, and J. Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the International Conference on World Wide Web (WWW)*, 2007.

[4] P. S. Bradley, K. Bennett, and A. Demiriz. Constrained k-means clustering. Technical Report MSR-TR-2000-65, Microsoft research, 2000.

[5] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. Technical Report MSR-TR-98-12, Microsoft Research, 1998.

[6] J.-W. Byun, A. Kamra, E. Bertino, and N. Li. Efficient k-anonymization using clustering techniques. In *Proceedings of the International Conference on Database Systems for Advanced Applications (DASFAA)*, 2007.

[7] J. Canny. Collaborative filtering with privacy. In *Proceedings of the IEEE Symposium on Security and Privacy*, May 2002.

[8] J. Canny. Collaborative filtering with privacy via factor analysis. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, August 2002.

[9] G. Cormode, D. Srivastava, T. Yu, and Q. Zhang. Anonymizing bipartite graph data using safe groupings. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, 2008.

[10] C. Dwork. Differential privacy. In *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP)*, 2006.

[11] C. Dwork, F. Mcsherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Theory of Cryptography Conference (TCC)*, 2006.

[12] S. Funk. Netflix update: Try this at home. http://sifter.org/∼simon/journal/20061211.html, 2006.

[13] G. Ghinita, Y. Tao, and P. Kalnis. On the anonymization of sparse high-dimensional data. In *Proceedings of the International Conference on Data Engineering (ICDE)*, pages 715–724, 2008.

[14] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[15] K. Y. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Journal of Information Retrieval*, 4(2):133–151, 2001.

[16] N. Good, J. B. Schafer, J. A. Konstan, A. Borchers, B. Sarwar, J. Herlocker, and J. Riedl. Combining collaborative filtering with personal agents for better recommendations. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 1999.

[17] M. Hay, G. Miklau, D. Jensen, D. Towsley, and P. Weis. Resisting structural identification in anonymized social networks. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, 2008.

[18] C.-L. Hsieh, J. Zhan, D. Zeng, and F. Wang. Preserving privacy in joining recommender systems. In *Proceedings of the International Conference on Information Security and Assurance (ISA)*, 2008.

[19] V. S. Iyengar. Transforming data to satisfy privacy constraints. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 279–288, 2002.

[20] D. Kifer and J. Gehrke. Injecting utility into anonymized datasets. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2006.

[21] S. P. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 1982.

[22] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *Proceedings of the International Conference on Data Engineering (ICDE)*, 2006.

[23] J. MacQueen. Some methods for classification and analysis of multivariate observation. In *Proceedings of the Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

[24] F. Mcsherry. Mechanism design via differential privacy. In *Proceedings of the Annual Symposium on Foundations of Computer Science (FOCS)*, 2007.

[25] B. Meyer. Netflix recommender framework. http://benjamin-meyer.blogspot.com/2006/10/netflix-prize-contest.html?program=NetflixRecommenderFramework.

[26] A. Narayanan and V. Shmatikov. How to break anonymity of the netflix prize dataset. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2008.

[27] H. Polat and W. Du. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, 2003.

[28] V. Rastogi, D. Suciu, and S. Hong. The boundary between privacy and utility in data publishing. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, 2007.

[29] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW)*, pages 175–186, 1994.

[30] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information. In *Proceedings of the Symposium on Principles of Database Systems (PODS)*, 1998.

[31] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. Application of dimensionality reduction in recommender system - a case study. In *ACM WebKDD Workshop*, 2000.

[32] B. M. Sarwar, J. A. Konstan, A. Borchers, J. Herlocker, B. Miller, and J. Riedl. Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW)*, 1998.

[33] L. Sweeney. k-anonymity: a model for protecting privacy. *Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.

[34] B. Thompson and D. Yao. Union-split clustering algorithm and social network anonymization. In *Proceedings of the ACM Symposium on Information, Computer and Communications Security (AsiaCCS)*, 2009.

[35] J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A. Fu. Utility-based anonymization using local recoding. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, 2006.

[36] B. Zhou and J. Pei. Preserving privacy in social networks against neighborhood attacks. In *Proceedings of the International Conference on Data Engineering (ICDE)*, 2008.

APPENDIX

Theorem 5.2

PROOF. Suppose an adversary, using a label-based attack, is able to identify the anonymization group containing user u. Based on the existence of a link to item o in the released review graph, the adversary would like to infer whether user u gave a rating for o in the original dataset. However, the existence of the link in the anonymized graph only implies that at least one user in that anonymization group had rated o. With no additional prior knowledge, the adversary can only infer that user u had rated o with probability at least $\frac{1}{k}$. □

Theorem 5.3

PROOF. Suppose an adversary, using a label-based attack, is able to identify the anonymization group containing user u. Based on the existence of a link to item o in the released review graph, the adversary would like to infer whether user u gave a rating for o in the original dataset. Let $Pr(u, o)$ denote the probability that user u rated item o in the original dataset, and let $Pr(C, o)$ be the unconditional probability that at least one user in anonymization group C rated o. We wish to calculate $Pr((u, o)|(C, o))$, the probability that user u rated item o, given that the edge exists in the released anonymized review graph.

By Bayes' Rule, we have that $Pr((u, o)|(C, o)) = Pr((C, o)|(u, o)) * Pr(u, o)/Pr(C, o)$. First note that $Pr((C, o)|(u, o)) = 1$ directly from our anonymization procedure, and we also have that $Pr(u, o) = p$. Furthermore, since we have assumed that each user has rated item o independently with probability p, we can find a bound on $Pr(C, o)$ as follows:

$$\begin{aligned} & Pr(C,o) = 1-(1-p)^k \\ = \quad & 1-(1-kp+\binom{k}{2}p^2-o(p^3)) \\ = \quad & kp-\binom{k}{2}p^2+o(p^3) \geq kp-\binom{k}{2}p^2 \end{aligned}$$

Combining these results, we get that

$$\begin{aligned} & Pr((u,o)|(C,o)) \leq \frac{1\cdot p}{kp-\binom{k}{2}p^2} \\ = \quad & \frac{1}{k}\left(\frac{kp}{kp-\binom{k}{2}p^2}+\frac{-\binom{k}{2}p^2}{kp-\binom{k}{2}p^2}\right)+\frac{1}{k}\left(\frac{\binom{k}{2}p^2}{kp-\binom{k}{2}p^2}\right) \\ = \quad & \frac{1}{k}+\frac{1}{k}\left(\frac{\frac{1}{2}k(k-1)p^2}{kp-\frac{1}{2}k(k-1)p^2}\right) \\ = \quad & \frac{1}{k}+\frac{1}{k}\left(\frac{(k-1)p}{2-(k-1)p}\right) \leq \frac{1}{k}+\frac{p}{2-kp} \end{aligned}$$

□