Privacy Preserving Neighbourhood-based Collaborative Filtering Recommendation Algorithms

Zhigang Lu

School of Computer Science

The University of Adelaide

A thesis submitted for the degree of

Master of Philosophy

August, 2015

To my loving parents and fiancée …

Declaration

I , Zhigang Lu, certify that this work contains no material which has been accepted for the award of any other degree or diploma in my name, in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. In addition, I certify that no part of this work will, in the future, be used in a submission in my name, for any other degree or diploma in any university or other tertiary institution without the prior approval of the University of Adelaide and where applicable, any partner institution responsible for the joint-award of this degree.

I give consent to this copy of my thesis when deposited in the University Library, being made available for loan and photocopying, subject to the provisions of the Copyright Act 1968. The author acknowledges that copyright of published works contained within this thesis resides with the copyright holder(s) of those works.

I also give permission for the digital version of my thesis to be made available on the web, via the University's digital research repository, the Library Search and also through web search engines, unless permission has been granted by the University to restrict access for a period of time.

Signature: _____ Date: _____

Acknowledgements

This MPhil study was a truly challenge and amazing life experience for me, it would not be possible without the support and guidance that I received from many people.

First and foremost, I would like to express my special appreciation and thanks to my principal supervisor, Prof. Hong Shen, for his great and patient supervision over the past two years. Under Prof. Shen's supervision, I have learned how to write research paper, how to make academic presentation, how to be a teaching assistant and so on, but the most important thing I learned from Prof. Shen is how to think as a researcher, such as the ability to find unsolved research problems and address them in a novel way. Moreover, Prof. Shen is the one who leads me to the filed of research, encourages me to continue my research career in coming years.

I would also like to express my gratitude to my supervisor, Dr. Nickolas Falkner, for his help, support in my research.

I own my parents and fiancée a lot, for their unconditional love, support and encourage. Especially, without the financial support from my parents, I even do not have the chance to study at the University of Adelaide.

I would like to thank some of the university staff, my colleagues and friends who provide the selfless help during my two years MPhil study: Ms. Jo Rogers, Ms. Julie Mayo, Ms. Sharyn Liersch, Ms. Lenka Hill, Ms. Sue Fiedler, Dr. Yuval Yarom, Ms. Alison Black, Ms. Rosie Wilkes, Dr. Crusher Wong, Dr. Kewen Liao, Yongrui, Dr. Xiaoqiang Qiao, Yihong, Scott, Ali, Javier, Zheng, Jack, Dr. Yingpeng Sang, Lingjing, Dr. Sergey Polyakovskiy, Dr. Jian Kang, Dr. Mingkui Tan, Wenjie, Dung.

Finally, I thank the School of Computer Science for financially supporting my conference travels. Thanks also go to the staff in Adelaide Graduate Centre and the anonymous examiners for their efforts on this thesis.

Preface

During my two years MPhil study at the University of Adelaide, I have produced one journal article and two conference papers related to this thesis. The thesis is based on the content presented in the following papers.

Journal Publication:

- Zhigang Lu and Hong Shen. (2015). An accuracy-assured privacy-preserving recommender system for Internet commerce. *Computer Science and Information Systems*. (minor revision)

Conference Publications:

- Zhigang Lu and Hong Shen. (2015). A security-assured accuracy-maximised privacy preserving collaborative filtering recommendation algorithm. In *Proceedings of the 19th International Database Engineering & Applications Symposium*, IDEAS'15, pages 72-80, New York, NY, USA. ACM.
- Zhigang Lu and Hong Shen. (2015). A fast algorithm to build new users similarity list in neighbourhood-based collaborative filtering. to appear in *Proceedings of the 16th International Conference on Parallel and Distributed Computing, Applications and Technologies*, PDCAT'15, Jeju, Korea.

Abstract

Recommender systems, which recommend users the potentially preferred items by aggregating similar interest neighbours' history data, show an increasing importance in various Internet applications. As a well-known method in recommender systems, neighbourhood-based collaborative filtering has received considerable attention recently because of its easy implementation and high recommendation accuracy. However, the risks of revealing customers' privacy during the process of filtering have attracted noticeable public concern. Specifically, the *k*NN attack discloses the target user's sensitive information by creating *k* fake nearest neighbours by non-sensitive information. Among the current solutions against the *k*NN attack, probabilistic methods showed a powerful privacy preserving effect. However, the existing probabilistic methods neither guarantee enough prediction accuracy due to the global randomness, nor provide assured security enforcement against the *k*NN attack.

To overcome the problem of recommendation accuracy loss, we propose a novel approach called Partitioned Probabilistic Neighbour Selection. In this thesis, we define the sum of *k* neighbours' similarity as the accuracy metric α, the number of user partitions, across which we select the *k* neighbours, as the security metric β. We consider two versions of the Partitioned Probabilistic Neighbour Selection schemes. Firstly, to ensure a required prediction accuracy while maintaining high security against the *k*NN attack, we propose an accuracy-assured Partitioned Probabilistic Neighbour Selection algorithm. We select neighbours from each exclusive partition of size *k* with a decreasing probability. Theoretical and experimental analysis show that to provide an accuracy-assured recommendation, our method yields a suitable trade-off between the recommendation accuracy and system security. Secondly, to ensure a required security guarantee while achieving the optimal prediction accuracy against the *k*NN attack, we propose a security-assured accuracy-maximised Partitioned Probabilistic Neighbour Selection algorithm. We select neighbours from each partition with exponential differential privacy to reduce the magnitude of noise. Theoretical and experimental analyses show that to achieve the same security guarantee against the *k*NN attack, our approach ensures an optimal prediction accuracy.

In addition, as the core of neighbourhood-based CF, the task of dynamically maintaining users' similarity list is challenged by the cold-start problem and the scalability problem. Recently, several methods have been proposed for solving the two problems. However, these methods require mn steps to compute the similarity list against the kNN attack, where n and m are number of users and items respectively. Observing that the k new users from the kNN attack, with enough recommendation data, share the same rating list, we present a faster algorithm, TwinSearch, to avoid computing and sorting the similarity list for each new user repeatedly to save the time complexity. The computation cost of our algorithm is $\frac{1}{125}$ of the existing methods. Both theoretical and experimental results show that the TwinSearch Algorithm achieves better running time than the traditional method.

Table of Contents

List of Figures

List of Tables

List of Symbols

$N_k(u_a)$ User u_a's k nearest neighbours set, descending order of similarity

α Accuracy metric

$\bar{r}_i$ User u_i's average rating on every rated item

β Security metric

$\mathscr{S}_a$ User u_a's similarity list

$f_\beta(i)$ The number of neighbours selected from partition No. i with the given security metric β

kNN k Nearest Neighbour

r_{is} User u_i's rating on item t_s

$sim(a,i)$ Similarity between user u_a and u_a's ith neighbour in $N_k(u_a)$

sim_{ij} Similarity between user u_i and user u_j

CF Collaborative Filtering

Chapter 1

Introduction

1.1 Research Background

1.1.1 Privacy Preserving CF Recommendation Algorithms

Recommender systems aim to predict user's potential ratings or preferences on the items which have not yet been purchased by the user [52]. Recently, an increasing importance of recommender systems has been shown in various Internet applications [23, 30, 58, 64, 66, 70, 71]. For example, Amazon has been receiving benefits for a decade from the recommender systems by recommending products to their customers, and Netflix posted a one million U.S. dollars award for improving their recommender system to make their business more profitable [17, 29, 60]. Currently, in recommender systems, Collaborative Filtering (CF) is a famous technology with three main popular techniques [35], i.e., neighbourhood-based methods [24], association rules based prediction [57], and matrix factorisation [32]. Among these techniques, neighbourhood-based methods are widely used in the industry because of the easy implementation and high prediction accuracy.

One of the most popular neighbourhood-based methods is k Nearest Neighbour (kNN) which provides recommendations by aggregating the opinions of a user's k nearest neighbours [2]. Although kNN recommender systems efficiently present very good performance on recommendation accuracy, the risk of revealing users' private information during the process of filtering is still a growing concern [76], e.g., the kNN attack presented by Calandrino et al. [9] exploits the property that the users are more similar when sharing same rating on corresponding items to reveal user's private data. Thus presenting an efficient privacy preserving neighbourhood-based CF algorithm against the kNN attack, which achieves a trade-off between the system security and recommendation accuracy, has been a natural research interest.

The literature in CF recommender systems has developed several approaches to preserve users' privacy. Generally, cryptographic, obfuscation, perturbation, probabilistic methods and differential privacy are applied [75]. Among them, cryptographic methods [18, 19, 45, 73] provide the most reliable security but the unnecessary computational cost cannot be ignored [76]. Obfuscation methods [11, 48, 49, 61, 68] and Perturbation methods [4, 5, 22, 31] introduce designed random noise into the original matrix to preserve customers' sensitive information; however the magnitude of noise is hard to calibrate in these two types of methods [16, 76]. The probabilistic methods [1] provided a similarity based weighted neighbour selection of the k nearest neighbours. Similar to perturbation, McSherry and Mironov [42] presented a naive differential privacy method which adds calibrated noise into the covariance (similarity between users/items) matrix. Similar to the probabilistic neighbour selection [1], Zhu et al. [76] proposed a Private Neighbour Selection to preserve privacy against kNN attack by introducing differential privacy in selecting the k nearest neighbours randomly (also adding noise into covariance matrix with differential privacy). Although the methods in [1, 42, 76] successfully preserve users' privacy against kNN attack, the low prediction accuracy due to the global randomness should be noted. Even worse, Zhu et al. [76] failed to maintain differential privacy in the process of neighbour selection. Therefore, none of the existing privacy preserving CF recommender systems can provide enough utility while preserving users' private information. Moreover, as privacy preserving CF recommendation algorithms, none of the existing randomised methods provide an assured security enforcement before the process of filtering.

1.1.2 Reducing Computational Cost against the kNN attack

The core of neighbourhood-based CF methods is the computation of a sorted similarity list for every user. The task of dynamically maintaining a similarity list is quite challenging in a neighbourhood-based recommender system, as the creation of new users and the rate updates of old users will result in the frequent updates of the similarity list. Accordingly, there are two main research problems in recommender systems, one is the cold-start problem, the other is the scalability problem [46]. The cold-start problem concerns the issue that the system cannot provide reliable recommendations to the new users who do not have enough rating data. Recent research [1, 7, 34, 36] on addressing the cold-start problem focus on improving the prediction accuracy with the limited rates information. While, the scalability problem concerns the point that, because of the frequent rate updates of old users, the system requires a large amount of computational cost to rebuild the similarity list for updating the recommendations. Some of the solutions [28, 37, 47, 72] to the scalability problem work on

decreasing the computational cost by incrementally updating the similarity lists from the existing lists, rather than recomputing the similarity lists by the updated ratings.

Different from the two classic problems, we notice a special case where the methods listed above do not work well. In this case, we assume the new users have enough rating data to build reliable similarity list. Additionally, the new users have totally the same ratings list (the *k*NN attack is an example for this case). The methods aim to solve the cold-start problem or the scalability problem will treat this special case as a normal input of recommender systems, then apply an $O(mn)$ algorithm to compute the new users' similarity list, where m is the number of items and n is the number of users in a recommender system. Considering the the value of m and n is usually very large, the computational cost of the above methods will be very large.

1.2 Overview of Contributions

In this thesis, we have three main works. Firstly, to overcome the problem of low prediction accuracy in existing probabilistic approaches against the *k*NN attack, we proposed an Accuracy-assured Partitioned Probabilistic Neighbour Selection algorithm. Secondly, to overcome the problems of unsatisfactory prediction accuracy and unassured security guarantee in the existing probabilistic approaches against the *k*NN attack, we proposed a Security-assured Accuracy-maximised Partitioned Probabilistic Neighbour Selection algorithm. Thirdly, to address the problem of large computational cost caused by a special case (e.g. the *k*NN attack), we proposed a faster algorithm to build new user's similarity list, TwinSearch algorithm. The main contributions of this thesis are:

In Accuracy-assured Partitioned Probabilistic Neighbour Selection algorithm:

- We expand the classic *k*NN attack to a more general case, β-*k*NN attack, which flexibly adjusts the size of fake user's set to improve the attack effectiveness. β is essentially regarded as a security measure denoting the degree of difficulty for an attacker to break the neighbourhood-based CF recommender systems. We are the first to consider the case when $\beta > 1$.

- To protect users' data privacy against β-*k*NN attack, we propose a novel differential privacy preserving neighbourhood-based CF method, which ensures a required prediction accuracy while achieving a better trade-off between the system security and recommendation accuracy against the *k*NN attack.

- To the best of our knowledge, we are the first to propose a theoretical analysis of the recommendation accuracy and system security on the recommendation results from any randomised neighbour selection methods in the neighbourhood-based CF

recommender systems. Previous related work only gave the experimental analysis on the same issues.

In Security-assured Accuracy-maximised Partitioned Probabilistic Neighbour Selection algorithm:

- We define performance metrics clearly in both prediction accuracy and system security to theoretically analyse the performance of privacy preserving CF method. Specifically, we define the sum of k neighbours' similarity as the accuracy metric α, the number of user partitions, across which we select the k neighbours, as the security metric β.

- We propose a novel differential privacy preserving method, which achieves the optimal prediction accuracy α with a given desired system security β among all of the existing developments of randomised neighbourhood-based CF recommendation algorithms.

- We show that, compared with the related methods, the proposed security-assured accuracy-maximised partitioned probabilistic neighbour selection method performs consistently well across various experimental settings. For example, we compare the accuracy performance on different datasets; we design the experiments on both user-based and item-based neighbourhood-based CF; we examine the accuracy performance in the scenario with and without kNN attack.

In TwinSearch algorithm:

- We consider a new case in CF recommender systems which may cause high computational cost for the current methods. To the best of our knowledge, we are the first to consider this special case in recommender systems.

- We design a faster algorithm, TwinSearch Algorithm, to avoid computing and sorting the similarity list for the new users repeatedly to save the computational resources. We introduced the probabilistic theory into the algorithm time complexity analysis, we prove that the computation cost of TwinSearch Algorithm is $\frac{1}{125}$ of the traditional similarity computation methods.

- We compare the running time of TwinSearch algorithm and the traditional similarity computation method in two real-world data sets on both user-based and item-based CF. The experimental results show that the TwinSearch algorithm achieves better running time than the traditional method.

1.3 Outline

The thesis is organised as follow:

- **Chapter 2: Background.** In this chapter, we briefly discuss some of the research literature in both privacy preserving neighbourhood-based CF recommender systems and the cold-start problem and the scalability problem solutions. For the privacy preserving neighbourhood-based CF recommendation methods, we categories the current methods into two groups: traditional methods and differential privacy application, according to how these methods protect customer's privacy. For the solutions to the cold-start problem and the scalability problem, we discuss their weaknesses when applying these solutions against the *k*NN attack.

- **Chapter 3: Preliminaries.** In this chapter, we firstly introduce the foundational concepts and mathematical model related with this thesis in collaborative filtering, differential privacy, and Wallenius' non-central hyper-geometric distribution. Afterwards, we introduce a popular attack, k nearest neighbour attack, then we expand its concept to a general attack, β-k nearest neighbour attack. Finally, we provide two performance metrics on the privacy preserving neighbourhood-based CF recommender systems against the β-*k*NN attack.

- **Chapter 4: An Accuracy-Assured Privacy-Preserving CF Algorithm.** In this chapter, to overcome the problem of recommendation accuracy loss in existing probabilistic methods against the *k*NN attack, we propose a novel method, Accuracy-assured Partitioned Probabilistic Neighbour Selection, to ensure a required prediction accuracy while maintaining high security against the *k*NN attack. Theoretical and experimental analysis show that to provide an accuracy-assured recommendation, our Partitioned Probabilistic Neighbour Selection method yields a better trade-off between the recommendation accuracy and system security. This chapter consists of our journal paper "An Accuracy-Assured Privacy-Preserving Recommender System for Internet Commerce".

- **Chapter 5: A Security-Assured Accuracy-Maximised Privacy-Preserving CF Algorithm.** In this chapter, to overcome the problems of unsatisfactory prediction accuracy and unassured security guarantee in the existing probabilistic methods against the β-*k*NN attack, we propose a novel method, Security-Assured Accuracy-Maximised Partitioned Probabilistic Neighbour Selection, to ensure a required system security while achieving the maximum prediction accuracy against the β-*k*NN attack. Theoretical and experimental analysis show that to achieve the same security guarantee against the β-*k*NN attack ($\beta \geq 1$), our approach ensures the optimal prediction accuracy. This

chapter consists of our conference paper "A Security-assured Accuracy-maximised Privacy Preserving Collaborative Filtering Recommendation Algorithm".

- **Chapter 6: A Fast Algorithm to Build New Users Similarity List in Neighbourhood-based Collaborative Filtering.** In this chapter, to overcome the current related research's problem of large computational cost caused by a special case: the new users, with enough recommendation data, have the same rating list, we design a faster algorithm, TwinSearch Algorithm, to avoid computing and sorting the similarity list for the new users repeatedly to save the computational resources. Both theoretical and experimental results show that the TwinSearch Algorithm achieves better running time than the traditional similarity computation method. This chapter consists of our conference paper "A Fast Algorithm to Build New Users Similarity List in Neighbourhood-based Collaborative Filtering".
- **Chapter 7: Conclusion.** In this chapter, we summarise the contribution of this thesis.

Chapter 2

Background

2.1 Privacy Preserving CF Recommendation Algorithms

A noticeable number of literature has been published to preserve customers' private data in recommender systems. However, Calandrino et al. [9] proposed a neighbourhood-based CF attack, the *k*NN attack, which is a serious privacy threat to the neighbourhood-based CF recommender systems in e-commerce, e.g., Amazon. In this section, we briefly discuss some of the research literature in privacy preserving neighbourhood-based CF recommender systems.

2.1.1 Traditional Privacy Preserving Recommender Systems

Amount of traditional privacy preserving methods have been developed in CF recommender systems [75], including cryptographic [18, 19, 45, 73], obfuscation [11, 48, 49, 61, 68], perturbation [4, 5, 22, 31] and probabilistic methods [1].

Erkin et al. [18] applied homomorphic encryption and secure multi-party computation in privacy preserving recommender systems, which allows users to jointly compute their data to receive recommendation without sharing the true data with other parties. Zhan et al. [73] proposed homomorphic encryption approach and scalar product approach for large-scale privacy-preserving recommendation applications. Specifically. the homomorphic encryption approach and scalar product approach preserves users' privacy by introducing homomorphic encryption to the computation of Pearson correlation similarity, then we can use the encrypted similarity for further recommendation computation without releasing the real value. Erkin et al. [19] proposed privacy preserving recommendation framework by applying homomorphic encryption. There are three parties in their framework: user, privacy service provider, and service provider. The two providers compute the ciphertext (submitted from a target user)

for the similarity between the target user and other users, then return the encrypted pack of neighbour number, n, and sum of neighbour ratings, r, to the target user. After decrypting the encrypted pack, the target user compute the value of $\frac{r}{n}$ as the recommendation. Nikolaenko et al. [45] proposed a protocol by the combination of public-key encryption and garbled circuits to a famous recommendation technique, matrix factorization. Their protocol provide recommendations by learning items profile without knowing the user ratings in database. Particularly, the original data (item + rating) are encrypted before sending to a trusted third party. The encrypted data will be processed with decryption, removing ratings, build item profiles in the third party. Then the recommender system gives recommendation based on the item profiles from the trusted third party. The Cryptographic methods provide the highest guarantee for both prediction security ans system security by introducing encryption rather than adding noise to the original record. Unfortunately, unnecessary computational cost impacts its application in industry [76].

Obfuscation and perturbation are two similar data processing methods. In particular, obfuscation methods aggregate a number of random noises with real users rating to preserve user's sensitive information. Parameswaran and Blough [49] proposed an obfuscation framework by using a proposed data obfuscation approach, Nearest Neighbour Data Substitution (NeNDS) [48]. In NeNDS, a two-step process is applied. At the first step, several similar users, who have closed profile information, are selected to be a subset. Then, in the selected subset, users non-identification are exchanged with each other. Once the data are processed by NeNDS, namely, the sets of similar items are obfuscated, the new users profile will be submitted to the central CF server. Shokri et al. [61] provided a distributed obfuscation CF mechanism by allowing local user profile modification while hiding the real profile to an untrusted central server. Particularly, the users keep the original profile offline, but aggregate part of their profile with similar users. The modified profile will be uploaded to the untrusted central server to give users recommendation. Weinsberg et al. [68] added extra reasonable ratings into user's profile against inferring user's sensitive information. Item selection and rating assignment are two steps in [68]. At the stage of item selection, three strategies are applied to select new items into user's profile: random strategy (select random items), sampled strategy (sample unrated items with user's existing rating distribution), and greedy strategy (select the most popular items). At the stage of rating assignment, the new item will be assigned rating by users' average ratings or predicted rating. The above strategies in both two steps guarantees the added ratings are reasonable. Casino et al. [11] proposed a microaggregation-based privacy preserving CF which achieves k-anonymity to hide users' sensitive data. Specifically, the microaggregation method firstly cluster the dataset to ensure

each cluster contains at least k most similar users, then within the cluster, the original data are replaced with the mean of each cluster.

Perturbation methods modify the user's original ratings by a selected probability distribution before using these ratings. Gong [22] proposed a perturbation recommendation approach with secure multiparty computation to achieve privacy preservation in distributed environment. In the process of data submission, the randomised noise is added to protect the original data, while, in the process of providing recommendation, the secure multiparty computation is introduced to guarantee the security of data shared between each distributed server. Bilge and Polat [5] modified the original DWT-based CF scheme[53] by adding uniform (or Gaussian) distribution noise into the original ratings to improve the privacy preserving performance of DWT-based CF scheme. After disguising the rating data, the original DWT-based CF method is applied to provide recommendation. Experimental results show the proposed scheme achieves a good trade-off between accuracy and privacy. Basu et al. [4] introduce Gaussian noise to individual ratings in Slope One predictor[33]. Theoretically evaluation in [4] proves that the Gaussian distribution noise has nearly no impact on the prediction accuracy, but the noise can hide user's real data successfully. [31] presented a perturbation based recommendation methods. Specifically, the users disguised the original data by adding random noise before submitting the data to recommender systems. Since the disguised data will impact the prediction accuracy, [31] introduced the posterior probability distribution to rebuild the user-item rating matrix. As a result, the proposed scheme achieves close prediction accuracy of the non-privacy recommendation method, while maintaining user's privacy. Both perturbation and obfuscation obtain good trade-off between prediction accuracy and system security due to the tiny data perturbation, but the magnitude of noise or the percentage of replaced ratings are not easy to be calibrated [16, 76].

The probabilistic method [1] applied the weighted sampling in neighbour selection which preserves users' privacy against the kNN attack successfully. In Adamopoulos and Tuzhilin [1], the similarity between candidates and a target user is regarded as the sampling weight. The potential neighbours are selected based on their weight, the candidate who has higher weight has a higher probability to be selected as the target user's neighbour. However, it cannot provide enough accuracy due to its global randomness. We suppose the set of k nearest neighbours as the highest quality neighbour set, the randomised weighted selection process will return neighbours with lower similarity with a high probability. Because the performance of the neighbourhood-based CF methods largely depends on the quality of neighbours, the prediction accuracy will be impacted significantly [76].

Therefore, according to the above analysis, achieving a trade-off between privacy and utility, while calibrating the adding noise are difficult tasks for these techniques.

2.1.2 Differential Privacy Recommender Systems

As a well-known privacy definition, the differential privacy technology [14] has been applied in the research of privacy preserving recommender systems. For example, McSherry and Mironov [42] provided the first differential privacy neighbourhood-based CF recommendation algorithm. In fact, their naive differential privacy protects the neighbourhood-based CF recommender systems against the *k*NN attack successfully, as they added Laplace noise into the covariance (similarity between users/items) matrix globally, so that the output k neighbours is no longer the original k nearest neighbours. However, the global noise decreases the accuracy of their recommendation algorithms significantly.

Another differential privacy neighbourhood-based CF recommender systems algorithm is proposed by Zhu et al. [76] which inspired this study. It aims to provide better prediction accuracy than McSherry and Mironov [42] while keeping differential privacy at both neighbour selection stage and rating prediction stage. They proposed a Private Neighbour Collaborative Filtering (PNCF) by introducing exponential differential privacy [43] to the process of neighbour selection to guarantee the system security against the *k*NN attack. After selecting the k neighbours, same with McSherry and Mironov [42], they also added Laplace noise into the similarity matrix to make the final prediction.

Unlike the k nearest neighbour method which selects the k most similar candidates, the PNCF method [76] randomly selects the k neighbours with each candidate u_i's weight ω_i. According to exponential mechanism of differential privacy, the selection weight is measured by a score function and its corresponding sensitivity as follow,

$$\omega_i = \exp\left(\frac{\varepsilon}{4k \times RS} q_a(U(u_a), u_i)\right), \tag{2.1}$$

where q is the score function, RS is the Recommendation-Aware Sensitivity of score function q for any user pairs u_i and u_j, ε is differential privacy parameter, and $U(u_a)$ is the set of user u_a's candidate list. For a user u_a, the score function q and its Recommendation-Aware Sensitivity are defined as follows:

$$q_a(U(u_a), u_i) = sim_{ai}, \tag{2.2}$$

$$RS = \max\left\{\max_{s \in S_{ij}}\left(\frac{r_{is} \cdot r_{js}}{\|r'_i\|\|r'_j\|}\right), \max_{s \in S_{ij}}\left(\frac{r_{is} \cdot r_{js}\left(\|r_i\|\|r_j\| - \|r'_i\|\|r'_j\|\right)}{\|r_i\|\|r_j\|\|r'_i\|\|r'_j\|}\right)\right\}, \tag{2.3}$$

where r_{is} is user u_i's rating on item t_s, sim_{ai} is the similarity between user u_a and u_i, r_i is user u_i's average rating on every item, S_{ij} is the set of all items co-rated by both users i and j, i.e., $S_{ij} = \{s \in S | r_{is} \neq \varnothing \ \& \ r_{js} \neq \varnothing\}$.

Actually, the above naive differential privacy neighbour selection is nearly the same to the probabilistic neighbour selection [1]. To address the above problem of low prediction accuracy in [1], a truncated parameter λ was introduced in [76]. Simply speaking, the candidates whose similarity is greater than $(sim(a,k)+\lambda)$ are selected to the neighbour set, while, whose similarity is less than $(sim(a,k)-\lambda)$ will not be selected, where $sim(a,k)$ denotes the similarity of user u_a's kth neighbour. Theorem 3.1 in [76] provided an equation to calculate the value of λ, i.e. $\lambda = \min(sim(a,k), \frac{4k \cdot RS}{\varepsilon} \ln \frac{k(n-k)}{\rho})$, where ρ is a constant, $0 < \rho < 1$.

However, we observe that the above idea in [76] has three weaknesses. Firstly, it adds random noise in the process of neighbour selection twice; however, it is not necessary. Because we can preserve privacy against the kNN attack successfully only by introducing randomness once, the extra randomness will decrease the prediction accuracy significantly. Secondly, the value of λ may not be achievable. This is because when computing the value of λ by ρ, it results in a good theoretical recommendation accuracy, but does not yield a good experimental recommendation accuracy on the given test datasets in [76]. So the PNCF method [76] will actually be a method of Global Probabilistic Neighbour Selection [1] and cannot guarantee any recommendation accuracy. Thirdly, the PNCF scheme breaks differential privacy in the process of neighbour selection. Suppose there is a tiny change in the dataset, then the value of similarity between target user u_a and other users u_i in the candidate list will change. There may exist a user u_c whose probability of being selected may change from 0 to $x > 0$, then the ratio between the two probabilities will be 0 or infinite, none of which satisfy Definition 1 in Section 3.2.

2.2 Reducing Computational Cost

The cold-start problem and the scalability problem are two famous issues in the research field of recommender systems, both of the two problems impact the performance of recommendation. In this section, we will discuss some of the research literature on addressing the two issues, then explain the reason why these solutions are not suitable in our proposed case.

Some of the methods on the cold-start problem focus on the improvement of prediction accuracy, due to the lack of enough rating data of new users. Theses methods gain better prediction performance by applying different strategy. For example, Bobadilla et al. [7] presented a new similarity measure with optimization based on neural learning, which shows the much better results than current metrics, such as cosine similarity measure. Liu et al. [36] showed an interesting phenomenon that to link a cold-start item to inactive users will give this new item more chance to appear in other users' recommendation lists. Adamopoulos and

Tuzhilin [1] applied probabilistic method to select the k neighbours from the entire candidate list, rather than the k nearest candidate, to avoid the low prediction accuracy due to the lack of rates data. Lika et al. [34] proposed an approach which incorporates classification methods in a pure CF system while the use of demographic data help for the identification of other users with similar behaviour.

Some of the solutions to the scalability problem proposed the methods based on incremental updates of user-to-user and item-to-item similarity. These methods achieve faster and high-quality recommendation than the traditional CF. Papagelis et al. [47] proposed an incremental method which quickly updates user's similarity list when the user adds/rates new items in the recommender systems. Liu et al. [37] presented the temporal relevance measure for ratings at different time steps and developed online evolutionary collaborative filtering algorithms by introducing this measure into the kNN algorithms and incrementally computing neighbourhood similarities, which achieve both better time and space complexity. Inspired by [47], Yang et al. [72] developed the user-based incremental similarity update method to an corresponding item-based method. Huang et al. [28] proposed a practical item-based CF algorithm on big data environment, with the super characteristics such as robust to the implicit feedback problem, scalable incremental update and real-time pruning.

Unfortunately, the current solutions to the cold-start problem and the scalability problem do not work well on a special case: the new users, with enough recommendation data, have the same rating list (the k Nearest Neighbour (kNN) attack [9] can be taken as an example of our special case, which creates k same fake users with at least 8 rated items into the recommender system). The reasons are the solutions to the cold-start problem only work on the new users which have not been gathered sufficient information, and the methods concentrating on the scalability problem only work on the old users who have already have a similarity list. Naturally, when facing the special case, the above methods have to apply the traditional similarity computation method which yields in $O(mn)$ time complexity, where m is the number of items and n is the number of users in a recommender system. Considering the value of m and n are usually very large, the computational cost of the above method will be very large. Therefore, it is necessary to have a faster algorithm to build the new users similarity list in our special case.

Chapter 3

Preliminaries

In this chapter, we firstly introduce the foundational concepts and mathematical model related with this thesis in collaborative filtering, differential privacy, and Wallenius' non-central hyper-geometric distribution. Afterwards, we introduce a popular attack, the k nearest neighbour attack, then we expand its concept to a general attack, β-k nearest neighbour attack. Finally, we provide two performance metrics on the privacy preserving neighbourhood-based CF recommender systems against the β-kNN attack ($\beta \geq 1$).

3.1 k Nearest Neighbour Collaborative Filtering

A collaborative filtering based recommender system predicts users' potential preferences by aggregating the relevant historical data [2, 10, 25, 26, 62]. Collaborative filtering, a popular technique in recommender systems, is in three categories: neighbourhood-based methods, association rules based methods, and matrix factorisation methods [35]. The neighbourhood-based methods generally provides recommendations by combining the opinions of a user's k nearest neighbours [2, 6, 59]. According to the recommendation target, the neighbourhood-based CF can be further classified to two methods [76]: user-based CF [65, 74] and item-based CF [21, 57]. In this thesis, without loss of generality, we use the user-based CF method to show the process of collaborative filtering.

Neighbour Selection and Rating Prediction are two main stages in the neighbourhood-based CF [8, 13, 44, 51, 76]. Suppose we predict a target user u_a's potential rating on an item t_x. At the Neighbour Selection stage, the k most similar neighbours of u_a are selected from u_a's candidate list $\mathscr{S}_a$, where similarities between users are calculated by a selected similarity measurement metric. Two of the most popular similarity measurement metrics are Pearson correlation coefficient and Cosine-based similarity [2, 41, 54, 56]. In this thesis,

we use the Cosine-based similarity [50] as the similarity measurement metric because of its lower complexity.

(1) Pearson Correlation Coefficient (user-based):

$$sim_{ij} = \frac{\sum_{s \in S_{ij}} (r_{is} - \bar{r}_i)(r_{js} - \bar{r}_j)}{\sqrt{\sum_{s \in S_{ij}} (r_{is} - \bar{r}_i)^2 \sum_{s \in S_{ij}} (r_{js} - \bar{r}_j)^2}}, \tag{3.1}$$

(2) Cosin-based Similarity (user-based):

$$\begin{aligned} sim_{ij} &= \cos(\boldsymbol{r}_i, \boldsymbol{r}_j) = \frac{\boldsymbol{r}_i \cdot \boldsymbol{r}_j}{\|\boldsymbol{r}_i\| \times \|\boldsymbol{r}_j\|} \\ &= \frac{\sum_{s \in S_{ij}} r_{is} r_{js}}{\sqrt{\sum_{s \in S_{ij}} r_{is}^2} \sqrt{\sum_{s \in S_{ij}} r_{js}^2}}, \end{aligned} \tag{3.2}$$

where r_{is} is user u_i's rating on item t_s, $r_{is} \in \mathscr{R}$, $\mathscr{R}$ is the user-item rating dataset, sim_{ij} is the similarity between user u_i and user u_j, $\bar{r}_i$ is user u_i's average rating on every rated item, i.e. $\bar{r}_i = \frac{1}{|S_i|} \sum_{t_j \in S_i} r_{ij}$, S_i is the set of all items rated by user u_i, i.e. $S_i = \{s \in S | r_{is} \neq \varnothing\}$, S_{ij} is the set of all items co-rated by both users i and j, i.e., $S_{ij} = \{s \in S | r_{is} \neq \varnothing \ \& \ r_{js} \neq \varnothing\}$.

At the stage of Rating Prediction in user-based CF methods, the predicted rating $\hat{r}_{ax}$ of user u_a on item t_x is calculated as an aggregation of other users' rating on item t_x [2, 76]. The prediction of $\hat{r}_{ax}$ is computed as follow:

$$\hat{r}_{ax} = \text{aggr}_{u_i \in N_k(u_a)}(r_{ix}), \tag{3.3}$$

where, $N_k(u_a)$ is a set which contains user u_a's k nearest neighbours. Specifically, there are three main technique to implement Equation (3.3) [2]. We list them as follow:

$$\hat{r}_{ax} = \frac{1}{k} \sum_{u_i \in N_k(u_a)} r_{ix}. \tag{3.4}$$

$$\hat{r}_{ax} = \frac{\sum_{u_i \in N_k(u_a)} sim(a,i) r_{ix}}{\sum_{u_i \in N_k(u_a)} |sim(a,i)|}. \tag{3.5}$$

$$\hat{r}_{ax} = \bar{r}_a + \frac{\sum_{u_i \in N_k(u_a)} sim(a,i)(r_{ix} - \bar{r}_i}{\sum_{u_i \in N_k(u_a)} |sim(a,i)|}. \tag{3.6}$$

Among the above three equations to predict user u_a's potential rating on item t_x, Equation (3.4) is the simplest one; however, Equation (3.5) is the most popular method in user-based CF recommendation, Equation (3.5) is the most common used approach in item-based CF recommendation [2]. Since we use user-based CF in this thesis, Equation (3.5) would be the

prediction equation in the following chapter's performance analysis. Specifically, in Equation (3.5), the prediction of $\hat{r}_{ax}$ is calculated as a weighted sum of the neighbours' ratings on item t_x, usually, we use similarity as the weight.

3.2 Differential Privacy

Informally, differential privacy [14, 15] is a scheme that minimises the sensitivity of output for a given statistical operation on two neighbouring (differentiated in one record to protect) datasets. Specifically, differential privacy guarantees no matter whether one specific record appears in a database, the privacy mechanism will shield the specific record to the adversary. The strategy of differential privacy is adding a random noise to the result of a query function on the database.

To understand the spirit of differential privacy clearly, several items will be introduced in advance. Firstly, $X(x_1, x_2, \cdots, x_n)$ and $X'(x'_1, x'_2, \cdots, x'_n)$ are two databases with n entries which differ in only one entry, where x_i and x'_1 are the ith entry of X and X'. We call X and X' are neighbouring dataset. Secondly, $f(X)$ is the query function on database X, the respond is the combination of the real answer $a = f(X)$ and a chosen random noise. Thirdly, the privacy mechanism $\mathscr{T}$, namely, the respond, is computed by $\mathscr{T}(X) = f(X) + Noise$.

In differential privacy, the basic assumption is the adversary owns a database X' which contains the information of all the entries except one record in a database X owned by a trusted data holder. The differential privacy ensures the probability ratio of the respond of a same query to X and X' approximate to 1. That is to say, the two databases are not distinguishable. Therefore, the specific record is shielded from the adversary by differential privacy mechanism. A formal definition of Differential Privacy is shown as follow:

Definition 1 (ε-Differential Privacy [14])**.** *A randomised mechanism $\mathscr{T}$ is ε-differential privacy if for all neighbouring datasets X and X', and for all outcome sets $S \subseteq Range(T)$, $\mathscr{T}$ satisfies:* $\Pr[\mathscr{T}(X) \in S] \leq exp(\varepsilon) \cdot \Pr[\mathscr{T}(X') \in S]$*, where ε is the privacy budget.*

The privacy budget ε is set by the database owner. Usually, a smaller ε denotes a higher privacy guarantee because the privacy budget ε reflects the magnitude of difference between two neighbouring datasets.

To achieve differential privacy, it is necessary to add random noise whose magnitude is chosen as a function of the largest change a single participant could have on the output to the query function. We refer to this quantity as the *sensitivity* of the function.

Definition 2 (Sensitivity of Query Function [14])**.** *For any query function $f : \mathscr{D}^n \to \mathscr{R}^d$, the sensitivity of f, $\Delta f = \max |f(X) - f(X')|$, for all neighbouring datasets X, $X' \in \mathscr{D}^n$.*

There are two main applications of the randomised mechanism $\mathscr{T}$: the Laplace mechanism [14] and the Exponential mechanism [43]. The Laplace mechanism comes from Laplace distribution, a continuous probability distribution. To apply the Laplace distribution in differential privacy, we modify the standard Laplace distribution as the following equation:

$$Lap(\frac{\Delta f}{\varepsilon}, Noise) = \Pr(Noise = y) = \frac{\varepsilon}{2\Delta f}\exp(-\frac{|y|\cdot\varepsilon}{\Delta f}). \tag{3.7}$$

Dwork [14] proved that Equation (3.7) satisfies the principle of differential privacy. Therefore, we have the formal definition of Laplace mechanism of differential privacy as blow:

Definition 3 (Laplace Mechanism [14])**.** *Given a function f: $\mathbb{R} \to \mathbb{R}^d$, the mechanism $\mathscr{T}$ provides the ε-differential privacy, if $\mathscr{T}(X) = f(X) + Lap^{-1}(\frac{\Delta f}{\varepsilon}, \Pr)^d$, where the sensitivity of function f, Δf is defined in Definition 2, and d represents the dimension.*

Laplace mechanism mainly addresses the numeric queries, to solve the non-numeric queries problems, McSherry and Talwar [43] proposed the exponential mechanism. The exponential mechanism introduces a score function $q(X,x)$ which reflects how appealing the pair (X,x) is, where X denotes a dataset, x is the respond. In this thesis, since we target at the neighbour selection from a recommender system, we regard the customer's neighbour similarity list as the dataset X, the neighbour selected from the list as the respond x. The formal definition is show as follow:

Definition 4 (Exponential Mechanism [43])**.** *Given a score function of a database X, $q(X,x)$, which reflects the score of query respond x. The exponential mechanism $\mathscr{T}$ provides ε-differential privacy, if $\mathscr{T}(X)$ = {the probability of a query respond $x \propto \exp(\frac{\varepsilon\cdot q(X,x)}{2\Delta q})$}, where the sensitive of score function $q(X,x)$, Δq, is defined in Definition 2.*

3.3 Wallenius' Non-central Hyper-geometric Distribution

Wallenius' non-central hyper-geometric distribution is a distribution of weighted sampling without replacement [20]. In the basic settings of Wallenius' non-central hypergeometric distribution, there are c distinct categories in the population, category i contains m_i same individuals, who have the same weight ω_i. We sample an individual with the probability which is proportional to its weight. Let $\boldsymbol{x}_v = (x_{1v}, x_{2v}, \ldots, x_{cv})$ denote the total number of the individuals in each category sampled after the first v draws [20]. The probability that the next draw gives a individual of colour i is:

$$p_{i(v+1)}(\boldsymbol{x}_v) = \frac{(m_i - x_{iv})\omega_i}{\sum_{j=1}^{c}(m_j - x_{jv})\omega_j}. \tag{3.8}$$

The weighted sampling process without replacement is repeatedly until k individuals have been retained, namely, $k = \sum_{i=1}^{c} x_i$, where x_i denotes the number of individuals sampled from category i by Wallenius' non-central hypergeometric distribution.

Wallenius [63] proposed the probability mass function for this distribution in the univariate case $(c = 2)$. Chesson [12] expanded Wallenius [63]'s solution to the multivariate case $(c > 2)$. In this thesis, we focus on the multivariate Wallenius' non-central hyper-geometric distribution's probability mass function because we regard one user/item in a recommender system as one individual in Wallenius' non-central hyper-geometric distribution. The multivariate probability mass function (PMF) is shown as blow:

$$PMF = \Lambda(\boldsymbol{x})I(\boldsymbol{x}), \tag{3.9}$$

where $\Lambda(\boldsymbol{x}) = \prod_{i=1}^{c} \binom{m_i}{x_i}, I(\boldsymbol{x}) = \int_0^1 \prod_{i=1}^{c} (1-t^{\omega_i/d})^{x_i} \mathrm{d}t, d = \boldsymbol{\omega} \cdot (\boldsymbol{m}-\boldsymbol{x}) = \sum_{i=1}^{c} \omega_i(m_i - x_i)$, $\boldsymbol{x} = (x_1, x_2, \ldots, x_c)$, $\boldsymbol{m} = (m_1, m_2, \ldots, m_c)$, $\boldsymbol{\omega} = (\omega_1, \omega_2, \ldots, \omega_c)$.

The mean $\boldsymbol{\mu}_v = (\mu_{iv}, \cdots, \mu_{cv})$ of $\boldsymbol{x}_v$ can be approximated by the following approximation formula [20]:

$$\mu_{iv} \approx \mu_{i(v-1)} + p_{iv}(\boldsymbol{\mu}_{v-1}), \quad \mu_{i0} = 0. \tag{3.10}$$

Manly [40] gave the approximated solution $\boldsymbol{\mu}^* = (\mu_1^*, \mu_2^*, \ldots, \mu_c^*)$ to the mean $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_c)$ of $\boldsymbol{x} = (x_1, \cdots, x_c)$ after the final draw. In Chapter 3, we mainly use Equation (3.11) to evaluate different probabilistic neighbour selection approaches in recommender systems:

$$\left(1 - \frac{\mu_1^*}{m_1}\right)^{1/\omega_1} = \left(1 - \frac{\mu_2^*}{m_2}\right)^{1/\omega_2} = \ldots = \left(1 - \frac{\mu_c^*}{m_c}\right)^{1/\omega_c}, \tag{3.11}$$

where $\sum_{i=1}^{c} \mu_i^* = k$, $\forall i \in C : 0 \leq \mu_i^* \leq m_i$.

Fog [20] presented three reasons which support the utilisation of Equation (3.11) in this thesis. Firstly, when $\forall i \in C : m_i > 0$ and $\omega_i > 0$, the approximated mean given by Equation (3.11) is accurate. Secondly, in general, Equation (3.11) is an excellent approximated solution. Thirdly, when all individuals in the population have same weight, Equation (3.11) provides the exact mean, rather than the approximated mean.

3.4 A Generalised Privacy Attack for Recommender Systems

In this section, we firstly introduce a popular attack, the k nearest neighbour attack, then we expand the concept to a general attack, β-k nearest neighbour attack.

3.4.1 The k Nearest Neighbour Attack

Calandrino et al. [9] stated a user-based attack called *k Nearest Neighbour* (kNN) attack. Simply, the kNN attack exploits the property that the users are more similar when sharing same rating on corresponding items to reveal user's private data.

In the kNN attack, it is supposed that an attacker's background knowledge consists of both the recommendation algorithm (the kNN CF recommendation) and its parameter k. Furthermore, a target user u_a's partial non-sensitive history ratings, i.e., the ratings on m items that u_a voted, are known to the attacker. Calandrino et al. [9] stated that practically $m \approx 8$.

The aim of the kNN attack is to disclose u_a's sensitive transactions that the attacker does not know about. To achieve this goal, the following steps are applied. Firstly, the attacker registers k fake users in a kNN CF recommender system. Then, the attacker uses the k fake users account to rate the m items with the same opinions of u_a. Afterwards, the CF recommender system will automatically give the new users (the k fake users) recommendation on the items, which are not rated yet, by aggregating the ratings of the k nearest neighbours. According to Equation (3.2), with a high probability, each fake user's k nearest neighbours set $N_k(\text{fake user})$ will include the other $k-1$ fake users and u_a. Because u_a is the only user who rated the potential recommendation items, u_a's rating will be the only useful r_{ix} in Equation (3.5).

3.4.2 β-k Nearest Neighbours Attack

According to the existing privacy preserving neighbourhood-based CF recommendation methods, we expand the kNN attack to a more general case, named β-k Nearest Neighbour (β-kNN) attack.

As we know, to preserve the target user u_a's private information against the kNN attack, we should avoid selecting the true k nearest neighbours, so the existing methods applied the randomness techniques. However, suppose the final k neighbours are selected from the top βk users of u_a's candidate list, also the parameters β and k are known to the attacker, the attacker would catch u_a's private data with a high probability by creating βk fake users. When β is

not great enough, it is still not difficult to break the privacy preserving neighbourhood-based CF recommender systems. Therefore, the β-kNN attack can flexibly adjust the size of fake user's set to improve the attack effectiveness. Actually, the kNN attack can be regarded as 1-kNN attack in the expanded case of the β-kNN attack.

In the β-kNN attack, β can be treated because a security measure as a greater value of β represents a higher fraud cost.

3.5 Performance Metrics

3.5.1 Accuracy

Naturally, in any neighbourhood-based CF recommender systems, aggregating the ratings of more similar users yields more reliable prediction. Therefore, we define the accuracy performance metric α as the similarity sum of the k neighbours of a target user u_a. Obviously, the greatest value of α would be the similarity sum of the k nearest candidates of a target user.

It is simple to compute α in the deterministic neighbourhood-based CF algorithms, e.g. the kNN CF recommendation algorithm, because the k neighbours selected by the deterministic algorithms are determined. So in the case of deterministic algorithms, we compute α by the following equation,

$$\alpha = \sum_{i \in N_k(u_a)} sim(a,i), \tag{3.12}$$

where $N_k(u_a) = \{\text{the } k \text{ nearest neighbours of user } u_a\}$.

While, in the randomised neighbourhood-based CF algorithms, because of the randomisation, we should calculate the value of α as the expected similarity sum of the k neighbours by

$$\alpha = \mathbb{E}(\sum_{i \in N_k(u_a)} sim(a,i)), \tag{3.13}$$

where $N_k(u_a) = \{\text{user } u_a\text{'s } k \text{ neighbours selected by weighted sampling}\}$.

However, it is difficult to compute Equation (3.13) directly, as we need to find all the possible k-neighbour combinations and their corresponding probabilities. So we give another

way to compute the expectation in Equation (3.13), shown in blow:

$$\begin{aligned} \alpha &= \mathbb{E}(\textstyle\sum_{i\in N_k(u_a)} sim(a,i)) \\ &= \textstyle\sum_{i=1}^{n} sim(a,user_i)\mathbb{E}(x_i) \\ &= \textstyle\sum_{i=1}^{n} sim(a,user_i)\mu_i, \end{aligned} \tag{3.14}$$

where $\sum_{i=1}^{n} \mu_i = k$, $\mu_i \in [0,1)$. Section 3.3 introduced the definition of x_i and μ_i.

Actually, when $user_i$ is selected as a neighbour of the target user u_a, $\mu_i = 1$, while when $user_i$ is not a neighbour of u_a, $\mu_i = 0$. Namely, in this thesis, deterministic algorithms (Equation (3.12)) is a special case of randomised algorithms (Equation (3.14)). Therefore, we compute the accuracy metric α by the following equation in both deterministic and randomised neighbourhood-based CF recommendation algorithms:

$$\begin{cases} \alpha = \sum_{i=1}^{n} sim(a,user_i)\mu_i, \\ k = \sum_{i=1}^{n} \mu_i. \end{cases} \tag{3.15}$$

3.5.2 Security

According to the property of the *k*NN attack, the purpose of a privacy preserving neighbourhood-based CF recommendation algorithm is to avoid the target user being the only real user in the final k neighbours set. Thus, the existing probabilistic privacy preserving solutions select the k neighbours across the partial/entire candidate list. It is obvious that the number of candidates who may be selected into the k neighbours set decides the success probability of the *k*NN attack (we call these candidates as potential neighbours). Namely, the more potential neighbours result in the less probability that the target user is the only real user in the final k neighbours set. On the other side, the attacker needs to create enough fake users to cover the potential neighbours set, so that the target user can be the only real user. That is to say, the more potential neighbours yield the higher attacking cost. In conclusion, in this thesis, because we partition the candidate list by the given k, we define the number of user partitions, across which we select the k neighbours, as the security metric β.

Chapter 4

An Accuracy-Assured Privacy-Preserving Collaborative Filtering Recommendation Algorithm

In this chapter, to overcome the problem of recommendation accuracy loss in existing probabilistic methods against the *k*NN attack, we propose a novel method, Partitioned Probabilistic Neighbour Selection, to ensure a required prediction accuracy while maintaining high security against the *k*NN attack. This chapter consists of our journal paper "An Accuracy-Assured Privacy-Preserving Recommender System for Internet Commerce".

Lu, Z. & Shen, H. (2015). An accuracy-assured privacy-preserving recommender system for Internet commerce.
Computer Science and Information Systems, v. 12 (4), pp. 1307-1326

Chapter 5

A Security-Assured Accuracy-Maximised Privacy-Preserving Collaborative Filtering Recommendation Algorithm

In this chapter, to overcome the problems of unsatisfactory prediction accuracy and unassured security guarantee in the existing probabilistic methods against the β-kNN attack, we propose a novel method, Security-Assured Accuracy-Maximised Partitioned Probabilistic Neighbour Selection, to ensure a required system security while achieving the maximum prediction accuracy against the β-kNN attack. This chapter consists of the paper "A Security-assured Accuracy-maximised Privacy Preserving Collaborative Filtering Recommendation Algorithm" in *Proceeding of 19th International Database Engineering & Applications Symposium* [38].

5.1 Introduction

Current research [1, 42, 76] on privacy preserving neighbourhood-based CF recommender systems applied different randomised strategies to improve the prediction accuracy, while ensure the security against the kNN attack by selecting the k neighbours across a target user's partial/entire candidate list. Among these randomised strategies, differential privacy is a better privacy preserving mechanism as it provides calibrated magnitude of noise.

Actually, since the information collected by recommender systems is always the customers' personal data [9], preserving the users' sensitive information should be the kernel

issue of recommender systems. But none of the existing privacy preserving neighbourhood-based CF recommendation algorithms ensure a successful security-assured privacy preservation against the β-kNN attack before the process of CF recommendation. So in this chapter, we present a security metric to measure the level of system security.

In addition, the prediction accuracy should also be considered carefully with the guarantee of assured security, otherwise, the recommender systems would be useless to the non-malicious users who are the majority of customers. However, because of the introduction of global noise, the current randomised methods cannot guarantee the prediction accuracy either. To provide enough prediction utility, we have to decrease the noise as much as possible. Since there is no need to add noise into both the stage of neighbour selection and rating prediction, we may simply add Laplace noise [14] to the final prediction rating after a regular kNN CF. Unfortunately, as Sarathy and Muralidhar [55] reported the security risk about the Laplace mechanism for numeric data, the above idea should be rejected. So we focus on adding noise at the stage of neighbour selection. Instead of global neighbour selection, we partition the order candidate list, so that we can control magnitude of noise inside each partition.

Therefore, in this chapter, we aim to propose a partitioned probabilistic (differential privacy) neighbour selection method, which guarantees an assured system security, then achieves the maximum prediction accuracy with the assured security against the β-kNN attack, without any perturbations in the process of rating prediction. To achieve our goals, we will start from a special case: security-assured accuracy-maximised privacy preserving CF against the kNN attack, then study a general case: security-assured accuracy-maximised privacy preserving CF against the β-kNN attack

5.2 Partitioned Probabilistic Neighbour Selection Scheme against the kNN attack

To achieve our goal, we will firstly provide the objective function with its constraints based on the discussions on both two performance metrics. Then, we propose the security-assured accuracy-maximised privacy preserving recommendation method by solving the objective function according to its constraints.

According to the security metric β and the properties of the kNN attack, we partition the entire candidate list of a user by the given k, i.e., the size of each partition (group) is k. Before providing the objective function, we introduce some variables in advance. We use $f_\beta(i)$ to denote the number of neighbours selected (weighted sampling with exponential differential

privacy) from partition No. i with the given security metric β, $i \in [1,\beta]$. Additionally, α_i denotes the prediction accuracy of partition No. i against the kNN attack. Therefore, we have a general equation for α,

$$\alpha = \sum_{i=1}^{\beta} \alpha_i. \tag{5.1}$$

To solve the Equation (5.1) for the optimal α with the given security metric β against the kNN attack, we select one random fake user as the user who receives the system recommendation in a kNN attack. We suppose the candidate list of the fixed fake user has already in a descending order of similarity. Figure 5.1 shows the fixed fake user's candidate list, where N_i denotes to the user set in partition i, $i \in [2,\beta]$, u_a is the attacker's target user.

Partition Number	1	2	$\cdots$	$\beta-1$	β
Partition Content	Fake users + u_a	N_2	$\cdots$	$N_{\beta-1}$	N_β

Fig. 5.1 Candidate list against the kNN attack

According to formulas (3.15) and Figure 5.1, we have

$$\alpha_i = \sum_{j=1}^{k} sim_{j,N_i} \mu_{j,N_i}, \tag{5.2}$$

where sim_{j,N_i} denotes the similarity between jth candidate in partition No. i and the fixed fake user, μ_{j,N_i} denotes the corresponding mean μ, $in \in [1,\beta]$. Moreover, because we aim to select $f_\beta(i)$ neighbours from partition No. i, $\sum_{j=1}^{k} \mu_{j,N_i} = f_\beta(i)$.

Combining Equation (5.1) and Equation (5.2), we have

$$\alpha = \sum_{i=1}^{\beta} \sum_{j=1}^{k} sim_{j,N_i} \mu_{j,N_i}. \tag{5.3}$$

Since the similarity between the candidates in partition No. 1 and the fixed fake users is absolutely one, we rewrite the above equation as

$$\alpha = f_\beta(1) + \sum_{i=2}^{\beta} \sum_{j=1}^{k} sim_{j,N_i} \mu_{j,N_i}. \tag{5.4}$$

Obviously, the Equation (5.4) is our objective function against the kNN attack.

Now we give the constraints of Equation (5.4). Since we need to select the k neighbours across the top β partitions, we should select at least one neighbour from partition No. β,

i.e., $f_\beta(\beta) = \sum_{i=1}^{k} \mu_{i,N_\beta} \geq 1$. As the candidate list is in a descending order of similarity, and we select one neighbour from the partition No. β, to cover all the top β partitions, the attacker needs to create at least βk fake users, no matter how many neighbours are selected from the partition No. i, $i \in [1, \beta - 1]$. So we can select zero neighbour from the partition No. i, $i \in [1, \beta - 1]$. In addition, because $f_\beta(\beta) \geq 1$ and $\sum_{i=1}^{\beta} f_\beta(i) = k$, $f_\beta(i) \leq k - 1$ for $i \in [1, \beta - 1]$. Recalling the other constraints we presented previously, we have the final objective function with constraints as follow:

$$\begin{aligned} \text{maximise} \quad & \alpha = f_\beta(1) + \sum_{i=2}^{\beta} \sum_{j=1}^{k} sim_{j,N_i} \mu_{j,N_i} \\ \text{subject to} \quad & \sum_{j=1}^{k} \mu_{j,N_i} = f_\beta(i) \\ & \sum_{i=1}^{\beta} f_\beta(i) = k \\ & f_\beta(i) \in \begin{cases} [1,k], & i = \beta \\ [0,k-1], & i \in [1,\beta) \end{cases} \end{aligned} \tag{5.5}$$

Then, we solve Linear Programming (5.5) as a Knapsack Problem with the property of Equation (3.11). The solution, that is the partitioned probabilistic neighbour selection method which guarantees the optimal expectation of prediction accuracy α with a given security metric β against the kNN attack is:

$$f_\beta(i) = \begin{cases} k-1, & i = 1 \\ 1, & i = \beta \\ 0, & i \in (1,\beta) \end{cases} . \tag{5.6}$$

Note that because $\forall\ \beta \geq 1$, the candidate list of any user is in a descending order of similarity, formula (5.6) will always be the optimal solution to Linear Programming (5.5) for any $\beta \geq 1$.

Algorithm 2 demonstrates the Partitioned Probabilistic Neighbour Selection (PPNS) method. From line 1 to line 5, we compute the necessary parameters by Equation (3.2), (2.3), (2.2) and (2.1). We select the k neighbours from each partition with exponential differential privacy by Partitioned Probabilistic Neighbour Selection (Equation (5.6)) in line 6. Next, once we have the k neighbours of target user u_a, we compute the prediction rating of u_a on a item r_x, r_{ax}, by Equation (3.5) in line 7. Finally, we return the neighbour set $N_k(u_a)$ and the prediction rating r_{ax}.

Algorithm 2 Partitioned Probabilistic Neighbour Selection.

Input:
Original user-item rating set, $\mathscr{R}$;
Target user, u_a and prediction item, t_x;
Number of neighbours, k;
Differential privacy parameter, ε;
Security metric, β.

Output:
Target user u_a's k-neighbour set, $N_k(u_a)$;
Prediction rating of u_a on t_x, r_{ax}.

1: Compute the similarity array for target user u_a, $\mathscr{S}_a$;
2: Sort $\mathscr{S}_a$ in descending order, $\mathscr{S}'_a$;
3: Compute exponential differential privacy sensitivity, RS;
4: Compute each user u_i's selection weight, ω_i;
5: Partition the sorted $\mathscr{S}'_a$ by k;
6: Select k neighbours from top β partitions;
7: Compute r_{ax} by $N_k(u_a)$;
8: **return** $N_k(u_a)$, r_{ax};

5.3 Partitioned Probabilistic Neighbour Selection Scheme against the β-kNN attack

According to the performance metrics of accuracy and security in Section 3.5, we rewrite our goal as proposing an algorithm which provides the optimal prediction accuracy α with a given β_0 (namely, against the β_0-kNN attack). To achieve the goal, we will firstly propose a neighbour selection method which is possible to guarantee the optimal prediction accuracy α against the β-kNN attack. Then with the analysis the β_0-kNN attack, we provide the objective function with its constraints based on the discussions on both two performance metrics. Finally, we propose the security-assured accuracy-maximised accuracy privacy preserving recommendation method by solving the objective function.

On the basis of what have been discussed in Chapter 1, because of the global noise, the prediction accuracy is impacted significantly in existing research on neighbourhood-based CF recommender systems. To reach the optimal α, we have to decrease the noise as much as possible. Unfortunately, we cannot just add Laplace differential privacy noise [14] to the final prediction rating after a regular kNN CF recommendation. Since Sarathy and Muralidhar [55] reported the security risk about Laplace differential privacy for numeric data. Therefore, to decrease and calibrate the magnitude of noise, it is natural to present a partitioned (rather

than global) probabilistic (differential privacy) neighbour selection method without any perturbations in the process of rating prediction.

Because of the given security metric β_0, we actually assume the the attacker's attacking limit is the β_0-kNN attack. Clearly, it is possible for the attacker to attack the system with a series of x-kNN attack, where $x \in [1, \beta_0]$, so, in this chapter, we assign an attacking probability $\Pr_i$ for each i-kNN attack, $i \in [1, \beta_0]$. Obviously, $\sum_{i=1}^{\beta_0} \Pr_i = 1$. Therefore, the potential partitioned probabilistic neighbour selection algorithm should achieve the optimal prediction accuracy α against not only the β_0-kNN attack, but the series of x-kNN attack, where $x \in [1, \beta_0]$, with the regarding of attacking probability $\Pr_x$.

According to the properties of the β-kNN attack, naturally, we partition the entire candidate list of a user by the given k, i.e., the size of each partition is k. Before providing the objective function, we introduce some variables in advance. We use $f_\beta(i)$ to denote the number of neighbours selected (weighted sampling with exponential differential privacy) from partition i with the given security metric β. To guarantee the prediction accuracy against the series of x-kNN attack, we should select the neighbours who is the most closed to the target user, so in this chapter the range of i in $f_{\beta_0}(i)$ is $i \in [1, \beta_0 + 1]$. Additionally, α_i denotes the prediction accuracy against the specific i-kNN attack, α_{ij} denotes prediction accuracy of partition j against the specific i-kNN attack. Therefore, we have a general equation for α,

$$\begin{cases} \alpha = \sum_{i=1}^{\beta_0} \Pr_i \times \alpha_i, \\ \alpha_i = \sum_{j=1}^{\beta_0+1} \alpha_{ij}. \end{cases} \tag{5.7}$$

To solve the Equation (5.7) for the optimal α with the given security metric β_0, we will start from the study of α_1, then deduce the computation of α_i from α_1. We clarify that in the computation of each α_i, $i \in [1, \beta_0]$, all of the candidate lists belong to a fixed fake user of target user u_a, so u_a will be in the partition of the top k neighbours (partition No. 1) of the fixed fake user with other $k-1$ fake users; all the candidate lists are sorted in a descending order of similarity.

(1) In the case of α_1, the fixed fake user's candidate list is shown in Figure 5.2, where N_i denotes to the user set in partition i, $i \in [2, \beta_0 + 1]$.

Partition No.	1	2	$\cdots$	β_0	$\beta_0 + 1$
Partition content	Fake + u_a	N_2	$\cdots$	N_{β_0}	N_{β_0+1}

Fig. 5.2 Candidate list in the 1-kNN attack

(2) In the general case of α_i, the fixed fake user's candidate list is shown in Figure 5.3. Because $(i-1)k$ more fake users are added to the candidate list, all of the user sets N_i in case

of α_1 are moved backward, i.e., the user set N_2 is now in partition No. $(i+1)$, the user set N_3 is now in partition No. $(i+2)$, ..., the user set N_{β_0-i+2} is now in partition No. β_0+1.

Partition No.	1	$\cdots$	i	$i+1$	$i+2$	$\cdots$	β_0+1
Partition content	Fake + u_a	$\cdots$	Fake	N_2	N_3	$\cdots$	N_{β_0-i+2}

Fig. 5.3 Candidate list in the i-kNN attack

According to Equation (3.15) and Equation (5.7), moreover, because the similarity between the fixed fake user and users in fake user partition is definitely 1, i.e., $\underset{N_s}{sim_j} = 1$, $s \in [1,i]$, we have

$$\begin{cases} \alpha_i &= \sum_{j=1}^{i} f_{\beta_0}(j) + \sum_{s=2}^{\beta_0-i+2} \sum_{j=1}^{k} \underset{N_s}{sim_j} \mu_{j(s+i-1)} \\ f_{\beta_0}(j) &= \sum_{s=1}^{k} \mu_{sj} \end{cases}. \tag{5.8}$$

where μ_{ij} denotes the mean μ_i (definition in Section 3.3) in partition j, $\underset{N}{sim_i}$ denotes the corresponding similarity in user set N to μ_i, $i \in [1,k]$, $j \in [1,\beta_0+1]$.

Therefore, combining Equation (5.7) and (5.8), we have the following equation to compute the prediction accuracy α as our objective function:

$$\begin{cases} \alpha &= \sum_{i=1}^{\beta_0} \Pr_i \left(\sum_{j=1}^{i} f_{\beta_0}(j) + \sum_{s=2}^{\beta_0-i+2} \sum_{j=1}^{k} \underset{N_s}{sim_j} \mu_{j(s+i-1)} \right) \\ f_{\beta_0}(j) &= \sum_{s=1}^{k} \mu_{sj} \end{cases}. \tag{5.9}$$

Now we give the constraints of Equation (5.9). Since we aim to preserve customers' privacy against the series of x-kNN attack ($x \in [1,\beta_0]$), we should select at least one neighbour from partition No. (β_0+1), i.e., $f_{\beta_0}(\beta_0+1) = \sum_{i=1}^{k} \mu_{i(\beta_0+1)} \geq 1$. In addition, because $f_{\beta_0}(\beta_0+1) \geq 1$ and $\sum_{i=1}^{\beta_0+1} f_{\beta_0}(i) = k$, $f_{\beta_0}(i) \leq k-1$ for $i \in [1,\beta_0]$. Recalling the other constraints we presented previously, we have the final objective function with constraints as follow:

$$\begin{aligned} \text{maximise} \quad & \alpha = \sum_{i=1}^{\beta_0} \Pr_i \left(\sum_{j=1}^{i} f_{\beta_0}(j) + \sum_{s=2}^{\beta_0-i+2} \sum_{j=1}^{k} \underset{N_s}{sim_j} \mu_{j(s+i-1)} \right) \\ \text{subject to} \quad & \sum_{i=1}^{\beta_0} \Pr_i = 1 \\ & \sum_{j=1}^{k} \mu_{ji} = f_{\beta_0}(i) \\ & \sum_{i=1}^{\beta_0+1} f_{\beta_0}(i) = k \\ & f_{\beta_0}(i) \in \begin{cases} [1,k], & i = \beta_0+1 \\ [0,k-1], & \text{otherwise} \end{cases} \end{aligned} \tag{5.10}$$

Then, we solve Equation (5.10) as a Knapsack Problem with the property of Equation (3.11). The solution, that is the partitioned probabilistic neighbour selection method which guarantees the optimal expectation of prediction accuracy α with a given security metric β against the series of x-kNN attack, where $x \in [1, \beta]$, is:

$$f_\beta(i) = \begin{cases} k-1, & i = 1 \\ 1, & i = \beta + 1 \\ 0, & \text{otherwise} \end{cases} . \tag{5.11}$$

We clarify that because $\forall\ \beta_0 \geq 1$, $\sum_{i=1}^{\beta_0} \Pr_i \equiv 1$, Formula (5.11) will always be the solution to Equation (5.10) for any $\beta_0 \geq 1$.

Algorithm 3 demonstrates the Partitioned Probabilistic Neighbour Selection (PPNS) method. From lines 1-5, we compute the necessary parameters by Equation (3.2), (2.3), (2.2) and (2.1). We select the k neighbours from each partition with exponential differential privacy by Partitioned Probabilistic Neighbour Selection (Equation (5.11)) in line 6. Next, once we have the k neighbours of target user u_a, we compute the prediction rating of u_a on a item r_x, r_{ax}, by Equation (3.5) in line 7. Finally, we return the neighbour set $N_k(u_a)$ and the prediction rating r_{ax}.

Algorithm 3 Partitioned Probabilistic Neighbour Selection.

Input:
Original user-item rating set, $\mathscr{R}$;
Target user, u_a and prediction item, t_x;
Number of neighbours, k;
Differential privacy parameter, ε;
Security metric, β_0.

Output:
Target user u_a's k-neighbour set, $N_k(u_a)$;
Prediction rating of u_a on t_x, r_{ax}.

1: Compute the similarity array for target user u_a, $\mathscr{S}_a$;
2: Sort $\mathscr{S}_a$ in descending order, $\mathscr{S}'_a$;
3: Compute exponential differential privacy sensitivity, RS;
4: Compute each user u_i's selection weight, ω_i;
5: Partition the sorted $\mathscr{S}'_a$ by k;
6: Select k neighbours from top $\beta_0 + 1$ partitions;
7: Compute r_{ax} by $N_k(u_a)$;
8: **return** $N_k(u_a)$, r_{ax};

5.4 Experimental Evaluation

In this section, we use the real-world dataset to evaluate the performance of both accuracy and security of our Partitioned Probabilistic Neighbour Selection method. We begin by the description of the dataset, then introduce the evaluation metrics, finally perform a comparative analysis of our method and existing privacy preserving neighbourhood-based CF recommendation algorithms.

5.4.1 Datasets and Experimental Settings

In the experiments, we use two real-world datasets, MovieLens dataset[1] and Douban[2] (one of the largest rating websites in China) film dataset[3]. The MovieLens dataset consists of 100,000 ratings (1-5 integral stars) from 943 users on 1682 films, where each user has rated at least 20 films, each film has been rated by $20-250$ users. The Douban film dataset contains 16,830,839 ratings (1-5 integral starts) from 129,490 unique users on 58,541 unique films [39].

To measure the quality of recommendations by different recommendation algorithms, we apply a famous measurement metric, Mean Absolute Error (*MAE*) [3, 26, 27, 69], in the experiments:

$$MAE = \frac{1}{UI}\sum_{i=1}^{U}\sum_{j=1}^{I}|r_{ij} - \hat{r}_{ij}|, \tag{5.12}$$

where r_{ij} is the real rating of user u_i on item t_j, and $\hat{r}_{ij}$ is the corresponding predicted rating from recommendation algorithms, U and I denote the number of users and items in the experiments. Specifically, in user-based experiments, we compute the *MAE* of ratings from 200 random users ($U = 200$) on all the items ($I = 1682$ or $I = 58,541$) in the two datasets, while, in item-based experiments, we compute the *MAE* of ratings on 200 random items ($I = 200$) from all the users ($U = 943$ or $U = 129,490$) in the two datasets. In addition, we only predict the $\hat{r}_{ij}$ for the $r_{ij} \neq 0$. Obviously, a lower *MAE* denotes a higher prediction accuracy, e.g., $MAE = 0$ means the prediction is totally correct because the prediction ratings equal to the real ratings, but no privacy guarantee against the kNN attack.

[1] http://www.grouplens.org/datasets/movielens/
[2] http://www.douban.com
[3] https://www.cse.cuhk.edu.hk/irwin.king/pub/data/douban

5.4.2 Experimental Results

Experimental Results against the kNN attack

In this section, we show the accuracy performance from different perspectives of four main neighbourhood-based CF methods, i.e., the k Nearest Neighbour (kNN) CF, naive Probabilistic Neighbour Selection (nPNS) [1], Private Neighbour CF (PNCF) [76] and our method, Partitioned Probabilistic Neighbour Selection (PPNS). Due to the similarity metric (Cosine-based similarity, Equation (3.2)) used in this chapter, in the second half of a candidate list, a large number of candidates' similarity will be zero which is useless for prediction. So in the experiments, we set the upper bound of β as $U/2k$ (user-based prediction) or $I/2k$ (item-based prediction).

Accuracy performance

We design three experiments (Figure 5.4 - Figure 5.6) to examine the user-based and item-based CF prediction accuracy on MovieLens dataset and Douban film dataset. As seen in Figure 5.4 to Figure 5.6, we notice that our privacy preserving method (PPNS) achieves much better accuracy performance than the two global methods (nPNS and PNCF) in both the two datasets on both user-based and item-based CF. Moreover, as a trade-off between the prediction accuracy and system security in PPNS, a greater security metric β results in a greater MAE which means a worse prediction accuracy. Specifically, when $\beta = 1$, PPNS achieves the same prediction accuracy with the kNN CF method which is regarded as the baseline neighbourhood-based CF recommendation method in this chapter.

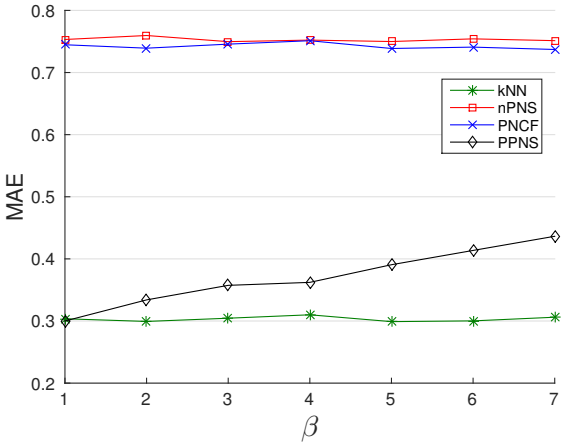

Fig. 5.4 Item-based prediction accuracy on MovieLens ($\varepsilon = 1$, $k = 100$)

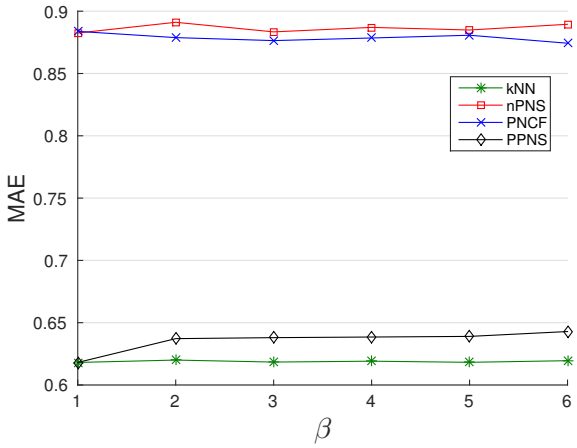

Fig. 5.5 User-based prediction accuracy on MovieLens ($\varepsilon = 1$, $k = 100$)

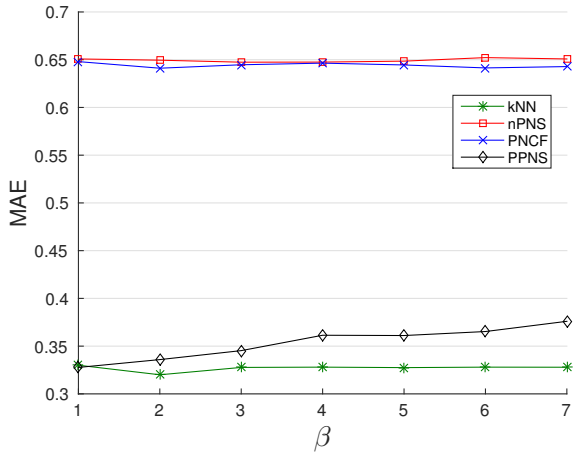

Fig. 5.6 User-based prediction accuracy on Douban film ($\varepsilon = 1$, $k = 100$)

Accuracy performance against the kNN attack

To examine the accuracy performance of the four methods against the kNN attack with the same security guarantee, we introduce a fixed security metric β to the three privacy preserving CF algorithms (nPNS, PNCF, PPNS). That is, we randomly select k neighbours from the βk nearest candidates with weighted sampling in nPNS; we calculate λ as $sim_k - sim_{\beta k}$ in PNCF; and we select the k neighbours across the top β partitions by Algorithm 2 in PPNS. The experiments are run on user-based CF because the kNN attack is a user-based attacking.

Figure 5.7 shows that to ensure the same security guarantee against the kNN attack, PPNS performs much better on the prediction accuracy than the other privacy preserving CF methods (nPNS and PNCF). Moreover, the MAE performance of the kNN CF method indicates that the kNN CF does not provide any security guarantee against the kNN attack.

Additionally, as we regard β as security metric, we observe that we achieve a trade-off between accuracy and security, because the greater β yields a greater *MAE* which denotes less prediction accuracy.

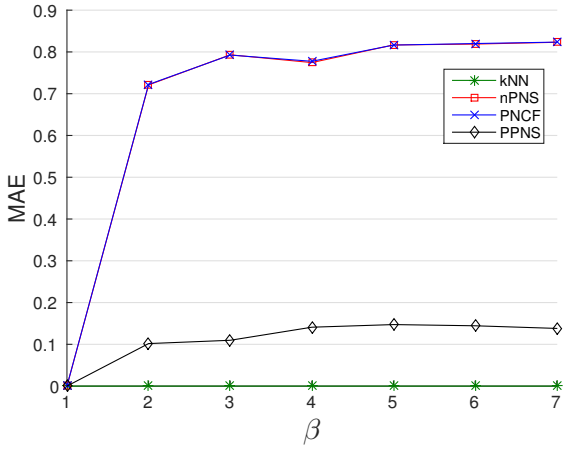

Fig. 5.7 Prediction accuracy on MovieLens against the *k*NN attack ($\varepsilon = 1$, $k = 50$, $m = 8$)

Figure 5.8 demonstrates the impacts of recommendation parameter k on the prediction accuracy. We examine the value of k from 10 to 100, which is a popular range for the recommendation parameter k. From Figure 5.8, we can see that a larger size of neighbour set (or the size of partition in PPNS) denotes the better prediction accuracy of PPNS method against the *k*NN attack.

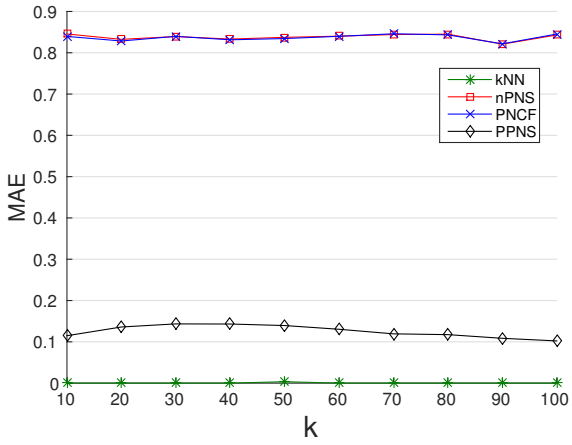

Fig. 5.8 Impacts of k on prediction accuracy against the *k*NN attack on MovieLens ($\varepsilon = 1$, $m = 8$, $\beta = 7$)

Figure 5.9 illustrates the impacts of differential privacy budge ε on the prediction accuracy. It is observed that as ε increases, the *MAE* performance improves in the two differential

privacy methods (PNCF and PPNS). So to achieve a better prediction accuracy, it is suggested to set a greater ε against the kNN attacks.

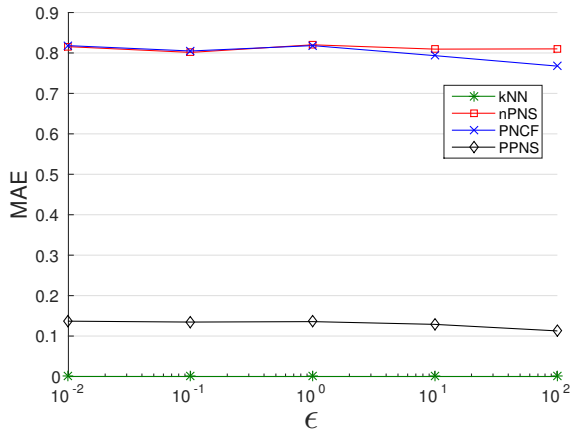

Fig. 5.9 Impacts of ε on prediction accuracy against the kNN attack on MovieLens ($k = 50$, $m = 8$, $\beta = 7$)

Figure 5.10 presents the impacts of attacking parameter m on the prediction accuracy. we can note that to reveal a target customer's privacy by the kNN attack, the attacker needs at least 2^3 real ratings of the target customer as auxiliary information, since when $m \geq 8$, the *MAE* of a non-privacy preserving CF (kNN CF) method is zero. When the attacker has more background knowledge, the prediction will be closer to the real ratings for all of the neighbourhood-based CF systems, but none of privacy preserving algorithms releases the customer's privacy.

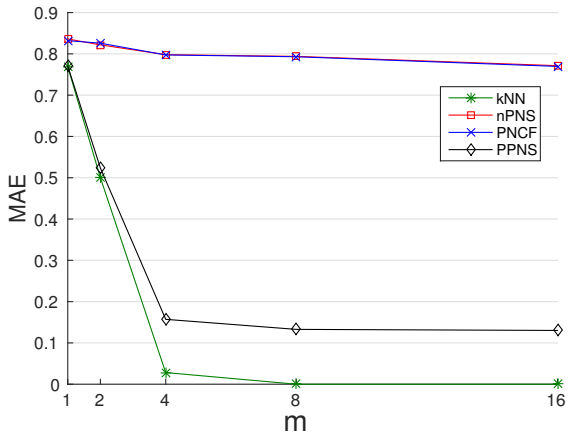

Fig. 5.10 Impacts of m on prediction accuracy against the kNN attack on MovieLens ($\varepsilon = 1$, $k = 50$, $\beta = 7$)

Experimental Results against the β-kNN attack

Our experiments examine the prediction performance of the existing methods (the k Nearest Neighbour (kNN) CF, naive Probabilistic Neighbour Selection (nPNS) [1], Private Neighbour Selection (PNS) [76]) and our method (Partitioned Probabilistic Neighbour Selection (PPNS)) against the series of x-kNN attack, where $x \in [1, \beta_0]$. That is to say, we introduce a fixed security metric β_0 for all the four CF algorithms: we randomly select k neighbours from the $(\beta_0+1)k$ nearest users with weighted sampling in nPNS; we calculate λ as $sim(a,k) - sim(a,(\beta_0+1)k)$ in PNS; and we select the k neighbours across the top β_0+1 partitions by Algorithm 3 in PPNS. The experiments are run on user-based CF because both the kNN attack and the β-kNN attack are user-based attacking. Moreover, we assume the attacker does not know the value of β_0 in the experiments.

Figure 5.11 shows the impacts of β_0 on prediction accuracy. In this experiment, we compute the expected prediction accuracy against the series of x-kNN attack, $x \in [1, \beta_0]$. We can see that our method (PPNS) shows a much better prediction accuracy against the series of x-kNN attack than the two global methods (nPNS and PNS). Moreover, the kNN CF method does not provide any security guarantee against the β-kNN attack, because its MAE value keeps zero. Additionally, the PPNS is less sensitive to β_0 than nPNS and PNS, thus it is more flexible for us to select the value of β_0 in PPNS.

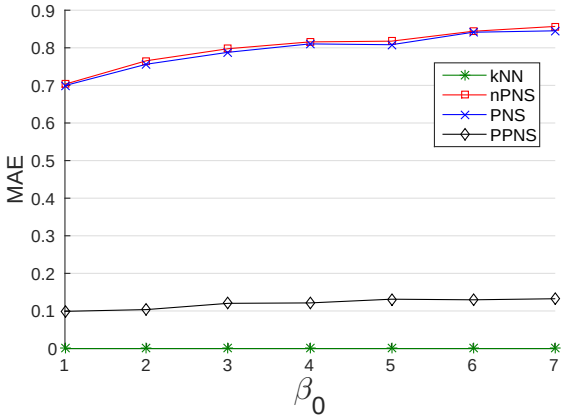

Fig. 5.11 Impacts of β_0 on prediction accuracy against the β-kNN attack ($\varepsilon = 1$, $k = 50$, $m = 8$)

Figure 5.12 shows the impacts of each x-kNN attack on prediction accuracy with a fixed β_0, $x \in [1, \beta_0]$. We observe that when the value of x is less than the β_0+1, the privacy preserving methods (nPNS, PNS, PPNS) protect user's privacy successfully, but when the value of x is greater than β_0, none of these methods guarantee the system security against

the x-kNN attack. Therefore, the safety of β_0 will be the most important issue to a security-assured privacy preserving CF recommendation algorithm.

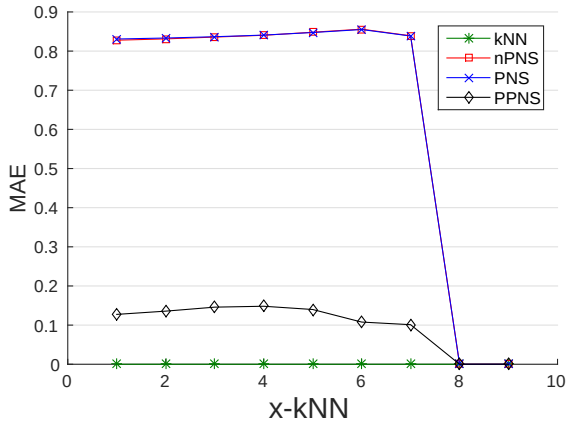

Fig. 5.12 Impacts of each x-kNN attack on prediction accuracy against the β-kNN attack ($\varepsilon = 1$, $k = 50$, $\beta_0 = 7$, $m = 8$)

Figure 5.13 shows the impacts of m on prediction accuracy. We notice that to reveal customer's privacy, the attacker needs at least 8 real ratings as auxiliary information against a non-privacy preserving CF system with the β-kNN attack. In addition, when $m < 8$, with the increasing value of m, PPNS perform better and better on prediction accuracy than both nPNS and PNS.

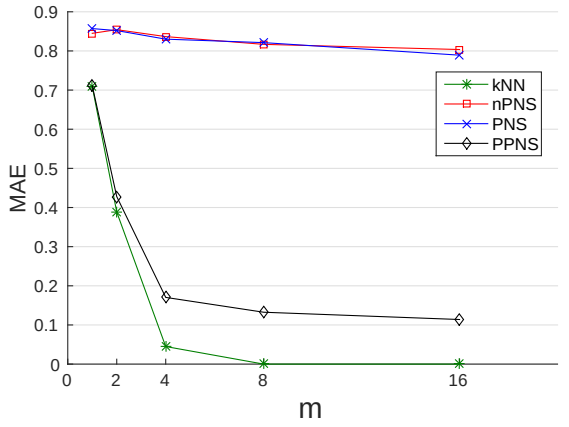

Fig. 5.13 Impacts of m on prediction accuracy against the β-kNN attack ($\varepsilon = 1$, $k = 50$, $\beta_0 = 7$)

Figure 5.14 illustrates the impacts of k on prediction accuracy. Actually, the size of neighbour set (or the size of partition in PPNS) does not impact the prediction accuracy significantly against the series of x-kNN attack, $x \in [1, \beta_0]$. Therefore, to improve the

recommendation efficiency against the β-kNN attack, a good strategy is to select a small k, as a greater k denotes longer recommendation time.

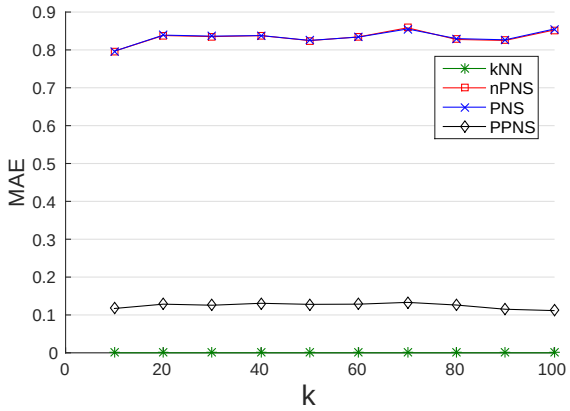

Fig. 5.14 Impacts of k on prediction accuracy against the β-kNN attack ($\varepsilon = 1$, $\beta_0 = 7$, $m = 8$)

Figure 5.15 demonstrates impacts of privacy budge ε on prediction accuracy. It is showed that in the two exponential differential privacy methods (PNS and PPNS), the greater value of *epsilon* denotes smaller *MAE* value. Because the greater value of ε will enlarge the differences between the original similarities. Namely, to achieve a better prediction accuracy in exponential differential privacy neighbour selection method with a given security enforcement, we should set a larger ε.

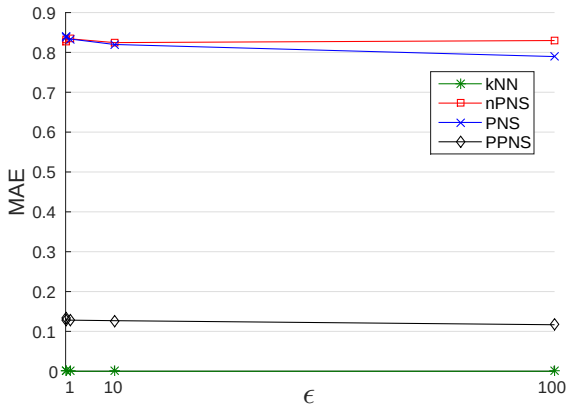

Fig. 5.15 Impacts of ε on prediction accuracy against the β-kNN attack ($\beta_0 = 7$, $k = 50$, $m = 8$)

Chapter 6

A Faster Algorithm to Build New Users Similarity List in Neighbourhood-based Collaborative Filtering

In this chapter, to overcome the current related research's problem of large computational cost caused by a special case: the new users, with enough recommendation data, have the same rating list, we design a faster ($\frac{1}{125}$ time complexity of the existing methods) algorithm, TwinSearch Algorithm, to avoid computing and sorting the similarity list for the new users repeatedly to save the computational resources. This chapter consists of the paper "A Fast Algorithm to Build New Users Similarity List in Neighbourhood-based Collaborative Filtering", to appear in *Proceeding of The 16th International Conference on Parallel and Distributed Computing, Applications and Technologies*.

6.1 Introduction

In the research field of recommender systems, there are two classic problems impact the prediction accuracy and computational cost significantly, one is cold start problem, the other one is scalability problem. Several research papers have been published to address these two problems successfully. However, when focusing on the privacy-preserving recommendation algorithms against the kNN attack [9] and the generalised β-kNN attack [38], we notice that both the kNN attack and the β-kNN attack impact the time complexity of the current methods on addressing cold-start problem and scalability problem. The reasons are the solutions to cold-start problem only work on the new users which have not been gathered sufficient information, and the methods concentrating on scalability problem only work on the old

users who have already have a similarity list. Naturally, when facing the special case, the above methods have to apply the traditional similarity computation method which yields in $O(mn)$ time complexity, where m and n is the number of items and users respectively in a recommender system. Considering the large value of m and n, the computational cost of the above method will be very large. Therefore, it is necessary to gain a faster algorithm to build the new users similarity list in our special case.

6.2 The TwinSearch Algorithm

6.2.1 Algorithm Design

In this section, we define the users who have the same rating list as *twin users*. To address the large computational cost due to the special case: the new users, with enough recommendation data, have the same rating list, we aim to avoid computing and sorting the similarity list for the new users repeatedly to save the computational resources. Since the new users are the same, our strategy to avoid repeated computation is searching the twin user from the system, then copying the twin user's similarity list to the new user directly.

According to the properties of the similarity in recommender systems, we know that if two users are twin user, i.e., $u_a = u_b$, then the similarity between an arbitrary user u_i and u_a, u_b are equal, i.e., $sim_{ai} = sim_{bi}$. Based on the definition of twin user, the ratings on any item i of twin user are equal, i.e., $r_{ai} = r_{bi}$. Therefore, we have the following relationships:

$$u_a = u_b \Rightarrow sim_{ai} = sim_{bi} \tag{6.1}$$

$$u_a = u_b \Leftrightarrow r_{ai} = r_{bi} \tag{6.2}$$

Relationship 6.1 helps us to find the potential twin users from the system, Relationship 6.2 helps us to find the exact twin user from the potential ones. Now we design the TwinSearch Algorithm to find and copy the twin user's similarity list to the new user by relationship 6.1 and 6.2.

In line 4, we search the potential twin users by Relationship 6.1. In line 9, we narrow the size of the final potential twin user set Set_0 by intersecting the c bigger potential twin user set Set_i. The for loop in lines 10-15 find the twin user from the potential twin users' set by Relationship 6.2. Our algorithm can be worked in both user-based and item-based CF, in this section, we present the TwinSearch algorithm from the perspective of the user-based methods, and this can be applied to item-based methods in a straightforward way.

Algorithm 4 TwinSearch Algorithm.

Input:

A user-item rating set, $\mathscr{R}$, with n users and m items; a user-user sorted similarity matrix, $\mathscr{S}$; a new user, u_0, with several ratings on different items; a constant, $c \in \mathbb{Z}^+$.

Output:

The new user u_0's similarity list.

1: Select c random users, u_i^*, $i \in [1,c]$;
2: **for** $i = 1$ to c **do**
3: compute similarity between user u_0 and u_i^*, sim_{0i};
4: search u_i^*'s similarity list $\mathscr{S}_i$ for a $Set_i = \{u_x | sim_{ix} = sim_{0i}\}$;
5: **if** $sim_{0i} = 1$ **then**
6: add u_i^* to Set_i;
7: **end if**
8: **end for**
9: Compute the intersection Set_0 of the c Set_is, $Set_0 = \bigcap_{i=1}^{c} Set_i$;
10: $count \leftarrow 0$;
11: **for** $i = 1$ to $|Set_0|$ **do**
12: **if** $r_{ij} = r_{0j}$, $j \in [1,m]$ **then**
13: copy the similarity list of $u_i \in Set_0$ to u_0;
14: *break*;
15: **end if**
16: **for** $j = 1$ to m **do**
17: **if** $r_{ij} \neq r_{0j}$ **then**
18: $count++$;
19: **end if**
20: **end for**
21: **if** $count = 0$ **then**
22: copy the similarity list of $u_i \in Set_0$ to u_0;
23: *break*;
24: **end if**
25: **end for**
26: **return** The new user u_0's similarity list.

6.2.2 Time Complexity Analysis

We select the c random users in line 1 in $O(c)$. The for loop in lines 2-8 executes exactly c times. Since the similarity computation in line 2 requires time $O(m)$, and we can find the Set_i in line 4 in $O(\log n)$ by binary search, the loop in lines 2-8 contributes $O(c(m+\log n))$ to running time. To compute the intersection Set_0 in line 9, it takes $O(cn)$ time. The for loop in lines 11-21 runs $|Set_0|$ times, where $|Set_0|$ is the size of the intersection Set_0. The for loop in lines 12-16 executes m times, within the for loop, lines 13-15 runs in $O(1)$ time. The running time of copying the similarity list depends on how we implement the

similarity matrix $\mathscr{S}$ data structure. We assume that we use the link list, since it is the fastest list copying implementation. So lines 17-20 takes $O(1)$ time. So the for loop in lines 11-21 requries $O(|Set_0|m)$ time. Therefore, the total running time of Algorithm 4 is $O(|Set_0|m + c(m + \log n))$.

Now we focus on the value of $|Set_0|$. Because $Set_0 = \bigcap_{i=1}^{c} Set_i$, $|Set_0| \leq \min\{|Set_i|\}$, i.e., $|Set_0| = \max\{\min\{|Set_i|\}\}$, $i \in [1, c]$. As the values in a specific Set_i are equal, Set_i must be included in one sub-list of the original similarity list. The sub-list is produced by partitioning the similarity list with the similarity value. For example, suppose that we have x sub-lists, then the similarity value in each sub-list is in the range of $[0, \frac{1}{x}), [\frac{1}{x}, \frac{2}{x}), \cdots, [1 - \frac{1}{x}, 1.0]$ correspondingly. Thus, the upper bound of $|Set_0|$ must be less than the size of largest sub-list.

Moreover, Wei et al. [67] showed that any user's similarity list obeys a specific Gaussian distribution in recommender systems. In this chapter, because of the value of similarity, we set the sample range in [0, 1.0]. The probability density function of each similarity list's Gaussian distribution is defined as:

$$N(\mu, \sigma^2) : f(sim) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(sim - \mu)^2}{2\sigma^2}), \quad sim \in [0, 1.0]. \tag{6.3}$$

Since for any Gaussian distributions, more than 99.99% samples are in the range of $[\mu - 4\sigma, \mu + 4\sigma]$, we fix the similarity value range [0, 1.0] within $\mu \pm 4\sigma$ in this chapter. Figure 6.1 shows the basic statistic settings of one similarity list, where the distance between the minimum (maximum) similarity value and mean μ of Gaussian distribution is $k_1\sigma$ ($k_2\sigma$), and the greatest size sub-list's similarity value range is between $[\mu - k_3\sigma, \mu + k_4\sigma]$. Therefore, we have the size of the sub-list with the most number of users, $s = \frac{\text{Area under the Gaussian distribution curve between } \mu - k_3\sigma \text{ and } \mu + k_4\sigma}{\text{Area under the Gaussian distribution curve between 0 and 1.0}} \times n$. According to the property of Gaussian distribution, we rewrite the expression of s as:

$$\begin{aligned} s &= \frac{\Phi(\frac{\mu + k_4\sigma - \mu}{\sigma}) - \Phi(\frac{\mu - k_3\sigma - \mu}{\sigma})}{\Phi(\frac{\mu + k_2\sigma - \mu}{\sigma}) - \Phi(\frac{\mu - k_1\sigma - \mu}{\sigma})} \times n \\ &= \frac{\Phi(k_3) + \Phi(k_4) - 1}{\Phi(k_1) + \Phi(k_2) - 1} \times n, \end{aligned} \tag{6.4}$$

where $\Phi(x)$ is the cumulative distribution function of standard Gaussian distribution. Our goal is to find the maximum value of s.

In fact, for a specific Gaussian distribution and a partition parameter x, the area under the Gaussian distribution curve between $\mu - k_3\sigma$ and $\mu + k_4\sigma$ is fixed. But, when $k_1 = k_3$, the area under the Gaussian distribution curve between 0 and 1.0 reaches the minimum value. Thus, when $k_1 = k_3$, the value of s is maximum. Then, we have the following linear

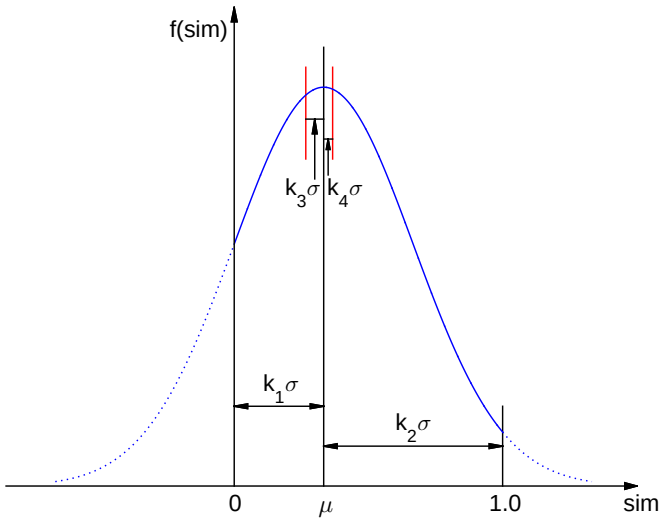

Fig. 6.1 Distribution of user's similarity list

programming:

$$\begin{aligned}
\text{maximise} \quad & s = \frac{\Phi(k_3)+\Phi(k_4)-1}{\Phi(k_1)+\Phi(k_2)-1} \times n \\
\text{subject to} \quad & \mu - k_1\sigma = 0 \\
& \mu + k_2\sigma = 1 \\
& \mu - k_3\sigma = 0 \\
& \mu + k_4\sigma = \tfrac{1}{x} \\
& 0 \le k_1 \le 4 \\
& 0 < k_2 \le 4 \\
& 0 \le k_3 \\
& 0 < k_4.
\end{aligned} \tag{6.5}$$

According to the properties of the cumulative distribution function of standard Gaussian distribution, we have the solution for linear programming (6.5): $k_1 = 0$, $k_2 = 4$, $k_3 = 0$, $k_4 = 0.01$. Then we have the maximum $s = \frac{1}{125}n$ which is the upper bound of $|Set_0|$. In this chapter, we assume $c \ll \frac{1}{125}n$. Therefore, the overall running time of the TwinSearch Algorithm is $\frac{1}{125}O(mn)$, which is much less than the running time ($O(mn)$) of traditional similarity computation method. In this chapter, we assume there are k new same users will be created in the system, so the total running time to build the k users in traditional similarity computation method is $O(kmn)$, while in the TwinSearch algorithm, it is $O((1+\frac{k-1}{125})mn)$.

6.3 Experimental Evaluation

In this section, we use the real-world datasets to evaluate the performance on time complexity of TwinSearch Algorithm and traditional similarity computation method. We begin by the description of the datasets, then perform a comparative analysis of our algorithm and the traditional similarity computation.

6.3.1 Dataset and Experimental Settings

In the experiments, we use two real-world datasets, MovieLens dataset[1] and Douban[2] (one of the largest rating websites in China) film dataset[3]. The MovieLens dataset consists of 100,000 ratings (1-5 integral stars) from 943 users on 1682 films, where each user has rated at least 20 films, each film has been rated by $20-250$ users. The Douban film dataset contains 16,830,839 ratings (1-5 integral starts) from 129,490 unique users on 58,541 unique films [39]. All the experiments are implemented in MATLAB 8.5 (64-bit) environment on a PC with Intel Core2 Quad Q8400 processor (2.67 GHz) with 8 GB DDR2 RAM.

6.3.2 Experimental Results

We design 4 experiments (Figure 6.2 to 6.5) to evaluate the running time for the k new user with same ratings on the above two data sets in both user-based and item-based CF. We use cosine similarity metric as the traditional similarity computation method, and set $k = 30$ in the four experiments. From the four figures, we can see that the TwinSearch algorithm achieves much better performance on time complexity than the traditional similarity computation method.

[1] http://www.grouplens.org/datasets/movielens/
[2] http://www.douban.com
[3] https://www.cse.cuhk.edu.hk/irwin.king/pub/data/douban

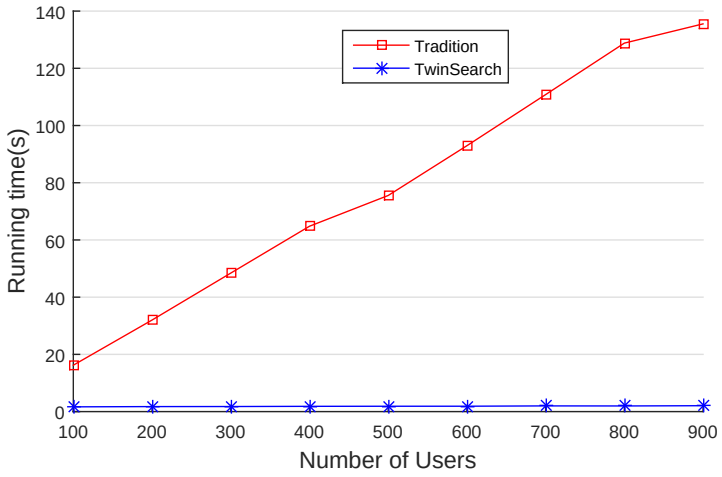

Fig. 6.2 Running time of User-based CF on MovieLens

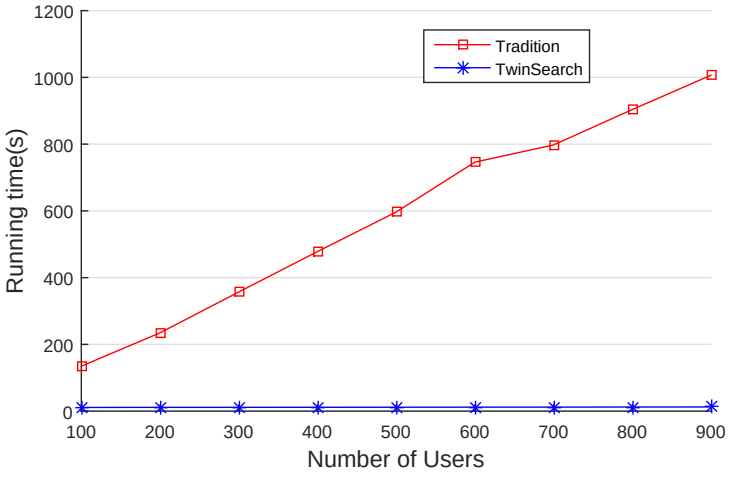

Fig. 6.3 Running time of User-based CF on Douban film

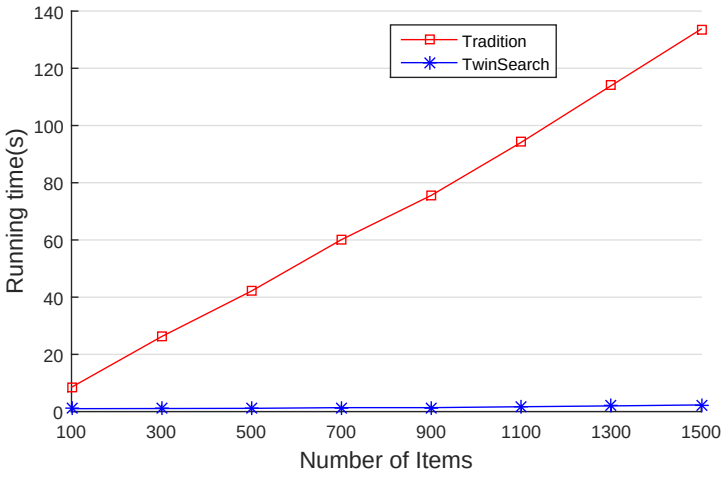

Fig. 6.4 Running time of Item-based CF on MovieLens

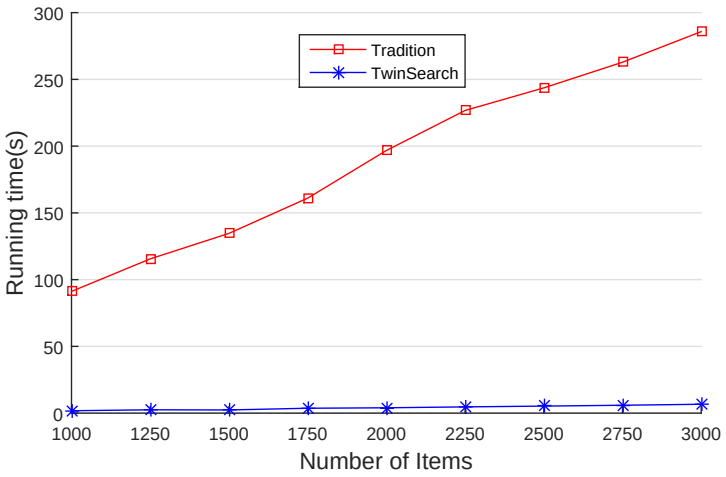

Fig. 6.5 Running time of Item-based CF on Douban film

Chapter 7

Conclusion

Recommender systems play an important role in Internet commerce since the first decade of 21st century. Among the approaches in recommender systems, neighbourhood-based Collaborative Filtering (CF) is one of the most popular approaches in industry, because of the easy implementation and high recommendation accuracy. To protect customers' private information against the kNN attack in the process of filtering, the existing privacy preserving neighbourhood-based CF recommendation methods [1, 42, 75, 76] introduced global noise into the covariance matrix and the process of neighbour selection. However, they neither ensure the prediction accuracy because of the global noise, nor guarantee an assured security enforcement before the collaborative filtering against the β-kNN attack [38] (generalisation of the kNN attack [9]).

To overcome the weaknesses of the current methods, we propose two novel privacy preserving neighbourhood-based CF method, Accuracy-assured Partitioned Probabilistic Neighbour Selection and Security-assured Accuracy-maximised Partitioned Probabilistic Neighbour Selection. The first method ensures a required recommendation accuracy while maintaining high system security against the kNN attack. Theoretical and experimental analysis show that to provide an accuracy-assured recommendation against the kNN attack, our Partitioned Probabilistic Neighbour Selection method yields a better trade-off between the recommendation accuracy and system security than the PNCF methods [75, 76] and Probabilistic Neighbour Selection [1]. While, the second method ensures a required security while achieving the maximum prediction accuracy against the β-kNN attack. The theoretical and experimental analysis show that achieving the same security guarantee against the β-kNN attack, our method ensures the optimal performance of recommendation accuracy among the current randomised neighbourhood-based CF recommendation methods [1, 75, 76].

In addition, two classic problems, the cold-start problem and the scalability problem, challenge the task of dynamically maintaining similarity list in neighbourhood-based CF.

Recently, several methods are presented on solving the two problems. However, these methods applied a traditional $O(mn)$ algorithm to compute the similarity list in a special case: the new users, with enough recommendation data, have the same rating list. To address the problem of large computational cost due to the special case, we design a faster algorithm, TwinSearch Algorithm, to build new users' similarity list, which avoids computing and sorting the similarity list to save the computational resources. The computation cost of our algorithm is $\frac{1}{125}$ of the existing methods. Both the theoretical and the experimental results show that our algorithm achieves better running time than the traditional method.

References

[1] Adamopoulos, P. and Tuzhilin, A. (2014). On over-specialization and concentration bias of recommendations: Probabilistic neighborhood selection in collaborative filtering systems. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, pages 153–160, New York, NY, USA. ACM.

[2] Adomavicius, G. and Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749.

[3] Antunes, P., Herskovic, V., Ochoa, S. F., and Pino, J. A. (2012). Structuring dimensions for collaborative systems evaluation. *ACM Computing Surveys (CSUR)*, 44(2):8.

[4] Basu, A., Vaidya, J., and Kikuchi, H. (2012). Perturbation based privacy preserving slope one predictors for collaborative filtering. In *Trust Management VI*, pages 17–35. Springer.

[5] Bilge, A. and Polat, H. (2012). An improved privacy-preserving dwt-based collaborative filtering scheme. *Expert Systems with Applications*, 39(3):3841–3854.

[6] Bobadilla, J., Hernando, A., Ortega, F., and Bernal, J. (2011). A framework for collaborative filtering recommender systems. *Expert Syst. Appl.*, 38(12):14609–14623.

[7] Bobadilla, J., Ortega, F., Hernando, A., and Bernal, J. (2012). A collaborative filtering approach to mitigate the new user cold start problem. *Knowledge-Based Systems*, 26(0):225 – 238.

[8] Breese, J. S., Heckerman, D., and Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, UAI'98, pages 43–52, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[9] Calandrino, J. A., Kilzer, A., Narayanan, A., Felten, E. W., and Shmatikov, V. (2011). "you might also like: " privacy risks of collaborative filtering. In *Proceedings of the 2011 IEEE Symposium on Security and Privacy*, SP '11, pages 231–246, Washington, DC, USA. IEEE Computer Society.

[10] Candillier, L., Meyer, F., and Boullé, M. (2007). Comparing state-of-the-art collaborative filtering systems. In Perner, P., editor, *Machine Learning and Data Mining in Pattern Recognition*, volume 4571 of *Lecture Notes in Computer Science*, pages 548–562. Springer Berlin Heidelberg.

[11] Casino, F., Domingo-Ferrer, J., Patsakis, C., Puig, D., and Solanas, A. (2013). Privacy preserving collaborative filtering with k-anonymity through microaggregation. In *e-Business Engineering (ICEBE), 2013 IEEE 10th International Conference on*, pages 490–497. IEEE.

[12] Chesson, J. (1976). A non-central multivariate hypergeometric distribution arising from biased sampling with application to selective predation. *Journal of Applied Probability*, 13(4):pp. 795–797.

[13] Delgado, J. and Ishii, N. (1999). Memory-based weighted majority prediction. In *SIGIR Workshop Recomm. Syst. Citeseer*. Citeseer.

[14] Dwork, C. (2006). Differential privacy. In Bugliesi, M., Preneel, B., Sassone, V., and Wegener, I., editors, *Automata, Languages and Programming*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin Heidelberg.

[15] Dwork, C. (2008). Differential privacy: A survey of results. In *Proceedings of the 5th International Conference on Theory and Applications of Models of Computation*, TAMC'08, pages 1–19, Berlin, Heidelberg. Springer-Verlag.

[16] Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography*, TCC'06, pages 265–284, Berlin, Heidelberg. Springer-Verlag.

[17] Ekstrand, M. D., Riedl, J. T., and Konstan, J. A. (2011). Collaborative filtering recommender systems. *Found. Trends Hum.-Comput. Interact.*, 4(2):81–173.

[18] Erkin, Z., Beye, M., Veugen, T., and Lagendijk, R. L. (2010). Privacy enhanced recommender system. In *31st Symposium on Information Theory in the Benelux, WIC 2010*, pages 35–42. IEEE Benelux Information Theory Chapter.

[19] Erkin, Z., Veugen, T., Toft, T., and Lagendijk, R. L. (2012). Generating private recommendations efficiently using homomorphic encryption and data packing. *Information Forensics and Security, IEEE Transactions on*, 7(3):1053–1066.

[20] Fog, A. (2008). Calculation methods for wallenius' noncentral hypergeometric distribution. *Communications in Statistics—Simulation and Computation*, 37(2):258–273.

[21] Gao, M., Wu, Z., and Jiang, F. (2011). Userrank for item-based collaborative filtering recommendation. *Inf. Process. Lett.*, 111(9):440–446.

[22] Gong, S. (2011). Privacy-preserving collaborative filtering based on randomized perturbation techniques and secure multiparty computation. *International Journal of Advancements in Computing Technology*, 3(4):89–99.

[23] Hennig-Thurau, T., Malthouse, E. C., Friege, C., Gensler, S., Lobschat, L., Rangaswamy, A., and Skiera, B. (2010). The impact of new media on customer relationships. *Journal of service research*, 13(3):311–330.

[24] Herlocker, J., Konstan, J. A., and Riedl, J. (2002). An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information retrieval*, 5(4):287–310.

[25] Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 230–237, New York, NY, USA. ACM.

[26] Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53.

[27] Hernández del Olmo, F. and Gaudioso, E. (2008). Evaluation of recommender systems: A new approach. *Expert Syst. Appl.*, 35(3):790–804.

[28] Huang, Y., Cui, B., Zhang, W., Jiang, J., and Xu, Y. (2015). Tencentrec: Real-time stream recommendation in practice. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, SIGMOD '15, pages 227–238, New York, NY, USA. ACM.

[29] Kabbur, S., Ning, X., and Karypis, G. (2013). Fism: factored item similarity models for top-n recommender systems. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 659–667. ACM.

[30] Keller, T. and Raffelsieper, M. (2014). Cosibon: an e-commerce like platform enabling bricks-and-mortar stores to use sophisticated product recommender systems. In *Proceedings of the 8th ACM Conference on Recommender systems*, pages 367–368. ACM.

[31] Kikuchi, H. and Mochizuki, A. (2012). Privacy-preserving collaborative filtering using randomized response. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*, pages 671–676. IEEE.

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

[33] Lemire, D. and Maclachlan, A. (2005). Slope one predictors for online rating-based collaborative filtering. In *SDM*, volume 5, pages 1–5. SIAM.

[34] Lika, B., Kolomvatsos, K., and Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4, Part 2):2065 – 2073.

[35] Liu, B., Mobasher, B., and Nasraoui, O. (2011). *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*. Data-Centric Systems and Applications. Springer.

[36] Liu, J.-H., Zhou, T., Zhang, Z.-K., Yang, Z., Liu, C., and Li, W.-M. (2014). Promoting cold-start items in recommender systems. *PLoS ONE*, 9(12):e113457.

[37] Liu, N. N., Zhao, M., Xiang, E., and Yang, Q. (2010). Online evolutionary collaborative filtering. In *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, pages 95–102, New York, NY, USA. ACM.

[38] Lu, Z. and Shen, H. (2015). A security-assured accuracy-maximised privacy preserving collaborative filtering recommendation algorithm. In *Proceedings of the 19th International Database Engineering & Applications Symposium*, IDEAS '15, pages 72–80, New York, NY, USA. ACM.

[39] Ma, H., Zhou, D., Liu, C., Lyu, M. R., and King, I. (2011). Recommender systems with social regularization. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 287–296, New York, NY, USA. ACM.

[40] Manly, B. F. J. (1974). A model for certain types of selection experiments. *Biometrics*, 30(2):281–294.

[41] McLaughlin, M. R. and Herlocker, J. L. (2004). A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, pages 329–336, New York, NY, USA. ACM.

[42] McSherry, F. and Mironov, I. (2009). Differentially private recommender systems: Building privacy into the net. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 627–636, New York, NY, USA. ACM.

[43] McSherry, F. and Talwar, K. (2007). Mechanism design via differential privacy. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '07, pages 94–103, Washington, DC, USA. IEEE Computer Society.

[44] Nakamura, A. and Abe, N. (1998). Collaborative filtering using weighted majority prediction algorithms. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 395–403, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[45] Nikolaenko, V., Ioannidis, S., Weinsberg, U., Joye, M., Taft, N., and Boneh, D. (2013). Privacy-preserving matrix factorization. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, pages 801–812, New York, NY, USA. ACM.

[46] Pagare, R. and Patil, S. A. (2013). Article: Study of collaborative filtering recommendation algorithm - scalability issue. *International Journal of Computer Applications*, 67(25):10–15.

[47] Papagelis, M., Rousidis, I., Plexousakis, D., and Theoharopoulos, E. (2005). Incremental collaborative filtering for highly-scalable recommendation algorithms. In *Proceedings of the 15th International Conference on Foundations of Intelligent Systems*, ISMIS'05, pages 553–561, Berlin, Heidelberg. Springer-Verlag.

[48] Parameswaran, R. and Blough, D. M. (2005). A robust data obfuscation approach for privacy preservation of clustered data. In *Workshop on Privacy and Security Aspects of Data Mining*, pages 18–25. Citeseer.

[49] Parameswaran, R. and Blough, D. M. (2007). Privacy preserving collaborative filtering using data obfuscation. In *Granular Computing, 2007. GRC 2007. IEEE International Conference on*, pages 380–380.

[50] Rajaraman, A. and Ullman, J. D. (2011). *Mining of Massive Datasets*. Cambridge University Press, New York, NY, USA.

[51] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, CSCW '94, pages 175–186, New York, NY, USA. ACM.

[52] Ricci, F., Rokach, L., and Shapira, B. (2011). Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer.

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

[54] Salton, G. and McGill, M. J. (1986). *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.

[55] Sarathy, R. and Muralidhar, K. (2011). Evaluating laplace noise addition to satisfy differential privacy for numeric data. *Trans. Data Privacy*, 4(1):1–17.

[56] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2000). Analysis of recommendation algorithms for e-commerce. In *Proceedings of the 2Nd ACM Conference on Electronic Commerce*, EC '00, pages 158–167, New York, NY, USA. ACM.

[57] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 285–295, New York, NY, USA. ACM.

[58] Sarwar, B. M., Karypis, G., Konstan, J., and Riedl, J. (2002). Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proceedings of the fifth international conference on computer and information technology*, volume 1. Citeseer.

[59] Schafer, J. B., Frankowski, D., Herlocker, J., and Sen, S. (2007). Collaborative filtering recommender systems. In Brusilovsky, P., Kobsa, A., and Nejdl, W., editors, *The Adaptive Web*, pages 291–324. Springer-Verlag, Berlin, Heidelberg.

[60] Schafer, J. B., Konstan, J., and Riedl, J. (1999). Recommender systems in e-commerce. In *Proceedings of the 1st ACM conference on Electronic commerce*, pages 158–166. ACM.

[61] Shokri, R., Pedarsani, P., Theodorakopoulos, G., and Hubaux, J.-P. (2009). Preserving privacy in collaborative filtering through distributed aggregation of offline profiles. In *Proceedings of the third ACM conference on Recommender systems*, pages 157–164. ACM.

[62] Su, X. and Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, 2009:4:2–4:2.

[63] Wallenius, K. T. (1963). Biased sampling: The non-central hypergeometric probability distribution. Technical Report 70, Stanford University.

[64] Wang, H.-F. and Wu, C.-T. (2012). A strategy-oriented operation module for recommender systems in e-commerce. *Computers & Operations Research*, 39(8):1837–1849.

[65] Wang, J., de Vries, A. P., and Reinders, M. J. T. (2006). Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 501–508, New York, NY, USA. ACM.

[66] Wei, K., Huang, J., and Fu, S. (2007). A survey of e-commerce recommender systems. In *Service Systems and Service Management, 2007 International Conference on*, pages 1–5. IEEE.

[67] Wei, Y. Z., Moreau, L., and Jennings, N. R. (2005). A market-based approach to recommender systems. *ACM Trans. Inf. Syst.*, 23(3):227–266.

[68] Weinsberg, U., Bhagat, S., Ioannidis, S., and Taft, N. (2012). Blurme: inferring and obfuscating user gender based on ratings. In *Proceedings of the sixth ACM conference on Recommender systems*, pages 195–202. ACM.

[69] Willmott, C. J. and Matsuura, K. (2005). Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. *Climate research*, 30(1):79.

[70] Xiao, B. and Benbasat, I. (2007). E-commerce product recommendation agents: use, characteristics, and impact. *Mis Quarterly*, 31(1):137–209.

[71] Xu, B., Zhang, M., Pan, Z., and Yang, H. (2005). Content-based recommendation in e-commerce. In *Computational Science and Its Applications–ICCSA 2005*, pages 946–955. Springer.

[72] Yang, X., Zhang, Z., and Wang, K. (2012). Scalable collaborative filtering using incremental update and local link prediction. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 2371–2374, New York, NY, USA. ACM.

[73] Zhan, J., Hsieh, C.-L., Wang, I.-C., Hsu, T.-S., Liau, C.-J., and Wang, D.-W. (2010). Privacy-preserving collaborative recommender systems. *Trans. Sys. Man Cyber Part C*, 40(4):472–476.

[74] Zhao, Z.-D. and Shang, M.-S. (2010). User-based collaborative-filtering recommendation algorithms on hadoop. In *Proceedings of the 2010 Third International Conference on Knowledge Discovery and Data Mining*, WKDD '10, pages 478–481, Washington, DC, USA. IEEE Computer Society.

[75] Zhu, T., Li, G., Ren, Y., Zhou, W., and Xiong, P. (2013). Differential privacy for neighborhood-based collaborative filtering. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '13, pages 752–759, New York, NY, USA. ACM.

[76] Zhu, T., Ren, Y., Zhou, W., Rong, J., and Xiong, P. (2014). An effective privacy preserving algorithm for neighborhood-based collaborative filtering. *Future Generation Computer Systems*, 36:142–155.