PRIVACY-PRESERVING DIMENSIONALITY REDUCTION-BASED COLLABORATIVE FILTERING

İbrahim YAKUT

Master of Science Thesis

Computer Engineering Program

May, 2008

JÜRİ VE ENSTİTÜ ONAYI

İbrahim YAKUT'un "**Privacy-Preserving Dimensionality Reduction-Based Collaborative Filtering**" başlıklı **Bilgisayar Mühendisliği** Anabilim Dalındaki, Yüksek Lisans Tezi 15.04.2008 tarihinde, aşağıdaki jüri tarafından Anadolu Üniversitesi Lisansüstü Eğitim-Öğretim Yönetmeliğinin ilgili maddeleri uyarınca değerlendirilerek kabul edilmiştir.

	Adı Soyadı	**İmza**
Üye (Tez Danışmanı)	**: Yard. Doç. Dr. HÜSEYİN POLAT**	**.....**
Üye	**: Doç. Dr. YUSUF OYSAL**	**.....**
Üye	**: Yard. Doç. Dr. NİHAT ADAR**	**.....**

Anadolu Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ..... tarih ve ..... sayılı kararıyla onaylanmıştır.

Enstitü Müdürü

ABSTRACT

Master of Science Thesis

PRIVACY-PRESERVING DIMENSIONALITY REDUCTION-BASED COLLABORATIVE FILTERING

İbrahim YAKUT

Anadolu University
Graduate School of Sciences
Computer Engineering Program

Supervisor: Asst. Prof. Dr. Hüseyin POLAT
2008, 86 pages

Collaborative filtering (CF) systems are widely used by many e-commerce sites. However, they fail to provide privacy measures. A significant amount of internet users do not feel comfortable to give their data for CF purposes due to privacy concerns. That is why it becomes a challenge to collect truthful and dependable data to perform CF services. Researches show that privacy concerns differ from user to user. Therefore, users might decide to hide their private data differently. Providing CF services on variably masked data is challenging. Two parties may need to combine their data for CF purposes for better recommendations. However, they do not want to integrate them due to privacy, legal, and financial reasons. If privacy measures are provided, they can combine their data. The question is then how they can offer CF services on integrated data without violating their privacy.

In this thesis, first, solutions are proposed to offer CF services based on Eigentaste algorithm without violating individual users' privacy. Second, it is shown how to provide recommendations using singular value decomposition (SVD)-based algorithms from inconsistently perturbed data. Finally, it is investigated how to achieve SVD-based CF on distributed data between two parties while preserving their privacy. To evaluate the overall performance of the proposed solutions, experiments are conducted using real data sets collected for CF purposes. The proposed schemes are analyzed in terms of accuracy, privacy, and additional costs. After explaining the solutions, conclusions are drawn and future directions are presented.

Keywords: Collaborative Filtering, Dimensionality Reduction, Singular Value Decomposition, Principal Component Analysis, Privacy

ÖZET

Yüksek Lisans Tezi

GİZLİLİĞİ KORUYARAK BOYUT İNDİRGEME TABANLI İŞBİRLİKÇİ FİLTRELEME

İbrahim YAKUT

Anadolu Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Yard. Doç. Dr. Hüseyin POLAT
2008, 86 sayfa

İşbirlikçi filtreleme (İF) sistemleri birçok elektronik ticaret sitesi tarafından kullanılmaktadır. Fakat bu sistemler gizlilik ölçütlerini sağlamada yetersiz kalmaktadırlar. Birçok internet kullanıcısı gizlilik endişelerinden dolayı işbirlikçi filtreleme amacıyla kendi bilgilerini rahatlıkla paylaşamamaktadırlar. Bu da İF servislerini yerine getirmek için doğru ve güvenilir veri toplanmasını güçleştirmektedir. Araştırmalar gizlilik endişelerinin kullanıcıdan kullanıcıya farklılık gösterdiğini göstermektedir. Bu yüzden kullanıcılar kendi gizli bilgilerini farklı şekillerde gizlemeyi tercih edebilirler. Bu şekilde saklanmış veriler üzerinde İF servislerini sağlamak çözülmesi güç bir problem haline gelmektedir. Bazı durumlarda iki elektronik ticaret firması verilerini biraraya getirip daha güvenilir ve doğru öneriler üretmek isteyebilir. Fakat bu firmalar gizlilik, hukuki ve finansal sebeplerden dolayı verilerini birleştirmek istemezler. Eğer gizlilik ölçütleri sağlanırsa bu firmalar verilerini birleştirebilirler. Bu durumda sorun onların gizliliklerine zarar vermeden birleştirilmiş veri üzerinden İF hizmetlerini nasıl gerçekleştirecekleridir.

Bu çalışmada ilk olarak kişisel gizliliğe zarar vermeden Eigentaste algoritmasına dayalı İF hizmetleri sunmak için çözümler ileri sürülmüştür. İkinci olarak, farklı yollarla saklanmış veriden tekil değerlerine ayrıştırma (TDA) tabanlı algoritmalar kullanılarak nasıl öneri üretileceği gösterilmiştir. Son olarak, dağıtık veri üzerinden iki taraf arasında tarafların gizliliklerini koruyarak TDA tabanlı işbirlikçi filtrelemenin nasıl gerçekleştirileceği araştırılmıştır. Öne sürülen çözümlerin performanslarını ölçmek için İF amacıyla toplanmış gerçek veriler kullanılarak deneyler yapılmıştır. Önerilen yöntemlerin doğruluk, gizlilik ve ek maliyet analizleri yapılmıştır. Çözümler anlatıldıktan sonra sonuçlar çıkarılmış ve öneriler sunulmuştur.

Anahtar Kelimeler: İşbirlikçi Filtreleme, Boyut İndirgeme, Tekil Değerlerine Ayrıştırma, Temel Bileşenler Analizi, Gizlilik

ACKNOWLEDGMENTS

I would like to thank to my thesis advisor Assist. Prof. Dr. Huseyin POLAT for his guidance, encouragement, and especially for his patience. His support and endurance contributes greatly to the quality of this study. It is pleasure and fruitful to work with him. Also, I would like to thank my fellow workers Cihan KALELI and Muzaffer DOGAN for their scientific and technical support. Finally, I would like to thank my wife for her moral contributions all the time.

İbrahim YAKUT

May, 2008

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Progression in the Internet and information technologies has been brought about some problems. The main problem is information overload. To overcome this problem, information filtering techniques are proposed. Collaborative filtering (CF) is a recent technique used for filtering and especially providing recommendations. By this technique, e-commerce sites provide recommendations to users about products for sale considering like-minded users' preferences. By providing CF services, online vendors increase the amount of the sales while contributing users' purchase process.

In order to perform CF, user-product databases including ratings of products either users already bought or rated according to their tastes or interests are required. However, due to privacy risks, collecting this type of data from users becomes a problem. Significant amount of users may not want to share their personal data at all. They might decide to share selectively or untruly in consideration of their individual privacy. To provide a more appropriate recommendation, data quality among truthfulness and comprehensiveness of several kinds of user profiles must be considered as an inevitable factor. To preserve their privacy, users first mask their data and send the perturbed data to a CF site or a data collector.

Users have different levels of concerns about their privacy. Data sensitivity and its value might differ from a user to another. To achieve required levels of privacy, users might decide to perturb their data variably. When each user disguises her data differently, it becomes a challenge for CF systems to still offer predictions based on such variably masked data.

In some applications, datasets are desired to be used as integrated by different data holders in order to provide better recommendations. Combining data is advantageous for online vendors because it is more likely to produce accurate and dependable recommendations on integrated data. However, due to privacy, legal, and financial reasons, they fail to integrate their data. If privacy measures are introduced, they can combine their data without revealing them to each other. In this thesis, the answers of the following questions are looked for:

How can CF systems provide recommendation with decent accuracy on masked data without greatly exposing users' privacy? Is it still possible to perform CF services based on variably masked data? How can data owners integrate their data for better CF purposes while preserving their privacy?

In Section 1.1, CF is explained while privacy-preserving collaborative filtering (PPCF) is introduced in Section 1.2. While necessary definitions in the thesis are introduced in Section 1.3, CF process is discussed in Section 1.4. After explaining privacy-preserving methods in Section 1.5, organization of the thesis is presented in Section 1.6.

1.1 Collaborative Filtering

Collaborative filtering (CF), one of the information filtering techniques, has been recently grown. CF concept was firstly appeared as a need for filtering number of e-mail messages came up at Xerox Palo Alto Research Center, in early 1990s. For this purpose, Tapestry system was designed and worked well for the small community of users who also knew each other [9]. CF is technique mainly used for providing referrals about an item such as e-mail, book, movie, and so on to any user using other users' preferences.

GroupLens Research, researchers from University of Minnesota, dealt with CF issues as scientific problem firstly. Usenet news is one of the earliest and largest bulletin board systems whose value is being severely diminished by the volume of low quality and uninteresting information posted in its newsgroups [35]. To make Usenet useful again, GroupLens introduce a CF system, which filters information according to neighborhood constructed by correlation between users [47]. Shardanand and Maes [52] modify GroupLens algorithm to recommend music by Ringo, which uses a word of mouth recommendation mechanism. The terminology "social information filtering" was used instead of CF by [52]. Ringo determines the similarity of users based on user rating profiles. Similarly, Belcore Video Recommender expands upon the same algorithm [26]. Maltz and Ehrlich [34] employ an approach using the term active CF, where recommendations are triggered by other peer actions by providing pointers to recommended documents. Several CF systems have been designed and

implemented depending on neighborhood-based methods since early 90's [5, 28, 48, 53].

CF techniques have been proven to provide satisfying recommendations to users [26, 52]. MovieLens is a movie recommendation system based on GroupLens technology [36]. Recommendation Tree (RecTree) is one method using divide-and-conquer approach to improve correlation-based CF and performing clustering on movie ratings from users [14]. WebWatcher has been designed for assisting information searches on the World Wide Web [4]. WebWatcher suggests users which hyperlinks would lead to the information that users want. The general function serving as the similarity model is generated by learning from a sample of training data logged from users. Yenta is a multi-agent matchmaking system implemented with the clustering algorithm and the referral mechanism [21].

Many methods, algorithms, and models have been proposed to resolve the similarity decisions in CF-based recommendation systems. One of the most common methods to determine the similarity is the cosine angle computation. Amazon.com recommendation system [33] uses this cosine measure to decide the similarity between every two items bought by each customer and to establish the item matrix, which contains item-to-item relationships. Several algorithms that combine the knowledge from artificial intelligence [37], Bayesian network [15], and other fields, have also been implemented in the recommendation systems. Genetic algorithm along with naïve Bayes classifier (NBC) is to define the relationships among users and items [30]. Genetic algorithm first completes clustering for discovering relationships among system users to find the global optimum. On the other hand, NBC defines the association rules of items. Then, similarity decisions would be performed to match the clusters of users or clusters of items, and the system can decide the final user profiles. The user profiles only consist of associated rules. To improve the recommendation quality of the generalized CF, demographic data is utilized with singular value decomposition (SVD) [54].

Two popular approaches, the coefficient correlation computation and the nearest-neighbor algorithm, have their limitations on scalability and sparsity.

Clustering [11], principal component analysis (PCA) [22], SVD [50], and discrete wavelet transform (DWT) [49] are introduced to CF-based recommendation systems to break these barriers. Eigentaste and genetic algorithms enable the constant time computations for online processes. Item-based CF algorithms are proposed to further decrease the computation time [33].

CF schemes are proposed for both server-based and distributed environment. To provide more effective and more adaptive recommendations, Cho et al. [16] propose a method that forms dual recommender groups and it then analyzes each group's influence on the target customers for the target product categories. There is also a fully distributed CF method that is self-organizing and operates in a distributed way [55]. It is a promising technique to facilitate filtering for relevant multimedia data in peer-to-peer (P2P) networks.

The CF algorithms can be examined in two general classes of collaborative filtering algorithms: Memory-based algorithms operate over the entire user database to make predictions. These systems generally depend on statistical techniques. Model-based CF, in contrast, uses the user database to estimate or learn a model, which is then used for predictions. The model building process is performed by different machine learning algorithms such as SVD, PCA, Bayesian network, clustering, and rule-based approaches. Model-based algorithms consist both online and offline computation parts.

CF has many important applications in e-commerce, direct recommendations, and search engines [12, 13]. Via CF, users can get referrals about many of their interests and activities; including, but not limited to restaurants, movies, books, jokes, and interesting things to do in a city. CF systems are used on the Internet to help consumers find the products they wish to buy at e-commerce sites [50]. A CF system on Amazon.com (www.amazon.com) suggests books to users based on other books the customers have told Amazon they like [33]. This recommendation system incorporates a matrix of the item similarity. Launch, music on Yahoo, Cinemax.com, Moviecritic, TV Recommender, Video Guide and the suggestion box, and CDnow.com are other successful examples of CF-based recommendation systems in the entertainment domain.

1.2 Privacy-Preserving Collaborative Filtering

Data mining (DM) is the science of extracting useful information from large data sets or databases [24]. Large size of data can be obtained from especially electronic transactions such as e-commerce, e-surveys, or integration of databases of two or more organizations. Proposed DM techniques are grouped into knowledge discovery and prediction. Knowledge discovery provides explicit information that has a readable form and can be understood by a user. Forecasting provides predictions of future or users' preferences.

Several types of DM methods are proposed and experienced up to now. But, they are a great threat to individual privacy. For this reason, researchers tend to provide DM methods caring about users' privacy, which is called privacy-preserving data mining (PPDM). PPDM has started receiving increasing attention after the works by Aggrawal and Srikant [2] and Lindell and Pinkas [32]. In PPDM studies, privacy is preserved mainly in several ways:

a) Using anonymization techniques such as *k*-anonymity
b) Using randomization such as randomized perturbation and randomized response techniques.
c) Using cryptographic techniques such as homomorphic encryption, oblivious transfer protocol, and permutations.
d) Other techniques such as geometric transformations have been already experienced for providing privacy.

In such studies, data either centralized or held by two or more parties. Problems related to the latter data organization are also called secure multiparty computation (SMC) problems in the literature.

Privacy issues in CF services were first considered by Canny [12, 13]. He proposes two schemes in which users control all of their private data; a community of users can compute a public "aggregate" of their data without disclosing any individual user's data. Aggregate data is constructed employing homomorphic encryption, which allows sums of encrypted vectors to be computed and decrypted without exposing individual data. The schemes are based on distributed computation of a certain aggregate of all users' data, which is treated as public data.

Polat and Du study privacy-preserving collaborative filtering (PPCF) issues. In one of their studies, randomized perturbation techniques (RPT) are employed on memory-based algorithms [42]. Perturbed ratings by adding random numbers are collected from users and recommendations are then generated based on these data. Another study is to produce recommendations privately using SVD-based algorithms [43]. In such schemes, they show that RPT can be employed to protect privacy on a model-based CF algorithm. Another study is about effects of inconsistently masked data using RPT on CF with privacy [44]. Moreover, they propose CF schemes, which can be performed on partitioned data. They present a scheme for binary ratings-based top-*N* recommendation on horizontally partitioned data in which two parties own disjoint sets of users' ratings for the same items while preserving data owners' privacy [46]. They also propose a privacy-preserving protocol for CF grounded on vertically partitioned data (VPD) [45]. As a modification on randomization techniques in PPCF, Zhang et al. [58] introduce a two-way communication privacy preserving scheme in which users perturb their ratings for each item based on the server's guidance instead of using an item-invariant perturbation.

Another way of disguising users' personal data is obfuscation techniques. Berkovsky et al. [7] describe a decentralized CF model in which user profiles are stored at the client side. In this approach, some of the personal data is replaced by some other data which is either constant or drawn from some distribution. In their follow-up work [8], they propose a decentralized recommendation generation scheme that is based on a hierarchical neighborhood topology. More specifically, users are organized into groups managed by super-peers. To enhance privacy, the super-peers choose only a random subset of their peers to form the neighborhood of similar users. To protect individual peers' privacy within a peer-group, the obfuscation techniques can be used and also only a subset of peers can be queried. Moreover, Parameswaran and Blough [41] propose methods that permute each column independently to provide recommendations while protecting users' privacy without causing accuracy losses.

Lam et al. [31] discuss questions relating security and privacy issues in recommendation systems to draw the attention of the information and

communication security community. Ahmad and Khokhar [3] propose an architecture that attempts to restore user trust in e-commerce recommender services by introducing the notion of 'Distributed Trust'. This essentially means that instead of trusting a single server, a coalition of servers is trusted. Distributions of trust make the proposed architecture fault resilient and robust against security attacks.

This thesis focuses on the usage of randomized perturbation techniques based on centralized data in which users do not participate in the CF process. This framework is more suitable for systems that provide online CF services such as Amazon, Google, Yahoo travel etc., while peer-to-peer case is more suitable in community-based CF systems. It is investigated how to achieve SVD-based recommendations on variably masked data while preserving users' privacy. In addition to investigating PPCF on existing databases, it is also going to be investigated how two online vendors that hold disjoint sets of data can provide referrals based on the integrated data without greatly sacrificing their privacy.

1.3 Definitions

Collaborative filtering (CF): The method of making automatic predictions (filtering) about the interests of a user by collecting taste information from many users (collaborating).

Active user (a): A user or a customer who looks for predictions for products that she has not experienced yet while the rest of the users are the non-active users.

Target item (q): The type of item for which a is looking for referrals to.

Rating (Vote): A value shows the preference of a user about an item or product. The users express their preference about items by rating them. Ratings might be numerical or binary. They can be discrete or continuous. In binary rating users rate items as *like* (1) or *dislike* (0). For better description, users can use a scale of numerical values for voting. Smaller values indicate that users do not like the items. Greater liking is indicated by increasing values.

Recommendation: A predicted preference for available products, which is the goal of the CF systems.

Data collector (Server): The entity that gathers ratings of items from many users for filtering purposes, and provides CF services to active users based on the collected data, using various algorithms.

Data owner: An online vendor or company that holds ratings gathered from many customers and performs filtering services with other companies by sharing data.

1.4 Collaborative Filtering Process

The goal of CF systems is to predict the utility of items to *a* based on a database of user votes from a sample or population of other users. Both *a*'s data and other users' data are taken as input for CF process, while prediction about *q* for *a* provided as an output as shown in Figure 1.1. The fundamental assumption is that if user u_1 and user u_2 rate j_s items similarly, they share similar tastes, and hence will rate other items similarly. Approaches to CF differ in how they define a "rating", j_s, and "similarly".

CF systems perform two types of tasks [50]. One of them is *prediction* of how much *a* will like *q*. The systems either tell how much a single item will be liked, or merely whether the item will be liked or disliked, in binary fashion. The other task is recommendation of a sorted list of items for *a*. This is known as *top-N recommendation* (*TN*). The system first forms a neighborhood for *a*, then focuses on items that the neighbors rated, and selects a list of *N* items that should be liked by *a.*

1.5 Privacy-Preserving Methods

To protect users' privacy, some methods such as randomization, anonymization, and cryptographic methods are employed. In this study, mainly randomized perturbation techniques (RPT), homomorphic encryption, and oblivious transfer protocol are utilized. In this section, these techniques are introduced.

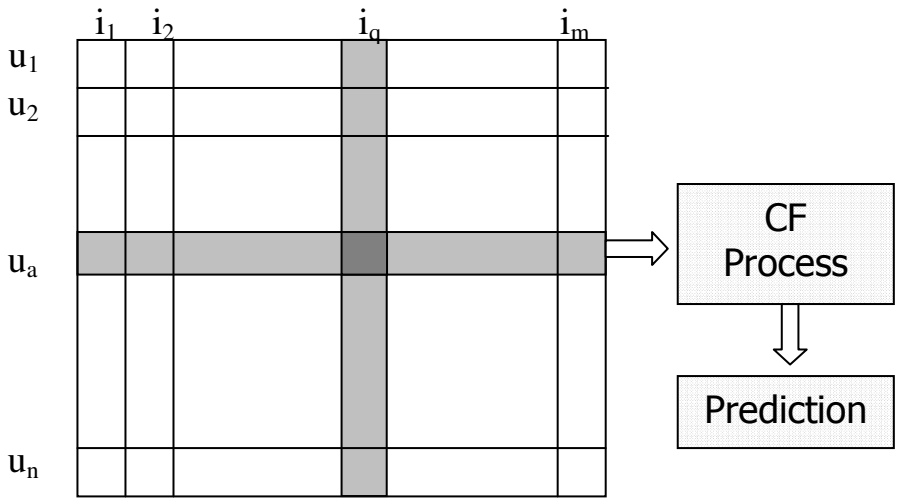

Figure 1. 1 Collaborative Filtering Process

1.5.1 Randomized Perturbation Techniques

A growing body of PPDM techniques is adopting randomization as a primary tool to "hide" information. The RPT attempts to preserve privacy of the data by modifying values of the sensitive attributes using a randomized process [42]. Simply by adding a number v, which is drawn from a certain distribution, the owner of a dataset returns a value $x_i + v$, where x_i is the original data. The n original data values $x_1, x_2, \ldots, x_n$ are viewed as realizations of n independent and identically distributed random variables x_i, where $i = 1, 2, \ldots, n$, each with the same distribution as that of a random variable X. To perturb the data, n independent samples $v_1, v_2, \ldots, v_n$, are drawn from a distribution V. Each owner or, say for this study, user provides the perturbed values to the data collector as in Figure 1.2. Commonly used distributions are the uniform and Gaussian distributions. While basic parameter, standard deviation, is taken as σ_u in Gaussian distribution, uniform distribution is taken over an interval $[-\alpha, \alpha]$, where α is a constant number and $\alpha = \sqrt{3}\sigma_u$.

1.5.2 Homomorphic Encryption Schemes

Homomorphic encryption is a semantically-secure public-key encryption, which has the additional property that there exists an encryption $E(A*B)$ such that

$$E(A) * E(B) = E(A*B),$$

where * is either addition or multiplication, A and B are original values, while $E(A)$ and $E(B)$ are given encrypted data.

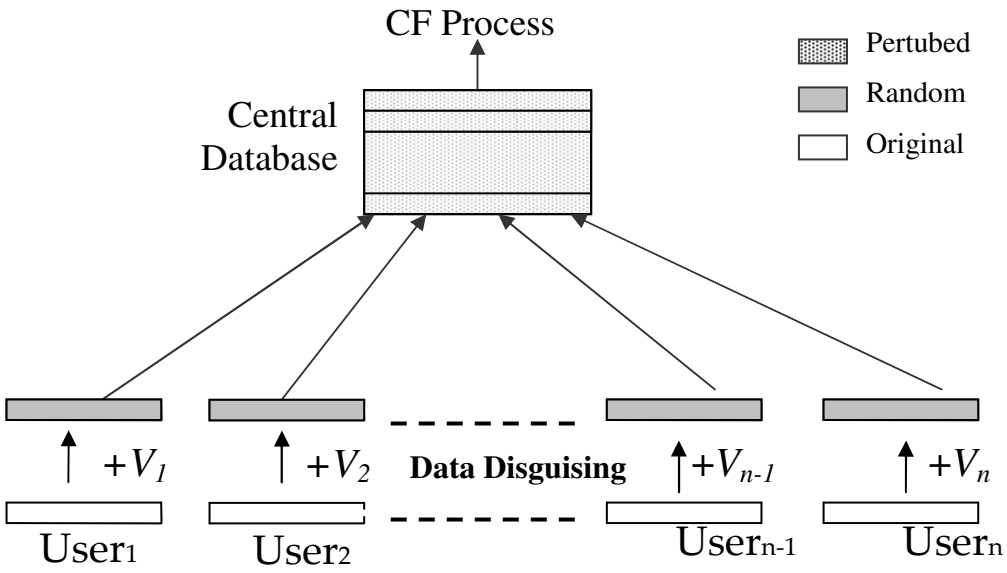

Figure 1. 2 Collaborative Filtering Using Randomized Perturbation Techniques

Homomorphic encryption systems can be very useful because it allows a third party to operate on encrypted values without knowing the plaintext. Thus, it can provide a setting for operation on encrypted values by someone else such that only the person who knows the key can decrypt the result. Examples of them are the systems proposed by Benaloh [6], Naccache and Stern [38], Okamoto and Uchiyama [39], and Paillier [40].

1.6 Organization of the Thesis

In the following chapter, privacy-preserving Eigentaste-based CF is studied. While schemes about privacy-preserving SVD-based CF on variably masked data are proposed in Chapter 3, it is investigated how referrals can be generated using SVD-based CF algorithms on distributed data in Chapter 4. Finally, in Chapter 5, conclusions are drawn and future research directions are introduced.

2. PRIVACY-PRESERVING EIGENTASTE-BASED COLLABORATIVE FILTERING

Many algorithms have been employed for CF purposes; and Eigentaste [22] is one of them. Eigentaste has a constant online computation time cost and provides flexibly usage of several clustering algorithms. Without privacy concerns, it is an easy task to offer Eigentaste-based CF services to customers. However, due to privacy concerns, customers refuse to contribute their ratings at all; or they might decide to give false data. Providing truthful referrals based on such inadequate and false data is impossible. If privacy measures are introduced, users feel more comfortable to give their true data. Therefore, providing privacy measures is vital for collecting truthful data and producing accurate recommendations.

In this chapter, how to achieve Eigentaste-based CF tasks without greatly exposing users' privacy is discussed. To protect users' privacy, RPT is utilized. Original Eigentaste algorithm is modified and/or simplified in such a way to provide private referrals efficiently with decent accuracy. The proposed schemes are investigated in terms of privacy. To evaluate the overall performance of proposed schemes, experiments are conducted using real data sets. Then, obtained outcomes are analyzed and finally some suggestions are provided.

After presenting introductory information in Section 2.1, PCA and Eigentaste-based CF algorithms are introduced in Section 2.2. In Section 2.3, privacy-preserving Eigentaste-based CF is studied, while privacy and overhead cost analysis of the proposed algorithms are done in Section 2.4. Experiments are presented in Section 2.5. Finally, the chapter is summarized in Section 2.6.

2.1 Introduction

To provide accurate referrals efficiently in terms of computational complexity, many algorithms have been proposed. Eigentaste [23] is a CF algorithm that uses universal queries on a common set of items and applies PCA. It requires constant time to compute predictions, given a database of n users. Some of the computations in Eigentaste can be done offline, while

recommendation computations are done online. Recursive rectangular clustering (RRC) is employed to cluster users

There is a great potential for individuals to share all kinds of information about places and things to do, see and buy; but the privacy risks are many and severe. E-commerce personalization poses various privacy risks and many users are concerned about the privacy of their personal information [18]. Due to such concerns, it becomes a challenge to collect data, especially truthful and trustworthy data, for CF purposes. Providing privacy measures plays a vital role to collect truthful data and to produce accurate referrals. *How can customers give their data for CF purposes without jeopardizing their privacy? Is it still possible to generate recommendations based on perturbed data?*

The main merit of Eigentaste algorithm is that it is a linear time algorithm and different kinds of clustering algorithms can be used with it. The goal is to provide referrals with privacy via an algorithm that can utilize various types of clustering algorithms in linear time. CF algorithms should be accurate and efficient [22]. Moreover, when privacy is an issue, they should preserve users' privacy, as well. On the other hand, privacy, accuracy, and efficiency are conflicting goals. Increasing one or two of them decreases the other(s). Therefore, it is desired to propose solutions to find equilibrium among them while achieving privacy-preserving Eigentaste-based CF. Proposed schemes are analyzed in terms accuracy, privacy, and efficiency. Experiments are conducted to assess the overall performance of the proposed schemes.

2.2 Principal Component Analysis and Eigentaste-based Collaborative Filtering

PCA is a closely-related factor analysis technique and reduces dimensionality by optimally projecting highly correlated data along a smaller number of orthogonal dimensions [22]. PCA facilitates dimensionality reduction for offline clustering of users and rapid computation of referrals. Goldberg et al. [22] apply eigen-analysis to solve for matrices E and Λ such that

$$C = E^T \Lambda E \tag{2.1}$$

and

$$\Lambda = ECE^T, \tag{2.2}$$

where C is correlation matrix, E is orthogonal matrix of eigenvectors of C, and Λ is diagonal matrix of eigenvalues of C. After finding eigenvectors, they keep principal eigenvectors only. The number of eigenvectors to retain depends on eigenvalues and it is small. If v eigenvectors are retained, data is projected along the first v principal eigenvectors:

$$x = AE_v^T, \tag{2.3}$$

where A is normalized matrix of users ratings of items in gauge set. The popular choice is to set v at 2, so that data are projected onto eigen-plane. Eigentaste algorithm can be described, as follows: Given an $n \times m$ matrix of raw ratings from n users and m items, Goldberg et al. [22] select k of these items to form a gauge set. They normalize the gauge set to produce A, which is an $n \times k$ matrix. Each rating is normalized by subtracting its mean rating over all users, and then dividing by its standard deviation. They define the global correlation matrix C over all users:

$$C = \frac{1}{n-1} A^T A, \tag{2.4}$$

where C is symmetric and positive definite. After projecting data onto eigenplane, users can be clustered. They implement RRC to cluster users. Each cell is treated as a cluster of neighbors in the eigen-plane. For each cluster, the mean for each non-gauge item is computed based on the number of users who rated that item. Sorting the non-gauge items in order of decreasing mean ratings yields a lookup table of recommendations for that cluster. The computations so far are done offline.

To compute recommendations online, a new user (an active user, a) contribute her ratings for all items in the gauge set. Using principal components, the data entered is projected onto the eigen-plane. After finding representative cluster, recommendations are presented with the help of lookup table.

2.3 Privacy-Preserving Eigentaste-based Collaborative Filtering

The goal is to provide private referrals efficiently with decent accuracy. Efficiency can be explained, as follows: Additional online costs like storage,

communication, and computation costs should be small and negligible because offline costs are not critical to overall performance. Accuracy can be defined, as follows: Predictions computed based on perturbed data should be close to true rating values. And finally, although it is a challenge to define privacy clearly, it can be explained in the context of CF, as follows: The server or the data collector should not be able to learn the true ratings and the rated items of users including active users. Users do not want to reveal their true ratings about products they bought or showed interest. Moreover, it might be more damaging to disclose the purchased products than revealing the true votes. For example, nobody wants to reveal that she bought pornographic magazine or visited pornographic sites. Therefore, besides perturbing true ratings, users disguise unrated items' cells, as well. Since privacy, accuracy, and efficiency are conflicting goals, it is aimed to find a good balance between them in this study.

2.3.1 Modified Eigentaste-based Collaborative Filtering Algorithm

Original Eigentaste algorithm was modified in such a way to achieve the goals explained previously, as follows: To normalize ratings into z-scores, user-mean votes are used rather than item-mean ratings. Each user is able to compute z-scores of her ratings without the help of other users if user-mean votes are used for normalization, as follows:

$$z_{uj} = \frac{v_{uj} - \overline{v_u}}{\sigma_u}, \tag{2.5}$$

where v_{uj} is the true rating of user u on item j, $\overline{v_u}$ is the mean rating of user u, σ_u is the standard deviation of user u's ratings, and z_{uj} is the z-score value of user u on item j.

Besides employing the RRC, as done in Eigentaste algorithm, there are alternative clustering algorithms like *k*-means, fuzzy-*C* means, etc. It is possible to apply different clustering algorithms to achieve privacy while providing accurate referrals efficiently. *K*-means clustering algorithm is proposed to use to cluster the projected data into 57 clusters because recent results show that PCA components provide solutions to *k*-means clustering [19, 29].

In Eigentaste, to create the lookup table, non-gauge items' means are used. In this algorithm, z-score values' means is used to create the lookup table. Once the cluster of a is found, the mean value of z-scores of the target item (q) is provided to a. Then, a can de-normalize it and finds the prediction (p_{aq}) for q, as follows:

$$p_{aq} = \overline{v_a} + \sigma_a \times \overline{z_q}, \tag{2.6}$$

where $\overline{v_a}$ is the mean rating of a's ratings, σ_a is the standard deviation of a's ratings, and $\overline{z_q}$ is the mean z-score value of q.

In addition to employing lookup table, alternative methods can be applied to calculate the referrals online. Using lookup table is advantageous due to small computation time. On the other hand, it might be possible to increase accuracy while sacrificing on computation complexity. It is possible to use well-known memory-based CF algorithms to compute the referrals online. After finding a's representative cluster, predictions can be found by computing the weighted sum of co-rated items between a and users in that cluster.

2.3.2 Data Masking

To disguise the private data, different approaches can be employed. One common solution to conceal the private is to employ randomized perturbation techniques (RPT). Although information from each individual user is masked, if the number of users and/or items is significantly large, the aggregate data of these users can be estimated with decent accuracy. For computations on aggregate data, it can be still generated meaningful outcomes without knowing the exact values of individual data items. RPT has been employed to provide PPCF services using memory-based and SVD-based algorithms [42, 43]. Since Eigentaste-based CF is based on aggregate information rather than individual data items, the RPT can also be applied to it. Therefore, in this study, the RPT and Eigentaste CF algorithm is integrated to provide accurate recommendations with privacy. After generating random data, users disguise their data by adding those random data to their corresponding ratings vector's cells. The data masking process can be summarized, as follows:

1. Each user u computes z-score values of their ratings.

2. Users and the server decide standard deviation range upper bound γ, distribution type determining threshold θ, and maximum percentage of cells going to be filled over unrated cells (δ values).
3. Each user u selects the standard deviation of the random numbers (σ_u) uniformly randomly over the range [0, γ]. Users generate the random numbers from a distribution with μ being 0 and σ_u. Each user u then uniformly randomly selects a random number r_u over the range [0, 1]. If $r_u \leq \theta$, uniform perturbing data; otherwise Gaussian perturbing data is used for data perturbation. Each user u finally uniformly randomly selects an integer x_{er} over the range [0, δ]. x_{er} is defined as the percentage of unrated items' cells to be filled with noise data. They then randomly select the x_{er} percent of their unrated items' cells to be disguised.
4. Each user u creates m_u number of random numbers using uniform or Gaussian distribution with the selected σ_u values, where m_u is the number of noise data required to disguise the user u's private data including ratings and unrated cells. Such value depends on the number of ratings and the number of unrated items' cells to be disguised.
5. Users mask their private data by adding noise data to each of the cells to be perturbed. Finally, they send the disguised data to the server.

To obtain a balance between accuracy, privacy, and efficiency, γ, θ, and δ values can be adjusted.

2.3.3 Eigentaste-based Collaborative Filtering with Privacy

After collecting perturbed data from users, the server creates a disguised user-item matrix, D', which is an $n \times m$ matrix including disguised z-scores collected from n users for m items. k of those m items are selected to form the gauge set, A', which is an $n \times k$ matrix. Note that all users rated all items in gauge set. Therefore, A' is a dense matrix. The server then starts providing recommendations based on this perturbed matrix.

To provide recommendations, the server first estimates correlation matrix from perturbed data offline, as follows:

$$C' = \frac{1}{n-1} A'^{T} A', \tag{2.7}$$

where C' is the estimated correlation matrix from perturbed data. It should be showed that the server can estimate C' from masked data. The entries other than the diagonal ones are computed, as follows:

$$\begin{aligned} &\frac{1}{n-1}(A'^{T}A')_{fg} = \frac{1}{n-1}\sum_{u=1}^{n}(z_{uf}+r_{uf})(z_{ug}+r_{ug}) = \\ &\frac{1}{n-1}\left[\sum_{u=1}^{n} z_{uf} z_{ug} + \sum_{u=1}^{n} z_{uf} r_{ug} + \sum_{u=1}^{n} z_{ug} r_{uf} + \sum_{u=1}^{n} r_{uf} r_{ug}\right] \approx \frac{1}{n-1}\sum_{u=1}^{n} z_{uf} z_{ug}, \end{aligned} \tag{2.8}$$

where n is the number of users, f and g show the row and column numbers, respectively, and $f \neq g$. Since random values r_{uf}s and r_{ug}s are independent and drawn from a distribution with $\mu = 0$, the expected value of $\frac{1}{n-1}\sum_{u=1}^{n} r_{uf} r_{ug}$ is 0. Similarly, the expected values of $\frac{1}{n-1}\sum_{u=1}^{n} z_{uf} r_{ug}$ and $\frac{1}{n-1}\sum_{u=1}^{n} r_{uf} z_{ug}$ are 0. However, since the scalar product is computed between the same vectors for the diagonal entries ($f = g$), they can be estimated, as follows:

$$\begin{aligned} &\frac{1}{n-1}(A'^{T}A')_{ff} = \frac{1}{n-1}\sum_{u=1}^{n}(z_{uf}+r_{uf})(z_{uf}+r_{uf}) = \\ &\frac{1}{n-1}\left[\sum_{u=1}^{n} z_{uf}^{2} + 2\sum_{u=1}^{n} z_{uf} r_{uf} + \sum_{u=1}^{n} r_{uf}^{2}\right] \approx \frac{1}{n-1}\left[\sum_{u=1}^{n} z_{uf}^{2} + \sum_{u=1}^{n} r_{uf}^{2}\right]. \end{aligned} \tag{2.9}$$

Again, the expected value of $\frac{1}{n-1}\sum_{u=1}^{n} z_{uf} r_{uf}$ is 0 due to the same reason. However, since $\frac{1}{n-1}\sum_{u=1}^{n} z_{uf}^{2}$ values are only needed for diagonal entries, it is necessary to get rid of $\frac{1}{n-1}\sum_{u=1}^{n} r_{uf}^{2}$ in (2.9), as follows, assuming $n \approx n-1$ for large n values:

$$\frac{1}{n-1}(A'^{T}A')_{ff} = \frac{1}{n-1}\left[\sum_{u=1}^{n} z_{uf}^{2} + \sum_{u=1}^{n} r_{uf}^{2}\right] - \sigma_{r}^{2} \approx \frac{1}{n-1}\sum_{u=1}^{n} z_{uf}^{2}, \tag{2.10}$$

where σ_r is the average standard deviation of random numbers. As explained before, with increasing n or in the long run, since random numbers are independently generated and drawn from distributions with μ being 0, the relative errors due to random numbers will converge to zero.

Since users disguise their private data using random numbers generated from some distribution with μ being 0 and σ_u, the server is able to estimate C' from perturbed data offline, as explained previously. After estimating C', the server can apply eigen-analysis and estimate E', which is a $k \times k$ orthogonal matrix of eigenvectors of C'. It only keeps the v principal eigenvectors. v is usually small and generally set to 2. It then projects the data along the first $v = 2$ principal eigenvectors, as follows:

$$x' = A' E_v'^T , \tag{2.11}$$

where x' is an $n \times 2$ estimated matrix. The entries of x' can be estimated for $i = 1, 2,..., n$ and for all $j = 1\ ,\ 2$, as follows, where e values represent the entries of E and R values represent the contributions of random values to true values of eigenvectors because they are estimated from perturbed data:

$$x'_{ij} = \sum_{l=1}^{k} (z_{il} + r_{il})(e_{lj} + R_{lj}) = \sum_{l=1}^{k} \left(z_{il} e_{lj} + z_{il} R_{lj} + r_{il} e_{lj} + r_{il} R_{lj} \right) \approx \sum_{l=1}^{k} z_{il} e_{lj} . \tag{2.12}$$

Due to the same reasons explained previously, the expected values of the last three summations in (2.12) will converge to zero. Since summations are computed over k, with increasing number of gauge items (k), the relative errors due to random numbers will become smaller. Therefore, the server is able to project the disguised data along with the first two principal eigenvectors. After projecting the masked data onto eigen-plane, the server clusters the projected data into various clusters. It is preferred to use k-means clustering over the RRC because recent results show that PCA components provide solutions to k-means clustering [19, 29]. The server finally generates lookup tables offline. For each cluster, it finds the mean of the z-scores for each non-gauge items. Such average z-score values are stored in corresponding lookup tables. The computations so far are conducted offline. Once the server creates the model (generating clusters and lookup tables) offline, it can start providing CF services to users. When a wants recommendations, she sends her z-score values of items in gauge set together with a query to the server. Three different methods are proposed to preserve a's privacy and they are explained in the following subsection. The server can easily project a's data and find her corresponding cluster when a uses the first method to protect her privacy because a sends her true z-scores vector together with random

vectors. When a uses the second or the third scheme to disguise the z-scores of items in the gauge set, the server first can project her z-scores, as follows, for all j = 1, 2 :

$$x'_{aj} = \sum_{l=1}^{k} (z_{al} + r_{al})(e_{lj} + R_{lj}) \tag{2.13}$$

Since random numbers are drawn from some distribution with μ being 0, the following approximation of (2.13) can be written:

$$x'_{aj} = \sum_{l=1}^{k} (z_{al} + r_{al})(e_{lj} + R_{lj}) \approx \sum_{l=1}^{k} z_{al} e_{lj} \tag{2.14}$$

After projecting a's data, the server can now find the representative cluster. It finally sends the required average z-score for the target item q to a, who can de-normalize such value and estimates prediction for q.

2.3.4 Preserving Active Users' Privacy

Three methods are proposed to protect active users' privacy, in this study. The first method (M1) is based on the 1-out-of-n Oblivious Transfer protocol [10, 20], which refers to a protocol where at the beginning of the protocol one party, Bob has n inputs X_1, ..., X_n and at the end of the protocol the other party, Alice, learns one of the inputs X_i for some $1 \leq i \leq n$ of her choice, without learning anything about the other inputs and without allowing Bob to learn anything about i. By combining efficient protocols, the 1-out-of-n Oblivious Transfer protocol could be achieved with poly-logarithmic (in n) communication complexity. When a uses this method, she sends Y -1 randomly generated vectors and her true z-scores vector including z-scores of items in the gauge set to the server. After finding referrals, the server uses the 1-out-of-n Oblivious Transfer protocol to send them to a that receives only one prediction instead of Y recommendations. Although this method is advantageous for the server in terms of business purposes, it introduces additional computation costs due to the 1-out-of-n Oblivious Transfer protocol. Moreover, it also introduces additional communication costs.

In the second and the third methods, active users also perturb their data as other users do. Note that a only sends the ratings for the items in the gauge set to the server. In the second method (M2), a generates k random numbers drawn from

a distribution with μ being 0. a then adds them to her ratings in the gauge set, and sends the disguised ratings in the gauge set to the server to get recommendations. In the third method (M3), a creates m_a random numbers, where m_a represents the number of rated items by a. After that, a uses the first k random numbers to disguise her ratings in the gauge set. After perturbing ratings in gauge set, a sends them to the server. The last two solutions do not introduce additional communication costs. Additional computation costs are negligible. Due to random numbers, they make accuracy worse compared to the first solution.

2.3.5 Providing Referrals

After estimating the correlation matrix C' from the masked data, the server clusters the users and finds the lookup tables for each cluster offline. When an active user a wants a prediction for a single item q or top-N recommendations for her unrated items, she sends her data in a private manner to the server together with a query. The server first projects a's data and finds the representative cluster for a. Then, it finds the estimated value, which is called as P', from the corresponding lookup table. It sends it to a, who can now de-normalize P' and finds prediction p'_{aq} for q, as follows: $p'_{aq} = \overline{v_a} + \sigma_a \times P'$. The server will only be able to know the estimated value of P' because it does not have the mean rating and the standard deviation of a who is looking for prediction. Therefore, it will not be able to learn how much a likes or dislikes q. To provide top-N recommendations, a sends a query stating that she is looking top-N recommendations for N_a items, where $N < N_a < m - m_r$, and m_r is the number of items rated by a. The server then computes P' values for all N_a items and sorts them decreasingly. Finally, it selects the first N items and sends the list to user a as top-N recommendations. Since the same mean and standard deviation are used to de-normalize P' values, the server does not need them to find the sorted list.

2.4 Overhead Costs and Privacy Analysis

Offline costs are not critical to overall performance. Therefore, additional online costs rather than offline ones should be taken account in these schemes. Since privacy, accuracy, and efficiency are conflicting goals, providing privacy

measures makes accuracy and/or efficiency worse. How accuracy changes with privacy measures are shown in the subsequent section. It is now shown that how much additional cost introduced due to privacy concerns. The communication cost, in terms of number of communications, does not change due to privacy-preserving schemes when a protects its privacy using the second and the third methods. However, the amount of data to be sent increases due to the appended random values. The communication costs increase due to 1-out-of-n Oblivious Transfer protocol when a uses the first method for preserving privacy. Although the amount of data to be stored increases due to random values, the server stores the collected perturbed data into the same $n \times m$ matrix. Additional computation costs due to privacy concerns are also small. Moreover, data disguising is performed without the help of a third party. The only additional computation cost is the computations that the server conducts to get rid of the contribution of random values in diagonal entries of correlation coefficient matrix.

The server tries to figure out the actual values of the ratings and the rated items. The server wants to obtain true ratings from masked z-scores. Since users perturb their z-score values, it becomes difficult to obtain true ratings from perturbed z-scores. To obtain such values, the server needs the standard deviations and means of users' ratings, which are only known by the users. It should know the type of the perturbing data, as well. Since each user u employs uniform or Gaussian perturbing data based on θ and r_u, the server can guess that uniform or Gaussian perturbing data is used with probability θ and 1-θ, respectively. The standard deviations of random numbers are also critical. The server does not know such values, because they are uniformly randomly generated over the range [0, γ] by the users. It only knows the γ, not σ_u values. The privacy measure should indicate how closely the original value of an item can be estimated from the perturbed data. To quantify privacy introduced the RPT, various measures are employed [1, 2]. Privacy introduced due to uniform or Gaussian perturbing data is analyzed in [42] and can be similarly analyzed. As expected, different perturbing data provide various privacy levels. Moreover, with increasing level of perturbation (with increasing σ_u values), privacy improves due to increasing randomness. It can be said that the proposed schemes improve privacy because

the server does not know the means and the standard deviations of users'' true ratings, type of perturbing data, and the standard deviations of random numbers.

The proposed method can be analyzed for preventing the server from learning the rated items, as follows: The server does not know the rated items due to appended random numbers. However, it can guess the randomly selected unrated items' cells. The probability of guessing such cells for the server should be computed. The probability of guessing the correct x_{er} value for the server is 1 out of δ. After guessing such value, the server can figure out the number of filled unrated items' cells (d) with the help of empty cells in the perturbed vector because the number of empty cells equals 1-x_{er} percent of unrated items' cells. After finding d, the server can guess the randomly filled unrated items' cells. If the number of filled cells is m_u including rated items' cells and filled unrated items' cells, the probability of guessing d randomly selected unrated items' cells is 1 out of $C_d^{m_u}$, $C_d^{m_u}$ represents the number of ways of picking d unordered outcomes from m_u possibilities. Therefore, the probability of guessing the randomly selected unrated items' cells of one user is 1 out of $(\delta \times (C_d^{m_u}))$.

2.5 Experiments

To assess the overall performance of the proposed schemes, experiments were performed using real data. The outcomes are also analyzed and summarized.

2.5.1 Datasets and Evaluation Metrics

Jester dataset is used in these experiments. It is a well-known dataset. It is a web-based joke recommendation system, developed at the University of California, Berkeley [23]. In Jester, users rate a core set of jokes, and then receive recommendations about others that they should like. The database has 100 jokes and records of 17,988 users. Almost 50% of all possible ratings are present. The ratings range from -10 to +10 and the scale is continuous.

Several evaluation criteria for CF have been used in literature [12, 25, 52]. The obtained results from the experiments are analyzed using well-known error measures such as Mean Absolute Error (MAE) and Normalized Mean Absolute Error (NMAE). The MAE is a measure of the deviation of recommendations from

their true user-specified values [51]. If $p_1, p_{2,\ldots,} p_c$ are true ratings, and $p'_1, p'_2, \ldots, p'_c$ are predicted ratings from integrated data with privacy concerns, then $\{\xi_1, \xi_2, \ldots, \xi_c\} = \{p'_1 - p_1, p'_2 - p_2, \ldots, p'_c - p_c\}$ represents errors. MAE is computed, as follows:

$$\text{MAE} = \frac{\sum_{i=1}^{c} |\xi_i|}{c} \tag{2.15}$$

As used in Goldberg et al. [22], the other error measure, which is going to be used is NMAE, which can be defined, as follows, where r_{max} and r_{min} represent the maximum and the minimum ratings, respectively:

$$\text{NMAE} = \frac{\text{MAE}}{|r_{\max} - r_{\min}|} \tag{2.16}$$

2.5.2 Methodology

Firstly, the data set is randomly divided into training and test sets, where training set contains 9,000 users' ratings, while test set includes the remaining 8,988 users' ratings. These randomly selected 9,000 users' data are used as training data. For testing, 5,000 users are selected from the test set. For each test user (active user, *a*), 10 rated items are randomly selected from non-gauge items. The train and test users' ratings are normalized by converting them into z-scores using user-mean votes. The private data is perturbed, as explained previously. To obtain trustworthy results, data disguising is run for 100 times. Throughout the experiments, *k* is set at 10 (there are 10 items in gauge set), *v* at 2, and cluster users into 57 clusters using *k*-means clustering. For each test user, one of the test items' rating is withheld and prediction is tried to be found. Predictions are compared with true ratings. The MAE and the NMAE values are computed. The final overall average values are finally displayed.

2.5.3 Experimental Results

Note that user-mean votes are employed rather than item-mean votes to normalize the ratings into z-scores. Then, *k*-means clustering is used instead of recursive rectangular clustering. Finally, lookup tables are generated based on z-scores. Therefore, experiments are performed at first to evaluate the overall

performance of the modified and/or simplified Eigentaste algorithm. For this purpose, 9,000 users are used for training while 5,000 test users. After finding recommendations for each test user and test item, they are compared with the withheld true ratings. The MAE and the NMAE values are calculated. The final values are displayed together with the results for Eigentaste found by [22] in Table 2.1.

Table 2. 1 Eigentaste vs. Modified Eigentaste

	MAE	**NMAE**
Eigentaste	3.740	0.187
Modified Eigentaste	3.334	0.167

As seen from Table 2.1, modified Eigentaste gives better results than Eigentaste algorithm. The proposed variations improve accuracy by 10%. In the following, experiments are performed using modified Eigentaste while considering privacy concerns.

After evaluating the overall performance of the proposed alternative variations to the original algorithm, trials are then performed to assess the proposed privacy-preserving Eigentaste schemes. Since three different schemes are proposed to protect a's privacy, experiments are conducted to compare those three schemes in terms of accuracy. Again, 9,000 training and 5,000 test users are used. To disguise private data, the parameters θ, γ, δ are set at 0.5, 4 and 100, respectively. Predictions are found for withheld items for each a based on perturbed data, where a's privacy is protected using three different methods. Overall MAE and NMAE values are computed and displayed in Table 2.2, where M1, M2, and M3 represent the first, second, and third methods, respectively.

Table 2. 2 Comparing Methods for Protecting a's Privacy

	M1	**M2**	**M3**
MAE	3.3508	3.4710	3.4807
NMAE	0.1676	0.1735	0.1741

The first method gives the best results, as seen from Table 2.2. However, it is based on the 1-out-of-n Oblivious Transfer protocol, which introduces

additional communication and computation costs. The second and the third methods provide similar results. Compared to the first method, the additional costs they introduce are negligible. Therefore, the second method is used in the following experiments to protect *a*'s privacy. The MAE, 3.3508, for the first method is very close to the one for the modified Eigentaste, which is 3.334. This means that the errors due to model creation based on perturbed data is very small. However, the second and the third methods also give decent results.

To show how accuracy changes with varying numbers of users (n), experiments are performed, where n varies from 500 to 8,000. The same 5,000 test users are used. To protect *a*'s privacy, the second method is used. Again, parameters are set as θ at 0.5, γ at 4, and δ at 100. The MAE and the NMAE values are calculated and displayed in Table 2.3.

As expected and seen from Table 2.3, the results are getting better with increasing n values. The reason for this phenomenon can be explained, as follows: In estimation of the matrix C', the relative errors due to random numbers will converge to zero with increasing n values. Although the results are worse for n values less than 2,000, the results for n values bigger than or equal to 2,000 are promising. Therefore, it is possible to provide accurate recommendations without violating users' privacy when there are sufficient data.

Table 2. 3 Accuracy vs. Varying Numbers of Users (n)

n	**500**	**1000**	**2000**	**4000**	**8000**
MAE	4.678	4.242	3.832	3.6243	3.483
NMAE	0.234	0.212	0.192	0.181	0.174

To prevent the server from learning the rated items, the users perturb some of their randomly selected unrated items' cells by inserting random values. It is hypothesized that accuracy varies with different numbers of perturbed unrated items' cells. Therefore, experiments are conducted using 9,000 and 5,000 training and test users, respectively. Intentionally, δ is varied from 0 to 100. When δ is 0, users do not disguise any empty cells, while it is 100; the number of disguised cells is the most. With increasing δ, randomness increases. The MAE and the NMAE values are computed for various δ values and displayed them in Table 2.4.

Table 2. 4 Accuracy with Varying Percentage of Filled Cells (δ)

δ	0	35	70	100
MAE	3.4460	3.4567	3.4615	3.4710
NMAE	0.1723	0.1728	0.1730	0.1735

As expected, accuracy slightly becomes better with decreasing δ values due to diminishing randomness. However, the gain is very small. This can be explained with the density of Jester data set. This scheme gives acceptable results even with larger δ values. Therefore, these privacy-preserving Eigentaste-based schemes provide referrals with decent accuracy with varying δ values.

Accuracy and privacy also depend on some other factors other than n, δ, and the methods to protect a's privacy, such as the perturbing data, level of perturbation, σ selection methods, and so on. Since it has been shown that how accuracy and privacy vary with these factors when the RPT is employed [42], experiments are not conducted to evaluate the proposed schemes with varying of such factors.

2.5.3 Conclusions

The proposed schemes allow the users and the server to adjust the parameters of data perturbation methods to achieve required levels of privacy and accuracy. With increasing γ and δ values, more randomness is added to original data. With increasing randomness, privacy improves, while accuracy diminishes. Therefore, the server and the users should select γ and δ values in such a way to obtain a balance between them. Although these schemes provide recommendations with decent accuracy for γ and δ being 4 and 100, respectively, accuracy even further can be improved by using smaller γ and δ values. On the other hand, privacy diminishes with decreasing such values. Users and the server are able to decide the values of such parameters in such a way to achieve required levels of privacy and accuracy. Although Gaussian perturbing data slightly provides more privacy and accurate results than uniform perturbing data, the results are still very close to each other. The users can select either of them to mask their data while still obtaining good results in terms of accuracy and privacy.

Although three methods are proposed to protect *a*'s privacy, the second and the third methods should be preferred over the first one. When considering accuracy and efficiency together, the last two methods are better because they do not introduce additional online communication and computation costs while sacrificing little on accuracy. Kargupta et al. [27] state that when standard deviation (σ) of perturbing data is less than or equal to 1, actual data can be predicted from disguised data reasonably well. However, as seen the experiment results, proposed schemes can provide recommendations with decent accuracy when σ is bigger than 1.

2.6 Summary of the Chapter

It is shown that it is possible to achieve private recommendations using Eigentaste algorithm. Schemes are proposed to produce accurate referrals without jeopardizing users' privacy. The overall performance of the schemes is evaluated in terms of accuracy conducting various experiments based on real data. The outcomes of the experiments are analyzed. Moreover, the proposed schemes are analyzed in terms of additional costs and privacy. It will be studied whether other clustering methods can be employed or not besides *k*-means clustering. In the future, how to improve recommendation qualities by using well-known correlation-based CF algorithms after clustering will be investigated. Also, providing private recommendations using Eigentaste algorithm based on partitioned data between various parties can be another study issue.

3. PRIVACY-PRESERVING SVD-BASED COLLABORATIVE FILTERING ON INCONSISTENTLY MASKED DATA

To protect users' privacy, the RPT can be employed. However, privacy concerns might vary from a user to another. Data sensitivity and the value of the information may also differ among different users. Moreover, there are various factors in data disclosure like sharing of users' data with others, type of data, and the purpose for which data is collected. Due to these reasons, users are not necessarily mask their data in the same way and might decide to disguise the private data differently using various parameters.

In this chapter, how to provide CF services from inconsistently masked data is discussed. Privacy-preserving schemes to achieve such tasks are proposed. The ultimate goal of these schemes is to prevent the server or data collector from learning the true values of the ratings provided by all of the users who have different level of privacy concerns. Experiments using real data sets to evaluate the overall performance of proposed schemes are performed. Likewise, these schemes are analyzed in terms of accuracy and privacy with varying factors.

Introductory information is given in Section 3.1, while existing SVD-based CF algorithms are introduced in Section 3.2. After inconsistently data masking is explained in Section 3.3, how to achieve PPCF on inconsistently masked data is discussed in Section 3.4. Privacy and overhead cost analysis of proposed algorithms are done in Section 3.5. Section 3.6 includes experiments and their results. Finally, summary of the chapter is presented in Section 3.7.

3.1 Introduction

Polat and Du [42, 43] show that it is possible to provide accurate predictions using the RPT while preserving users' privacy. In their schemes, data disguising is done by each user using the same method. However, users might want to perturb their private data in such a way that they can achieve the level of privacy they want because privacy concerns might be different from user to user. To observe internet users' attitudes toward privacy a survey was conducted in 1999 [17]. The results of the research shows that 17% of respondents are privacy

fundamentalists who refuse to share her personal data even privacy protection is guaranteed. According to the same study, 56% of the respondents are pragmatic majority who has privacy concerns less than fundamentalists. Thus, their contribution increases by the provision of privacy measures. The remaining 27% of respondents are marginally concerned; and they are generally willing to provide private data under almost any condition in spite of often expressing it in a mild general concern about privacy.

To provide recommendations and to collect trustworthy data from users who have different levels of privacy concerns, methods, which allow a user provide private information while ensuring privacy at her choice can be applied. The issue is how to produce accurate referrals on incompatibly masked data, where users can control some part of perturbation process. Providing predictions based on inconsistently disguised data using memory-based algorithms is discussed in [44]. It is shown that it is possible to produce predictions on inconsistently perturbed data using memory-based algorithms. Here, it is studied how to achieve accurate predictions from inconsistently masked data using model-based algorithms. Various ways to disguise private data using RPT is discussed in this chapter. How inconsistently perturbed data affects the recommendation quality compared to consistently perturbed ones is investigated. The CF algorithm proposed by Sarwar et al. [50] is modified; and then it is used to show whether accurate referrals can be provided based on variably perturbed data or not. Various experiments are performed using real data sets based on different data disguising settings. The results are compared and analyzed.

3.2 Singular Value Decomposition and SVD-based Collaborative Filtering

Singular value decomposition (SVD) is a well-known matrix factorization technique that factors an $n \times m$ matrix A into three matrices, as follows:

$$A = U \cdot S \cdot V^T, \tag{3.1}$$

where U and V are two orthogonal matrices of size $n \times y$ and $m \times y$, respectively; y is the rank of the matrix A; and S is a diagonal matrix of size $y \times y$, having all singular values of matrix A as its diagonal entries. All entries of S are positive, and stored in decreasing order of their magnitude. It is possible to

reduce the $y \times y$ matrix S to have only the largest k diagonal values to obtain a matrix S_k, $k < y$. SVD provides the best lower rank approximations of the original matrix A. The basic steps of SVD are, as follows:

- Find the eigenvalues (λ) of the matrix $A^T A$ and arrange them in descending order.
- Find the number of nonzero eigenvalues (λ) of the matrix $A^T A$.
- Find the orthogonal eigenvectors of the matrix $A^T A$ corresponding to the obtained eigenvalues and arrange them to form the column-vectors of the matrix V.
- Form the diagonal matrix S, placing on its leading diagonal the square roots of eigenvalues, $s_i = \sqrt{\lambda_i}$ for $i = 1...y$.
- Find the column-vectors of matrix U using $u_i = s_i^{-1} A v_i$ for $i = 1...y$ and arrange them to form the matrix U with size $n \times y$.

Sarwar et al. [50] propose an SVD-based CF algorithm, which is the best choice for large and sparse databases. SVD is used to cope with sparsity, scalability, and synonymy problems that correlation-based CF algorithms have. The sparse user-item ratings matrix (A) is filled using the average ratings for users, or the average ratings for items, to capture a meaningful, latent relationship. The filled matrix is normalized by converting ratings to z-scores or subtracting customer average from each rating. The normalized matrix (A_{norm}) is factored into U, S, and V using SVD. Then the matrix S_k is obtained by retaining only the largest k singular values. Accordingly, the dimensions of matrices U and V are also reduced. Then, $U_k \sqrt{S_k}$ and $\sqrt{S_k} V_k^T$ are computed.

These resultant matrices can be used to compute the prediction for any user u on q. To compute the prediction, the scalar product of the u^{th} row of $U_k \sqrt{S_k}$ (denoted as $U_k \sqrt{S_k}(u)$) and the q^{th} column of $\sqrt{S_k} V_k^T$ (denoted as $\sqrt{S_k} V_k^T(q)$) is calculated and the result is de-normalized, as follows:

$$p_{uq} = \overline{v_u} + \left(U_k \sqrt{S_k}(u) \cdot \sqrt{S_k} V_k^T(q) \right), \tag{3.2}$$

where $\overline{v_u}$ is mean rating for user u.

3.3 Inconsistent Data Masking

Since SVD-based CF is based on aggregate information rather than individual data items, the RPT can be applied to them. The challenge is that whether it is still possible to generate accurate referrals when data is perturbed differently or not.

3.3.1 Data Disguising Ways

Users are worried about their privacy with various levels. Sensitivity of the private data and the value of the information might be different. The factors in data disclosure are various. Therefore, due to such reasons, users might perturb their data differently to achieve various privacy levels. The ways of data disguising can be explained, as follows:

1. First, users can be classified into two main groups: One group includes the users who have no concerns so that they divulge their data to the server. Small number of users falls into this group. Other group consists of those users who are worried about disclosing the private data so that they perturb their data and contribute masked data. Great majority of people fall into this group.

2. Further, the users can be classified who perturb their data into subgroups on how and how much data they mask

a. Perturbing Data: Users can employ either uniform or Gaussian distribution with μ being 0 to generate random data for data perturbation.

i. Gaussian Distribution: Noise data is created using Gaussian distribution with the standard deviation (σ).

ii. Uniform Distribution: Random data is created using uniform distribution over the range [$-\alpha, \alpha$], where $\alpha = \sqrt{3}\,\sigma$.

b. Level of Perturbation: Users might use different σ values to achieve required privacy levels. The range of random numbers is critical for overall performance. With increasing range, more randomness is added; thus, privacy increases, while accuracy decreases. To achieve required privacy and accuracy, users are able to select the σ values to reach such goals. Users and the server agree on a range, $(0, \gamma)$, and users choose the σ values uniformly randomly over the range $(0, \gamma)$.

c. Standard Deviation (σ) Selection: Generally speaking, when users and the server agree on a σ value, each user creates random numbers based on the given value, which it can be called as predetermined method. To select the σ values, an alternative way, which called as random scheme, can be used. This time, instead of using the same σ, each user uniformly randomly selects the σ over the range $(0, \sigma)$; and uses it to create noise data.

d. Number of Items: Users might decide to mask different numbers of rated and/or unrated items to conceal true ratings and rated items.

i. Masking Ratings: Since the type, sensitivity, and the value of data might be different for users, users might decide to perturb different numbers of ratings.

A. Some users mask all of their ratings.

B. On the other hand, some of them might perturb some randomly selected ratings.

ii. Hiding Rated Items: Users may want to conceal rated items because it might be more damaging revealing which items users rated.

A. To hide which items are rated, users might decide to fill all unrated items' cells with random number

B. Rather than filling all unrated items' cells, users randomly select some unrated items' cells to fill with noise data.

iii. Given a ratings vector, which is usually sparse, users might decide to choose randomly some of the ratings vector's cells to mask. These chosen cells consist of ratings and empty cells of unrated items. The number of chosen cells depends on how much accuracy and privacy users want to achieve.

3.3.2 Masking Process

Those users, who have no concern revealing their private data, send the true data to the server. However, when the server receives data from a user, it does not know whether it is a true data or masked data due to underlying inconsistent data perturbation. Once users decide to perturb the private data, they conduct the following steps:

1. Users first normalize their ratings and each user decides the perturbing data to generate random numbers.
2. Each user decides the level of randomness. For this purpose, each user chooses the σ and decides how to select σ (use predetermined or random method).
3. Users decide the amount of data to perturb and choose the number of ratings and unrated cells to disguise.
4. Finally, each user creates random numbers on how much data to be masked using the selected parameters.

Users send inconsistently perturbed data to the server, which creates a differently masked user-item matrix, A'. An example of such matrix is shown in Table 3. 1. The matrix A' includes disguised and undisguised data because some users send perturbed data while others send true data. Since the amount of data to be masked might be different for users, A' also consists of empty cells. Some cells include only noise data because remember that users fill some unrated items' cells with random data to prevent the server from learning rated items. The server does not know the type of perturbing data, the σ values, and the σ selection methods due to underlying inconsistent data masking.

Table 3. 1 Inconsistently Masked User-Item Matrix

	Item$_1$	**Item$_2$**	**Item$_3$**	**...**	**Item$_{m-1}$**	**Item$_m$**
User$_1$	z_{11}	z_{12}	z_{11}	…	$z_{1(m-1)}$	z_{1m}
User$_2$		$z_{12}+ r_{12}$		…		$z_{2m}+ r_{2m}$
User$_3$	$z_{31}+ r_{31}$		r_{33}	…	$r_{3(m-1)}$	
. . .				. . .		
User$_{n-1}$		$z_{(n-1)2}+ r_{(n-1)2}$		…		z_{n-1m}
User$_n$	$z_{11}+ r_{11}$	$z_{11}+ r_{11}$		…		$z_{11}+ r_{11}$

3.4 Providing Recommendations on Inconsistently Masked Data

To utilize inconsistently data effectively, some modifications must be done on original algorithm. Modifications, how to estimate to the original matrix, and how to provide referrals are discussed in the following sections.

3.4.1 Modified SVD-based Collaborative Filtering Algorithms

SVD-based CF algorithms can be modified and/or simplified to achieve private referrals with decent accuracy. Rather than factoring the filled matrix, sparse matrix is factored. When empty cells are filled with default ratings, since users' ratings vectors are sparse, their standard deviations become smaller; thus, data ranges become larger after converting them into z-scores. User mean votes are employed for normalization because each user is able to normalize her ratings without needing other users' data. The following modified and/or simplified algorithms are proposed to use:

Modified Algorithm 1 (MA1): Ratings are first normalized by converting them into z-scores, where user-mean ratings are employed, and normalized user-item matrix (A_{norm}) is found. Then, the steps explained previously to find $U_k\sqrt{S_k}$ and $\sqrt{S_k}V_k^T$ matrices are conducted. Since z-scores are used, predictions can be computed, as follows:

$$p_{uq} = \overline{v_u} + \sigma_u \times \left(U_k\sqrt{S_k}(u) \cdot \sqrt{S_k}V_k^T(q)\right) \tag{3.3}$$

where σ_u is the standard deviation of user u's ratings.

Modified Algorithm 2 (MA2): Ratings are first converted into normalized votes using bias-from-mean approach, where user-mean is employed rather than item-mean, and normalized user-item matrix (A_{norm}) is found. Then, the steps explained previously to find $U_k\sqrt{S_k}$ and $\sqrt{S_k}V_k^T$ matrices are conducted. Predictions can be computed, as follows:

$$p_{uq} = \overline{v_u} + \left(U_k\sqrt{S_k}(u) \cdot \sqrt{S_k}V_k^T(q)\right) \tag{3.4}$$

3.4.2 Estimating SVD from Inconsistently Masked Data

The range of random numbers should be broad enough to preserve users' privacy. Such range is critical for overall performance. With increasing range,

privacy increases, while accuracy diminishes. When ratings are normalized using bias-from-mean approach, the range of random numbers is larger than the one when z-score normalization method is used. In [43], it is shown how to estimate SVD of A', as follows, when users mask private data in the same way and disguise all ratings and empty cells:

The server first computes $A'^T A'$. Each entry of $A'^T A'$ is computed by calculating the scalar product of rows of matrix A'^T and the columns of the matrix A'. The entries other than the diagonal ones are computed, as follows:

$$(A'^T A)_{fg} = \sum_{u=1}^{n} (z_{uf} + r_{uf})(z_{ug} + r_{ug}) \approx \sum_{u=1}^{n} z_{uf} z_{ug} , \qquad (3.5)$$

where n is the number of users, f and g show the row and column numbers, respectively, and $f \neq g$. Since random values are independent and drawn from a distribution with $\mu = 0$, the expected value of $\sum_{u=1}^{n} r_{uf} r_{ug}$ is 0. Similarly, the expected values of $\sum_{u=1}^{n} z_{uf} r_{ug}$ and $\sum_{u=1}^{n} r_{uf} z_{ug}$ are 0. However, since the scalar product is computed between the same vectors for the diagonal entries ($f = g$), they can be estimated, as follows:

$$(A'^T A')_{ff} = \sum_{u=1}^{n} (z_{uf} + r_{uf})(z_{uf} + r_{uf}) = \sum_{u=1}^{n} z_{uf}^2 + 2\sum_{u=1}^{n} z_{uf} r_{uf} + \sum_{u=1}^{n} r_{uf}^2 \qquad (3.6)$$

Again, the expected value of $\sum_{u=1}^{n} z_{uf} r_{uf}$ is 0. However, since $\sum_{u=1}^{n} z_{uf}^2$ values are only needed for diagonal entries, it is necessary to get rid of $\sum_{u=1}^{n} r_{uf}^2$ in (3.6), as follows:

$$(A'^T A')_{ff} = \sum_{u=1}^{n} z_{uf}^2 + \sum_{u=1}^{n} r_{uf}^2 - n \times \sigma_r^2 \approx \sum_{u=1}^{n} z_{uf}^2 , \qquad (3.7)$$

where σ_r is the standard deviation of random numbers. σ_r is multiplied by n to get rid of the contribution of the random numbers because all users disguise all cells. After estimating the matrix $A'^T A'$, the server can find the rank and the eigenvalues, which are used to find eigenvectors that form the matrix V'. It then finds the matrix S' using the eigenvalues estimated from $A'^T A'$.

Finally, the server needs to calculate the first y column vectors of U using $b_i = s_i^{-1} A v_i$ for for $i = 1 \ldots y$, where v_is are column-vectors of V. Similarly, b_i

vectors can be estimated using A', s_i', and v_i' vectors, where $s_i's$ and $v_i's$ are estimated from the matrix $A'^T A'$. The entries of b_i' vectors can be estimated, as follows, where $j = 1...\ n$:

$$b_i(\mathrm{j}) = \frac{1}{\mathrm{s_i'}} \sum_{\mathrm{l}=1}^{m} (z_{jl} + r_{jl}) v_{il}' \approx \frac{1}{\mathrm{s_i'}} \sum_{\mathrm{l}=1}^{m} z_{jl} v_{il}' \,, \tag{3.8}$$

where the expected value of $\sum_{l=1}^{\mathrm{m}} r_{jl} v_{il}'$ is 0.

It is possible to estimate the SVD of A' when it consists of consistently masked data and all cells are masked. However, it becomes a challenge when users disguise their data using various ways. In this section, it is elucidated that how to estimate the SVD of A' when it consists of differently perturbed data. The estimations mentioned above are based on the scalar product and sum computations. Moreover, as seen from (3.7), to get rid of the contribution of the random numbers, n and σ_r values are needed. It should be first shown meaningful outcomes using these computations from differently masked data can still be provided. Then, it should be investigated whether contribution of the random numbers in (3.7) can be got rid of. As shown in [44], it is still possible to estimate the results of scalar product and sum computations on inconsistently masked data, considering different scenarios, explained before:

First, suppose that some users mask their ratings, while others send true data. In this case, it is possible to estimate the sum and the scalar product of two vectors, one is masked one is not, because random numbers are drawn from some distributions with μ being 0. Let X be a size of n undisguised vector and Y' be size of n vector, which is disguised as $Y + V$, then

$$X \cdot Y' = \sum_{i=1}^{n} (x_i)(y_i + v_i) = \sum_{i=1}^{n} x_i y_i + \sum_{i=1}^{n} x_i v_i \approx \sum_{i=1}^{n} x_i y_i \quad , \tag{3.9}$$

and also

$$Y' = \sum_{i=1}^{n} y_i + v_i = \sum_{i=1}^{n} y_i + \sum_{i=1}^{n} v_i \approx \sum_{i=1}^{n} y_i \,, \tag{3.10}$$

The effects of noise data in (3.9) and (3.10) can be removed. To get rid of the contributions of random numbers in (3.7), σ_r is set at the expected standard deviation of noise data considering only number of users who disguise their

ratings. Such number of disguising users can be determined via examining conducted surveys on the privacy concern characteristics of users.

Second, suppose that some users employ uniform, while others use Gaussian perturbing data for data masking. Let X' and Y' be disguised as $X + U$, uniformly and $Y + V$, normally, then:

$$X \cdot Y' = \sum_{i=1}^{n} (x_i + u_i)(y_i + v_i) = \sum_{i=1}^{n} \left(x_i y_i + x_i v_i + u_i y_i + u_i v_i \right) \approx \sum_{i=1}^{n} x_i y_i \qquad (3.11)$$

Since random numbers are drawn from distributions with μ being 0, again, it is possible to estimate the sum and scalar product of two vectors as in (3.11), one is disguised with uniformly generated noise data and the other is masked by Gaussian perturbing data. Since σ_r will be same whether users employ uniform or Gaussian, (3.7) holds for this case and the number of ratings masked by users is considered.

Third, users are able to employ different σ values and they can select such values by employing predetermined or random schemes. In both cases, each user generates random numbers using different σ values. The sum and the scalar product can be estimated from data disguised by users who employed various σ values due to μ being 0. When users select σ values uniformly randomly over the range $(0, \gamma)$, then

$$\sigma_r = \mathrm{E}(\sigma_r) = \frac{1}{n_d} \sum_{i \in n_d} \sigma_i = \frac{0 + \gamma}{2} = \frac{\gamma}{2}, \qquad (3.12)$$

where n_d is the number of commonly disguised cells and $\mathrm{E}(\sigma_r)$ expected value of σ_r. Since all users mask their ratings, the number of rated items is considered when trying to get rid of the contributions of random numbers.

Finally, users might decide to mask various numbers of cells, where the number of cells to be perturbed depends on their privacy concerns. Such cells may include ratings only, ratings and some unrated items' cells, or randomly selected ratings and unrated items' cells, as explored before. Let X' and Y' be disguised where $X' = X + U$ and $Y' = Y + V$, providing that different numbers of cells are selected randomly to perturb. According to (3.10), the effects of noise on non-commonly disguised cells are removed. So Let d_s is the indices of commonly disguised cells then,

$$X' \cdot Y' = \sum_{i \in d_s} (x_i + u_i)(y_i + v_i) = \sum_{i \in d_s} \left(x_i y_i + x_i v_i + u_i y_i + u_i v_i \right) \approx \sum_{i \in d_s} x_i y_i \,, \qquad (3.13)$$

Due to the same reasons, it is possible to estimate the sum and scalar product when users disguise different numbers of cells. When trying to get rid of the contributions of random numbers, average number of disguise cells is considered where σ_r is the standard deviation of random numbers. Such scenarios can be accumulated and it can be shown that scalar product and sum based on them can be estimated.

3.4.3 Providing Referrals

To get a prediction for target item q, the user u sends a query to the server, which computes the P' by calculating the scalar product of the u^{th} row of $U_k \sqrt{S_k}$ and the q^{th} column of $\sqrt{S_k} V_k^T$, and sends it to the user u, who can now calculate the p'_{uq}. The server will only learn P' because it does not know $\overline{v_u}$ and σ_u values. Therefore, it will not be able to learn how much the user u likes or dislikes q.

To provide top-N recommendations, the user u sends a query stating that she is looking top-N recommendations for N_u items, where $N < N_u < m\text{-}d$, and d is the number of items rated by u. The server then computes P' values for all N_u items and sorts them decreasingly. Finally, it selects the first N items and sends the list to user u as top-N recommendations. Since the same mean and standard deviation are used to de-normalize P' values, the server does not need them to find the sorted list.

3.5 Privacy and Overhead Cost Analysis

The privacy measure should indicate how closely the original value of an item can be estimated from the perturbed data. To quantify privacy, various measures are employed [1, 2]. Privacy introduced due to uniform or Gaussian perturbing data for consistently data masking is analyzed in [42] and can be similarly analyzed. Since users employ inconsistent data masking, privacy improves. The server tries to figure out the true ratings and rated items. Due to randomly inserted noise data, it will not be able to learn rated items. Since users

disguise normalized ratings, it becomes difficult for the server to obtain original votes without knowing the mean ratings and standard deviations of the ratings, which are only known by users. Due to inconsistent data perturbation, the server will not be able to learn how and how much data is masked. To obtain true ratings, the server should know the type and the parameters of the perturbing data and the amount of disguised data. However, it does not know such information due to inconsistent data masking. Therefore, it can be said that the inconsistent data perturbation improves privacy.

Proposed schemes do not introduce additional storage and communication costs, while they cause an increase in computation cost. However, extra computation costs due to privacy concerns are negligible. The server conducts additional computations to get rid of the contribution of noise data in diagonal entries of $A'^{T}A'$ matrix. Data perturbation is easily done by each user without the help of a third party.

3.6 Experiments

To show whether accurate referrals can be achieved from inconsistently masked data or not; and to show how accuracy changes with varying data masking ways, experiments using well-known real data sets are performed.

3.6.1 Datasets and Evaluation Metrics

Jester and MovieLens public (MLP) data sets are used in these experiments. MLP was collected by the GroupLens Research Project at the University of Minnesota (www.cs.umn.edu/research/Grouplens). Each user has rated at least 20 movies in MLP, and ratings are made on a 5-star scale. It consists of 100,000 ratings for 1,682 movies made by 943 users.

The most common criteria are the Mean Absolute Error (MAE) and the Average Relative Error (ARE). They were also employed as the choice of evaluation criteria in these trials. The MAE and the ARE should be minimized. The lower the MAE and the ARE, the more accurate the scheme is. To show what percentage of the MAEs introduced due to the proposed privacy-protecting schemes. Another employed metric is ARE can be defined, as follows:

$$\text{ARE} = \frac{|E_d - E_u|}{E_d} \times 100, \tag{3.10}$$

where E_d and E_u represent the MAEs of the predicted ratings from disguised data and undisguised data, respectively.

3.6.2 Methodology

Average ratings for users and the standard deviations of them are computed to normalize the ratings and normalized data is obtained. Random values to disguise the normalized ratings are generated. The random numbers are created based on various data disguising ways to achieve different privacy levels. Those noise data is added to the private values and/or empty cells, and variably perturbed user-item matrix, A' is obtained.

Data is divided into training and testing, as follows: While all users' data in MLP is employed, 1,000 users are selected randomly and used their data for training from Jester. Randomly, 10% of the available ratings are selected to use in the experiments as test data, where k is set at 10 for both data sets. For test items, their ratings are withheld and it is tried to predict their values given all other votes using proposed schemes. The predictions found based on disguised data are compared with the withheld ratings. Data disguising is run for 100 trials and found predictions. Overall MAEs and AREs are obtained and final values are displayed.

3.6.3 Experimental Results

Following experiments are done using the modified SVD-based algorithm 1 (MA1):

Experiment 1: Note that some users might send masked data, while others may send true data. Experiments are performed to show how accuracy changes with varying numbers of users who disguise their data, where they perturb their ratings only. x_u is defined as the percentage of the users who disguise their private data and experiments are conducted while varying x_u from 0 to 100. When x_u is 0, all users send true data, while every user sends disguised data when it is 100. To generate random numbers, Gaussian distribution is employed with predetermined σ being 3. Both data sets are used and the overall MAEs are computed; and they

are shown in Table 3.2. As expected, accuracy worsens with increasing x_u values due to increasing numbers of users send masked data. However, as seen from Table 3.2, the results are still promising when all users disguise their data, where the ARE is only 7.20 % for MLP.

Table 3. 2 Accuracy with Varying x_u

x_u (%)	0	30	60	100
MLP	0.7723	0.8043	0.8193	0.8322
Jester	3.4192	3.6174	3.7836	3.9847

Experiment 2: Users employ either uniform or Gaussian perturbing data for data masking. To show how accuracy changes with various perturbing data, experiments are conducted using both data sets, where uniform or Gaussian distributions with σ being 2 is utilized to generate random numbers. x_g is defined as the percentage of users who perturb their data using Gaussian, while the remaining users use uniform perturbing data. Experiments are run while varying x_g from 0 to 100 to show how accuracy changes with varying x_g values. x_g is selected randomly as the percentage of the users who employ Gaussian distribution to create noise data. When x_g is 0, all users employ uniform perturbing data, while they absolutely employ Gaussian perturbing data when it is 100. The overall MAEs and AREs are computed for both data sets and since the outcomes are similar, MAEs are just shown for Jester in Figure 3.1. As seen from Figure 3.1, accuracy improves with increasing numbers of users employing uniform perturbing data. On the other hand, Gaussian perturbing data introduces more privacy than uniform perturbing data [42]. Moreover, the accuracy loss due to using Gaussian perturbing data is small. Most importantly, it is still possible to provide accurate referrals when users employ different perturbing data.

Experiment 3: Users can choose different σ values employing various ways to select them. In these experiment sets, how accuracy changes with varying level of perturbation is shown. With increasing randomness, accuracy diminishes, while privacy improves. Users are able to select σ values to achieve required privacy and accuracy levels. After the server and the users agree on a predetermined γ value, each user u uniformly randomly selects the σ_u over the

range [0, γ] in which γ is big enough to achieve required privacy levels. Although the server knows γ, it will not be able to learn the σ values randomly selected by the users. Experiments were performed for γ being 1, 2, 3, and 4 to show how accuracy changes with varying level of perturbation and various σ selection ways. As expected, the results on uniformly randomly selected σ values are better than the ones when users employ predetermined values. The reason for this phenomenon is that more randomness is added to actual data when all users employ the same predetermined σ values.

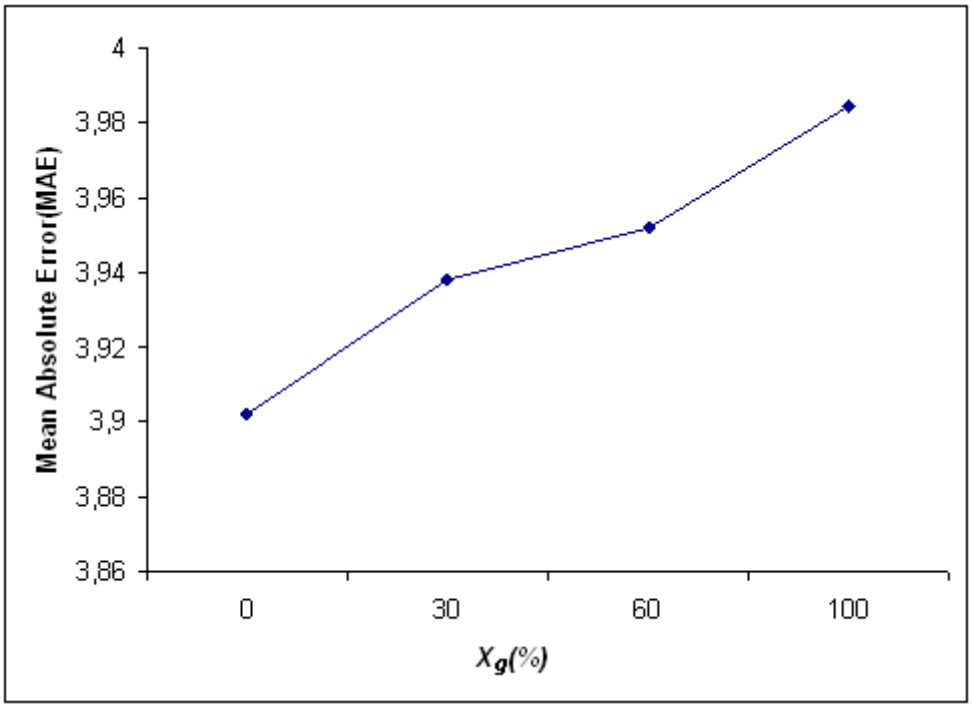

Figure 3. 1 Accuracy vs. Perturbing Data

On the average, the σ will be $\gamma = 2$ because it is uniformly selected over the range [0, γ]. The bigger the randomness, the less accurate the results are. Therefore, the results get better when users randomly select the σ values. In addition, the server is not able to learn such values, which are only known by the users. Experiments are performed using both data sets, with Gaussian perturbing data. The overall MAEs are computed and the outcomes are shown in Table 3.3. As seen from Table 3.3 and as expected, obtained results become better with decreasing levels of perturbation due to less randomness. Although accuracy worsens with increasing σ values, the results are still promising when γ is 4, where the AREs are 8.14 % and 17.11 % for MLP and Jester, respectively.

Table 3. 3 Accuracy vs. Level of Perturbation

γ	**1**	**2**	**3**	**4**
MLP	0.7798	0.7984	0.8283	0.8408
Jester	3.4679	3.7422	3.8751	4.1254

Experiment 4: In these set of experiments, how accuracy changes with varying numbers of masked cells is shown. Note that three different scenarios are explained previously: First, users might decide to perturb different numbers of ratings. Second, in addition to masking all ratings, they may add noise data to various numbers of empty cells to prevent the server learning the rated items. And finally, they randomly select different numbers of cells from their ratings vectors to disguise, where such selected cells consist of ratings and empty cells. As expected, accuracy worsens with increasing disguised data and/or cells. Instead of performing different experiments for these three different cases, since they relate with each other and give similar results, experiments are performed to show how varying amounts of perturbed cells (including ratings and empty cells) affect accuracy. Given a ratings vector including ratings and empty cells, users randomly select different numbers of cells to disguise. Experiments are conducted using both data sets, where x_c is defined as the percentage of the cells to be perturbed. Gaussian distribution is used with σ being 2 to generate random numbers. x_c is varied from 0 to 100. When x_c is 0, users do not mask anything, while they perturb all of their ratings vectors' cells when it is 100. The overall MAEs are computed, and since the outcomes are similar, MAEs for MLP only are shown in Fig. 3.2. With increasing x_c values, more cells are disguised and that makes accuracy worse. Such cells include ratings and empty cells. Since ratings vectors are usually very sparse, such randomly selected cells mostly contain empty cells. When they are masked, noise data is inserted. That is why, accuracy worsens with increasing numbers of perturbed cells.

In addition to the experiments using the MA1, trials are conducted to show how accurate the results are when the modified SVD-based CF algorithm 2 (MA2) is employed. In this case, since bias-from mean normalization is used, the range of the private data becomes larger compared to z-scores. Therefore, larger

random numbers should be used to mask the private data. The same methodology is followed. The results are only shown when different numbers of users perturb their ratings. Gaussian distribution is used with σ being 3 to generate random numbers. x_u is varied from 0 to 100. The overall MAEs are computed and outcomes are displayed in Table 3.4 for MLP only. Although the results are worse than the ones in Table 3.2, they are still promising. When x_u is 100, the ARE is only 10.61 %, which is acceptable.

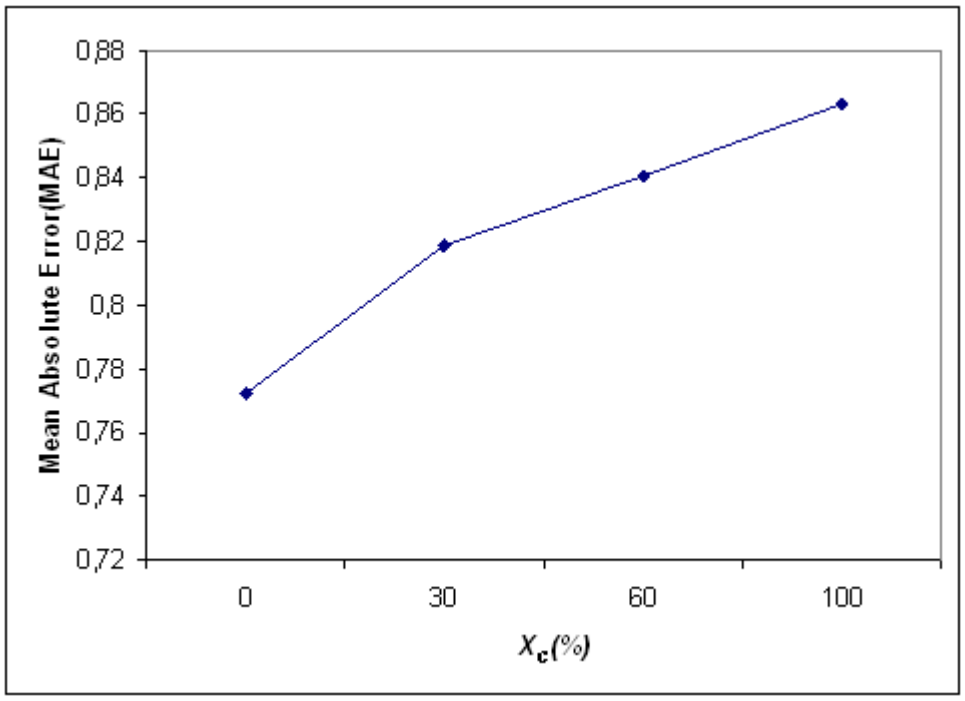

Figure 3. 2 Accuracy vs. x_c Values

3.6.5 Conclusions

Proposed schemes in this study allow users to mask their data employing different perturbing ways to achieve required levels of privacy and/or accuracy. As expected, with decreasing amount of perturbed data, accuracy improves. Although perturbing private data by employing Gaussian or uniform disguising data gives similar results in terms of accuracy, Gaussian perturbing data achieves higher privacy. Therefore, users should prefer Gaussian perturbing data for data masking. Since random scheme improves accuracy, users should randomly select the σ values instead of using the predetermined values. Prediction qualities diminish with increasing number of disguised items. As expected, accuracy

decreases with increasing level of perturbation. On the other hand, users are able to improve their privacy employing bigger σ values. They should select σ in such a way to achieve a balance between accuracy and privacy.

Table 3. 4 Accuracy vs. x_u Values for MA2

x_u (%)	0	30	60	100
MLP	0.7695	0.8203	0.8449	0.8609

3.7 Summary of the Chapter

It is shown that how to achieve CF tasks using SVD-based algorithms on differently perturbed data. Various ways of data disguising are elucidated. The SVD-based CF algorithms are modified/simplified in such a way to accomplish inconsistently masked data-based CF. Experiments are performed to evaluate the overall performance of proposed schemes. Obtained results show that it is still possible to offer accurate CF services when private data is masked differently. These schemes are analyzed in terms of privacy and accuracy. Recommendations for masking data are provided to achieve required levels of privacy and accuracy. How to extend these schemes to other CF algorithms will be studied. It will be studied that how to increase accuracy when some aggregate information is disclosed. The schemes proposed for the modified SVD-based algorithm 2 (MA2) will be evaluated in deeply in future studies.

4. PRIVACY-PRESERVING SVD-BASED COLLABORATIVE FILTERING ON DISTRIBUTED DATA

SVD-based CF systems offer reliable and accurate predictions when they own large enough data. Data collected for CF purposes, however, might be split between different companies, even competing ones. Some vendors, especially newly established ones, might have problems with available data. To increase mutual advantages, provide richer CF services, and overcome problems caused by inadequate data, companies want to integrate their data. However, due to privacy, legal, and financial reasons, they do not want to combine their data.

In this chapter, it is explored how to achieve SVD-based recommendations based on integrated data without jeopardizing data owners' privacy. Data might be horizontally or vertically split between different parties. Both partitioning schemes-based CF using SVD is investigated. Various methods are proposed to provide recommendations with satisfying accuracy while preserving data owners' privacy. To observe overall performance of the proposed schemes, experiments on real data are performed and the obtained outcomes are also analyzed.

Preliminary information is presented in Section 4.1, while distributed data-based CF schemes are proposed in Section 4.2. Privacy and overhead cost analysis are performed in Section 4.3 and Section 4.4, respectively. After explaining experiments and their results in Section 4.5, the chapter is summarized in Section 4.6.

4.1 Introduction

Data from many users is needed to provide CF services. However, data might be split between different e-companies, even competing vendors. Besides, some companies especially those newly established ones might have limited amount of data. Inadequate data might cause problems and restricts CF systems to provide referrals for only limited number of items. To increase mutual advantages, conduct richer CF tasks, overcome problems caused by inadequate data, and so on, e-companies might decide to combine their data for CF purposes. However, due to privacy reasons, they do not want to disclose their private data to

each other. Moreover, they do not want to integrate their data due to financial and legal reasons if privacy measures are not in place. Data owners might be worried about that if they reveal their data, their data might be transferred to other parties and lose their competitive edge.

It is hypothesized that if privacy measures are provided, online vendors, even competing companies, might decide to integrate their data. Privacy measures should allow the companies to conduct CF tasks based on combined data, while protecting their privacy. In other words, such companies should not be able to learn the true rating values and rated items in each other's databases while providing the same CF services to their customers using the joint data.

It is investigated how to achieve SVD-based recommendations based on integrated data without jeopardizing data owners' privacy. The proposed schemes should be able to offer accurate referrals efficiently with privacy. However, accuracy, privacy, and efficiency are conflicting goals. Therefore, the schemes should provide a balance between them. Proposed solutions are analyzed in terms of privacy and overhead costs introduced due to underlying privacy concerns. Additional computation, communication, and storage costs should be negligible. To assess the overall performance of the schemes, real data-based experiments are performed and their results are examined.

4.2 Partitioned Data-based Collaborative Filtering Using SVD with Privacy

CF systems usually operate on existing databases, which include ratings for items collected from many users. However, in some cases, data collected for CF purposes might be split horizontally or vertically between different parties. In horizontal partitioning, different online vendors hold disjoint sets of users' preferences for the same items, while in vertical partitioning, they own disjoint sets of items' ratings collected from the same users. To make the data sharing possible, the identity of the products and customers can be established across the data holders' databases. This data exchange between vendors can be achieved offline.

The goal is to investigate how to achieve SVD-based CF tasks from partitioned data between different parties with privacy. Providing accurate

referrals efficiently without jeopardizing vendors' privacy is challenging. Achieving privacy, accuracy, and efficiency together is not an easy task, because they conflict each other. The proposed schemes should allow data owners to find equilibrium between them. E-commerce sites should not be able to find out the true rating values and the rated items in their databases. With privacy protection measures, data owners will not be able to learn such information related to each other's databases. For customers, obtaining accurate referrals efficiently is important. Customers prefer to return those e-commerce sites, which offer accurate and dependable recommendations. Due to privacy protection measures, accuracy might get worse. However, recommendations calculated with privacy concerns should be as close as possible to those computed without privacy concerns. CF systems produce referrals to many users in real time. During an online interaction, users get referrals from CF systems. Online computation time for providing recommendations should be small enough so that many users can obtain referrals without wasting too much time. Due to privacy measures, additional costs are expected. However, such costs should be negligible. Therefore, in the proposed schemes, additional costs introduced due to privacy concerns should be small enough. In CF, some computations are conducted off-line while others are done online. Since offline costs are not critical to overall performance, the proposed schemes in terms of additional online costs.

With the evolution of the Internet, e-commerce has become very popular. Many users sell or buy products over the Internet. E-commerce has brought many advantageous for customers. To increase their sales and profits, online vendors try different methods. One of such methods is CF. To recruit new customers and keep the current ones, providing truthful and dependable recommendations efficiently are important. Customers buy products based on the recommendations provided to them by online vendors. They prefer returning to those sites with accurate and reliable predictions. On the other hand, false and unreliable referrals lead angry customers. Therefore, for both online vendors and customers, providing true and trustworthy recommendations is important.

Online vendors collect ratings from users to offer referrals. Truthful and dependable referrals can only be produced from enough ratings. To find large

enough neighborhood, there should be large enough number of users. Moreover, to compute similarities between users, there should be enough number of commonly rated items. Users can buy items from different online vendors. Their ratings for the same products might be held by various sites. Each site might own ratings collected from a limited number of users. When data is horizontally partitioned, it becomes difficult for vendors to form a large enough reliable neighborhoods. In some cases, users buy products from different sites. Their ratings for various items are split between parties. When data is vertically partitioned, data holders have problems finding enough commonly rated items by users and it becomes a challenge to find dependable matching between users. Therefore, it is important to integrate horizontally or vertically partitioned data for offering truthful and reliable recommendations. By combining partitioned data, it is more likely to offer richer CF services. Inadequate data might cause cold start problem and it restricts the CF systems and e-commerce sites to generate referrals for only a limited number of items. Those sites with insufficient data might overcome the cold start problem by integrating the partitioned data. By combining horizontally or vertically partitioned data, e-companies will be able to provide referrals to more users and for more items using their SVD-based models computed from combined data.

In the following subsections, HPD- and VPD-based privacy-preserving schemes to provide SVD-based CF services are investigated. Looking at the steps in SVD-based CF scheme and SVD itself, there are four major issues that should be handled when data is partitioned. They are how to remove sparsity of the matrix A using product average, how to normalize ratings using deviation from mean approach with user mean values, how to compute A^TA, and how to find column vectors of U using $b_i = s_i^{-1} Rv_i$ for $i = 1 \ldots r$ from horizontally or vertically partitioned data.

4.2.1 Horizontally Partitioned Data-based Collaborative Filtering Using SVD with Privacy

Data collected for CF purposes is stored in an $n \times m$ matrix A, which is a sparse user-item matrix, as follows:

$$R = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}$$

In horizontal partitioning, company P and company Q own ratings matrices with sizes $n_1 \times m$ and $n_2 \times m$, respectively, where $n = n_1 + n_2$. They want to conduct SVD-based CF services using their combined data, which is an $n \times m$ matrix, where n and m represent the number of users and items, respectively. User-item matrices held by P and Q can be shown, as follows:

$$A_{PH} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n_1 1} & x_{n_1 2} & \cdots & x_{n_1 m} \end{bmatrix} \text{ and}$$

$$A_{QH} = \begin{bmatrix} x_{(n_1+1)1} & x_{(n_1+1)2} & \cdots & x_{(n_1+1)m} \\ x_{(n_1+2)1} & x_{(n_1+2)2} & \cdots & x_{(n_1+2)m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}.$$

In the following, it is shown how to handle four issues mentioned before when data is horizontally partitioned between two vendors.

a. Removing Sparsity: To fill the sparse matrix A, product averages are used. Since data is horizontally partitioned, to calculate the item means, data owner needs each other's data. Data holders can remove sparsity without jeopardizing their privacy using the following privacy-preserving item mean computation on HPD scheme:

i. Both parties calculate user means for users they hold. They are able to compute such mean values without the need of each other's data.

ii. They decide an integer θ, where $0 < \theta \leq 100$.

iii. Each party j (j is P or Q) uniformly randomly selects an integer θ_j over the range $(0, \theta]$.

iv. They uniformly randomly select θ_j% of the empty cells in their user-item matrices, and fill their entries with corresponding user mean values.

v. Data owners then compute the column sum and the number of ratings for each item; and exchange these aggregate data.

vi. After such data exchanging, they can finally calculate the product means from integrated data.

Since data owners do not know the user mean values for users each other's hold, randomly selected θ_j values, and empty cells by each other, they will not be able to derive data about each other's data from exchanged aggregate data.

b. Normalization: In HPD-based schemes, normalization can be done easily without violating data holders' privacy. Since data is horizontally partitioned, each party can compute user means for those users they hold without the help of the other party. After computing user mean values, each party can normalize their ratings using such values and the deviation from mean approach.

c. A^TA Computation: After obtaining a filled and normalized sub-matrix, the next step is to compute A^TA based on integrated data. A^TA is an *m x m*, symmetric matrix. Since it is a symmetric matrix, to find the entries of it, data owners should compute *m x (m + 1) / 2* values rather than *m x m* values. Such values are calculated based on the data held by both parties. The A^TA matrix is shown, as follows:

$$A^T A = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1m} \\ X_{21} & X_{22} & \cdots & X_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ X_{m1} & X_{m2} & \cdots & X_{mm} \end{bmatrix}.$$

The entries of A^TA can be computed, as follows:

$$X_{11} = x_{11}x_{11} + \cdots + x_{n_1 1}x_{n_1 1} + x_{(n_1+1)1}x_{(n_1+1)1} + \cdots + x_{n1}x_{n1} = X_{11P} + X_{11Q}$$
$$X_{12} = x_{11}x_{12} + \cdots + x_{n_1 1}x_{n_1 2} + x_{(n_1+1)1}x_{(n_1+1)2} + \cdots + x_{n1}x_{n2} = X_{12P} + X_{12Q}$$
$$\vdots$$
$$X_{m(m-1)} = x_{1m}x_{1(m-1)} + \cdots + x_{n_1 m}x_{n_1(m-1)} + x_{(n_1+1)m}x_{(n_1+1)(m-1)} + \cdots + x_{nm}x_{n(m-1)} = X_{m(m-1)P} + X_{m(m-1)Q}$$
$$X_{mm} = x_{1m}x_{1m} + \cdots + x_{n_1 m}x_{n_1 m} + x_{(n_1+1)m}x_{(n_1+1)m} + \cdots + x_{nm}x_{nm} = X_{nmP} + X_{nmQ}$$

where X_{ijP} and X_{ijQ} can be written, as follows, for $i = 1, \ldots, m$ and $j = 1, \ldots, m$:

$$X_{11P} = x_{11}x_{11} + \cdots + x_{n_1 1}x_{n_1 1} \qquad X_{11Q} = x_{(n_1+1)1}x_{(n_1+1)1} + \cdots + x_{n1}x_{n1}$$
$$X_{12P} = x_{11}x_{12} + \cdots + x_{n_1 1}x_{n_1 2} \qquad X_{12Q} = x_{(n_1+1)1}x_{(n_1+1)2} + \cdots + x_{n1}x_{n2}$$
$$\vdots \qquad \text{and} \qquad \vdots$$
$$X_{m(m-1)P} = x_{1m}x_{1(m-1)} + \cdots + x_{n_1 m}x_{n_1(m-1)} \qquad X_{m(m-1)Q} = x_{(n_1+1)m}x_{(n_1+1)(m-1)} + \cdots + x_{nm}x_{n(m-1)}$$
$$X_{mmP} = x_{1m}x_{1m} + \cdots + x_{n_1 m}x_{n_1 m} \qquad X_{mmQ} = x_{(n_1+1)m}x_{(n_1+1)m} + \cdots + x_{nm}x_{nm}$$

As seen from the equations given above, to compute the entries of A^TA, both parties' data is needed. The X_{ijP} and X_{ijQ} values can be computed from data held by P and Q, respectively. Since A^TA is symmetric, the parties should exchange $m\ x\ (m + 1)\ /\ 2$ aggregate values. Party P and party Q hold $n_1\ x\ m$ and $n_2\ x\ m$ unknowns, respectively. Since both parties know the equations to obtain those aggregate values, they can solve such equations for unknowns that other party holds. When the number of unknowns is bigger than the number of equations, data owners will not be able to solve the equations for such unknowns. However, when the number of users held by each party is less than $(m + 1)\ /\ 2$, they are able to solve the equations for unknowns held by each other. They then can learn the true data items in each other's databases from given aggregate values.

To prevent the parties from learning each other's private data while computing A^TA, the following privacy-preserving protocol (called **Protocol I)** is proposed:

i. Data owners decide an integer value γ, where $0 < \gamma \leq 100$. Each party j uniformly randomly selects an integer value γ_j over the range $(0, \gamma]$.

ii. In equations to calculate X_{ijP} and X_{ijQ} values, they find those entries of the equations that represent the multiplication of two filled cells. They then uniformly randomly select $\gamma_j\%$ of them and removed them from the equations.

iii. After removing those randomly chosen entries, finally, each party computes the new aggregate values and exchanges them.

Data holders will not be able to learn the true data items reside in each other's databases due to randomly selected γ_j and randomly removed cells. After these data exchanges, they now have the estimated A^TA matrices. Each party can now find the eigenvalues (λ), the number of nonzero eigenvalues of the A^TA matrix it has, and the orthogonal eigenvectors of the A^TA matrix it estimated corresponding to the obtained eigenvalues. Each party j finally forms the matrix V_j.

d. Computation of the column vectors of U_j: The final step in SVD computation is finding the column-vectors of matrix U_j using $b_i = s_i^{-1}Rv_i$ for $i = 1 \ldots r$ and arranging them to form the matrix U_j with size $n\ x\ r$. Both parties already have estimated S_j and V_j matrices. Since each party owns its singular values, for

the sake of simplicity, the entries of U_j can be computed, as follows, without considering the multiplication by singular values right now:

$$U_{11} = x_{11}v_{11} + x_{12}v_{21} + \cdots + x_{1m}v_{m1}$$
$$U_{12} = x_{11}v_{12} + x_{12}v_{22} + \cdots + x_{1m}v_{m2}$$
$$\vdots$$
$$U_{n_1 r} = x_{n_1 1}v_{1r} + x_{n_1 2}v_{2r} + \cdots + x_{n_1 m}v_{mr}$$
$$U_{(n_1+1)1} = x_{(n_1+1)1}v_{11} + x_{(n_1+1)2}v_{21} + \cdots + x_{(n_1+1)m}v_{m1}$$
$$\vdots$$
$$U_{n(r-1)} = x_{n1}v_{1(r-1)} + x_{n2}v_{2(r-1)} + \cdots + x_{nm}v_{m(r-1)}$$
$$U_{nr} = x_{n1}v_{1r} + x_{n2}v_{2r} + \cdots + x_{nm}v_{mr}$$

As seen from the equations above, to compute n_1 *x* r and n_2 *x* r entries of U_j, company P and Q do not need each other's data, respectively, because data is horizontally partitioned. However, to compute the remaining entries, they need each other's data. They can find such entries of U_j without jeopardizing their privacy using the following privacy-preserving protocol (called **Protocol II**), which is a variant of the one proposed by [45]:

i. Both parties first compute $s_i^{-1}v_i$ for $i = 1 \ldots r$ and find V'_j matrices.

ii. Data holders then horizontally divide m *x* r size V'_j matrices into w sub-matrices, whose size is (m / w) *x* r. Note that there are r column vectors in each sub-matrices. To improve privacy, each party disguises data in each sub-matrix independently. If the other party learns data in one of the sub-matrices, it will not be able to obtain data hidden in the remaining sub-matrices.

iii. For each sub-matrix w, each party conducts the followings:

1. Permutes all column vectors using a permutation function Π_j, where j is P or Q. Then, divides each permuted column vector, $\Pi_j (I_i)$, where $i = 1, \ldots, r$, into z_d random vectors, where $\Pi_j(I_i) = \sum_{d=1}^{z_d} Z_d$; z_d is a uniform random integer from a range $[1, \beta_j]$ and Z_d represents random vectors for $d = 1, \ldots, z_d$. Since parties do not know each other's β values, they will not be able to learn z_d values, as well.

2. Permutes all the randomly divided column vectors using a permutation function π_j. After permuting them, data holders exchange them.

3. Each party computes the scalar products between corresponding parts of data they hold and the received vectors to find the entries of the matrix U_j. Due to permutations and random division, they are not able to learn each other's data.

4. After finding scalar products, they encrypt the results with their public keys using homomorphic encryption schemes and exchange the encrypted aggregate values. Due to encryption, data holders cannot decrypt the encrypted values, because they can be decrypted only by using the other party's private key.

5. Since data owners know the permutation functions and the division process, they can calculate the final scalar product values using the homomorphic encryption property.

iv. After finding encrypted scalar product results for each sub-matrix, companies can now find the final scalar product results using homomorphic encryption property again because V'_j matrices were horizontally divided into sub-matrices.

v. Since those final scalar product results are encrypted by the other party, data owners should exchange them to get them decrypted. However, after decryption, each party can derive information about each other's data. Therefore, each party generates large enough random numbers, encrypts them with other party's public key using homomorphic encryption schemes, and adds them to scalar product results. They then exchange them.

vi. Each party decrypts the encrypted values and exchanges them. Since scalar product results are perturbed with random numbers, which are known by the other party, each party is not able to obtain the true scalar product results.

vii. Finally, each party subtracts the added random numbers from disguised scalar product results and finds the true scalar product results.

Using the **Protocol II**, the parties will be able to find the column vectors of U_j and they can form the estimated U_j matrices without learning true values in

V'_j matrices. Although they will not be able to learn the true values reside in V'_j matrices, they can learn the values in each other's user-item matrices. Therefore, to overcome this problem, they can use **Protocol III**, similar scheme to the **Protocol I,** as follows:

i. Data owners decide an integer value δ, where $0 < \delta \leq 100$, where $0 < \gamma \leq 100$.

ii. They then uniformly randomly select an integer value δ_j over the range $(0, \delta]$.

iii. They finally uniformly randomly select δ_j% of the filled cells in their user-item matrices, and remove them.

They find scalar products based on the new user-item matrices. Due to the **Protocol III** or δ_j values and randomly removed values, data owners will not be able to derive data about each other's user-item matrices from scalar product results.

After secure data exchanging, both parties now have the estimated U_j, S_j, and V_j matrices. They then find required matrix multiplications to offer referrals. Finally, they start providing CF services based on these resultant matrices.

4.2.2 Vertically Partitioned Data-based Collaborative Filtering Using SVD with Privacy

In vertical partitioning, company P and company Q own ratings matrices with sizes $n \times m_1$ and $n \times m_2$, respectively, where $m = m_1 + m_2$. User-item matrices held by P and Q can be shown, as follows:

$$A_{PV} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m_1} \\ x_{21} & x_{22} & \cdots & x_{2m_1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm_1} \end{bmatrix} \text{ and}$$

$$A_{QV} = \begin{bmatrix} x_{1(m_1+1)} & x_{1(m_1+2)} & \cdots & x_{1m} \\ x_{2(m_1+1)} & x_{2(m_1+2)} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n(m_1+1)} & x_{n(m_1+1)} & \cdots & x_{nm} \end{bmatrix}.$$

Data owners want to conduct SVD-based CF services using their combined data. In the following, it is shown how to handle four issues mentioned before when data is vertically partitioned between two vendors.

a. Removing Sparsity: Since data is vertically partitioned, both parties can easily compute item or column mean values without the need of other party's data. They fill empty cells in their databases with corresponding item mean values and remove sparsity.

b. Normalization: After removing sparsity, data owners have dense user-item matrices. Since user means are needed to normalize the ratings and data is vertically partitioned, data owners should find and exchange the sum of the ratings in each row. They do not need to send the number of values available in each row, because user-item matrices held by them are filled; and m_1 and m_2 are known by them. The sum of the ratings in each row is aggregated. It is difficult to derive true ratings and rated items from aggregate values. Data owners do not know the item mean values for those items held by each other, number of filled cells, and which cells are filled. Therefore, they will not be able to learn the rated items and their true ratings from these exchanged aggregate values.

c. A^TA Computation: After finding filled and normalized user-item matrices, the next step is to calculate the matrix A^TA. In horizontal partitioning, to compute the each entry of the matrix A^TA, both parties' data is needed. However, in vertical partitioning, some entries of the matrix A^TA can be computed from data holders' data alone, while the remaining entries are computed based on both parties' data. The entries of A^TA can be computed, as follows:

$$X_{11} = x_{11}x_{11} + x_{21}x_{21} + \cdots + x_{n1}x_{n1}$$
$$X_{12} = x_{11}x_{12} + x_{21}x_{22} + \cdots + x_{n1}x_{n2}$$
$$\vdots$$
$$X_{1m_1} = x_{11}x_{1m_1} + x_{21}x_{2m_1} + \cdots + x_{n1}x_{nm_1}$$
$$X_{1(m_1+1)} = x_{11}x_{1(m_1+1)} + x_{21}x_{2(m_1+1)} + \cdots + x_{n1}x_{n(m_1+1)}$$
$$\vdots$$
$$X_{n(m-1)} = x_{1m}x_{1(m-1)} + x_{2m}x_{2(m-1)} + \cdots + x_{nm}x_{n(m-1)}$$
$$X_{nm} = x_{1m}x_{1m} + x_{2m}x_{2m} + \cdots + x_{nm}x_{nm}$$

As seen from the equations above, in vertical partitioning, $(m/2) \times (m/2+1)/2$ values can be computed by the data holders by themselves using their own data only. To compute $(m/2)^2$ entries, both data owners' data is involved. The remaining $(m/2) \times (m/2+1)/2$ entries can be calculated using only the other party's data. Therefore, in the view of one party, it can compute some entries without the need of the other party, to compute some entries, it needs only the other party's data, and to calculate the remaining entries, both parties' data is needed. The privacy-preserving scheme for vertical partitioning to compute A^TA can be explained, as follows:

1. For those scalar product results, which do not require the other party's data, data holders can easily compute them by themselves using their own data only. They do not need each other's data.

2. To compute the scalar products that require the other party's data only, data owners employ the **Protocol I**. Using the **Protocol I**, each party computes such aggregates and sends them to the other party. As a result, both parties will have such values to compute the matrix A^TA. Due to the randomly selected values and randomly removed cells, the parties will not be able to derive information from aggregates calculated using the **Protocol I**.

3. For those scalar product computations requiring both parties' data, the parties horizontally divide their $n \times m_1$ and $n \times m_2$ size matrices into t sub-matrices. Remember that there are m_1 and m_2 column vectors in data holders' user-item matrices. They can use the similar idea with the **Protocol II**. They follow the steps *iii* through *vii* of the **Protocol II** to calculate the remaining entries of A^TA requiring both parties' data. However, they can derive data about each other's data sets from the scalar product results. Therefore, before starting computing scalar products, they employ the **Protocol III**. After removing some of the filled cells in their user-item matrices, they finally compute scalar products based on new user-item matrices.

Due to the **Protocol II** and the **Protocol III,** data owners will not be able to learn information about each others' data.

d. Computation of the column vectors of U_j: The final step in SVD computation is finding the column-vectors of matrix U_j using $b_i = s_i^{-1}Rv_i$ for $i = 1$

… r and arranging them to form the matrix U_j with size n x r. Both parties already know the estimated S_j and V_j matrices. Using such known information and A, which vertically partitioned; they can calculate the entries of U_j, as follows,

$$U_{11} = x_{11}v_{11} + \cdots + x_{1m_1}v_{m_1 1} + x_{1(m_1+1)}v_{(m_1+1)1} + \cdots + x_{1m}v_{m1}$$
$$U_{12} = x_{21}v_{11} + \cdots + x_{2m_1}v_{m_1 1} + x_{2(m_1+1)}v_{(m_1+1)1} + \cdots + x_{2m}v_{m1}$$
$$\vdots$$
$$U_{n(r-1)} = x_{n1}v_{1(r-1)} + \cdots + x_{nm_1}v_{m_1(r-1)} + x_{n(m_1+1)}v_{(m_1+1)(r-1)} + \cdots + x_{nm}v_{m(r-1)}$$
$$U_{nr} = x_{n1}v_{1r} + \cdots + x_{nm_1}v_{m_1 r_1} + x_{n(m_1+1)}v_{(m_1+1)r} + \cdots + x_{nm}v_{mr}$$

Company P and company Q do not need the other party's data to find the first m_1 and last m_2 pieces of the equations above, respectively. To calculate the remaining items, they need each other's data. The parties can use the same privacy-preserving scheme for computation of the column vectors of U_j when data is horizontally partitioned. They follow the same steps as in the computation of the column vectors of U_j when data is horizontally partitioned. They can use the **Protocol II** and the **Protocol III** to calculate the values requiring both parties' data. However, there is one difference. Instead of dividing the V'_j matrices, party P horizontally divides V'_P matrix's lower half (the m_2 x r sub-matrix of V'_P including rows from $m_1 + 1$ to m), while party Q horizontally divides V'_Q matrix's upper half (the m_1 x r sub-matrix of V'_Q including rows from 1 to m_1). The entries of U_j matrices then can be computed by following the same steps that are used to compute the entries of the U_j matrices requiring both parties' data when data is horizontally partitioned.

In both HPD- and VPD-based schemes, after estimating U_j, S_j, and V_j matrices, the parties can start providing SVD-based CF services to their customers based on the integrated data. Although the parties will not get the same U_j, S_j, and V_j matrices, because they exchange aggregate values estimated from their data items, they will get models constructed from integrated data sets, rather than split data sets alone. Since partitioned data is combined, those companies having problems with available data will overcome the problems caused by inadequate data.

4.2.3 Providing Collaborative Filtering Services on Integrated Data

After estimating the SVD-based models for CF, the companies can now start providing predictions for single items and top-*N* recommendations to their customers. Note that after combining data, each party is able to provide CF services to more users and for more items. Moreover, such services offered based on combined data are likely to be more accurate and dependable. To get a prediction for a target item q, a user u sends a query to one of the data owners. The party computes the prediction using the estimated model from combined data with privacy concerns; and sends it to the user. To get top-*N* recommendations, a user u sends a query to one of the parties stating she is looking for top-*N* recommendations. The party computes referrals for all unrated items by that user u, sorts them decreasingly, and sends the first *N* items as top-*N* recommendations. In conclusion, data owners can provide predictions and top-*N* recommendations using proposed SVD-based privacy-preserving schemes.

The proposed schemes are analyzed in terms privacy, accuracy, and efficiency in the following. Those schemes should prevent the data owners from learning the true ratings and/or rated items in each other's databases. The proposed schemes should not cause too much online additional costs, such as storage, communication, and computation costs. And finally, recommendations generated based on integrated data with privacy concerns should be as close as possible to those ones generated on combined data without privacy concerns.

4.3 Privacy Analysis

The proposed HPD- and VPD-based schemes are analyzed separately. For each of them, removing sparsity, normalization, and computation of A^TA and U matrices in terms of privacy protection are scrutinized. Since three different protocols are proposed to use, while analyzing these schemes in terms of security, the proposed protocols are also analyzed.

4.3.1 Analysis of HPD-based Schemes

Data holders are not able to derive information about each other's data sets during removing sparsity due to the following reasons. They do not know the true

ratings, the rated items in each other's data sets, and the mean votes for users that each other hold. Moreover, they do not know θ_j values and filled cells selected uniformly randomly by each other. Therefore, it becomes difficult to derive the true ratings and the rated items from exchanged aggregate values. Note that such aggregates are the sum of ratings including the inserted user mean ratings for each item and the number of ratings including the inserted ones in each column. Data owners are able to guess θ_j values and the randomly selected cells with probabilities 1 out of θ and 1 out of $C\left(\begin{pmatrix} n_j \times m \\ \left(\frac{\theta_j n_j \times m}{100}\right) \end{pmatrix}\right)$, respectively, where $C\begin{pmatrix} f \\ g \end{pmatrix}$ represents the number of ways of picking g unordered outcomes from f possibilities. However, such probabilities are small and even if they guess them, they will not be able to learn the true ratings from sum of such ratings.

The parties can normalize their ratings without the need of each other's data. Therefore, during normalization, they cannot derive information about each other's ratings and the rated items.

The computation of A^TA matrix is secure due to the **Protocol I**. The security of the **Protocol I** can be explained, as follows: The parties do not know the γ_j value selected randomly by each other. Moreover, since they do not know the filled cells in each other's data sets, they do not know the items representing two filled cells multiplication in equations to calculate X_{ijP} and X_{ijQ} values. And finally, even when they guess γ_j values, they will not learn the removed items randomly selected γ_j% of the all items from equations to calculate X_{ijP} and X_{ijQ} values. Data owners are able to guess γ_j values and the randomly selected and removed items with probabilities 1 out of γ and 1 out of $C\left(\begin{pmatrix} n_j \times \left(\frac{m^2+m}{2}\right) \\ \left(\left(\frac{\gamma_j n_j}{100}\right) \times \left(\frac{m^2+m}{2}\right)\right) \end{pmatrix}\right)$, respectively. To further improve security, data holders might decide to select γ_j values independently for each equation. In this case, the parties should guess γ_j values for each equation.

The parties find U_j matrices without jeopardizing their privacy due to the **Protocol II** and the **Protocol III**. The **Protocol III** is secure due to the same reasons given for the **Protocol I**. The parties will not learn information about each other's data while performing the **Protocol II** due to the permutation functions (Π_j and π_j), the random division or the β_j values, encryption, and the random numbers.

4.3.2 Analysis of VPD-based Schemes

Since the parties do not need each other's data to remove sparsity, they do not exchange anything. Therefore, they can remove sparsity without violating their privacy. Furthermore, they will not be able to learn the ratings and the rated items reside in each other's data sets while normalizing their ratings. Because they filled their sparse matrices with corresponding item mean ratings and they do not know the mean ratings for items that the other party owns. Additionally, since exchanged aggregates are the sum of the all ratings including the inserted column means, it is impossible for them to figure out the rated items, the column mean values, and the true ratings.

As explained for the HPD-based schemes, during the computation of A^TA matrix, it becomes impractical for the parties to derive information about each other's data due to the **Protocol I**, **II**, and **III**.

It can be said that those proposed schemes to compute U_j matrices when data is vertically partitioned are secure due to the same reasons explained previously. Note that to find such matrices; the data holders employ the **Protocol II** and **III**, which are shown secure. Finally, data owners should update their models periodically. They can update their model when they have large enough new data so that they will not be able to derive information.

4.4 Online Additional Costs

SVD-based CF schemes have both off-line and online costs. Unlike online costs, offline costs are not critical to the overall performance. Similarly, additional off-line costs introduced due to privacy concerns or integrating split data are not critical, either. CF systems should be able to provide referrals to users in real time. Therefore, online costs are critical to the overall performance. The schemes cause

additional offline costs, but they are not critical. Therefore, additional online costs are the points of interest in this analysis.

These privacy-preserving schemes do not introduce any extra online communication costs. As in partitioned data without privacy concerns, users send a query to the data holders asking predictions for target items in these combined data-based schemes with privacy. Data owners then send predictions or recommendations to users. Similarly, the proposed schemes do not cause any additional online storage costs.

And finally, these schemes do not introduce any additional online computation costs due to privacy concerns and integrating data, because the number of multiplications required to find predictions will be the same. Remember that predictions are computed based on reduced sub-matrices. For both cases, providing predictions from split data alone and from integrated data using these proposed schemes, number of multiplications to offer a single prediction is the same and equals to k.

In conclusion, these proposed schemes do not cause any extra online communication, storage, and computation costs. By combining split data, online vendors will be able to provide more accurate and dependable referrals to more users and for more items without sacrificing on online costs using the SVD-based models estimated from integrated data while protecting their privacy.

4.5 Experiments

To assess the overall performance of the proposed schemes, experiments are conducted based on real data sets.

4.5.1 Datasets and Evaluation Metrics

Jester and MLP datasets are utilized in the following trials. To evaluate the schemes in terms of accuracy, ARE and MAE are employed.

4.5.2 Methodology

The whole MLP dataset is used in the experiments. However, since Jester is a large data set, 1,600 users are randomly selected and their ratings are used for the experiments. For testing, 10% of the ratings are randomly selected, their

entries are replaced with null, and predictions are tried to be found for them. Then the predictions found for them are compared with their true withheld ratings. Finally, the MAEs and the AREs are computed, and the final values are displayed. For the experiments, k is set at 14, which happens to be the optimum value [51].

4.5.3 Experimental Results

Experiment I. Firstly, experiments are conducted to show how integrating partitioned data affect accuracy without privacy concerns. For this purpose, trials are performed to show how accuracy changes with combining varying amounts of HPD. As expected, accuracy improves with increasing number of users (n). It is desired to show how accuracy improves by integrating HPD. Experiments are also performed when the sparse user-item matrices are not filled but normalized to show how the results change when sparse user-item matrices are used. Both data sets are used while varying number of users held by one company. It is assumed that the numbers of users held by each company are n_1 and n_2, respectively, where $n_1 = n_2$ and $n = n_1 + n_2$. Number of users belongs to first data holder n_1 is varied from 100 to 480 and 800 for MLP and Jester, respectively. Note that the integrated data contains n users' data. Therefore, experiments are conducted for split data (varying n_1) and integrated data (n). After running the trials 100 times for partitioned data sets alone and the integrated data, the MAEs are computed; and the final values are displayed in Table 4.1 and Table 4.2 for MLP and Jester, respectively. Remember that 10% of the available ratings are randomly selected for testing.

Table 4. 1 Accuracy with Varying Amounts of HPD (MLP)

n_1		**60**	**120**	**240**	**480**
Split Data	*Filled*	0.8717	0.8704	0.8320	0.8030
	Not Filled	0.8736	0.8712	0.8379	0.8052
n		**120**	**240**	**480**	**943**
Integrated Data	*Filled*	0.8704	0.8320	0.8030	0.7790
	Not Filled	0.8712	0.8379	0.8052	0.7831

As seen from Table 4.1 and Table 4.2, combining HPD improves accuracy. With increasing number of users or integrating split data, accuracy gets better when filled or not filled user-item matrices are employed. In MLP, when each data owner holds 120 users' data and merges their data, accuracy improves by 4.41%. For Jester, combining split data when n_1 is 100, accuracy improves by 3.89%. Therefore, by merging split data, besides offering recommendations to more users and for more items, accuracy improves, as well. As seen from Table 4.1 and Table 4.2, obtained results are better when the sparse user-item matrices are filled than when they are not filled. However, the point is that the results become better by joining split data.

Table 4. 2 Accuracy with Varying Amounts of HPD (Jester)

***n*1**		**50**	**100**	**200**	**400**	**800**
Split Data	*Filled*	3.9826	3.9824	3.8239	3.6908	3.6023
	Not Filled	4.0905	4.0610	3.8976	3.7572	3.6495
n		**100**	**200**	**400**	**800**	**1600**
Integrated Data	*Filled*	3.9824	3.8239	3.6908	3.6023	3.5417
	Not Filled	4.0610	3.8976	3.7572	3.6495	3.5872

Experiment II. To show how accuracy changes with combining varying amounts of VPD without privacy concerns, experiments are performed while varying number of items held by each party. The quality of recommendations gets better with increasing numbers of items (m). It is tried to show how much improvement the data owners gain by combining their VPD. For this purpose, experiments are conducted using MLP data set only because Jester has only 100 items. It is assumed that the numbers of items held by each company are m_1 and m_2, respectively, where $m_1 = m_2$ and $m = m_1 + m_2$. Then, m_1 is varied from 100 to 800 and recommendations are found based on split data sets alone and integrated data. Note again that the integrated data contains ratings of m items. Therefore, trials are performed for split data (varying m_1) and integrated data (m). The trials

are repeated for 100 times, the MAEs are calculated and the final values are displayed in Table 4.3.

Table 4. 3 Accuracy with Varying Amounts of VPD

m_1	**100**	**200**	**400**	**800**
Split Data	0.8089	0.8053	0.7992	0.7895
m	**200**	**400**	**800**	**1682**
Integrated Data	0.8053	0.7992	0.7895	0.7990

As expected, the quality of recommendations develops with merging split data. When data is vertically partitioned between parties, data holders are able to provide more accurate referrals if they integrate their data. Although the improvements in accuracy might seem to be low, it is likely to provide more dependable recommendations if partitioned data is united.

After showing how overall performance changes with combining varying amounts of partitioned data without privacy concerns, experiments are then conducted to show how these proposed privacy-preserving schemes affect the overall performance. There are various factors that affect the accuracy of the recommendations such as θ, γ, and δ values. Note that in the HPD-based schemes, empty cells, selected randomly based on θ values, are filled with user mean values. In the **Protocol I**, some items of the equations to find the entries of the A^TA matrices are randomly selected based on γ values. And finally, in the **Protocol III**, which is used to find the column vectors of U_j matrices, data holders uniformly randomly select some of the filled cells from their user-item matrices and remove them based on δ values. Since inserting user mean ratings or non-personalized ratings into some empty cells and removing ratings from some of the filled cells or values from scalar product computation equations might affect accuracy, various experiments are conducted to show how accuracy changes with varying of such factors. After deciding θ, γ, and δ values, data holders uniformly randomly select θ_j, γ_j, and δ_j values over the ranges $(0, \theta]$, $(0, \gamma]$, and $(0, \delta]$, respectively. Therefore, in the following experiments, data disguising is repeated 100 times. Each time, data holders are expected to select different θ_j, γ_j, and δ_j values; and consequently, they select different numbers of cells and different cells

to be filled or removed. After running experiments 100 times, the overall average outcomes are displayed. Moreover, the number of users (n) and items (m), and the number of test items are fixed while θ, γ, and δ values are varied, where it is assumed that n and m represent the number of users and items of integrated data, respectively.

Experiment III. To show how accuracy changes with varying θ values, experiments are conducted using both data sets while changing θ values from 0 to 100. Randomly selected 1,600 users' data are used from Jester, while the all 943 users' data are employed from MLP. Recommendations are produced for randomly selected 10% of the available ratings while varying the θ values. The trials are repeated 100 times. To calculate the MAEs and the AREs, the predictions are compared with true withheld votes. Since obtained results are similar for both data sets, the MAEs and the AREs for MLP only are displayed in Table 4.4. Note that when θ is 0, the scheme can be considered the one without privacy concerns.

Table 4. 4 Accuracy with Varying θ Values

θ (%)	**0**	**30**	**60**	**100**
MAE	0.7790	0.8072	0.8106	0.8056
ARE (%)	0.00	3.49	3.89	3.30

To protect their data against each other, data owners employ various privacy-protecting methods. With privacy concerns, accuracy might become worse. As seen from Table 4.4, with increasing θ values, accuracy becomes worse. Although up to θ being 60, the quality of recommendations diminishes with increasing θ, accuracy gets better for θ being 100. However, accuracy is still worse when θ is 100 compared to θ being 0 or without privacy concerns. To see how much accuracy the data holders lose with varying θ values, the AREs are also computed. Such values show that accuracy losses due to θ or privacy concerns are small and the AREs are usually less than 4% for all θ values. The reason why the results get better when θ is bigger than 60 can be explained, as follows: In the long run, inserting user mean values might make accuracy better because they are considered as default votes and filled user-item matrices give better results than

not filled ones. Although the results are better when the whole user-item matrix is filled with default votes, accuracy becomes worse when some of the empty cells filled with such votes. In conclusion, e-companies can decide the value of θ to achieve a required level of privacy and accuracy.

Experiment IV. γ is another factor that affects accuracy. To assess the overall performance of these schemes with varying γ values, experiments are conducted using both data sets. In these experiments, 120 and 400 users are used for training randomly selected from MLP and Jester, respectively. γ is varied from 0 to 100 and found predictions for test items, which are 10% of available ratings selected randomly. The trials are repeated 100 times. Produced referrals are compared with privacy concerns with those true votes. After calculating the overall MAEs, they are displayed in Table 4.5.

Table 4. 5 Accuracy with Varying γ Values

γ (%)	0	30	60	100
MLP	0.87040	0.870439	0.87047	0.87052
Jester	3.6908	3.6909	3.6910	3.6913

Data owners uniformly randomly select some of the pieces of the equations to calculate the entries of A^TA matrix based on γ values. There are n pieces in each such equation. Note that such randomly removed pieces are those representing the multiplication of the two filled cells. As seen from Table 4.5, although accuracy becomes worse with increasing γ values, the amount of accuracy loss is negligible. These schemes achieve almost the same level of accuracy when data owners employ γ values to protect their privacy. The reason why the parties achieve such level of accuracy while protecting their privacy is that the number of pieces removed from the equations is very small compared to the total number of pieces.

Experiment V. Finally, experiments are performed while varying δ values from 0 to 100. The same training data is used as utilized in Experiment III. Predictions are generated with varying δ values for test items, they are compared with the true ratings, the MAEs are calculated, and the final average outcomes are displayed in Table 4.6.

Table 4. 6 Accuracy vs. δ Values

δ (%)	0	30	60	100
MLP	0.7790	0.7701	0.7658	0.7672
Jester	3.5417	3.5291	3.5126	3.5020

Generally it is expected that the quality of the recommendations would become worse due to privacy-protecting schemes before conducting experiments. This is generally true for experiments up to the fifth one. As seen from Table 4.6, unlike these expectations, accuracy slightly becomes better when employing δ values. Remember that data owners remove the values of the randomly chosen filled cells while they compute U matrices. For MLP, when δ is 100, accuracy worsens compared to the δ values less than or equal to 60. Although accuracy generally improves with increasing δ values, the gain is small. It is interesting why these results become better up to some δ values and start getting worse especially for MLP.

It will be deeply studied why obtained outcomes get better up to some δ values and starts getting worse after that. For now, it can be said that the data holders are able to select δ values to accomplish required level of privacy and accuracy.

4.5.4 Conclusions

It is shown that it is possible to provide accurate predictions using the proposed schemes. As explained previously, the proposed methods also allow data holders to provide top-N recommendations in addition to providing predictions for single items. Since online time to provide recommendations is critical, instead of calculating predictions for all unrated items of the user who is looking for referrals, that user can ask top-N recommendations for some of her unrated items. This method improves online computation time because predictions are computed for small number of unrated items.

Data collected for CF purposes might be horizontally or vertically split between more than two parties. Schemes are proposed to provide CF services by combining partitioned data between two parties. The proposed schemes can be

extended to multi-party schemes. Since offline additional costs are not critical and the proposed schemes do not cause any supplementary online costs, by following the similar steps for two party-based schemes, it is possible to estimate SVD-based CF models from split data between more than two parties.

4.6 Summary of the Chapter

Online vendors might have insufficient data to provide CF services to their customers. It is important to offer accurate and dependable referrals to keep the current customers and to recruit the new ones. It is not always possible to offer such services from insufficient data. Data collected for CF purposes might be split between various parties and the parties might not want to combine their data due to privacy, legal, and financial reasons. However, they want to integrate their data to overcome problems caused insufficient data and to provide richer CF services if privacy measures are provided.

It is shown how to offer SVD-based CF services using integrated data without jeopardizing data owners' privacy. By combining horizontally or vertically partitioned data, it is likely to produce more accurate and trustworthy referrals. Based on combined data, online vendors can provide referrals to more users and for more items. By combining HPD, data owners can offer predictions to more users. Similarly, by integrating VPD, data holders can produce referrals for more items. The experiment results show that e-vendors can provide referrals with decent accuracy using these privacy-preserving schemes. The schemes do not bring in any supplementary online costs, which are critical to overall performance. The schemes are analyzed in terms of privacy and it is shown that they are secure. The schemes can be used to produce top-N referrals besides offering predictions. Furthermore, it is possible to extend the proposed methods to multi-party schemes. Data owners are able to select θ, γ, and δ values in such a way to achieve equilibrium between accuracy and privacy. Therefore, proposed schemes make it possible to find a good balance between accuracy and privacy. Since such values affect both accuracy and privacy, the parties can select them based on the required levels of accuracy and privacy.

5. CONCLUSIONS AND FUTURE WORK

In this thesis, how to preserve privacy while still produce accurate recommendations using dimensionality reduction-based CF algorithms is investigated. Various privacy-preserving schemes are proposed and it is shown that they are secure and are able to generate referrals with decent accuracy. Conventional model-based CF algorithms are modified considering privacy risks. To provide recommendation services to real world internet users having different privacy concerns, some discussions are taken place and methods are introduced. Furthermore, providing recommendations on distributed data while ensuring data holders' privacy are also investigated. In this chapter, the study is concluded and the future directions are introduced.

5.1 Challenges and Results

During the study, the key point is to provide satisfactory predictions while protecting users' privacy. At the same time, online computation time must be considered as a critical factor. To achieve conflicting goals such as privacy, accuracy, and efficiency together is the major challenge. However, optimum methods balancing the response level of these goals can be a solution. For this reason, it is demonstrated that randomization techniques can be employed. By this way, privacy is preserved with little additional communication and computation costs; but some accuracy losses in negligible amount should be expected.

The study and its results can be briefly explained, as follows. First of all, it is shown that private recommendations with decent accuracy can be generated using Eigentaste-based algorithms [56]. RPT are applied to Eigentaste-based algorithms to provide referrals while protecting users' privacy. The algorithm is modified in order to offer predictions on masked data. To evaluate the proposed schemes, experiments are executed on real datasets and satisfactory results are obtained.

Secondly, it is shown how to provide referrals without violating users' privacy based on variably masked data using SVD-based CF algorithms [57]. Users' concerns about their privacy might differ among various users and there

are different factors affecting users' attitudes towards privacy. That is why they might decide to perturb their data differently. Various data disguising ways are explained. It is shown that it is still possible to offer accurate referrals on variably masked data. Several experiments are performed and their results are analyzed.

Finally, to solve insufficient data problem in SVD-based recommendation algorithms, methods running on horizontally or vertically privately integrated model are introduced and discussed. The proposed schemes make it possible for online vendors to combine their integrated data to provide richer CF services without violating their privacy. They are examined in terms of accuracy by performing real data-based trials. Moreover, it is shown that they are secure and they do not introduce significant overhead costs.

5.2 Future Directions

In each chapter, related future directions are briefly introduced. Apart from them, this section presents future directions discussed in three categories: methodology, privacy, and applicability issues.

At first, in methodological perspective; there remains works to be done to study whether it is possible to apply the proposed solutions to other memory or model-based CF algorithms or not. For example, randomized perturbation techniques can be applied to discrete wavelet transformation (DWT)-based recommendation algorithms because DWT might handle masked data effectively. Additionally, in the second chapter, *k*-means clustering algorithm can be optimized on masked data or other clustering algorithms can be employed to obtain better results. The proposed two-party schemes can be extended to multi-party schemes. It should be extensively investigated how to extend the proposed schemes to multi-party methods. In addition to horizontal and vertical partitioning, data might be hybrid partitioned between parties. It should be studied whether there are privacy-preserving schemes for hybrid distributed data or not. It is assumed that no overlapping occurs between users or items when data is distributed. However, overlapping may take places in practical applications. Thus, handling overlapping data can be another future work.

Secondly, schemes can be extended considering privacy and security enhancing issues. A recommender system might use value of information (VOI) metric to bound the amount of information collected about a user to some optimal level with respect to both privacy and recommendation quality. The schemes can be enhanced with VOI metric. By this way, no redundant personal information would be needed to collect. Moreover, in all of the studies, users' trustworthiness is out of interest. Considering malicious user existence would get interesting discussions grow up.

Finally, some directions about the applicability of the schemes can be introduced. The sparsity of the input data affects accuracy. In the discussions, sparsity is not taken place. Some studies investigating the data masking and sparsity together would be one of the future topics. Furthermore, server-based CF systems have been very successful and useful in e-commerce and in direct recommendation applications. The schemes can be employed on server-based applications, effectively. However, after designing systems based on the proposed schemes and providing referrals to internet users, some surveys and studies should be done on users' trust of these kinds of systems and the quality of information collected by this way.

REFERENCES

[1] Agrawal D. and Aggarwal C. C., "On the design and quantification of privacy-preserving data mining algorithms", *Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, Santa Barbara, CA, USA, 247-255, 2001.

[2] Agrawal R. and Srikant R., "Privacy-preserving data mining", *Proceedings of the 2000 ACM SIGMOD on Management of Data*, Dallas, TX, USA, 439-450, 2000.

[3] Ahmad W. and Khokhar A., "An architecture for privacy-preserving collaborative filtering on Web portals", *Proceedings of the 3rd International Symposium on Information Assurance and Security*, Manchester, England, 273-278, 2007.

[4] Armstrong R., Freitag D., Joachims T., and Mitchell T., "Webwatcher: A learning apprentice for the World Wide Web", *Proceedings of the AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments*, Stanford, CA, USA, 6-12, 1995.

[5] Balabanovic M. and Shoham Y., "Fab: Content-based collaborative recommendation", *Communications of the ACM*, **40(3)**, 66-72, 1997.

[6] Benaloh J., "Dense probabilistic encryption", *Proceedings of the Workshop on Selected Areas of Cryptography*, Kingston, Ontario, Canada, 120-128, 1994.

[7] Berkovsky S., Eytani Y., Kuflik, T., and Ricci, F., "Privacy-enhanced collaborative filtering" *Proceedings of the PEP05, UM05 Workshop on Privacy-Enhanced Personalization*, Edinburgh, UK, 75-84, 2005.

[8] Berkovsky S., Eytani Y., Kuflik, T., and Ricci, F., "Hierarchical neighborhood topology for privacy-enhanced collaborative filtering" *Proceedings of the Workshop on Privacy-Enhanced Personalization,* Montreal, Canada, 6-13, 2006.

[9] Borchers A., Herlocker J., Konstan J., and Riedl J. T., "Ganging up on information overload", *IEEE Computer*, **31(4)**, 106-108, 1998.

[10] Brassard G., Crépeau C. and Robert J. "All-or-nothing disclosure of secrets", *Lecture Notes in Computer Science*, **263**, 234-238, 1987.

[11] Breese J., Heckerman D., and Kadie C., "Empirical analysis of predictive algorithm for collaborative filtering", *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence,* Madison, WI, USA, 43-52, 1998.

[12] Canny J. "Collaborative filtering with privacy via factor analysis", *Proceedings of the 25th ACM SIGIR'02 Conference*, Tampere, Finland, 238-245, 2002.

[13] Canny J., "Collaborative filtering with privacy", *Proceedings of IEEE Symposium on Security and Privacy*. Berkeley, CA, USA, 45-57, 2002.

[14] Chee S. H. S., Han J., and Wang K., "RecTree: An efficient collaborative filtering method", *Lecture Notes in Computer Science,* **2114**, 141-151, 2001.

[15] Chien Y. H. and George E. I., "A Bayesian model for collaborative filtering", *Proceedings of the 7th International Workshop on Artificial Intelligence and Statistics*, San Francisco, CA, USA, 1999.

[16] Cho J., Kwon K., and Park Y., "Collaborative filtering using dual information sources", *IEEE Intelligent Systems*, **22(3)**, 30-38, 2007.

[17] Cranor L. F., Reagle J., and Ackerman M. S. *Beyond concern: Understanding net users' attitudes about online privacy*, AT&T Labs-Research Technical Report, No: 99.4.3, 1999.

[18] Cranor L. F., " 'I didn't buy it for myself' privacy and e-commerce personalization", *Proceedings of the 2003 ACM Workshop on Privacy in the Electronic Society*, Washington, DC, USA, 111-117, 2003.

[19] Ding C. and He X., "*K*-means clustering via principal component analysis", *Proceedings of the 21th International Conference on Machine Learning*, Banff, Alberta, Canada, 225-232, 2004.

[20] Even S., Goldreich O., and Lempel A., "A randomized protocol for signing contracts", *Communications of the ACM*, **28(6)**, 637-647, 1985.

[21] Foner L., "Yenta: A multi-agent, referral-based matchmaking system", *Proceedings of the 1st International Conference on Autonomous Agents*, Marina Del Rey, CA, USA, 301-307, 1997.

[22] Goldberg K., Roeder T., Gupta D., and Perkins C., "Eigentaste: A constant time collaborative filtering algorithm", *Information Retrieval*, **4(2)**, 133-151, 2001.

[23] Gupta D., Digiovanni M., Narita H., and Goldberg K. "Jester 2.0: A new linear-time collaborative filtering algorithm applied to jokes", *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Berkeley, CA, USA, 291-292, 1999.

[24] Hand D., Mannila H., and Smyth P., *Principles of Data Mining,* MIT Press, Cambridge, MA, USA, 2001.

[25] Herlocker J. L., Konstan J. A., Borchers A., and Riedl J. T., "An algorithmic framework for performing collaborative filtering", *Proceedings of the 22nd ACM SIGIR'99 Conference*, Berkeley, CA, USA, 230-237, 1999.

[26] Hill W., Stead L., Rosenstein M., and Furnas G., "Recommending and evaluating choices in a virtual community of use", *Proceedings of ACM Conference on Human Factors in Computing Systems CHI'95*, Denver, CO, USA, 194-201, 1995.

[27] Kargupta H., Datta S., Wang Q., and Sivakumar K. "Random data perturbation techniques and privacy-preserving data mining", *Proceedings of the 3rd IEEE International Conference on Data Mining*, Melbourne, FL, USA, 99-106, 2003.

[28] Kautz H., Selman B, and Shah M., "Combining social networks and collaborative filltering", *Communications of the ACM*, **40(3)**, 63-65, 1997.

[29] Kim D. and Yum B. "Collaborative filtering based on iterative principal component analysis" *Expert Systems with Applications*, **28(4)**, 823-830, 2005.

[30] Ko S. J. and Lee J. H., "Discovery of user preference through genetic algorithm and Bayesian categorization for recommendation", *Lecture Notes in Computer Science,* **2465**, 471-484, 2001.

[31] Lam S. K., Frankowski D., and Riedl J. T., "Do you trust your recommendations? An exploration of security and privacy issues in recommender systems", *Lecture Notes in Computer Science,* **3995**, 14-29, 2006.

[32] Lindell Y. and Pinkas B., "Privacy-preserving data mining", *Lecture Notes in Computer Science*, **1880**, 36-54, 2000.

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

[34] Maltz D. and Ehrlich K., "Pointing the way: Active collaborative filtering", *Proceedings of ACM Conference on Human Factors in Computing Systems*, Denver, CO, USA, 202-209, 1995.

[35] Miller B. N., Riedl J. T., and Konstan J. A., "Experiences with GroupLens: Making Usenet useful again", *Proceedings of the USENIX 1997 Annual Technical Conference*, Anaheim, CA, USA, 219-233, 1997.

[36] Miller B. N., Albert I., Lam S., Konstan J. A., and Riedl J. T., "MovieLens unplugged: Experiences with an occasionally connected recommender system", *Proceedings of ACM 2003 International Conference on Intelligent User Interfaces*, Miami, FL, USA, 263-266, 2003.

[37] Mobasher B., Jin X., and Zhou Y., "Semantically enhanced collaborative filtering on the Web", *Lecture Notes in Computer Science*, **3209**, 57-76, 2004.

[38] Naccache D. and Stern J., "A new public key cryptosystem based on higher residues", *Proceedings of the 5th ACM Conference on Computer and Communications Security*, San Francisco, CA, USA, 59-66, 1998.

[39] Okamoto T. and Uchiyama S., "A new public-key cryptosystem as secure as factoring", *Proceedings of EUROCRYPT 1998*, Espoo, Finland, 308-318, 1998.

[40] Paillier P., "Public-key cryptosystems based on composite degree residuosity classes", *Proceedings of EUROCRYPT 1999*, Prague, Czech Republic, 223-238, 1999.

[41] Parameswaran R. and Blough D. M., "Privacy-preserving collaborative filtering using data obfuscation" *Proceedings of the 2007 IEEE International Conference on Granular Computing*, Silicon Valley, CA, USA, 380-386, 2007.

[42] Polat H. and Du W. "Privacy-preserving collaborative filtering" *International Journal of Electronic Commerce*, **9(4)**, 9–36, 2005.

[43] Polat H. and Du W., "SVD-based collaborative filtering with privacy", *Proceedings of the 20th ACM Symposium on Applied Computing, Special Track on E-commerce Technologies*, Santa Fe, NM, USA, 791-795, 2005.

[44] Polat H. and Du W. "Effects of inconsistently masked data using RPT on CF with privacy", *Proceedings of the 22nd ACM Symposium on Applied Computing,Special Track on E-commerce Technologies*, Seoul, Korea, 649-653, 2007.

[45] Polat H. and Du W., "Privacy-preserving collaborative filtering on vertically partitioned data", *Lecture Notes in Computer Science*, **3721**, 651-658, 2005.

[46] Polat H. and Du W., "Privacy-preserving top-*N* recommendation on horizontally partitioned data", *Proceedings of International Conference on Web Intelligence*, Compiegne, France, 725-731, 2005.

[47] Resnick P., Iacovou N., Suchak M., Bergstrom P., and Riedl J. T., "GroupLens: An open architecture for collaborative filtering of netnews", *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, Chapel Hill, NC, USA, 175-186, 1994.

[48] Resnick P. and Varian H.R., "Recommender systems", *Communications of the ACM*, **40(3)**, 56-58, 1997.

[49] Russell S. and Yoon V., "Applications of wavelet data reduction in a recommender system" *Expert Systems with Applications***, 34(4)**, 2316-2325, 2008.

[50] Sarwar B. M., Karypis G., Konstan J. A., and Riedl J. T., "Application of dimensionality reduction in recommender system – A case study", *Proceedings of the ACM WebKDD 2000 Web Mining for E-commerce Workshop*, Boston, MA, USA, 2000.

[51] Sarwar B. M., Konstan J. A., Borchers A., Herlocker J. L., Miller B. N., and Riedl J. T., "Using filtering agents to improve prediction quality in the GroupLens research collaborative filtering system", *Proceedings of the 1998 ACM Conference on Computer Supported Cooperative Work*, Seattle, WA, USA, 345-354, 1998.

[52] Shardanand U. and Maes P., "Social information filtering: Algorithms for automating 'word of mouth' ", *Proceedings of ACM Conference on Human Factors in Computing Systems*, Denver, CO, USA, 210-217, 1995.

[53] Terveen L., Hill W., Amento B., McDonald D., and Creter J. "Phoaks: A system for sharing recommendations", *Communications of the ACM*, **40(3)**, 59-62, 1997.

[54] Vozalis M. G. and Margaritis K. G., "Using SVD and demographic data for the enhancement of generalized collaborative filtering", *Information Sciences*, **177(15)**, 3017-3037, 2007.

[55] Wang J., Pouwelse J., Lagendijk R. L., and Reinders M. J., "Distributed collaborative filtering for peer-to-peer file sharing systems", *Proceedings of Symposium on Applied Computing 2006*, Dijon, France, 1026-1030, 2006.

[56] Yakut I. and Polat H., "Privacy-preserving Eigentaste-based collaborative filtering", *Lecture Notes in Computer Science*, **4752**, 169-184, 2007.

[57] Yakut I. and Polat H., "Achieving private SVD-based recommendations on inconsistently masked data", *Proceedings of International Conference on Security of Information and Network*, Gazimagusa, TRNC, 172-176, 2007.

[58] Zhang S., Ford J., and Makedon F., "A privacy-preserving collaborative filtering scheme with two-way communication", *Proceedings of the 7th ACM Conference on Electronic Commerce,* Ann Arbor, MI, USA, 316-323, 2006.