Differential Privacy and Its Application

By

Tianqing Zhu

Submitted in fulfilment of the requirements for the degree of
Doctor of Philosophy

Deakin University
March 2014

DEAKIN UNIVERSITY
ACCESS TO THESIS - A

I am the author of the thesis entitled

Differential Privacy and Its Application

submitted for the degree of

DOCTOR OF PHILOSOPHY

This thesis may be made available for consultation, loan and limited copying in accordance with the Copyright Act 1968.

'I certify that I am the student named below and that the information provided in the form is correct'

Full Name: .... Tianqing Zhu..
(Please Print)

Signed: .... Signature Redacted by Library ....

Date: .... 31/03/2014...

DEAKIN UNIVERSITY
CANDIDATE DECLARATION

I certify that the thesis entitled (10 word maximum)

Differential Privacy and Its Application
submitted for the degree of

DOCTOR OF PHILOSOPHY

is the result of my own work and that where reference is made to the work of others, due acknowledgment is given.

I also certify that any material in the thesis which has been accepted for a degree or diploma by any university or institution is identified in the text.

'I certify that I am the student named below and that the information provided in the form is correct'

Full Name: ....Tianqing Zhu...
(Please Print)

Signed: .... Signature Redacted by Library **.......................................**

Date: ....31/03/2014...

To my Son...

Table of Contents

Table of Contents **iv**

List of Tables **viii**

List of Figures **ix**

Acknowledgements **x**

Publication List **xi**

Abstract **xiv**

1 Introduction **1**
- 1.1 Differential Privacy 3
  - 1.1.1 Differentially Private Data Release 5
  - 1.1.2 Differentially Private Data Mining 6
- 1.2 Challenges and Research Objectives 7
  - 1.2.1 Challenges 7
  - 1.2.2 Research Objectives 8
- 1.3 Thesis Outline 10

2 Differential Privacy **12**
- 2.1 Preliminaries 12
  - 2.1.1 Notation 12
  - 2.1.2 Differential Privacy 13
  - 2.1.3 The Privacy Budget 14
  - 2.1.4 The Sensitivity 15
  - 2.1.5 The Principle Differential Privacy Mechanisms 16
- 2.2 Differentially Private Data Release 17

2.2.1 Interactive Data Release 18
2.2.2 Non-Interactive Data Release 25
2.2.3 Summary 34
2.3 Differentially Private Data Mining 35
2.3.1 Data Mining with Interface 35
2.3.2 Data Mining with Full Access 40
2.3.3 Summary 44
2.4 Applications of Differential Privacy 44
2.4.1 Privacy Preserving Recommender System 44
2.4.2 Privacy Preserving Tagging Recommender System 47
2.4.3 Differential Privacy for a Coupled Dataset 48
2.5 Summary 49

3 Differentially Private Neighborhood-based Collaborative Filtering 51
3.1 Introduction 51
3.2 Fundamentals of a Recommender System 55
3.2.1 Notation 55
3.2.2 Related Work 58
3.2.3 Summary 61
3.3 Private Neighbor Collaborative Filtering 62
3.3.1 The Private Neighbor Collaborative Filtering Algorithm . . . 62
3.3.2 The Private Neighbor Selection 65
3.4 Privacy and Utility Analysis 71
3.4.1 Privacy Analysis 71
3.4.2 Utility Analysis 72
3.5 Experiment and Analysis 75
3.5.1 Datasets and Measurements 77
3.5.2 Performance of *PNCF* 78
3.5.3 Performance of *Recommendation-Aware Sensitivity* 81
3.5.4 Performance of the *Private Neighbor Selection* Mechanism . . 85
3.5.5 Effect of Privacy Budget 86
3.6 Summary 87

4 Deferentially Private Tagging Recommender System 91
4.1 Introduction 91
4.2 Fundamentals of Tagging Recommender System 94
4.2.1 Notations 94
4.2.2 Tagging Recommender Systems 95
4.2.3 Related Work 96

4.3 Private Tagging Release 97
4.3.1 Private Tagging Release Algorithm Overview 98
4.3.2 Private Topic Model Generation 100
4.3.3 Topic Weight Perturbation 103
4.3.4 Private Tag Selection 105
4.4 Privacy and Utility Analysis 108
4.4.1 Privacy Analysis 108
4.4.2 Utility Analysis 109
4.5 Experiment and Analysis 113
4.5.1 Datasets 114
4.5.2 Semantic Loss Analysis 115
4.5.3 Performance of Tagging Recommendation 117
4.5.4 Impact of Privacy Budget 123
4.6 Summary 126

5 Coupled Differential Privacy for Non-IID Datasets 128
5.1 Introduction 128
5.1.1 Related Work 132
5.2 Problem Statement 135
5.2.1 An Example: Coupled Records in a Dataset 135
5.2.2 Coupled Differential Privacy Problem 137
5.2.3 Research Issues and Challenges 139
5.3 Coupled Dataset Analysis 140
5.3.1 Coupled Analysis 141
5.3.2 Coupled Sensitivity 142
5.4 Coupled Iteration Mechanism 145
5.4.1 Overview of Coupled Iteration Mechanism 145
5.4.2 Coupled Update Function 147
5.4.3 Parameters Discussion 149
5.5 Mechanism Analysis 152
5.5.1 Privacy Analysis 152
5.5.2 Utility Analysis 153
5.6 Experiment and Analysis 156
5.6.1 Datasets and Configuration 157
5.6.2 The Performance of *CIM* 158
5.6.3 *Coupled Sensitivity* vs. *Global Sensitivity* 162
5.6.4 Impact of Parameter T 164
5.6.5 Impact of Parameter η 168

5.7 Conclusions 170

6 Conclusion **172**

6.1 Contributions 172

6.2 Future Work 174

Bibliography **178**

List of Tables

2.1 Mechanism Comparsion 21
2.2 Frequency Table D in k Batch Queries 26
2.3 k Range Queries 27
2.4 k Range Queries 28
2.5 Comparison between *Interactive* and *non-Interactive* Settings 35
2.6 Comparison between Data Mining Algorithms 45

3.1 Overall Performance Comparison on *Netflix* 80
3.2 Paired-t-test for *PNCF* vs. DP on *Netflix* 81
3.3 Paired-t-test for RS&NM vs. GS&NM on *Netflix* 84

4.1 Privacy Budget Allocation in the *PriTop* Algorithm 108
4.2 Characteristics of the datasets 115
4.3 Paired-t-test for *PriTop* vs. *Tag Suppression* 123

5.1 Frequency Dataset D 135
5.2 The Maximal Size of Coupled Groups 162

List of Figures

1.1 Four Scenarios of *Differential Privacy* 4

3.1 Performance of *PNCF* and DP on the *MovieLens* Dataset 79
3.2 *Global Sensitivity* vs. *Recommendation-Aware Sensitivity* on *Netflix* . 82
3.3 *Global Sensitivity* vs. *Recommendation-Aware Sensitivity* on *Movielens* 84
3.4 Comparison between the *Exponential Mechanism* and *PNCF* on *Netflix* 89
3.5 Comparison between *Exponential Mechanism* and *PNCF* on *MovieLens* 90
3.6 Impact of ϵ on *PNCF* when $k = 35$ 90

4.1 *Semantic Loss* on Different Datasets 118
4.2 *FolkRank* Recall Result 119
4.3 *FolkRank* Precision Result 122
4.4 *FolkRank* Recall Result 124
4.5 *FolkRank* Precision Result 125

5.1 Effectiveness of *CIM* 160
5.2 Effectiveness of *Coupled Sensitivity* 163
5.3 Impact of T on Update Round 165
5.4 Impact of T on *CIM* Performance 167
5.5 Impact of η on the Maximal Update Round 169

Acknowledgements

I would like to thank a number of people who constantly provided me with advice, help, support and understanding during time at Deakin University working on my PhD.

Firstly, I would like to express my deepest gratitude and appreciation to my supervisors, Prof. Wanlei Zhou and Prof. Yang Xiang, for their suggestions and constant support during my research over the last three years. I am honoured to have received their supervision. Both guided me through difficulties when I struggled, and I gained much from them regarding knowledge about research, writing, presenting and teaching. I will always be very grateful for their guidance, support and encouragement.

I would also like to thank the academic and general research staff of the School of Information Technology, Deakin University. Special thanks to Dr. Gang Li, who guided me through a lot in regards to my research work. I also thank Ms. Alison Carr, Ms. Lauren Fisher and Ms. Kathy Giulieri for their great secretarial work that enabled me to attend academic conferences. I also owe a great deal of thanks to my research friends who helped me during my research and daily life during my PhD study: Yini Wang, Sheng Wen, Yongli Ren, Longxiang Gao, Jun Zhang, and so many others.

I also would like to thank my parents for their long distance support and love. The greatest debt I owe goes to my husband, Ping Xiong. Without their everlasting encouragement and understanding, this thesis wouldn't exist.

Melbourne, Australia Tianqing Zhu
March 2014

Publication List

Refereed Journal Articles

1. **Tianqing Zhu**, Yongli Ren, Wanlei Zhou, Jia Rong, Ping Xiong. An Effective Privacy Preserving Algorithm for Neighborhood-based Collaborative Filtering. Future Generation Computer Systems. August 2013. `http://dx.doi.org/10.1016/j.future.2013.07.019`.

Refereed Conference Papers

1. **Tianqing Zhu**, Gang Li, Wanlei Zhou, Ping Xiong, and Cao Yuan. Deferentially private tagging recommendation based on topic model. In Knowledge Discovery and Data Mining (PAKDD), The 18th Pacific-Asia Conference on, (In press) 2014. (ERA2010 ranking A Conference)

2. **Tianqing Zhu**, Gang Li, Yongli Ren, Wanlei Zhou, and Ping Xiong. Differential privacy for neighborhood-based collaborative filtering. In International Conference on Advances in Social Networks Analysis and Mining, ASONAM. IEEE Computer Society, page 752–759. 2013.

3. **Tianqing Zhu**, Ping Xiong, Yang Xiang, and Wanlei Zhou. An effective differentially private data releasing algorithm for decision tree. In Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 IEEE 12th

International Conference on. IEEE, page 388–395. 2013. (ERA2010 ranking A Conference)

4. **Tianqing Zhu**, Gang Li, and Wanlei Zhou. Privacy preserving for tagging recommender systems. In Web Intelligence and Intelligent Agent Technology (WI-IAT), 2013 IEEE/WIC/ACM International Conferences on, volume 1, page 81–88, IEEE, 2013.

5. Yongli Ren, **Tianqing Zhu**, Gang Li, Wanlei Zhou. Top-N Recommendations by Learning User Preference Dynamics. In proceedings of the 17th Pacific-Asia Conference on Knowledge Discovery and Data Mining. PAKDD, pages 390-401. 2013. (ERA2010 ranking A Conference)

6. Ping Xiong and **Tianqing Zhu**. An anonymization method based on tradeoff between utility and privacy for data publishing. In Management of e-Commerce and e- Government (ICMeCG), 2012 International Conference on, IEEE, pages 72-78. 2012.

7. Longxiang Gao, Ming Li, **Tianqing Zhu**, Alessio Bonti, Wanlei Zhou, and Shui Yu. Amdd : Exploring entropy based anonymous multi-dimensional data detection for network optimization in human associated dtns. In Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on, IEEE, pages 1245-1250. 2012.

Submitted Papers

1. **Tianqing Zhu**, Cao Yuan, Gang Li, Yongli Ren, Wanlei Zhou. Privacy Preserving Topic Model for Tagging Recommender Systems. IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, PART B: CYBERNETICS. (submitted)

2. **Tianqing Zhu**, Gang Li, Yongli Ren, Wanlei Zhou, Ping Xiong. Privacy Preserving Data Releasing for Tagging Recommendation. IEEE Transactions on Information Forensics and Security. (IEEE Trans, Impact Factor 1.895) (submitted)

3. **Tianqing Zhu**, Gang Li, Wanlei Zhou, Ping Xiong. Coupled Differential Privacy for Dataset with Non-independent Records. IEEE Transactions on Information Forensics and Security. (submitted)

4. **Tianqing Zhu**, Gang Li, Wanlei Zhou, Yongli Ren, Ping Xiong. Privacy Preserving Collaborative Filtering for KNN Attack Resisting Social Network Analysis and Mining. (Major Revision)

Abstract

Abstract: *Differential privacy* has become an important research area since the first publication about *information revealing* in 2003. Since then, extensive work has been done to develop this new concept because it constitutes a rigorous and provable privacy notion that can be implemented in various research areas. However, even though *differential privacy* is considered a promising solution in the privacy preserving community, it still suffers from several weaknesses. The most serious one is that in many applications high sensitivity queries will lead to large amounts of noise, which may significantly decrease the utility of the dataset. The second problem lies in the sparsity of the dataset, which will induce redundant noise when *differential privacy* utilizes the randomized mechanism in the domain. The third overlooked issue is *differential privacy* preserving in a coupled dataset. The coupled records will release more information than expected, and decrease the privacy guarantee of the dataset. Hence, these weaknesses prevent differential privacy from being applied in a broader range of real-world applications. This thesis aims to address these issues to make *differential privacy* more effective and applicable.

The first issue is overcome by proposing *application-aware sensitivity.* Traditional sensitivity is only determined by the query and calibrates the maximal changes when deleting one record in a dataset. We observe that if we take each record into consideration, the maximal change will not always occur. *Application-aware sensitivity* is determined by the change of each record instead of using the maximal change. Be using this novel strategy, we can significantly decrease the sensitivity of queries. Theoretical analysis proves the proposed application-aware *sensitivity* can retain the utility of applications while satisfying the requirement of *differential privacy*.

The second issue can be addressed by shrinking the randomized domain because the noise can decrease when the randomized range is limited. We adapt the private clustering and topic model to structure records into groups and limit the randomized domain within each group. After this, a better trade-off between *privacy* and *utility* can be obtained by taking advantage of the differentially private composition properties. Theoretical analysis and experimental results indicate the effectiveness of the proposed method.

For the coupled problem in differential privacy, we re-measure the sensitivity to capture the relation among records. As most coupled records are only partially coupled, we take advantage of the diverse coupled level between records and propose the notion of *coupled sensitivity*, which is less than the traditional global sensitivity. We also design an iteration based coupled releasing mechanism to save privacy budgets. Based on the novel sensitivity and the mechanism, we propose coupled differential privacy to answer a large number of queries for a coupled dataset as the solution.

In summary, this thesis makes three major contributions: 1) using application-aware sensitivity to decrease noise; 2) shrinking the randomized domain to cut redundant noise; 3) proposing a coupled differential privacy notion to enhance the privacy guarantee for a coupled dataset. These contributions enable differential privacy to be applied smoothly and effectively in various applications.

Keywords: Privacy Preserving, Differential Privacy, Data Mining

Chapter 1

Introduction

In the past decades, the collection of digital information by corporations, organizations and governments have created opportunities for researchers and decision makers. This collected data is usually related to a specific need. For example, in the medical arena, governments build health record systems for exchange of medical information [39]. In e-commerce, transaction data is generated from abundant activities including searching, browsing and online shopping [97]. A data collector, also known as the *curator*, is in charge of exchanging, releasing and sharing the datasets for further analysis [2].

However, most of the collected datasets are personally related and contain private or sensitive information. Even curators can apply some simple anonymization techniques, such as deleting `user name` or `ID number` to preserve privacy, with individuals sensitive information still having a high probability to of being re-identified from the released dataset. In the early years, Latanya Sweeney et al. provided an example of de-anonymization on a published medical dataset [81,87]. They showed that even with all the explicit identifiers removed, individuals can still be re-identified by linking with another public vote list dataset through the combination of `zip code`, `date of birth` and `gender`. The information the adversary can link with is defined as the `background information` [87]. Lots of literatures suggests that *with the background information, the combination of several attributes may re-identify an individual* [20, 41, 91, 92]. By doing this, Narayanan [67] partially de-anonymized the *Netfix*, a movie rating dataset, by linking with another movie dataset, *IMDB*. Mudhakar et al. [85] de-anonymized mobility traces by using social networks as their background information. Stevens et al. [47] exploited `Skype`, a popular P2P communication software to invade users' location and sharing of information. All of these examples show that simple anonymization is insufficient for privacy preserving. The adversary with background information will be able to identify the individual's records with high probability. A more rigorous privacy preserving is desired to guarantee sensitive personal information.

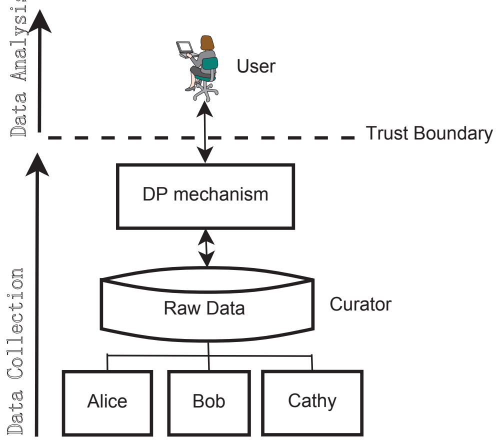
Research communities have proposed various methods to preserve privacy [2, 55, 89], and have designed a number of metrics to evaluate the privacy level of these methods. The methods and their privacy criteria are defined as the privacy model. The most popular privacy model is the *k-anonymity* model [81, 87]. It partitions the dataset into a number of *equivalence groups* in which every record has the same attribute values with at least $K-1$ other records. There are a number of other privacy models. For example, the *l-diversity* [57] privacy model ensures at least l diverse values exist for the sensitive attribute. The *t-closeness* [50, 51] model requires the sensitive attribute distribution in each group should not deviate from that of the whole dataset by more than t. The *δ-presence* [68] bounds the probability of inferring the presence of any individual's record within a specified range. However, the vulnerability of these models lies in the fact they can be easily compromised by uncontrollable background information [90]. Moreover, they can hardly prove the privacy they can preserve, namely, they lack a solid theoretical foundation to evaluate the privacy level of the released dataset.

Fortunately, *differential privacy* offers a rigorous definition of privacy with a set of technologies that satisfies the definition. It is a solid privacy model that provides a provable privacy guarantee against the worst-case *background information*. It assumes all records are independent and identically distributed (I.I.D) in a universe and that the adversary knows all other records in a dataset except one record. This *background information* can be considered the worst-case background because this is the maximal background information he can obtain for that unknown record. Under this assumption, *differential privacy* theoretically proved the adversary has low probability to figure out the unknown record even with this worst-case background [21]. Consequently, *differential privacy* has been considered a promising privacy preserving technique due to its strong and provable privacy guarantee.

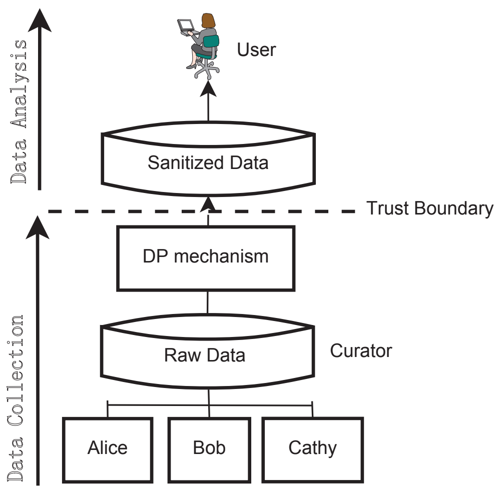
1.1 Differential Privacy

Differential privacy acquires the intuition that releasing an aggregated report should not reveal too much information on any individual record in the dataset [21]. This can be achieved using randomized mechanisms whose output distribution remains almost unchanged even with an arbitrary individual record deleted. More precisely, the randomized mechanism contains adding calibrated noise to the query output or randomizing all possible outputs in the domain. These mechanisms are usually implemented in four basic scenarios, as depicted in Fig. 1.1.

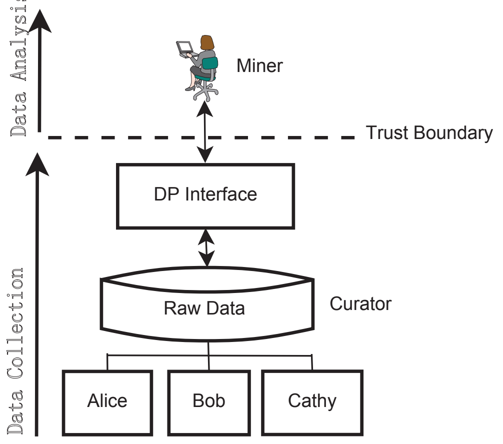
According to Fig. 1.1, all scenarios contain two phases: *data collection* and *data*



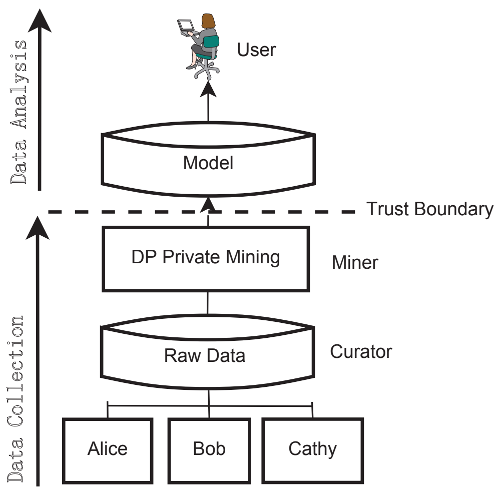
(a) Interactive Data Releasing



(b) Non-Interactive Data Releasing



(c) Interface Based Data Mining



(d) Fully Access Data Mining

Figure 1.1: Four Scenarios of *Differential Privacy*

analysis, separated by the *trust boundary.* In the *data collection* phase, the trustworthy curator collects personal information from individuals. In the *data analysis* phase, the collected dataset will be distributed to the untrusted users. Here, the *data analysis* usually includes *data release* and *data mining.* Accordingly, we have *Differentially Private Data Releasing* and *Differentially Private Data Mining.*

1.1.1 Differentially Private Data Release

In the scenario of *differentially private data release* (PPDR), after collecting personal information, the curator will release the dataset to users or a third party for further analysis. There are two settings to perform the releasing procedure: the *interactive* setting as described in Fig. 1.1a, and the *non-interactive* setting as described in Fig. 1.1b.

In the *interactive* setting, the curator puts an interactive differential privacy interface between the users and the dataset to resolve the queries from the users, to provide randomized query answers for the privacy purpose. In the *non-interactive* setting, the curator modifies the raw dataset and then releases a sanitized version to public users. After sanitation, the dataset is released for direct access or for queries in various forms, such as the *contingency table* or multi-dimensional *histograms.*

PPDR emphasizes the publishing of individual records without considering specific algorithms that public users might use. This research has been focused on how to modify the original dataset or the queries with the guarantee of *differential privacy* while preserving an acceptable dataset *utility.*

1.1.2 Differentially Private Data Mining

In the scenario of *differentially private data mining* (PPDM), the curator tries to learn the statistical facts about the data records to build data mining models for users. In most cases, the curator may not be a data mining expert, so he may deliver the dataset to the *data miner* for further analysis.

For different types of data miners, there are two possible solutions and both have to pre-determine the data mining tasks: The first solution considers the data miner as an untrusted party, so the curator only provides a differentially private interface to present basic operators to the data miner, who uses the noisy results to create models. This procedure is shown in Fig. 1.1c. Since the essential task in data mining is the construction of models on aggregated data, the data miner can construct approximately accurate models without access to the precise information on individual records.

Another solution is presented in Fig. 1.1d, which allows the trustworthy data miner to build a model from the original dataset using specific privacy preserving data mining algorithms. A *differential privacy* mechanism will be implemented in data mining algorithms to ensure the final model will not disclose any sensitive individual information.

PPDM emphasizes the accuracy of the data mining models and each differentially private mechanism will be adapted to a specified data mining algorithm. Research works has concentrated on how to modify the data mining algorithm to satisfy *differential privacy* while retaining a high mining accuracy [28].

1.2 Challenges and Research Objectives

1.2.1 Challenges

Even *differential privacy* is considered as a promising privacy preserving solution, it still has some challenges:

- The most serious challenge is that in many applications, queries with high sensitivity in many applications will lead to large amounts of noise, which may decrease the utility of the dataset significantly. The noise added by differential privacy is calibrated by the sensitivity of the query. Simple queries such as `count` or `sum` introduce minor noise, which has low effect on the utility. However, queries with high sensitivity are involved in many real world applications, such as the similarity measurement in the recommender system. How to decrease the noise for queries with high sensitivity is the essential challenge for differential privacy.

- The second challenge lies in the sparsity of the dataset. When *differential privacy* utilizes the randomized mechanism, the sparse dataset will induce redundant noise. Randomization is an essential differential privacy mechanism that allocates different probabilities to every possible output of a query and then samples one output according to the probabilities. If the dataset is sparse, there will be a lot of possible outputs and the randomization mechanism will introduce a massive amount of errors compared to the non-private mechanism.

- The third challenge is differential privacy preserving in a coupled dataset. Existing research on differential privacy assumes that in a dataset, data records are sampled independently. However, in real-world applications, data records in a dataset are rarely independent [14]. A traditional differentially private technique performed on a coupled dataset will disclose more information than expected, and this indicates a serious privacy violation [44]. Although recent research has been concerned with this new privacy violation, it does lack a solid solution for the coupled dataset. Moreover, how to decrease the large amount of noise incurred by differential privacy in a coupled dataset remains unexplored.

In summary, these challenges prevent differential privacy from being applied in a broader range of real-world applications. This thesis aims to address these issues to make *differential privacy* more effective and applicable.

1.2.2 Research Objectives

To address the above challenges, this thesis aims to *achieve a breakthrough in extending the theory of differential privacy to enable its effective applicability in real world applications.* The approaches will involve the extension of the data release mechanism to accommodate varies applications ranging from collaborative filtering and tagging recommendation to those involved with a coupled dataset, in addition to the development of *application-aware sensitivity* that is insensitive to the utility loss. More specifically, the research issues of this thesis are:

How to define *application-aware sensitivity*? The large amount of noise introduced by differential privacy in many applications is derived from queries with high sensitivity. Traditional *sensitivity* measurement may not be suitable for these queries due to high dimensional input. For example, in applications such as collaborative filtering and tagging recommendation, the similarity query plays a vital role but has a high sensitivity, and this leads to large utility loss when using a traditional sensitivity measurement. How to define appropriate sensitivity is an essential issue to be addressed.

How to shrink the randomized domain for a sparse dataset? The sparsity of a dataset will induce redundant noise when *differential privacy* utilizes the randomized mechanism in the domain. The most efficient way is to decrease the noise to shrink the scale of the randomization mechanism. Previous work applied dataset partition or clustering to limit the randomized scale. However, these methods fail to retain the structure or the semantics of the dataset. Especially for a tagging recommendation, that contains a unique structure among *users*, *tags* and *items*, the problem is how to shrink the domain in this structure and retain the semantics of the dataset.

How to accommodate coupled information in a differential privacy mechanism? A coupled differential privacy solution will involve the coupled information identification, sensitivity measurement and the design of data release mechanism. The difficulty is that the dataset may be coupled in different ways. How to identify this information and incorporate the coupled information into differential privacy remains unexplored. This thesis will investigate these problems by proposing an effective iterative based mechanism for coupled datasets.

1.3 Thesis Outline

This section aims to establish the structural organization of the thesis. According to three research issues of this thesis: *application-aware sensitivity*, *shrinking the domain* and *dealing with the coupled dataset*, the content of each chapter is organized as follows:

- Chapter 2 presents a comprehensive survey on the development of differential privacy, including relevant concepts, assumptions, and emerging techniques in this area. Efforts have been given to identify various research directions and emerging research issues. This chapter will also explore open challenges.

- Chapter 3 focuses on the *application-aware sensitivity* issue in the context of neighbourhood-based CF methods. This chapter specifically investigates how to design a differentially private recommender system by proposing a *Private Neighbor Collaborative Filtering (PNCF)* algorithm. The algorithm includes the *Private Neighbor Selection* to provide a private recommender system, and *Recommendation-Aware Sensitivity* is introduced to decrease the noise brought into the recommendation results.

- Chapter 4 proposes a novel *Private Topic Model* to address the *domain shrinking* issue for tagging recommendations. This chapter focuses on how to preserve privacy for a tagging recommender system by using a private topic model to randomize the released tagging dataset. This chapter also proposes a privacy preserving tag releasing algorithm, *PriTop*, which includes three privacy preserving operations: *Private Topic Model Generation* structures the uncontrolled tags, *Private Weight Perturbation* adds the *Laplace* noise into the weights to hide the numbers of tags, while *Private Tag Selection* finally finds the most suitable replacement tags to hide the original tags.

- Chapter 5 addresses the privacy issue for coupled datasets. It proposes an effective coupled differential privacy solution by defining the *coupled sensitivity* and designing a coupled data releasing mechanism. With consideration of the coupled degrees among records, the proposed *coupled sensitivity* can significantly decrease the noise compared to the traditional global sensitivity. The coupled data releasing mechanism is designed based on an iterative method to answer a large number of queries. Compared with the traditional method, the proposed coupled differential privacy solution enhances the privacy guarantee for coupled datasets with less accuracy costs.

- Chapter 6 summarizes the contributions of this thesis, and presents some possible suggestions and extensions for future research.

To maintain the readability, each chapter is organized in a self-contained format, and some essential contents, e.g. definition, are briefly recounted in related chapters.

Chapter 2

Differential Privacy

This chapter provides an extensive literature review on *differential privacy* by tracing research trends and chronologically reviewing the contributions along each research line regarding *differential privacy*.

2.1 Preliminaries

2.1.1 Notation

Let D be the dataset with n records and d attributes; and let variable r represent a record sampled from a universe $\mathcal{X}$; Two datasets D and D' are *neighboring* if they have the same cardinality but differ in only one record. The size of dataset D is defined as $|D|= \sum_{r\in\mathcal{X}} D(r)$, which will be typically regarded as an unordered set of n records from domain $\mathcal{X}$.

A query f is a function that maps dataset D to a real number: $f : D \to \mathbb{R}$. A group of queries is denoted as $\mathcal{F}$. *Differential privacy* provides a randomization mechanism $\mathcal{M}$ to mask the difference of query f between the *neighboring datasets* [23]. The maximal difference on the results of query f is defined as the *sensitivity* Δf, which determines how much perturbation is required for the answer. The $\mathcal{M}$ can be considered as a randomization algorithm that accesses the database and implements some functionality. For example, we add noise λ to the query result, and output the noisy answer denoted by a 'hat' over the notation: $\hat{f}(D)$ denotes the randomized answer of querying f on D.

2.1.2 Differential Privacy

Differential privacy acquires the intuition that releasing an aggregated result should not reveal too much information about any individual record in the dataset [21]. A formal definition of *differential privacy* is as follows [23]:

Definition 1 ((ϵ, τ)-Differential Privacy). A randomized algorithm $\mathcal{M}$ gives (ϵ, τ)-differential privacy for any pair of *neighboring datasets* D and D', and for every set of outcomes Ω, $\mathcal{M}$ satisfies:

$$Pr[\mathcal{M}(D) \in \Omega] \leq \exp(\epsilon) \cdot Pr[\mathcal{M}(D') \in \Omega] + \tau \tag{2.1.1}$$

where Ω denotes the output range of the algorithm $\mathcal{M}$. If $\tau = 0$, the $\mathcal{M}$ preserves ϵ-*differential privacy.*

(ϵ, τ)-*differential privacy* provides freedom to violate the strict ϵ-*differential privacy* for some low probability events. For example, a certain dataset may have zero probability on some outputs, but for neighboring datasets, the probability on these outputs can be non-zero. This violation probability will be limited by the parameter τ. The ϵ-*differential privacy* provides a stricter guarantee than (ϵ, τ)-*differential privacy* does. Since we do not intend to weaken the definition of ϵ-*differential privacy* substantially, for the rest of the thesis, we are interested in ϵ-*differential privacy.*

2.1.3 The Privacy Budget

In Definition 1, parameter ϵ is defined as the *privacy budget* [33], which controls the privacy guarantee level of mechanism $\mathcal{M}$. A smaller ϵ represents a stronger privacy. In practice, ϵ is usually set as less than 1, such as 0.1 or $\ln 2$.

Two *privacy budget* compositions are widely used in the design of mechanisms [63]: the *sequential composition* and the *parallel composition*, as defined in Def. 2 and Def. 3, respectively.

Definition 2. *Sequential Composition*: Suppose a set of privacy steps $\mathcal{M} = \{\mathcal{M}_1, ...\mathcal{M}_m\}$ are sequentially performed on a dataset, and each $\mathcal{M}_i$ provides ϵ privacy guarantee, the $\mathcal{M}$ will provide $(m \cdot \epsilon)$-*differential privacy.*

The *sequential composition* undertakes the privacy guarantee for a sequence of differentially private computations. When a series of randomized mechanisms have been performed sequentially on a dataset, the *privacy budgets* will be added up for each step.

Definition 3. *Parallel Composition* [63]: Suppose we have a set of privacy steps $\mathcal{M} = \{\mathcal{M}_1, ...\mathcal{M}_m\}$, if each $\mathcal{M}_i$ provides ϵ_i privacy guarantee on a disjointed subset of the entire dataset, the parallel of $\mathcal{M}$ will provide $\max\{\epsilon_1, ..., \epsilon_m\}$-*differential privacy.*

The *parallel composition* corresponds to a case where each $\mathcal{M}_i$ is applied on disjointed subsets of the dataset. The ultimate privacy guarantee only depends on the largest *privacy budget.*

2.1.4 The Sensitivity

Sensitivity is a parameter determining how much perturbation is required in the mechanisms. For example, when we release a specified query f of dataset D, the sensitivity will calibrate the noise for $f(D)$. Two types of sensitivity are employed in *differential privacy*: the *global sensitivity* and the *local sensitivity.*

The Global Sensitivity

The *global sensitivity* is only related to the type of query f. It considers the maximal differences between query results on neighboring datasets, and indicates how much the difference should be hidden in mechanisms. The formal definition is as below:

Definition 4 (Global Sensitivity). For query $f : D \rightarrow \mathbb{R}$, the *global sensitivity* of f is defined as

$$\Delta f_{GS} = \max_{D,D':d(D,D')=1} ||f(D) - f(D')||_1 \tag{2.1.2}$$

Global sensitivity works well when we release queries with low sensitivity, such as `count` or `sum` queries. For example, the `count` query normally has $\Delta f_{GS} = 1$, which is much smaller than the true answer. But for queries such as `median`, `average`, the *global sensitivity* yields high values. We will then resort to the *local sensitivity* for those queries [69].

The Local Sensitivity

Local sensitivity calibrates the record-based differences between query results on neighboring datasets and also satisfies the *differential privacy* definition [69]. The *local sensitivity* is defined as below:

Definition 5 (*Local sensitivity*). For query $f : D \to \mathbb{R}$, *local sensitivity* is defined as

$$\Delta f_{LS} = \max_{D'} ||f(D) - f(D')||_1, \tag{2.1.3}$$

Compared to the *global sensitivity*, *local sensitivity* has the following property:

$$\Delta f_{GS} = \max_{\delta} f_{LS}, \tag{2.1.4}$$

which generates less noise for queries with high sensitivity.

For queries such as `count` or `range`, the *local sensitivity* is identical to the *global sensitivity*. Because the significant literature is concerned with the `count` or `range` queries, for simplification, we do not differentiate *global sensitivity* and *local sensitivity* in this thesis.

2.1.5 The Principle Differential Privacy Mechanisms

Two popular mechanisms exist to achieve *differential privacy*: the *Laplace* mechanism and the *Exponential* mechanism. The first one is suitable for numeric queries and the latter is suitable for non-numeric queries.

The *Laplace* Mechanism

The *Laplace* mechanism relies on adding controlled noise to the query result before returning it to the user [25]. The noise is sampled from the *Laplace* distribution, which is centered at 0 with scaling b. The corresponding probability density function is:

$$p(x) = \frac{1}{2b} \exp(-\frac{\mid x \mid}{b}). \tag{2.1.5}$$

Definition 6 (*Laplace* mechanism). Given a function $f : D \to \mathbb{R}$ over a dataset D, the computation provides the ϵ-differential privacy.

$$\mathcal{M}(D) = f(D) + Lap(\frac{\Delta f}{\epsilon}) \tag{2.1.6}$$

where Δf represents the *sensitivity* of query f.

The ***Exponential*** Mechanism

The *Exponential* mechanism [63] focuses on non-numeric queries, pairing with a *score function* $q(D, \psi)$ that represents how good an output ψ is for dataset D. The choice of *score function* is application dependent. Different applications lead to various *score functions*. The *Exponential* mechanism is formally defined as below:

Definition 7 (*Exponential* mechanism). Let $q : (D, \psi)$ be a quality function of dataset D that measures the score of output $r \in \mathcal{R}$. Then an *Exponential* mechanism $\mathcal{M}$ is ϵ-differential privacy if:

$$\mathcal{M}(D) = \{\text{return r with the probability} \propto \exp(\frac{\epsilon q(D, r)}{2\Delta q})\} \tag{2.1.7}$$

where Δf represents the *sensitivity* of query f.

2.2 Differentially Private Data Release

The primary goal of a differentially private data release is to *release aggregate statistics on a dataset without disclosing any individual record*. Depending on the sequence of answers for the query set, there are two settings utilized in the mechanisms: the *interactive* setting and the *non-interactive* setting [22]. Subsections 2.2.1 and subsection 2.2.2 will present these two settings, respectively.

The goal of *non-interactive setting* research is to improve the accuracy of the query results. Existing research has been carried out along four research lines: the first line of research is associated with the query release, such as batch queries release [48] and group-based release [65]. The second research line is group-based techniques in the data release [65]. The third research line is interested in the contingency table release [4]. The fourth research line tries to release a synthetic dataset [11].

2.2.1 Interactive Data Release

Interactive data release can be described as follows: if a curator has a dataset D and a set $\mathcal{F}$ of queries, a data release mechanism is required to answer each query $f_i \in \mathcal{F}$ one by one until the privacy budget is used up. In other words, a query f_i will only be answered after the answer of the previous query f_{i-1} is released. The performance of the mechanism is measured by *accuracy*. The interactive data release focuses on how to answer as many queries as possible with a proper accuracy.

The accuracy of the mechanism output is measured by the *accuracy* parameter α, which represents the expected distance between the true answer and the randomized output. For any query $f \in \mathcal{F}$, the accuracy of mechanism $\mathcal{M}$ should satisfy the Eq. 2.2.1:

$$Pr_{f\in\mathcal{F}}[|f(D) - \mathcal{M}(f(D))| \leq \alpha] \geq 1 - \delta, \tag{2.2.1}$$

where δ is a constant less than $1/2$. In this equation, $f(D)$ represents the query result of f, while $\mathcal{M}$ represents the perturbation on query result $f(D)$. A less α leads to a higher accuracy.

The research interests in interactive data release have mainly been mechanism design and histogram release. We will present these in the following subsections.

Mechanism Design for Interactive Data Release

Mechanism design is crucial for interactive data release. Different mechanisms have various capabilities for answering a large number of queries, and have diverse accuracies regarding query results. Existing mechanisms can be categorized into three types: *Naive Laplace*, *Query Separation* and *Iteration*.

Naive Laplace *Naive Laplace* is the most popular mechanism that adopts the *Laplace* mechanism, in which the *Laplace* noise is added to each query result before being released [25]. It is widely considered as a state-of-the-art mechanism for data release because it can answer any type of query efficiently. However the maximal query number the mechanism can answer is limited in sub-linear n of the dataset. In addition, the noise magnitude will be dramatically large when dealing with the continuous attribute.

Query Separation Roth and Roughgarden presented the *Median* mechanism [80]. Compared to the *Naive Laplace*, the *Median* mechanism supports more queries with a fixed *privacy budget* by separating queries into hard and easy ones. Among them, easy queries can be answered by the *median* values of the hard query results so that queries do not cost any of the *privacy budget*. They estimated that among any k queries in domain X, there were $O(\log_2 k \log_2|X|)$ hard queries and the rest were easy queries. This result confirmed that the *Median* mechanism could answer exponentially more queries than the *Naive Laplace* mechanism. However, the drawback is that time complexity of the *Median* mechanism is exponential to dataset size n and the sample complexity is also super-polynomial.

Iteration Hardt et al. [34] proposed an iteration mechanism named *Private Multiplicative Weights* (PMW), which constructed a dataset sequence to answer queries by iteratively updating the datasets. PMW viewed datasets as a histogram with positive weight on each bin. The initial histogram x_0 was set as a uniform distribution over the domain. The mechanism then maintained a sequence of histogram x_0, x_1,..., x_t in t iterations, which gave increasing approximation to the original histogram x. The main advantage of PMW was that it saved the privacy budget and decreased the noise when confronting lots of queries. After the parameters are calibrated for the complexity and accuracy, this mechanism could run in polynomial time on each query with a sampling error approximate to $O(\sqrt{(\log k)/n})$.

Similarly, Gupta et al. [32] presented a general iteration framework *Iterative Database Construct* (IDC), which may implement other release mechanisms in this framework. In each iteration, when a given query witnesses a significant difference between the current dataset and the original one, the mechanism updates the current dataset for the next iteration. The update function was defined by the *Frieze/Kannan low-rank matrix decomposition* algorithm. The effectiveness of the framework was evaluated by `cut` queries in a social network graph dataset. Compared to *PMW*, IDC was a more general framework that could be incorporated into various other mechanisms.

Table. 2.1 lists the comparison between different mechanisms.

Table 2.1: Mechanism Comparsion

Type of Mechanism	Description	Advantage	Disadvantage	Algorithms
Naive Laplace	Using Laplace & Exponential mechanism directly	Easy to implement	Large Noise, small number of queries (Sub-linear to n)	Laplace
Query separation	Only adds noise to small part of queries	Noise can decrease	How to find weak queries	Median
Iteration	The datasets are sampled from the domain iteratively until converged or all queries have been answered	More queries, linear to n	Introduces extra computation complexity	PMW, IDC

Histogram Release

Histogram is a data structure that attracts a lot of research interest in differential privacy due to its naturally low sensitivity for lots of queries [22]. For example, when the histogram serves to support the `count` query, adding or removing a single record will affect, at most, one bin. Hence, the `count` query on the histogram has the *sensitivity* 1, and the magnitude of added noise on each bin will be relatively small.

We can map a dataset D with d attributes to a histogram x over the domain $\mathcal{X}$. Each bin b represents the combination of attributes, and the number of bins is denoted by N. The frequencies in a histogram are the fraction of the count of bins, which are denoted as $x(b_i), (i \in N)$. Formally, we define the histogram representation as below:

Definition 8 (Histogram Representation). A dataset D can be represented by a histogram x in a domain $\mathcal{X}$: $x \in N^{|\mathcal{X}|}$, where N consists of all possible combinations of attributes A_d in D.

Based on this definition, we can consider x as distribution over $\mathcal{X}$, with positive weight on each bin in x. Two datasets D and D' are defined as *neighboring datasets* if and only if their corresponding histograms x and x' satisfy $||x - x'||_1 \leq 1$.

To preserve differential privacy for histogram release, one direct mechanism is to add *Laplace* noise to N bins. When N is small, this mechanism can retain a high utility of the query result. However, if the original dataset contains multiple attributes, the combination of these attributes will lead to a large N. For some queries such as the `range` query (tries to obtain the answer of how many records are satisfied with the attribute failing into the interval from a to b), the answer will be meaningless due to the large amount of noise accumulated from the enormous number of bins. One way to tackle the problem is to partition the attributes into several mutual groups. For each group, we merge the related bins into a single bin and adopt an average frequency on this new bin. If we take a `range` query as an example, the sensitivity is still estimated as 1, but the total number of bins is reduced, leading to less noise in total. The partition method decreases the magnitude of noise on the query result, however, it introduces extra error when taking the average frequencies to the merged bins. How to optimize the partition method to obtain a trade-off between the *Laplace* noise and the introduction of average error is another challenge that needs to be addressed.

Xiao et al. [95] provided a *kd-tree* based partition method to generate nearly uniform partitions to decrease the average error. Their idea was to partition the original histogram and merge bins with similar frequencies together. The average frequency would then be close to those original frequencies, and would reduce the average error. They applied the *kd-tree* to identify bins with similar frequencies when answering the queries. A *kd-tree* was a binary tree so that every non-leaf node could be considered as a splitting hyperplane to divide the space into two half-spaces. Generating a balanced *kd-tree* on the histogram frequency could help to divide the original histogram into a few, almost uniform structures, which did achieve a better trade-off between the *Laplace* noise and the average error.

In their successive work [94], the algorithm was called as *DPCube* and the 2-dimensional histogram was extended into a k-dimensional ($k > 2$) one. They implemented a set of query matrix to generate an optimal query strategy on a k-dimensional histogram to test the performance. When the parameters (frequency closeness threshold, partition times) are estimated properly, the *DPCube* achieves a good balance between the query number and the errors.

Similarly, in exploration of the optimal partition method, Xu [96] provided two methods to generate optimal query results by minimizing the *Sum of Squared Error* (SSE). The first one, *NoiseFirst*, injected the *Laplace* noise to each bin before partitioning the attributes. Another algorithm, *StructureFirst*, used the *Exponential* mechanism to select an optimal partitioning strategy by adopting the SSE as the score function. Both algorithms tried to find an optimal partition method on the original histogram to support the range queries. Experiments showed the *NoiseFirst* method achieved a lower SSE for the `range` query with small intervals, while *StructureFirst* was more suited to the `range` query with larger intervals.

Another issue also involved violation of constraints. When adding noise to a query result, this may lead to inconsistency [36]. For example, suppose there exists a constraint of $count(A) + count(B) = count(C)$ in a histogram. If the noise answers of A, B and C does not hold this constraint, the consistency of the released result is compromised.

To maintain the consistency, Michael Hay et al. [36] defined a *constrained inference* to adjust the released output. Two types of consistency constraints have been explored. The first, *sorted constraints*, required query results to satisfy a particular sequence. The second, *hierarchical constraints*, predefined the sequence of a hierarchical interval set. The proposed approach provided the answer set $\widehat{\mathcal{F}}(x)$ to respond to query set $\mathcal{F}$ by using a standard differential privacy mechanism. The *constrained inference* step applied the linear combination method to estimate a set of approximate answers $\overline{\mathcal{F}(x)}$ that were close to $\widehat{\mathcal{F}}(x)$, which satisfied the consistency constraints. Both theory and experiment demonstrated the released $\overline{\mathcal{F}(x)}$ can retain the constraints, and can also decreases the query errors due to the consistency between queries.

2.2.2 Non-Interactive Data Release

Non-interactive data release manages to develop a mechanism that can answer query set $\mathcal{F}$ in a batch. It answers more queries than the interactive data release, and provides higher flexibility for data analysis. However, the major problem is that the non-interactive setting introduces too much noise to satisfy the *differential privacy*, and decreases the utility of the released data significantly. How to release more answers with a limited privacy budget while maintaining a better trade off between the utility and privacy is the essential research issue in non-interactive data release.

There are several research lines on the non-interactive data release that can be categorized to *k batch queries release*, *contingency table release*, *group-based release* and *sanitized dataset release*.

Batch Queries Release

Batch queries release refers to the non-interactive scenario that a fixed set of k queries is provided to the release mechanism and these need to be answered in a batch [48]. In this scenario, queries are correlated to each other, deleting a record will affect multiple query answers. According to the definition of the sensitivity, k batch queries have much higher sensitivity than a single query.

For example, Tab. 2.2 shows a frequency dataset D with 4 variables, and Fig. 2.3 contains all possible `range` queries $\mathcal{F} = f_1, ..., f_10$. Deleting any record in D will

Table 2.2: Frequency Table D in k Batch Queries

Grade	Count	Variable
$90 - 100$	12	x_1
$80 - 89$	24	x_2
$70 - 79$	6	x_3
$60 - 69$	7	x_4

change at most 6 query results (column contains x_2 in the Tab. 2.2) in $\mathcal{F}$. According to Def. 4, the sensitivity of $\mathcal{F}$ is 6.

In this situation, the *Laplace* noise can be added in two different mechanisms and both mechanisms will introduce high noise due to the large sensitivity. The first mechanism directly adds noise to each query result. As illustrated in the above example, the sensitivity of $\mathcal{F}$ is $O(n^2)$, and the variance of the noise per answer is $O(n^4/\epsilon^2)$. The second mechanism adds noise to the counts of variables in D and then generates query results from the noisy D. In this case, the sensitivity equal to 1, but the noise in $\mathcal{F}$ will accumulate quicker. Hence, both mechanisms lead to inaccurate answers, and *how to reduce the noise* is a challenge in batch queries release.

Projection is a type of mechanism to decrease the sensitivity of batch queries. In the projection mechanism, the original dataset is mapped to a new matrix. The sensitivity of the query set will be also adjusted. After noise is added to this matrix, the noisy matrix will be inverted to generate a new dataset. Queries will be performed on this new dataset to preserve privacy and utility guarantees.

Xiao et al. [93] developed a projection mechanism called *Privelet* that applied wavelet transformation on the dataset before adding noise to it. For a frequency

Table 2.3: k Range Queries

f_1	x_1	$+$	x_2	$+$	x_3	$+$	x_4
f_2	x_1	$+$	x_2	$+$	x_3		
f_3		$+$	x_2	$+$	x_3	$+$	x_4
f_4	x_1	$+$	x_2				
f_5		$+$	x_2	$+$	x_3		
f_6					x_3	$+$	x_4
f_7	x_1						
f_8			x_2				
f_9					x_3		
f_{10}							x_4

matrix M, they applied a wavelet transformation to generate a wavelet coefficient matrix C, in which each entry $c_i \in C$ was considered as a linear combination of entries in M. The *Privelet* added weighted *Laplace* noise to each wavelet coefficient to create C^* that ensured ϵ-differential privacy. The C^* was applied to answer some basic queries in F, and some other queries could be generated by the linear combination of basic queries. For example, Tab. 2.4 shows the basic queries C^* could answer. If we provide the query of $range(x_2, x_3) = x_2 + x_3$, the true answer can be generated by $range(x_2, x_3) = 0.5f_1 - 0.5f_3 - 0.5f_4$ and the sensitivity will decrease to 3. For this example, the sensitivity of the wavelet coefficient was estimated as $1 + \log_2 n$ and the variance of noise per answer was $O((\log_2 n)^3/\epsilon^2)$, which were much smaller than those in the *Laplace* mechanism. The *Privelet* mechanism can be extended to a multi-dimensional dataset in which the variance of noise per answer was $O((\log_2 n)^3 d/\epsilon^2)$.

Table 2.4: k Range Queries

f_1	x_1	$+$	x_2	$+$	x_3	$+$	x_4
f_2	x_1	$+$	x_2	-	x_3	-	x_4
f_3	x_1	-	x_2				
f_4					x_3	-	x_4

Other research has also used projection mechanisms in batch query release. For example, Li et al. [49] applied a *Matrix* mechanism to map the batch query into a new query set to decrease sensitivity. Hay et al. [36] projected the original dataset to a hierarchical structure for `range` queries.

Group-based Techniques in Data Release

Group-based techniques in data release combines traditional group-based technology with *differential privacy* [53,65]. *Differential privacy* makes an assumption that based on the background information, an adversary obtains every other records except the record he/she wants to know. Researchers argues this assumption is too rigorous to retain sufficient utility. On the other side, traditional group-based privacy models such as *k-anonymity* or *l-diversity* underestimate the background information. They assume an adversary only knows the background information of an individual he/she wants to identify. Even group-based techniques can maintain a higher utility, with the privacy guarantee for datasets insufficient. If a mechanism can combine differential privacy and group-based privacy, it is possible to obtain a trade off between the privacy and utility.

Li et al. [53] adopted random sampling to associate *k-anonymity* with *differential* *privacy*. Their insight underlying the paper is to add uncertainty to the k-*anonymity* model. Based on this idea, a safe k-*anonymization* algorithm was established. The algorithm first defined a function A to sample random records from dataset D and added the sampled record $A(D)$ to the original D: $S_D = D + A(D)$. It then deleted the records that appeared less than k times in the S_D and released the new dataset. Li et al. proved that if function A meets the requirement of *differential privacy*, the released dataset would satisfy *differential privacy*. Their research provided a new perspective on how to bridge the gap between k-anonymization and *differential privacy*. However, it did not evaluate the utility of the released dataset. From the perspective of differential privacy, the released dataset can hardly provide a satisfactory result for all types of analytical tasks. Li's work did not evaluate which type of tasks the released dataset would undertake.

Mohammed and Chen [65] presented an *anonymization* algorithm based on the *generalization* technique for classification. The general idea was to transfer the original dataset to a top most general representation, and specialize it under the differential privacy guarantee. Three steps exist in their algorithm: selecting a candidate attribution for specialization, determining the split value of an attribute and adding noise to the count in each group. In the first step after the dataset was generalized, the algorithm applied the *Exponential* mechanism to each attribute to select a specific value to replace the general one according to a predefined taxonomy tree. In the second step, the specified dataset was split into several *equivalence groups* in which every record had the same attribute values. Noise was then added to the count of records in each group in the last step. The algorithm eventually released the noise counts of records in each group for the classification task. The classification accuracy was demonstrated by a set of experiments. It could be proven these three steps were all guided using the *differential privacy* mechanism. Therefore, the final published noise count can fulfill the *differential privacy* requirement.

Contingency Table Release

Contingency table displays the frequencies of the combined attributes in a dataset. It is a popular data structure for data analysis in the *medical science*, and *social sciences* fields [27].

When a dataset contains n records and d binary attributes, the contingency table for D is a frequency matrix of 2^d possible combinations of these attributes. Normally, the curator does not release the entire contingency table because when d is large, the contingency table is likely to be sparse. Instead, the curator will release subsets containing parts of attributes that are defined as the *k-way marginal frequency* table ($k \leq d$). One contingency table may contain several overlapped marginal frequency tables.

There are two methods to privately release these *k-way marginal frequency* tables. The most intuitive one is adding noise to the frequency of the whole contingency table [43]. Users can create any *k-way marginal frequency* table from the noisy contingency table and maintain the consistency of all tables. However, this method leads to the noise magnitude of $O(2^d)$, when dimension d is large, the noise will increase dramatically and render the perturbation results unrealistic. Another method is to extract the marginal frequency tables from the original dataset and then add the noise to frequencies. This method yields excellent accuracy when n is large compared to the number of marginal tables. But it can lead to the violation of consistency.

For example, the counts of the same attributes in different marginal frequency tables may not be equal, or might violate common sense. The research issue in contingency table release is to retain the accuracy, as well as the consistency of the *k-way marginal frequency* table.

For the noise measurement, Kasiviswanathan et al. [43] investigated the minimal noise magnitude for releasing a set of *k-way marginal frequency* tables. They simulated attacks in polynomial time that allows an adversary to reconstruct sensitive information from a marginal table. The result proved that bound of $\tilde{\Omega}(\min\{\sqrt{n}, \sqrt{d^{k-1}}\})$ on the average magnitude of noise per record was insufficient to preserve privacy. Based on the attack result, they obtained a stronger lower bound of $\tilde{\Omega}(\min\{\sqrt{n}, \sqrt{d^{k}}\})$ noise for marginal table release. Their analysis can be extended to design a strong mechanism satisfying differential privacy. However, this measurement was only performed on a low-dimensional contingency table. For a high dimensional domain, the result was still unclear.

For the consistency issue, Barak et al. [4] proposed an algorithm to retain the consistency by transforming the contingency table into the *Fourier* domain. This transformation served as a non-redundant information encoding method for the marginal tables. Since any set of *Fourier* coefficients corresponds to one contingency table, adding noise in this domain will not violate the consistency for marginal frequency tables. Linear programming was then applied to obtain a non-negative, integrate marginal frequency table with the given *Fourier* coefficients. Finally, without compromising the differential privacy, they produced a set of consistent marginal frequency tables.

Synthetic Dataset Release

Synthetic dataset release refers to the scenario of releasing a dataset instead of query results. For a long time it was considered a difficult problem due to the large noise introduced. But with theoretical development in differential privacy, the inaccuracy problem can be solved by introducing computational learning theory to the synthetic dataset release.

From the perspective of computational learning theory, the main purpose of analyzing a dataset is to obtain information about a certain class. The observation can be applied in the synthetic dataset release. If the query on a dataset is limited within a particular class, the added noise will not affect the utility. Specifically, let $\widehat{D}$ be a synthetic dataset sampled from a domain $\mathcal{X}$, if we are only concerned about the Boolean prediction of a class $\mathcal{C}$, for every query f derived from $\mathcal{C}$, the ratio between the positive result in $\widehat{D}$ and in $\mathcal{X}$ is bounded by the accuracy parameter α. This conclusion provides an efficient way to obtain a trade off between utility and privacy in the synthetic dataset release. Hence, there are two research issues in the release procedure. The first issue is *how many records are needed in the synthetic dataset to achieve an α-accurate result*? The second issue is *can we develop an algorithm to obtain the synthetic dataset in polynomial time*?

Both research issues were first analysized by Blum et al. [11], who investigated the resources requirement in terms of query types, size of dataset D (sample complexity), the usefulness parameter α (accuracy) and time complexity. In their work, Blum et al. demonstrated the of releasing a synthetic dataset that was useful for queries from any given class over a discretized domain. Specifically, they employed the *Exponential* mechanism to search a *synthetic* dataset $\widehat{D}$ from the domain that could answer `interval` and `halfspace` queries for all classes in $\mathcal{C}$. In this searching procedure, several essential bounds on the resources were analyzed: Firstly, to measure the accuracy of the released synthetic dataset, an *usefulness* definition was provided as follows:

Definition 9 ((α,δ)-usefulness). A dataset mechanism $\mathcal{M}$ is (α,δ)-usefulness for queries in class C if with probability $1-\delta$, for every $f \in \mathcal{C}$, and every dataset D, when $\widehat{D} = \mathcal{M}(D)$, we have

$$|f(\widehat{D}) - f(D)| \leq \alpha \tag{2.2.2}$$

.

This definition can be applied to evaluate the $\mathcal{M}$ that reserves the utility of the final released data on a certain type of query. For their exponential searching mechanism, the low bound of useful parameter α is $O(n^{\frac{2}{3}} \log^{\frac{1}{3}} \log|C|)$, which is associated with the size of D and class $\mathcal{C}$.

Secondly, they presented the number of records needed to achieve an α,δ)-usefulness result. Kasiviswanathan et al. [42] claimed that, for any class of $\mathcal{C}$ and any dataset $D \geq \{0,1\}^d$, if the size of the dataset satisfied

$$|D| \geq O(\frac{dVCDIM(\mathcal{C})\log(\frac{1}{\alpha})}{\epsilon\alpha^3} + \frac{\log\frac{1}{\delta}}{\epsilon\alpha})$$

, we can successfully obtain a (α,δ)-usefulness dataset $\widehat{D}$ of D. Blum et al. [11] provided a tighter bound on the size of the dataset in their mechanism:

$$|D| \geq O(\frac{\delta\log\delta + \log(\frac{1}{\alpha\delta})}{\epsilon\alpha^3})$$

.

But for the time complexity, none of the mechanisms could perform on a polynomial time. The time cost is super-polynomial in the size of universe $\mathcal{X}$ and the concept class $\mathcal{C}$. Namely, it is $|\mathcal{X}|^{poly(n)\log|\mathcal{C}|}$. Blum et al. confirmed that if we require a polynomial time, the definition of usefulness should be relaxed.

There were other works that have investigated synthetic dataset release mechanisms. For example, Dwork et al. [26] divided class $\mathcal{C}$ into several subsets and iteratively proposed synthetic datasets for every subset. Hardt et al. [35] considered the synthetic data release as a learning procedure that approximated the query result by a threshold. Beimel et al. [5] provided a tighter bound for sample complexity: for a concept $c_j \in \mathcal{C} : \{0,1\}^d \rightarrow \{0,1\}$, the non-interactive synthetic dataset size is $\Omega(\frac{(d+\log(\frac{1}{\beta})}{\epsilon\alpha}))$.

In summary, the computational learning theory extends the research work on non-interactive data release, proving it is possible to maintain an acceptable utility of the synthetic dataset while preserving differential privacy. However, *how to reduce the computational complexity*, and *how to provide various types of queries on these datasets* remain a challenge.

2.2.3 Summary

Differentially private data release has attracted substantial attentions due to its advances in social network, data collection and storage technology. Along the research line of PPDR, both interactive and non-interactive settings have their advantages and disadvantages. The *interactive* setting induces lower noise per query result but can only answer a sublinear number of queries in total. On the contrary, a *non-interactive* setting can answer an exponential number of query, but with high complexity. Table. 2.5 compares their basic profiles in terms of the number of queries, the noise magnitude and the efficiency of their mechanisms.

Table 2.5: Comparison between *Interactive* and *non-Interactive* Settings

Setting	Query Number	Magnitude of Noise	Time Complexity
Interactive	Sub-linear of n	$o(n^{1/2})$	Polynomial time
Non-Interactive	Exponential of n	Larger than $o(n^{1/2})$	Super-polynomial time for most release mechanisms

2.3 Differentially Private Data Mining

Differentially private data mining aims to release an approximate accurate model while maintaining every individual's privacy in the dataset. It is usually associated with specific data mining tasks, such as *classification*, *frequent pattern mining* and *clustering*.

Two scenarios exist on PPDM: *data mining with interface*, which considers the data miner as an untrusted party and adopts an interface between the dataset and the data miner; *data mining with fully access*, in which the data miner can access the dataset freely and directly implement differential privacy mechanisms in data mining algorithms.

2.3.1 Data Mining with Interface

In this scenario, the data miner can only access the dataset through the interface to obtain the aggregate information. *Differentially privacy* mechanism is implemented on the interface to ensure the reply enforces each individual's privacy. Through the interface, the data miner need not worry about the privacy requirement, and the main research issues are with designing the interface for diverse data mining algorithms with a limited *privacy budget*.

Two essential interfaces provide the differential privacy: *Sub-Linear Queries* (SuLQ) framework [10, 17] and the *Privacy Integrated Queries platform* (PINQ) [60]. Both have been implemented in the data mining tasks.

The *Sub-Linear Queries* (SuLQ) interface contains an additive noise operator that adds noise to query results to preserve differential privacy. For each query $f : D \rightarrow [0, 1]$, SuLQ answers the query by $\sum_i f(r_i) + N(0, R)$, where $N(0, R)$ is a *Normal* distribution with zero mean and variance R. Based on the additive noise operator, SuLQ framework can support advanced functions such as *singular value decomposition*, *k-means*, *ID3* classifier, and other statistical queries [10]. One limitation of SuLQ lies in its lack of any *Exponential* mechanism. When dealing with non-numeric queries, SuLQ can not provide sufficient privacy guarantees [28].

On the other hand, the *Privacy Integrated Queries Platform* (PINQ) [60] provides more operators than SuLQ. It supports both the *Laplace* mechanism and the *Exponential* mechanism, and provides an operator `Partition` to execute queries on disjointed datasets that it can take advantage of parallel composition. The data miner can utilize the *privacy budget* more efficiently in this interface.

In the follow sub-sections, we will present different data mining algorithms with these interfaces.

Classification

Classification algorithms construct models (classifiers) from a training dataset to predict the categorical class labels for unobserved records [40]. *Differential privacy* classification aims to produce a classifier that can predict the record labels without compromising the privacy of individuals in the training dataset.

As one of the most popular classifiers, Iterative Dichotomous (ID3) was the first algorithm applied in differential privacy. ID3 recursively chooses attributes to create a decision tree by splitting the training records [75]. Assuming there is an input dataset D with d categorical attributes $\mathbf{a} = \{a_1, ...a_d\}$ and class label $\mathcal{L} = \{L_1, L_2, ...\}$. The decision tree is constructed from the root $\dashv$ that holds all the training records. The algorithm selects the attribute a_i that maximizes the *information gain* to partition the records into child nodes. The procedure performs recursively on each node until all the records in the node are with the same value or all the attributes are exhausted.

The SuLQ framework developed a differentially private ID3 algorithm [10]. It applied an additive noise operator on the *information gain* calculation and the private ID3 algorithm picked an attribute a_i whose noisy *information gain* was not more than a threshold. In this way, the private ID3 introduced uncertainty to the output and preserved the *differential privacy.*

The SuLQ has two major disadvantages when dealing with the ID3 algorithm. The first one involves the privacy budget arrangement. In SuLQ, *information gain* is evaluated separately for each attribute in each iteration. It means the *privacy budget* will be consumed several times in each iteration, which is wasteful use of the *privacy budget.* How to obtain a trade off for the *privacy* and the classification *accuracy* still remains a problem.

Another problem is dealing with continuous attributes. SuLQ only provides the *ID3* algorithm to deal with categorical attributes. For a continuous attribute, if we simply transfer it into intervals, the basic concept of *differential privacy* is violated, because the split values in continuous attributes reveal information on the records.

To alleviate these disadvantages, Friedman and Schuster [28] considered a classification algorithm within the framework of the PINQ [60]. They showed that a naive utilization of an additive operator in SuLQ would lead to inferior results. They improved the ID3 algorithm in two ways: Firstly, they implemented the *Exponential* mechanism in the attributes choosing step. The score function q in this *Exponential* mechanism was defined by the *information gain* or the *gain ratio*. The attribute with a top score has a higher probability of being selected. In this way, the *privacy budget* was consumed only once when finding the best splitting attribute, and less noise was introduced. Secondly, their algorithm could deal with continuous attributes. The attribute values were divided into ranges by every possible splitting value, which were considered candidate options selected by the *Exponential* mechanism. When the splitting values were decided, the continuous attribute was divided into intervals.

From these two algorithms, we can conclude that introducing the *Exponential* mechanism in the interface will enhance the performance of the data mining algorithm.

Clustering

As an unsupervised learning algorithm, *clustering* groups unlabel records into clusters so that all records in the same cluster are similar to each other. Assuming that the input dataset includes records $r_i \in \mathbb{R}^d$ and the clustering output are k centers $c_1, ..., c_k$, with each set of records close to these centers respectively. The target of *Differential*

Privacy clustering is to ensure the center c_i and the number of records in each cluster will not change dramatically if one record is removed from the dataset.

The SuLQ framework provided a differential privacy solution for *k-means* clustering [10]. The number of records in each cluster would be masked by adding noise. SuLQ approximates the number of points $\overline{s_j}$ in each cluster:

$$\overline{s_j} = \begin{cases} SuLQ(f(r_i)) := 1 & \text{if } j = arg\min_j \\ 0 & \text{otherwise} \end{cases}$$

Meanwhile, it approximates the center of each cluster c_j,

$$\overline{c_j} = \begin{cases} SuLQ(f(r_i)) := r_i & \text{if } j = arg\min_j \\ 0 & \text{otherwise} \end{cases}$$

In this way, SuLQ masked the center and the number of records in each clusters.

However, the SuLQ framework did not explicitly measure the sensitivity of cluster centers, which made noise calibrating a difficult task. In actual fact, adding noise to a cluster centers was impractical because by definition, the sensitivity of the center query was measured by the largest diameter of all clusters. The noise derived from this large sensitivity would essentially erase all centers.

To deal with the issue of large sensitivity in cluster centers, Nissim et al. [69] adopted the *local sensitivity* on *k-means* clustering. They circumvented this problem by relying on some intuitions: in a well-clustered scenario, a noisy record should have approximately the same center as its previous center. In addition, in "well-clustered" records, moving a few records would not eventually change the centers. Based on these intuitions, they define a *local sensitivity* to measure the record-based sensitivity of the cluster center, which was much lower than tractional global sensitivity. Since the value of *local sensitivity* and its smooth bound were difficult to measure, they further provided a sample-aggregate framework to approximate the *local sensitivity* and its smooth bound.

2.3.2 Data Mining with Full Access

Data mining with full access considers data miners as trusted parties who can obtain all records and implement mining algorithms freely. The target is to ensure mining models will not leak the information of individuals. This scenario provides more flexibility to data miners to choose private operators in the data mining procedure, however data miners should have prior knowledge on privacy preserving techniques.

Classification

A traditional classification problem can be described as follows: suppose a training dataset D contains records with binary labels L: for any $r_i \in D$, $(r_i, L_i) \in \mathbb{R} \rightarrow \{-1, 1\}$. A learning procedure $\mathcal{M}$ will construct classifier producing prediction labels in $-1, 1$ for observing records. When defining a loss function $l(D) \in \mathbb{R}$, a classifier can be represented by a vector $\mathbf{w}$ in $\mathbb{R}$ with minimal $l(r)$.

The differentially private classification algorithms ensure classifiers derived from neighbor datasets D and D' do not differ significantly by adding noise λ to the $\mathbf{w}$ or $l(D)$.

ID3 Jagannatham et al. [37] provided a random private decision tree with a ID3 algorithm. Unlike the traditional decision tree, the random decision tree would randomly select attributes for nodes instead of using any carefully defined criteria. The proposed private ID3 algorithm first created a tree in which all the leaves were on the same level, and then built a leaves count vector V with $M * T$

integers, where M is the number of leaf nodes and T is the number of possible labels for records in the training dataset. Once the independent *Laplace* noise was added to the count vector V, a differentially private random decision tree could be generated from the noisy V. Because neighbor datasets only differ in 1 on the count of leaf nodes, the sensitivity was measured to 1 and the noise was sampled from $Lap(1/\epsilon)$. This algorithm would be iteratively performed to produce multiple random decision trees. and the ensemble method would eventually combine these trees to make the final prediction. The private ID3 algorithm randomly chooses attributes to generate nodes without any additive noise. This step would save privacy budget ϵ, resulting in an accurate prediction result.

Logistic Regression Chaudhuri and Monteleoni proposed two private solutions on *logistic regression* [16] by bringing random noise into algorithms.

The first solution injected *Laplace* noise in classifier $\mathbf{w}$ to provide a private logistic regression algorithm. The sensitivity is $\frac{2}{n\phi}$, where ϕ is the regularization parameter. But with analysis, they argue the learning performance of this simple algorithm would degrade with the decreasing of ϕ.

They presented a second solution by adding noise to the loss function $l(D)$ before outputting the classifier. After analyzing the performance of both algorithms, the second solution outperformed the first one. This indicated that adding noise to the loss function will obtain a better tradeoff than perturbing the classifier.

Frequent Itemset Mining

Frequent itemset mining is a popular data mining task that aims to discover itemsets that frequently appear in a transactional dataset. Suppose D is a *transaction* dataset and I is a set of items, and an *itemset* refers to a subset that is sampled from I. Let each record $r_i \in D$, be denoted as a *transaction* that contains a set of items from I, and a frequent itemset refers to a set of items whose number of occurrences in transactions is above a threshold. The proportion of supporting transactions in the dataset is defined as the *frequency.* Let U represent all frequent itemsets, releasing $top{-}k$ most frequent itemsets in U will directly bring risk to the privacy of individuals. We should release the *top-k* frequent itemsets under a rigorous privacy guarantee.

The main challenge in differentially private frequent itemset mining is that the total number of itemsets is exponential to the number of items: If I contains m items, the number of all possible itemsets is $|U|= \sum_{i=1}^{k}\binom{m}{i}$. The differential privacy mechanism has to compute the frequency of all the item combinations and add noise to all frequencies, which is computationally expensive. How to decrease the number of candidate itemsets, is a major research issue in differentially private frequent itemset mining.

Bhaskar et al. [7] managed to solve the problem and present a *top-k* frequent itemsets mining algorithm $DiffFIM$. The $DiffFIM$ contains two major private steps: *private selection* and *frequency perturbation. Private selection* chose top k itemsets by applying a novel method, *truncated frequency*, which avoided explicitly enumerating through all itemsets in U. The *truncated frequency* of itemset $X \in U$ was defined as $\widehat{p(X)} = max(p(X), p_k - \gamma)$, where p_k was the frequency of the k-th most frequent itemsets and $\gamma \in [0, 1]$ was defined as an accurate parameter that controlled the value of the *truncated frequency.* Every itemset with a frequency greater than $p_k - \gamma$ was computed as its normal frequency $p(X)$ while the rest of them was considered as a superset. This superset used $p_k - \gamma$ as a lower bound of the frequency and used γ to ensure that itemsets with frequencies less than $p_k - \gamma$ would be selected with low probabilities. Eventually, the *frequency perturbation* released the frequencies of these k itemsets with the *Laplace* noise perturbation.

The advantage of the *truncate frequency* is that it could significantly decrease the computing complexity of the algorithm. However, it still has weaknesses. Li et al. [52] argued that *truncate frequency* was only applicable when k was small, otherwise, the accurate parameter γ might be larger than q_k. Another weakness was the *top-k* itemsets should be pre-defined as length m, which decreases the flexibility of the frequent itemset mining.

To address these weakness, Li et al. [52] proposed an algorithm, *PrivBasis*, introduced a new notion of *basis sets* to avoid the selection of *top-k* itemsets from a very large candidate set. More specifically, given some minimum support threshold, θ, they constructed a basis set $\mathcal{B} = B_1, B_2, ..., B_w$, so that any itemset with a frequency higher than θ was a subset of some basis B_i. We introduced techniques for privately constructing basis sets, and for privately reconstructing the frequencies of all subsets of B_i with reasonable accuracy. One could then select the most frequent itemsets from the reconstructed subsets.

The *PrivBasis* algorithm could release arbitrary length itemsets and ensure high accuracy. But generating *basis sets* B was not easy. Furthermore, when the length of itemset l and the number of *basis sets* w were large, the cardinality of the candidate set was still too big to deal with. Hence, how to efficiently decrease the size of the candidate set is still a challenge.

2.3.3 Summary

PPDM is a relative new research area that combines privacy and data mining to design private analogues of popular data mining algorithms. *Differential privacy* provides a theoretically framework to evaluate the privacy risks of releasing models, and this is a starting point for further research work on PPDM. Table 2.6 shows the comparison between different types of data mining algorithms, representing only a small part in the data mining research community. It is worthwhile to incorporate the differential privacy in various data mining algorithms in the future.

2.4 Applications of Differential Privacy

The concept of *differential privacy* was first applied in the statistical database community [21]. The principal motivation is to release statistical information without compromising the privacy of individual respondents. Because *differential privacy* provides a rigorous and provable privacy definition, researchers have shown an increased interest in differential privacy applications [19,30,61,62], such as *recommender system*, *tagging recommender systems* and *coupled differential privacy*.

2.4.1 Privacy Preserving Recommender System

Recommender systems are achieving widespread success on e-commerce Web sites, which are capable of recommending users the products they probably like. *Collaborative Filtering* (CF) is one of the most popular recommendation techniques as it

Table 2.6: Comparison between Data Mining Algorithms

Setting	Mining Method	Typical Algorithm	Advantage	Disadvantage
Interface	Classification	DiffP-ID3 [28], DiffP-C4.5 [28], SuLQ-based ID3 [10], PINQ-based ID3 [28]	Easy to implement; High classification accuracy	Iteration round has to be predefined, which makes it difficult to allocate the privacy budget.
	Clustering	SuLQ-based k-means [10], Sample Aggregate Framework [69]	Easy to implemented	High Sensitivity
Fully Access	Classification or Logistics	Random ID3 [37], Objective Perturbation [16]	High classification accuracy	High computing complexity
	Frequent Itemset Mining	FIM [7], PrivBasis [52]	Easy to implement	Large candidate itemsets

is insensitive to the product details. This is achieved by analyzing users' historical transaction data with various data mining or machine learning techniques, e.g. k nearest neighbor rule, the probability theory and matrix factorization. Accordingly, CF methods are generally categorized into the *neighborhood-based* methods and the *model-based* methods. However, there is a potential privacy leak risk in the recommendation process. Within the literature, it has demonstrated that continual observation of the recommendations with some background information makes it possible to infer the individual's rating or even transaction history, especially for the *neighborhood-based* methods [12]. This is usually referred to as the *KNN attack*, in which an adversary can infer the rating history of an active user by creating some fake neighbors based on background information [12].

As a prominent privacy definition, the *differential privacy* technique has been incorporated into the research on recommender systems. For example, McSherry et. al. [62] is the first group to introduce the *differential privacy* notion to CF. They add the *Laplace* noise into the covariance matrix, and use the *non-private* recommender algorithm on this matrix to predict ratings. Another example is the work provided by Banerjee et. al. [3]. They implement a private local recommendation algorithm and analyse the lower bounds of the sample complexity of their algorithm. However, both methods only focus on the untrusted recommender system.

Machanavajjhala et al. formalized trade-offs between accuracy and privacy of graph-based social recommendations in other research [58]. What they were concerned with was the graph network instead of traditional rating dataset.

When addressing the issues of privacy in recommender systems, the main issue is how to achieve a trade-off between the level of privacy and the performance of recommendations. Most traditional privacy preserving methods lack a rigid privacy guarantee while *differential privacy* suffers from unsatisfactory performance.

2.4.2 Privacy Preserving Tagging Recommender System

A *tagging system* allows Internet users to annotate resources with personalized tags. The connection among users, resources and annotations, often referred to as *folksonomy*, provides users the freedom to explore tags, and obtain recommendations. The release of tagging datasets accelerates both commercial and research work on recommender systems. However, a *tagging system* is usually confronted with serious privacy concerns, because adversaries may re-identify a user and their sensitive information from the tagging dataset with only a little background information.

Parra-Arnau et al. [71] made the first contribution towards the development of a privacy preserving tagging system by proposing the *tag suppression* approach. They first modeled the user's profile using a tagging histogram and eliminated sensitive tags from this profile. To retain utility, they applied a clustering method to structure all tags, and then suppressed the less represented ones. Finally, they analyzed the effectiveness of their approach in terms of the *semantic loss* of users. However, there were several limitations to *tag suppression*. It only releases an incomplete dataset, with parts of the sensitive tags deleted, and sensitive tags are considered subjective without any quantity measurement. Furthermore, if the dataset is publicly shared, users can be identified because the remaining tags still have the potential to reveal a user's identity. The privacy issue in tagging recommender systems remains largely unexplored, and we attempt to fill this void in Chapter 4.

2.4.3 Differential Privacy for a Coupled Dataset

Existing research on differential privacy assumes that records are sampled independently in a dataset. However, in real-world applications, records in a dataset are rarely independent. The relationships among records are referred to as coupled information and the dataset is defined as a coupled dataset. A differential privacy technique performed on a coupled dataset will disclose more information than expected, and this indicates a serious privacy violation.

Currently, only limited efforts has been given to this line of research. Kifer et al. [44] were the first to argue that differential privacy without considering the correlation between records would decrease the privacy guarantee on the coupled dataset. For example, suppose a record r has influence on a set of other records, and this set of records will provide evidence on r even though record r is deleted from the dataset. In this scenario, traditional differential privacy fails to provide sufficient privacy as it claims. To deal with the problem, their successive paper proposed a new privacy framework, *Pufferfish* [45], which allows application domain experts to add an extra data relationship to develop a customized privacy definition. However, the *Pufferfish* framework does not satisfy differential privacy, and the privacy guarantee for coupled datasets still calls for further investigation.

Chen et al. [18] considered the social network as a coupled dataset and easily dealt with the coupled problem by multiplying global sensitivity with the number of coupled records. They defined the maximal number of coupled records as $k = 25$, but this naive method was not optimal because it introduced a large amount of noise into the output, which overwhelms the true answer and demolishes the utility of the dataset.

Consequently, when addressing the privacy issue in a coupled dataset, the essential problem is how to model the extra background information introduced by coupled records. This is still an open issue that needs to be explored further.

2.5 Summary

Differential privacy is an innovative privacy model with a strong mathematical foundation for privacy preserving. The privacy model thrives because it is not domain-specific and can interplay with other sub-areas. Over the past decade, the research community has extended the theory of *differential privacy* to a variety of areas, such as *statistics*, *geometry*, *complexity theory*, *learning theory* and *machine learning* etc.

This chapter surveys recent progress in research on differential privacy in data release and data mining. The ever-increasing amount of various datasets in different contexts and environments presents both an opportunity for data release and data mining, but also a challenge to effectively preserve privacy for individuals. As a young and promising field, differential privacy still faces a number of unresolved problems. Here, we recall and highlight three issues that will be addressed in this thesis:

- The first is the *application-aware sensitivity* issue.
- The second is the issue of shrinking the randomized domain for the sparse dataset.
- The third is to propose a coupled differential privacy solution for coupled datasets.

Specifically, this thesis focuses on the *application-aware sensitivity*, as demonstrated in Chapter 3, with emphasis on neighborhood-based collaborative filtering methods. The issue of shrinking the randomized domain for sparse tagging datasets will be investigated in Chapter 4. Chapter 5 explores the coupled differential privacy solution for coupled datasets.

Chapter 3

Differentially Private Neighborhood-based Collaborative Filtering

3.1 Introduction

In this chapter, we aim to address the application-aware sensitivity issue in the context of neighborhood-based collaborative filtering. *Collaborative Filtering* (CF) is one of the most popular recommendation techniques because of its insensitivity to product details. The recommendation is usually achieved by analyzing the user's historical transactions with various data mining or machine learning techniques, e.g, *k nearest neighbor rule*, the *probability theory* and *matrix factorization* [79]. Accordingly, CF methods can be categorized into *neighborhood-based* methods and *model-based* methods [56]. However, there is potential for a breach of privacy in the recommendation process.

The literature has shown that with some background information, continual observation of recommendations makes it possible to infer the individual's rating or even the transaction history, especially for the *neighborhood-based* methods [12]. This is usually referred to as a *KNN attack*, in which an adversary can infer the rating history of an active user by creating fake neighbors based on background information [12]. In this chapter, we aim to address the privacy preserving issue in the context of *neighborhood-based* CF methods.

Typically, a collaborative filtering method employs certain traditional privacy preserving approaches, such as cryptographic, obfuscation and perturbation. Among them, *Cryptographic* is suitable for multiple parties but induces extra computational cost [13, 98]. *Obfuscation* is easy to understand and implement, however the utility will decrease significantly [6, 70]. *Perturbation* preserves high privacy levels by adding noise to the original dataset, but the magnitude of noise is subjective and hard to control [73]. Moreover, these traditional approaches suffer from a common weakness: the privacy notion is weak and hard to prove theoretically, thus impairing the credibility of the final result. In order to address these problems, *differential privacy* has recently been proposed because it is a more rigid notion that provides a strong and provable privacy definition that can quantify the privacy risk to individuals [21, 22].

Differential privacy was introduced into CF by McSherry et al. [62], who pioneered a study that constructed the private covariance matrix to randomize each user's rating before submitting to the system. Machanavajjhala et al. [58] presented a graph link-based recommendation algorithm and formalized the trade-off between accuracy and privacy. Both of them employed *Laplace* noise to mask accurate ratings so the actual opinions of an individual were protected.

Although *differentially privacy* is promising for privacy preserving CF due to its strong privacy guarantee, it still has some limitations and research barriers. More specifically, there are two weaknesses in existing work:

- Existing methods usually fail to hide similar neighbors, which makes CF vulnerable to *KNN attack*. This kind of attacks was first mentioned in Calandrino's work [12]. When CF provides similar users or items explicitly or implicitly, the adversary can infer the rating history of a target user by creating fake neighbors based on background information. The *KNN attack* is consequently referred as a serious privacy violation. Existing privacy methods only protect the rating of the users, but not the users themselves. In actual fact, neighbors can reveal sensitive information about a target user.

- *Differential privacy* usually induces a large noise that affects the quality of the selected neighbors. Existing work usually leads to significant accuracy loss when obtaining sufficient privacy. Large noise occurs for two reasons: the high *sensitivity* and the naive mechanism. Informally, *sensitivity* calibrates the information that needs to be hidden in a query when an individual is deleted in the dataset. It directly determines the size of the noise to be added to each query [21]. Unfortunately, the queries employed in recommendation techniques always have high *sensitivity*, followed by the addition of large noise. A naive mechanism is another issue that leads to high noise. Previous work directly uses the *differential privacy* mechanism and disregards the unique characteristics of recommendations, thus negatively affecting the recommendation performance.

To overcome these weaknesses in *neighborhood-based* CF methods, we propose a

Private Neighbor Collaborative Filtering (*PNCF*) algorithm in this chapter. This idea is based on two observations. Firstly, all possible privacy leakage should be considered. For example, both the ratings and the neighbors are targets that need protection. However, prior work ignores the protection of neighbors. Secondly, *sensitivity* and *mechanism* should be integrated with the requirement of applications. *Differential privacy* was initially proposed as a promising solution to private counting queries [22], whose *sensitivity* is much lower than operations in CF. Hence, an adaptive mechanism is expected. The proposed *PNCF* algorithm design is based on the two observations to *provide comprehensive privacy for individuals* while *minimizing the accuracy loss of recommendations*.

To achieve the objective, three research issues will be addressed in this chapter:

- *How to preserve neighborhood privacy?* Both a user's neighbors and the original ratings will be hidden in a private CF. How to protect neighbors is the primary issue that needs to be considered. We provide *Private Neighbor Selection* in our algorithm to reduce the probability that an adversary will infer similar users or items from candidates. Independent *Laplace* noise is then added to hide a user's original rating scores. These privacy preserving steps will protect both the neighbors and the ratings.

- *How to define sensitivity for recommendation purposes?* Traditional *sensitivity* measurement is not suitable for CF due to high dimensional input. How to define a new *sensitivity* is another issue to be addressed. To preserve the performance, we define a practical *Recommendation-Aware Sensitivity* for CF, which reduces the magnitude of noise when compared with traditional *sensitivity*.

- *How to design the exponential mechanism for CF?* The performance of *neighborhood-based* methods is largely dependent on the quality of selected neighbors. The third issue is how to enhance the quality of selected neighbors in a privacy preserving process. A naive *differentially private* mechanism leads to inferior quality neighbors. Enhancing the quality of neighbors is a promising way to improve performance. By re-designing the private selection mechanism, we retain the accuracy from the final output result.

The rest of this chapter is organized as follows. We present foundational concepts on the recommender system in Section 3.2, and propose the *PNCF* algorithm in Section 3.3. In this section, we also undertake theoretical analysis on *sensitivity* in the privacy preserving stage. Section 3.5 presents the empirical results and analysis, followed by the conclusion in Section 3.6.

3.2 Fundamentals of a Recommender System

In this section, we introduce the foundational concepts in collaborative filtering, and briefly review related work.

3.2.1 Notation

Let $U = \{u_1, u_2...u_n\}$ be a set of users and $I = \{t_1, t_2...t_m\}$ be a set of items. The *user* $\times$ *item* rating dataset R is represented as a $n \times m$ matrix, which can be decomposed into row vectors: $D = [\mathbf{u_1}, \mathbf{u_2}, ..., \mathbf{u_n}]^T$ and $\mathbf{u_a} = [r_{a1}, r_{a2}, ..., r_{am}]$. The row vector $\mathbf{u_a}$ corresponds to the user u_a's rating list, and r_{ai} denotes the rating that user u_a gave to item t_i. D can also be represented by column vectors: $D = [\mathbf{t_1}, \mathbf{t_2}, ..., \mathbf{t_m}]$

and $\mathbf{t_i} = [r_{1i}, r_{2i}, ..., r_{ni}]^T$. For each t_i, $s(i,j)$ represents its similarity with item t_j. $N_k(t_i)$ denotes the set of item t_i's k neighbors, and $U_{ij} = \{u_x \in U | r_{xi} \neq \oslash, r_{xj} \neq \oslash\}$ denotes the set of users, co-rating on both item t_i and t_j. $s(i,j)$ denotes the similarity between t_i with t_j.

Collaborative Filtering

Collaborative Filtering (CF) is a well-known recommendation technique that can be further categorized into *neighborhood-based methods* and *model-based methods* [78]. The *neighborhood-based* methods are generally based on the k nearest neighbor rule (KNN), and provides recommendations by aggregating the opinions of a user's k nearest neighbors [77].

Two stages are involved in *neighborhood-based methods*: the *Neighbor Selection* and the *Rating Prediction.* In the *Neighbor Selection* stage, the similarity between any two users or any two items is estimated, and corresponds to the *user-based* methods and the *item-based* methods. Various measurement metrics have been proposed to compute the similarity. Two of the most popular ones are the *Pearson Correlation Coefficient* (PCC) and *Cosine-based Similarity* (COS) [1]. Neighbors are then selected according to the similarity.

- Pearson Correlation Coefficient

$$sim(i,j) = \frac{\sum_{x \in U_{ij}} (r_{xi} - \bar{r}_i)(r_{xj} - \bar{r}_j)}{\sqrt{\sum_{x \in U_{ij}} (r_{xi} - \bar{r}_i)^2} \sqrt{\sum_{x \in U_{ij}} (r_{xj} - \bar{r}_j)^2}} \tag{3.2.1}$$

  where $\bar{r}$ is the average rating given by relative users.

- Cosine-based Simlarity

$$sim(i,j) = \frac{r_i \cdot r_j}{||r_i||_2 ||r_j||_2} \tag{3.2.2}$$

For any item t_i in the *Rating Prediction* Stage, all ratings on t_i by users in $N_k(u_a)$ will be aggregated into the predicted rating $\hat{r}_{ai}$ by user u_a. Specifically, the prediction of $\hat{r}_{ai}$ is calculated as a weighted sum of the neighbors' ratings on item t_i. Accordingly, the determination of a suitable set of weights becomes an essential problem because most work relies on the similarity between users or items to determine the weight. For example, for *item-based* methods, the prediction of $\hat{r}_{ai}$ is formulated as follows:

$$\hat{r}_{ai} = \frac{\sum_{j \in I_a} s(i,j) \cdot r_{a,j}}{\sum_{j \in I_a} |s(i,j)|}. \tag{3.2.3}$$

In user-based methods, the active user's prediction is made by the rating data from many other users whose rating is similar to the active user. The predicted rating of user v on item α is:

$$\hat{r}_{v\alpha} = \bar{r_v} + \frac{\sum_{u \in U_v} s_{vu}(r_{u\alpha} - \bar{r_u})}{\sum_u |s_{vu}|}$$

, where $\bar{r_v}$ and $\bar{r_u}$ is average rating given by user u and v respectively. Similarly, s_{vu} is the similarity between user v and u.

In the item similarity matrix, element $s_{\alpha\beta} \in \mathcal{S}$ denotes the similarity between item α to item β. For a certain item α, we use a vector $\mathbf{s}_\alpha$ to represent all similarities between item α to other items. In a user similarity matrix, element $s_{vu} \in \mathcal{S}$ denotes the similarity between user v to u.

KNN attack to CF

Calandrino et al. [12] recently presented the *KNN attack*. They claim that if a recommendation algorithm and its parameters are known by an attacker, and supposing he/she knows the partial ratings history of active user u_a on m items, then the attacker can infer user u_a's remaining rating history. The inference process can be summarized as follows.

The attacker initially creates k fake users known as *sybils*. He/she arranges each *sybil*'s history rating with the m items in the active user u_a's rating history. Then with high probability, the k nearest neighbors of each *sybil* will consist of the other $k-1$ *sybils* along with the active user u_a. The attacker inspects the lists of items recommended by the system to any of the sybils. Any item on the *sybils*, for example, lists not belonging to those m items, will be an item that u_a rates. The attacker will finally infer the ratings history of an active user and this process will be considered a serious privacy violation. While this is an example for *user-based* methods, similar inference can also be processed for item based methods.

A *KNN attack* can be performed efficiently in CF due to the sparsity of a typical rating dataset. Approximately, $m = O(\log n)$ is sufficient for an attacker to infer a user, where n is the total number of users in the rating dataset. For example, in a dataset with thousands of users, $m \approx 8$ is sufficient [12]. This is such a small number that can easily be collected by an attacker. Furthermore, an attack will be more serious if an attacker can adaptively change the rating history of his sybils by observing the output of CF. This can be easily implemented in a system that allows users to change previously entered ratings. How to hide similar neighbors is a major privacy issue that cannot be overlooked.

3.2.2 Related Work

Privacy Preserving Recommender Systems

A considerable amount of literature has been published on privacy violations in recommender systems. The study first concerned with this issue was undertaken by

Ramakrishnan et. al. [76], who claimed that users' rated items across disjointed domains could face a privacy risk through statistical database queries. This hypothesis was proven by Narayanan et al. [66, 67], who re-identified part of the users in the *Netflix* Prize dataset by associating it with the *International Movie DataBase* (IMDB) dataset. However, a more serious privacy violation was presented by Calandrino et. al. [12]. By observing temporal changes in the public outputs of a recommender system, they inferred a particular user's historical rating and behavior by using background information. Specifically, they carried out a *KNN attack* on *neighborhood-based* CF methods. It should be noted inference can be successfully performed from the public output of recommender systems, while prior work needs a complete ratings dataset. Their findings endanger some of the most popular recommender systems in e-commerce, including *Amazon*, *last.fm* and *Hunch*. This serious dilemma makes the privacy issue a vital factor in the application of CF techniques.

Several traditional privacy preserving methods have been employed in CF, including *cryptographic* [13, 98], *perturbation* [73] and *obfuscation* [6, 70]. Canny [13] proposed a multi-party computation method that allows users to compute a public aggregate of their data without disclosing their true data. Each user uses local computation to get personalized recommendations. Zhan et al. [98] solved a similar problem by applying homomorphic encryption and scalar product approaches. The *Cryptographic* method preserves high performance because it does not disturb the original record. However, extra computational cost and complicated security protocols makes it harder to apply widely and it appears to be used between recommender systems rather than normal users. *Perturbation* and *obfuscation* are both similar. In particular, *perturbation* will change a user's rating systematically by adding noise before submitting to the recommender system. For example, Polat et al. [73, 74] deployed a centralized server to store the perturbed ratings with uniform noise added to each rating before making a prediction. *Obfuscation* replaces a certain percentage of a user's rating by random values. Berkovsky et al. [6] decentralized rating profiles among multiple repositories and replaced some ratings with their mean. Both *perturbation* and *obfuscation* offer a high level of privacy because all or part of a user's ratings are not true. But the magnitude of noise or the percentage of replaced ratings are subjective and difficult to control. Therefore, how to obtain a tradeoff between privacy and utility is a difficult task for both techniques.

Moreover, one criticism of traditional private recommender systems is their near inability to measure privacy levels, thus impairing the credibility of the final result. A more rigid and provable privacy notion is required in current recommender systems.

Differential Private Recommender System

The concept of *differential privacy* was first applied in the statistical database community. The principal motivation was to release statistical information without compromising the privacy of individual respondents [21]. Because *differential privacy* provides a rigorous and provable privacy definition, researchers have shown an increasing interest in applying it to many applications [19, 30, 61].

As a prominent privacy definition, the *differential privacy* technique has been incorporated in research on recommender systems. For example, McSherry et al. [62] was the first to introduce the *differential privacy* notion to CF. They added *Laplace* noise into the covariance matrix, and used the *non-private* recommender algorithm on this matrix to predict ratings. Another example is the work of Banerjee et al. [3].

They implemented a private local recommendation algorithm and analyzed the lower bounds of the sample complexity of their algorithm. However, both methods only focused on the untrusted recommender system. As a result, their recommendation algorithms only dealt with synthetic rating data, which induces extra noise and fails to consider the *KNN attack* mentioned by Calandrino et al. [12] In this chapter, we mainly study the trusted recommender system which deals with the original ratings dataset. But the algorithm we propose ensures the user cannot observe sensitive information from the recommendation output, and therefore, the proposed algorithm will be immunized from a *KNN attack*. Other work from Machanavajjhala formalized trade-offs between accuracy and privacy of graph-based social recommendations [58]. What they care about is the graph network instead of the traditional rating dataset, which also differs from ours.

3.2.3 Summary

When addressing the issues of privacy in recommender systems, the main issue is how to obtain a trade-off between the level of privacy and the performance of recommendations. Most traditional privacy preserving methods lack a rigid privacy guarantee while *differential privacy* suffers from unsatisfactory performance. In addition, both traditional and *differential privacy* methods fail to resist *KNN attacks*, which have been shown to be a serious privacy violation in CF [12].

Hence, in this chapter, we propose a *Private Neighbor Collaborative Filtering* algorithm, with the aim of providing comprehensive privacy for individuals, as well as maximizing the accuracy of recommendations. More specifically, we attempt to address the following issues;

- How to hide neighbors so they can resist *KNN attack*;
- How to decrease *sensitivity* in CF;
- How to design a *differential privacy* mechanism for recommendation purposes;

These are investigated in the following sections.

3.3 Private Neighbor Collaborative Filtering

In this section, we propose a *Private Neighbor Collaborative Filtering* (*PNCF*) algorithm to address the privacy preserving issue in *Neighborhood-based* CF methods. Firstly, we present an overview of the algorithm, followed by a detailed discussion. We then provide a theoretical analysis on how *PNCF* achieves the ϵ-*differential privacy* preserving purposes while retaining the utility for recommendation purposes.

3.3.1 The Private Neighbor Collaborative Filtering Algorithm

For the privacy preserving issue in the context of *neighborhood-based* CF methods, the preserving targets differ between *item-based* methods and *user-based* methods due to the different perspectives regarding definition of similarity. In *item-based* methods, an adversary can infer who the neighboring users are by observing any changes in the item similarity matrix. Therefore, the objective is to protect the users' identity. In *user-based* methods, what an adversary can infer from the user similarity matrix is the item rated by the active user. The preserving objective is then to hide the historically rated items. The proposed *PNCF* algorithm can deal with both cases. To make it clear, we will present the *PNCF* algorithm from the perspective of the *item-based* methods, and this can be applied to *user-based* methods in a straightforward manner.

For traditional *non-private item-based* CF methods, the prediction of the active user's score is generated by previous ratings of this user on similar items. The first stage aims to identify the items of k nearest neighbor, and the second stage aims to predict the rating by aggregating the ratings on those identified neighbor items. To resist a *KNN attack*, the neighbor information in both stages should be preserved. We propose the *PNCF* algorithm to address this problem, which includes two private operations:

Private Neighbor Selection prevents the adversary from inferring "who is the neighbor". Specifically, the *exponential mechanism* is adopted to perform private selection on the item similarity matrix to find k neighbors $N_k(t_i)$. The exponential mechanism ensures that for a particular item, deleting a user has little impact on the chosen probability. Therefore, the adversary is unlikely to figure out who their neighbors are by continuously observing recommendations, and is unlikely to infer the rating history of an active user by creating fake neighbors.

Perturbation prevents the adversary from inferring "*what is the rating*" of a particular user on an item. It perturbs neighbors similarity by adding a zero mean *Laplace* noise to mask "*what is the rating given by a certain neighbor*". Noisy output is utilized as the weight in making predictions.

Details for the first operations is presented in Section 3.3.2, and the second operation and theoretical analysis on privacy preserving and utility retaining is provided in

Section 3.4.

Algorithm 1 provides the pseudocode for how to use the proposed *PNCF* algorithm in a traditional *neighborhood-based* CF method. In the algorithm, Step 1 and 4 are standard recommendations steps. Specifically, Step 1 computes the similarity between items. It is not a step to guarantee the privacy but will still play a vital role, because the result will be employed as a utility *score* in the next step. Step 4 provides the prediction for $\hat{r}_{ai}$ according to Eq. 3.2.3. and considers it as an output of recommendations. However, Step 2 and 3 implement two operations in the proposed *PNCF* algorithm. Compared to the *non-private* algorithm, the cost of this will be the prediction accuracy. The accuracy cost is defined as the utility loss of the algorithm. We will provide both a theoretical analysis and an experimental comparative analysis on the utility in Section 3.4 and Section 3.5, respectively.

Although the standard DP mechanism, the exponential mechanism, can be applied for private selection, it can not be directly applied to CF because the naive exponential mechanism induces abundant noise that significantly influences the performance of the prediction. Here, the main challenge is to enhance performance by decreasing the noise magnitude. Consequently, two issues will be addressed in Step 2 and 3 accordingly: a) decrease in the *sensitivity*, and b) increased accuracy. Section 3.3.2 provides details on how the proposed *PNCF* algorithm can address these two issues. Moreover, both of these operations consume an equivalent $\epsilon/2$. According to the composition rule [63], the algorithm satisfies ϵ-*differential privacy.*

Algorithm 1 Private Neighbor Collaborative Filtering($PNCF$)

```
Require: D, privacy parameter $\epsilon$, truncated parameter $w$, number of neighbors $k$,
    $u_a$, $t_i$
Ensure: $\hat{r}_{ai}$
    1. Computing item to item similarity Matrix $S$;
    2. Private Neighbor Selection: Select $k$ neighbors $N_k(t_i)$ from $I$;
    3. Perturbation: Perturb the similarity in $N_k(t_i)$ by adding; $Lap(\frac{2k \cdot LS(i,\cdot)}{\epsilon})$ noise;
    4. Predict $\hat{r}_{ai}$;
```

3.3.2 The Private Neighbor Selection

Private Neighbor Selection aims to privately select k neighbors from a list of candidates for the privacy preserving purpose. This is unlike the k nearest neighbor method which sorts all candidates by their similarities and selects the top k similar candidates. *Private Neighbor Selection* adopts the *exponential* mechanism to arrange probabilities for every candidate. The probability is measured by a score function and its corresponding *sensitivity*. Specifically, we use the similarity as the score function and the *sensitivity* is measured accordingly. For an item t_i, the score function q is defined as follows:

$$q_i(I, t_j) = s(i, j), \tag{3.3.1}$$

where $s(i, j)$ is the output of the score functions representing the similarity between t_i with t_j, I is item t_i's candidate list for neighbors, and t_j is the selected neighbor. The probability of selecting t_j will be arranged according to the $q_i(I, t_j)$.

The exponential mechanism uses the score function q to preserve *differential privacy*. However, the naive exponential mechanism fails to provide accurate predictions because it is too general, and therefore not suitable for recommendation purposes. Accordingly, two operations are proposed to address this obstacle. The first operation is to define a new *Recommendation-Aware Sensitivity* to decrease the noise, and the second operation is to provide a new exponential mechanism to enhance the accuracy. Both are integrated to form the proposed *PNCF* algorithm, which consequently obtains a better trade-off between privacy and utility.

Recommendation-Aware Sensitivity

In this section, we propose *Recommendation-Aware Sensitivity* based on the notion of *Local Sensitivity* to reduce the magnitude of noise introduced for privacy-preserving purposes.

Recommendation-Aware Sensitivity for score function q, $RS(i,j)$ is measured by the maximal change in similarity of two items when removing a user's rating record. Let $s'(i,j)$ denote the $s(i,j)$ after deleting a user, then $RS(i,j)$ captures the maximal difference if we test all the users' ratings:

$$RS(i,j) = \max_{i,j \in I} ||s(i,j) - s'(i,j)||_1. \tag{3.3.2}$$

The result varies on the different item pairs.

Take the *item-based Cosin* similarity as an example to generate the value of *Recommendation-Aware Sensitivity.* PCC similarity can be measured in the same way. Items are considered as a vector in the n dimensional user space. For example, let $\mathbf{r_i}$ and $\mathbf{r_j}$ be a rating vector pair that contains all ratings given to t_i and t_j, respectively. Firstly, we observe all the ratings for a single user u_x and then analyse his/her impact on $s(i,j)$. There are four possible rating pairs on both item t_i and t_j:

$$(0,0), (0, r_{xj}), (r_{xi}, 0), \text{and } (r_{xi}, r_{xj})$$

Please note that in *neighborhood-based* CF methods, similarity is only measured on the co-rated set between two items. This means the first three rating pairs have no impact on the similarity function. Let $U_{ij} = \{u_x \in U | r_{xi} \neq \oslash, r_{xj} \neq \oslash\}$ be a set of users who rated both t_i and t_j, with these two item rating vectors then be represented as $\mathbf{r_{Ui}}$ and $\mathbf{r_{Uj}}$, respectively. The length of the vector, $||r_{Ui}||$, is determined by both the rating and the number of co-rated users. When deleting a user, the rating vector pair will be transferred to a new pair of $\mathbf{r'_{Ui}}$ and $\mathbf{r'_{Uj}}$ and the similarity will change to $s'(i,j)$ accordingly. The measurement and the smooth bound of *Recommendation-Aware Sensitivity* are summarized in Lemma 1 and 2, respectively.

Lemma 1. For any item pair t_i and t_j, the score function $q(I, t_j)$ has *Local Sensitivity*

$$RS(i,j) = \max\left\{ \max_{u_x \in U_{ij}} \left(\frac{r_{xi} \cdot r_{xj}}{||r'_i|| \cdot ||r'_j||} \right), \max_{u_x \in U_{ij}} \left(\frac{r_{xi} \cdot r_{xj} (||r_i|| ||r_j|| - ||r_i|| ||r_j||)}{||r_i|| \cdot ||r_j|| \cdot ||r'_i|| \cdot ||r'_j||} \right) \right\}, \tag{3.3.3}$$

where u_x is the user that makes a great impact on the t_i and t_j similarity, and $RS(i,j) = 1$ when $|U_{ij}| = 1$.

PROOF.

$$\begin{aligned} RS(i,j) &= \max||s(i,j) - s'(i,j)||_1 \quad (3.3.4) \\ &= \frac{r_i \cdot r_j}{||r_i|| \cdot ||r_j||} - \frac{r'_i \cdot r'_j}{||r'_i|| \cdot ||r'_j||} \\ &= \frac{||r'_i|| \cdot ||r'_j|| \cdot r_i \cdot r_j - ||r_i|| \cdot ||r_j|| \cdot r'_i \cdot r'_j}{||r_i|| \cdot ||r_j|| \cdot ||r'_i|| \cdot ||r'_j||}. \end{aligned}$$

Thus,

$$RS(i,j) \leq \begin{cases} \frac{||r_i|| \cdot ||r_i \cdot (r_i \cdot r_j - r'_i \cdot r'_j)||}{||r_i|| \cdot ||r_j|| \cdot ||r'_i|| \cdot ||r'_j||}, & \text{if } ||r'_i|| \cdot ||r'_j|| \cdot r_i \cdot r_j \geq ||r_i|| \cdot ||r_j|| \cdot r'_i \cdot r'_j, \\ \frac{r_i \cdot r_j (||r_i|| \cdot ||r_j|| - ||r'_i|| \cdot ||r'_j||)}{||r_i|| \cdot ||r_j|| \cdot ||r'_i|| \cdot ||r'_j||}, & \textit{otherwise}. \end{cases} \tag{3.3.5}$$

Please note $\frac{||r_i||\cdot||r_i\cdot(r_i\cdot r_j-r'_i\cdot r'_j)||}{||r_i||\cdot||r_j||\cdot||r'_i||\cdot||r'_j||}=\max_{x\in U_{ij}}\frac{r_{xi}\cdot r_{xj}}{||r'_i||\cdot||r'_j||}$.

Thus, $sim(i,j)$ and $sim'(i,j)$ differ at most by

$$\max\left\{\max_{u_x\in U_{ij}}\left(\frac{r_{xi}\cdot r_{xj}}{||r'_i||\cdot||r'_j||}\right),\max_{u_x\in U_{ij}}\left(\frac{r_{xi}\cdot r_{xj}(||r_i||||r_j||-||r_i||||r_j||)}{||r_i||\cdot||r_j||\cdot||r'_i||\cdot||r'_j||}\right)\right\}, \tag{3.3.6}$$

which is largely determined by the length of rating vector pairs. □

Lemma 2. *Recommendation-Aware Sensitivity* with smooth bound is

$$B(RS(i,j))=exp(-\beta)\cdot RS(i,j). \tag{3.3.7}$$

Both *Lemma* 1 and *Lemma* 2 indicate that depending on the length of rating vector pairs, score function q will only change slightly in a normal case. Compared to *Global Sensitivity*, *Recommendation-Aware Sensitivity* can significantly decrease the noise magnitude. We use *Recommendation-Aware Sensitivity* in our *PNCF* algorithm. To simplify the notation, we will use $RS(i,j)$ or *Recommendation-Aware Sensitivity* to represent $B(RS(i,j))$ in the following formulas.

Private Neighbor Selection Implementation

Private Neighbor Selection is the major private operation in *PNCF*. In this section, we present the *Private Neighbor Selection* based on the exponential mechanism and show how it is introduced into CF. Given an active user u_a and a target item t_i, we have a candidate item list I and a corresponding similarity list $\mathbf{s(i)}=<s(i,1),...,s(i,m)>$, which consists of similarities between t_i and other $m-1$ items. According to Eq. 4.3.11, each element in $\mathbf{s(i)}$ is the score of the corresponding item. The goal of *Private Neighbor Selection* is to select k neighbors in candidate item list I, according to the score vector $\mathbf{s(i)}$. It should be noted that we will not select t_i in I as it is the target item.

However, even though we apply *Recommendation-Aware Sensitivity* to reduce noise, the naive exponential mechanism still yields low prediction accuracy. The reason is that performance of *neighborhood-based* CF methods is largely dependent on the quality of neighbors. If we assume the top k nearest neighbor as the highest quality neighbor, and we use s_k to denote the similarity to the k-th neighbor, the randomized selection process will have a high probability of picking up neighbors with low scores. The rating pattern of these low quality neighbors may be totally different from the pattern of the active user, which lowers the accuracy of the prediction. To address this problem, the best solution is to improve the quality of k neighbors under *differential privacy* constraints.

Motivated by this, we provide a new notion of *truncated similarity* as the score function to enhance the quality of selected neighbors. The truncated notion was first mentioned in Bhaskar et. al.'s work [7], in which they used truncated frequency to decrease the computational complexity. The same notion can be used in our score function q to find those high quality neighbors. Specifically, for each item in the candidate list I, if its similarity $s(i,j)$ is smaller than $s_k(i,\cdot) - w$, then $s(i,j)$ is truncated to $s_k(i,\cdot) - w$, where w is a truncated parameter in the score function; otherwise it is still preserved as $s(i,j)$. The truncated similarity can be denoted as

$$\hat{s}(i,j) = \max\left(s(i,j), s_k(i,\cdot) - w\right), \tag{3.3.8}$$

where truncated parameter w will be analyzed in the next section.

The *truncated similarity* ensures no item in $N_k(t_i)$ has a similarity less than $(s_k(i,\cdot)-w)$ and every item whose similarity is greater than $(s_k(i,\cdot)+w)$ is selected to the $N_k(t_i)$. Compared to the naive exponential mechanism in which some items with a similarity lower than $(s_k(i,\cdot) - w)$ may have a higher probability of being selected,

Algorithm 2 Private Neighbor Selection

```
Require: $\epsilon$, $k$, $w$, $t_i$, $I$ ,$\mathbf{s}(i)$
Ensure: $N_k(t_i)$
1: Sort the vector $\mathbf{s}(i)$;
2: $C_1 = [t_j | s(i,j) \geq s_k(i,j) - w, t_j \in I]$,
   $C_0 = [t_j | s(i,j) < s_k(i,j) - w, t_j \in I]$,
3: for N=1:k do
4:     for each item $t_j$ in $\mathbf{t}_i$ do
5:         Allocate probability as:
           $$\frac{\exp\left(\frac{\epsilon \cdot \hat{s}(i,j)}{4k \cdot RS(i,j)}\right)}{\sum_{j \in C_1} \exp\left(\frac{\epsilon \cdot \hat{s}(i,j)}{4k \cdot RS(i,j)}\right) + |C_0| \cdot \exp\left(\frac{\epsilon \cdot \hat{s}(i,j)}{4k \cdot RS(i,j)}\right)}.$$
6:     end for
7:     Sample an elements $t$ from $C_1$ and $C_0$ without replacement according to their
       probability;
8:     $N_k(t_i) = N_k(t_i) + t$;
9: end for
```

the *truncated similarity* can significantly improve the quality of selected neighbors. Based on the *Recommendation-Aware Sensitivity* and the *truncated similarity*, we present the *Private Neighbor Selection* operation as shown in Algorithm 2.

We divided item candidate list I into two sets: C_1 and C_0. Set C_1 consists of items whose similarities are larger than the truncated similarity $(s_k(i,\cdot)-w)$, and C_0 consists of the remaining items in I. In the *Private Neighbor Selection* operation, items in C_1 follow exponential distribution according to their similarities. Each of these have the probability proportion to $\exp\left(\frac{\epsilon \cdot s(i,j)}{4k \cdot RS(i,j)}\right)$. In C_0, we do not deal with the elements one by one. Instead, we consider C_0 as a single candidate with a probability proportion to $\exp\left(\frac{\epsilon \cdot \hat{s}(i,j)}{4k \cdot RS(i,j)}\right)$. When C_0 is chosen, the items in C_0 are selected uniformly. This does not violate *differential privacy* because we use C_0 as a single candidate and assign the weight according to the exponential mechanism. This means the probability for each element in C_0 still follows exponential distribution. The probability of providing the same output for neighboring datasets is still bounded by $exp(\epsilon)$.

The *Private Neighbor Selection* operation can provide high quality neighbors and guarantee the *differential privacy* simultaneously. The choice of w will influence the utility in a fixed privacy level. We will determine the value of w in our utility analysis in Section 3.4 and show its effectiveness in Section 3.5.

3.4 Privacy and Utility Analysis

In this section, we present a theoretical analysis for the privacy and the utility of the proposed *PNCF* algorithm.

3.4.1 Privacy Analysis

The proposed *PNCF* algorithm contains two private operations: the *Private Neighbor Selection* and the *Perturbation*. In this section, we analyse the privacy guarantee of each step.

The *Private Neighbor Selection* is essentially processing the exponential mechanism successively. An item is selected without replacement in each round until k distinct neighbors are chosen. The score *sensitivity* is calibrated by *Recommendation-Aware Sensitivity*.

From the definition of the *Exponential* mechanism, each selection round preserves $(\frac{\epsilon}{2k})$-differential privacy. The *sequential composition* in Definition 2 undertakes the privacy guarantee for a sequence of differentially private computations. When a series of private analysis is performed sequentially on a dataset, the *privacy budget* ϵ will be added for each step. According to the *sequential composition* definition, *Private Neighbor Selection* guarantees $\frac{\epsilon}{2}$-differential privacy as a whole.

In the *Perturbation* step, we add independent *Laplace* noise to the $N_k(t_i)$ chosen in the previous step. Given an item set $N_k(t_i)$, *perturbation* adds independent *Laplace* noise to their similarities. The noise is calibrated by $\epsilon/2$ and the *Recommendation-Aware Sensitivity* is:

$$s_{noise}(i,j) = s(i,j) + Lap\left(\frac{2 \cdot RS(i,j)}{\epsilon}\right). \tag{3.4.1}$$

According to the definition of the *Laplace* mechanism, this step satisfies $\epsilon/2$-differential privacy.

Consequently, when combining both operations, the proposed method preserves ϵ-differential privacy by applying *composition lemma* on the selection and perturbation step together.

3.4.2 Utility Analysis

The utility of *PNCF* is measured by the accuracy of the predictions. Since we focus on the *neighborhood-based* CF, the prediction accuracy depends highly on the quality of the neighbors. Given an input dataset R, we set the *non-private* neighborhood-based CF method as a baseline. By comparing the selected neighbors in *PNCF* with the corresponding neighbors in the baseline method, we can analyse the utility level of the proposed algorithms.

To predict the rating of t_i for the active user u_a, the *non-private* algorithm typically chooses the top k similar neighbors and then generates an integrated output as the prediction. Therefore, the closeness between top k similar neighbors in the baseline method and the privately selected neighbors in *PNCF* is the key factor that determines the utility level. When implementing the exponential mechanism, the randomized selection step will choose low similarity neighbors with high probability, which significantly lowers the accuracy of recommendations.

The proposed *PNCF* algorithm can preserve the quality of the neighbors in two aspects: every neighbor with a true similarity greater than $(s_k + w)$ will be selected, and no neighbor in the output has a true similarity less than $(s_k - w)$, where the true similarity refers to the original similarity without perturbation or truncation. Therefore, *Private Neighbor Selection* guarantees that with high probability, the k selected neighbors will be close to the actual top k nearest ones. Hence, the utility in *PNCF* will be better retained than in the naive exponential mechanism. We present two theorems and proofs to support the claims as follows.

Theorem 3.4.1. *Given an item t_i, let $N_k(t_i)$ be the k selected neighbors and $|v|$ be the maximal length of all the rating vector pairs. We also suppose RS as the maximal Recommendation-Aware Sensitivity between t_s and other items respectively, and suppose ρ is a small constant less than 1. Then, for all $\rho > 0$, with probability at least $1 - \rho$, the similarity of all items in $N_k(t_i)$ is larger than $s_k - w$, where $w = \min(s_k, \frac{4k \cdot RS}{\epsilon} \ln \frac{k \cdot (|v| - k)}{\rho})$.*

Proof. First, we compute the probability of selecting a neighbor with a similarity less than $(s_k - w)$ in each round of sampling. This occurs when there is an unsampled neighbor with similarity no less than s_k. Then, we prove that with the constraint of w and ρ, no neighbor in the output has true similarity less than $(s_k - w)$ after k rounds sampling for neighbor selection.

If a neighbor with similarity $(s_k - w)$ is still waiting for selection, the probability of picking a neighbor with similarity less than $(s_k - w)$ is $\leq \frac{\exp(\frac{\epsilon(s_k - w)}{4k \cdot RS})}{\exp(\frac{\epsilon s_k}{4k \cdot RS})} = \exp(-\frac{\epsilon \cdot w}{4k \cdot RS})$. Since there are at most $|v|$ neighbors with similarity less than $(s_k - w)$, according to the union bound, the probability of choosing a neighbor with similarity less than $(s_k - w)$ is at most $(|v| - k) \cdot \exp(-\frac{\epsilon \cdot w}{4k \cdot RS})$.

Furthermore, by the union bound in the sampling step, the probability of choosing any neighbor with similarity less than $(s_k - w)$ is at most $k \cdot (|v| - k) \cdot \exp(-\frac{\epsilon \cdot w}{4k})$.

Let $\rho \geq k \cdot (|v| - k) \cdot exp(-\frac{\epsilon \cdot w}{4k \cdot RS})$.

Then,

$$\begin{aligned} & -\frac{w \cdot \epsilon}{4k \cdot RS} \leq \ln(\frac{\rho}{k \cdot (|v| - k)}) \\ \Rightarrow \quad & \frac{w \cdot \epsilon}{4k \cdot RS} \geq \ln \frac{k \cdot (|v| - k)}{\rho} \\ \Rightarrow \quad & w \geq \frac{4k \cdot RS}{\epsilon} \ln \frac{k \cdot (|v| - k)}{\rho}. \end{aligned} \tag{3.4.2}$$

Thus, the probability that similarities less than $(s_k - w)$ will be chosen is less than ρ. As defined in Section 3.3.2, *Recommendation-Aware Sensitivity* is $O(\frac{1}{||v||^2})$. So for constant ρ, $s = O(\frac{k \cdot \ln(|v| - k)}{\epsilon ||v||^2})$ is sufficient. In practise, we have to ensure $s_k - w \geq 0$ in COS similarity or $s_k - w > -1$ in *PCC*, so $w = \min(s_k, \frac{4k \cdot RS}{\epsilon} \ln \frac{k \cdot (|v| - k)}{\rho})$. □

Theorem 3.4.2. *Given an item t_i, for all $\rho > 0$, with probability at least $1 - \rho$, the similarities of all neighbors $> s_k + w$ are present in $N_k(t_i)$, where $w = \min(s_k, \frac{4k \cdot RS}{\epsilon} \ln \frac{k \cdot (|v| - k)}{\rho})$.*

PROOF. Similar to Theorem 3.4.1, we first compute the probability of picking a neighbor with a similarity less than s_k in each round of sampling when an unsampled neighbor with similarity greater or equal than $(s_k + w)$ is not present in $N_k(t_i)$. Then we prove that with the constraint of w and ρ, all neighbors with similarity $\geq s_k + w$ have been chosen in $N_k(t_i)$.

Suppose a neighbor with a similarity greater than (s_k+w) has not been selected in $N_k(t_i)$, the conditional probability of picking any neighbor with similarity less than s_k is $\leq \frac{\exp(\frac{\epsilon(s_k)}{4k \cdot RS})}{\exp(\frac{\epsilon(s_k+w)}{4k \cdot RS})} = \exp(-\frac{\epsilon \cdot w}{4k})$. Therefore, the probability of not selecting any neighbor with similarity less than s_k in any of the k rounds of sampling is:

$$\left(1 - (|v|-k) \cdot \exp(\frac{\epsilon(s_k + w)}{4k \cdot RS})\right)^k \geq \left(1 - k \cdot (|v|-k) \cdot \exp(\frac{\epsilon(s_k + w)}{4k \cdot RS})\right). \qquad (3.4.3)$$

Let $1-\rho \leq (1-k \cdot (|v|-k) \cdot \exp(\frac{\epsilon(s_k+w)}{4k \cdot RS}))$. Thus, is similar to the proof of Theorem 3.4.1. When

$$w = \min\left(s_k, \frac{4k \cdot RS}{\epsilon} \ln \frac{k \cdot (|v|-k)}{\rho}\right), \qquad (3.4.4)$$

all neighbors with similarity $\leq s_k + w$ are present in $N_k(t_i)$. $\square$

3.5 Experiment and Analysis

In this section, we provide discussion concerning the experiments we conducted to evaluate the performance of the proposed *PNCF* algorithm and evaluate the performance of the proposed *PNCF* algorithm on the research issues we identified in this study:

- *How does PNCF perform on the privacy preserving issue from the perspective of neighbors?* As neighbors can be defined among users or items based on various similarity metrics, we evaluate the performance of *PNCF* in both a *user-based* and an *item-based* manner with two popular similarity metrics, PCC and COS.

We compare PCC and COS with traditional *neighborhood-based* CF methods. Moreover, to obtain a thorough comparison, we conducted the experiments on two benchmark datasets, *MovieLens* and *Netflix*, which will be described in Section 4.5.1.

- *How does the proposed Recommendation-Aware Sensitivity satisfy the recommendation purpose?* To answer this question, we compare *Recommendation-Aware Sensitivity* with the traditional *Global Sensitivity* in terms of recommendation accuracy. Specifically, we compare them in the standard naive exponential mechanism from both a *user-based* manner and an *item-based* manner on two popular similarity metrics, PCC and COS.

- *How does PNCF perform compared to other-related methods?*

  To retain the quality of the selected neighbors, *PNCF* redesigns the exponential mechanism for CF. Therefore, we compare *PNCF* with the naive mechanism (NM) as it is highly-related to the redesigned mechanism in *PNCF*. Specifically, in order to achieve a fair comparison, this set of experiments is deployed with the same *sensitivity*, but with a different neighbor selection mechanism.

- *How does the privacy budget affect the performance of PNCF?* In the context of privacy preserving, the privacy budget is a key parameter that controls the privacy level of the privacy preserving algorithms. We then examine the trade-off between the utility and the privacy of *PNCF* on two benchmark datasets by varying it in a wide range.

3.5.1 Datasets and Measurements

The datasets we experimented with are the popular `Netflix` dataset [1] and the `MovieLens` dataset. [2] The `Netflix` dataset was extracted from the *Netflix* Prize dataset, where each user rated at least 20 movies, and each movie was rated by $20 - 250$ users. The `MovieLens` dataset includes around 1 million ratings collected from $6,040$ users about $3,900$ movies. `MovieLens` is the standard benchmark data for collaborative filtering research, while the `Netflix` dataset is a real industrial dataset released by Netflix. Both datasets contain millions of ratings that last for several years and are sufficient for investigating the performance of the proposed method from both a research and industry perspective. Specifically, we apply the *All-But-One* strategy, which randomly selects one rating of each user, and then, predicts its value using all the left ratings in the dataset.

In this chapter, to measure the quality of recommendations, we apply a popular measurement metric, *Mean Absolute Error* (MAE) [1]:

$$MAE = \frac{1}{|T|} \sum_{a,i \in T} |r_{ai} - \hat{r}_{ai}|, \tag{3.5.1}$$

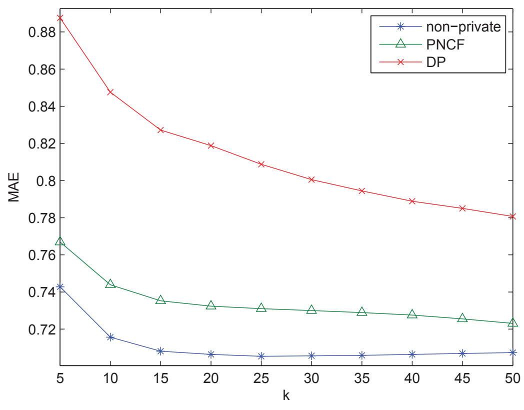
where r_{ai} is the true rating of user u_a on item t_i, and $\hat{r}_{ai}$ is the value of predicting the rating. T denotes the test dataset, and $|T|$ represents its size. A lower MAE means a higher prediction accuracy. Note that in each experiment, we consider the traditional *non-private* CF method as a baseline.

[1]http://www.netflixprize.com
[2]http://www.grouplens.org

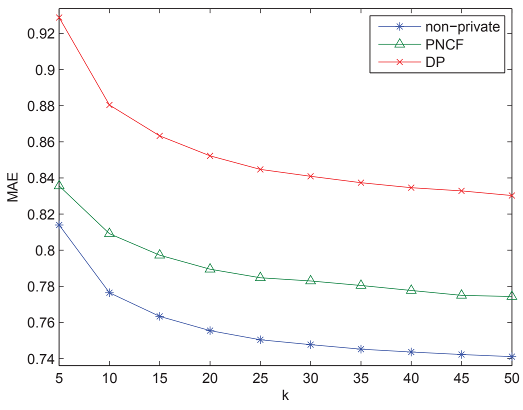
3.5.2 Performance of *PNCF*

In this section, we examine the performance of *PNCF* from the perspective of privacy preserving to neighbors. To obtain a thorough examination, we conduct experiments by defining neighbors from both the user and the item perspective on two popular measurement metrics, PCC and COS. Specifically, we apply the traditional *neighborhood-based* CF as the *non-private* baseline, and then compare the *PNCF* with the standard DP method in terms of the recommendation accuracy, as both quantify the privacy risk to individuals. Parameter k represent the number of the neighbors. Moreover, the truncated parameter w is set according to Lemma 3.4.1. The privacy budget ϵ is fixed to 1 to ensure the *PNCF* algorithm satisfies the *1-differential privacy*.

Table 3.1 shows the results on the *Netflix* data set. From this table, it is clear that *PNCF* significantly outperforms DP in all configurations. Specifically, in the *item-based* manner with the PCC metric, when $k = 40$, *PNCF* achieves a MAE of 0.7178. This outperforms DP by 13.60%. When $k = 10$, *PNCF* obtains a MAE of 0.7533, which outperforms DP by 13.07%. In the *user-based* manner with the PCC metric, when $k = 30$, *PNCF* outperforms DP by 8.71% in MAE. Similar trends are also observed when measuring neighbor similarity in COS. This indicates that *PNCF* performs better in terms of the recommendation accuracy than the standard *differential privacy*(DP) method that uses *Global Sensitivity* and the naive exponential mechanism. On the other hand, compared with the baseline *non-private* algorithm, the accuracy cost introduced by *PNCF* is much smaller than the cost introduced by DP. This is because *PNCF* introduces two novel operations to reduce the magnitude of introduced noise. Moreover, the two-tailed, paired t-test with a 95% confidence level has been applied to evaluate the performance of *PNCF* under all configurations.



(a) PCC-item



(b) PCC-user

Figure 3.1: Performance of *PNCF* and DP on the *MovieLens* Dataset

The detailed statistical results on the *Netflix* dataset are presented in Table 3.2. This table shows that the difference in performance between *PNCF* and DP is statistically significant.

Moreover, Fig. 3.1 shows the results on the *MovieLens* dataset. Specifically, Fig. 3.1a and 3.1b show the performance of *PNCF* and DP with the PCC metric under an *item-based* and *user-based* manner, respectively. It is clear that *PNCF* performs much better then DP. In addition, we also observe that as k increases, both *PNCF* and DP achieve better MAE. However, *PNCF* always performs better than DP across all k values. This is because *PNCF* achieves privacy preserving by distinguishing the quality of potential neighbors, and therefore always selects good-quality neighbors as analysed in Section 3.4.2. Moreover, *PNCF*'s MAE performance is very close to that of the *non-private* baseline. This indicates *PNCF* can retain the accuracy of recommendation while providing comprehensive privacy for individuals.

Table 3.1: Overall Performance Comparison on *Netflix*

		PCC			COS		
	k	*Non-private*	*PNCF*	DP	*Non-private*	*PNCF*	DP
Item-based	5	0.7504	0.7835	0.9153	0.7524	0.7893	0.8786
	10	0.7210	0.7533	0.8666	0.7240	0.7637	0.8301
	15	0.7121	0.7407	0.8499	0.7159	0.7486	0.8134
	20	0.7083	0.7343	0.8371	0.7137	0.7414	0.8012
	25	0.7070	0.7278	0.8329	0.7133	0.7401	0.7972
	30	0.7068	0.7244	0.8287	0.7137	0.7390	0.7927
	35	0.7072	0.7208	0.8244	0.7152	0.7385	0.7899
	40	0.7078	0.7178	0.8154	0.7169	0.7398	0.7835
	45	0.7086	0.7163	0.8062	0.7185	0.7392	0.7789
	50	0.7092	0.7146	0.8019	0.7199	0.7395	0.7770
User-based	5	0.7934	0.8025	0.8962	0.8041	0.8009	0.8661
	10	0.7641	0.7691	0.8538	0.7691	0.7708	0.8237
	15	0.7509	0.7551	0.8324	0.7553	0.7612	0.8042
	20	0.7428	0.7485	0.8229	0.7481	0.7564	0.7956
	25	0.7375	0.7435	0.8137	0.7434	0.7525	0.7867
	30	0.7339	0.7408	0.8115	0.7398	0.7512	0.7847
	35	0.7316	0.7398	0.8081	0.7371	0.7494	0.7812
	40	0.7298	0.7381	0.8059	0.7350	0.7480	0.7794
	45	0.7284	0.7368	0.8028	0.7331	0.7469	0.7764
	50	0.7273	0.7353	0.8015	0.7317	0.7456	0.7751

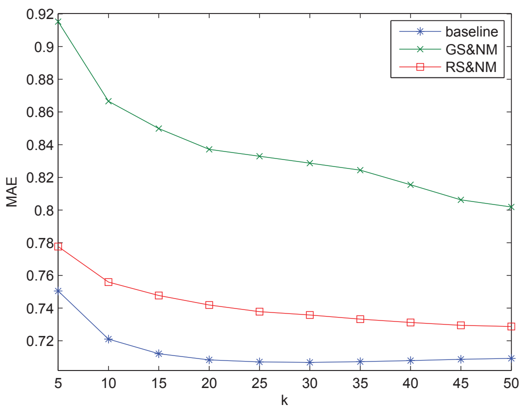
Table 3.2: Paired-t-test for *PNCF* vs. DP on *Netflix*

		df	*t*	*p*-value
Item-based	PCC	9	26.5135	< 0.0001
	COS	9	11.6696	< 0.0001
User-based	PCC	9	25.8884	< 0.0001
	COS	9	10.5045	< 0.0001

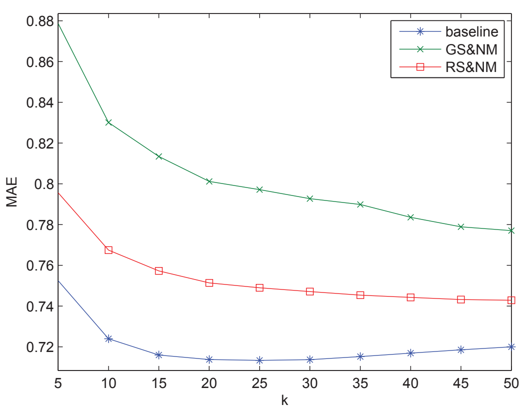
3.5.3 Performance of *Recommendation-Aware Sensitivity*

In this section, we examine the performance of *Recommendation-Aware Sensitivity* (RS) on recommendations in terms of MAE. Specifically, we compare RS with traditional *Global Sensitivity* (GS) on two well-known similarity metrics (PCC and COS) from both the *user-based* manner and the *item-based* manner. In order to show the impact of *Recommendation-Aware Sensitivity* on the private part of the algorithm, we select the traditional Naive exponential Mechanism (NM) and set privacy budget ϵ to 1. Therefore, in this configuration, we refer to the *Recommendation-Aware Sensitivity*-based algorithm and the *Global Sensitivity*-based algorithm as RS&NM and GS&NM, respectively. The results for *Netflix* and *MovieLens* are shown in Fig. 3.2 and 3.3, respectively.

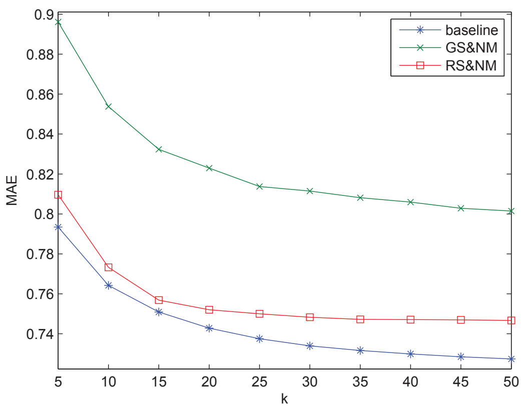
Fig. 3.2 shows the MAE results on the *Netflix* dataset. It was observed that the *Recommendation-Aware Sensitivity* significantly outperforms the traditional *Global Sensitivity* on both PCC and COS metrics under both the *user-based* and *item-based* manner. Specifically, as shown in Fig. 3.2a, when $k = 40$ in the *item-based* manner with the PCC metric, *Recommendation-Aware Sensitivity* achieves a MAE



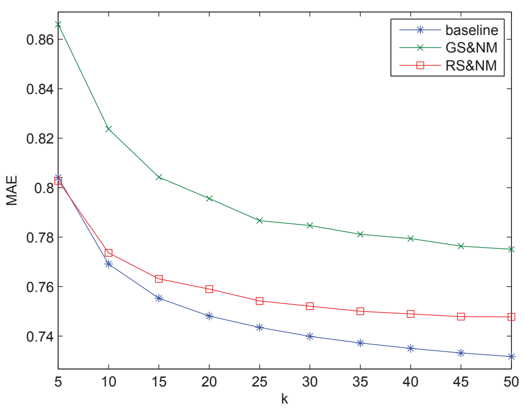
(a) PCC-item



(b) COS-item



(c) PCC-user



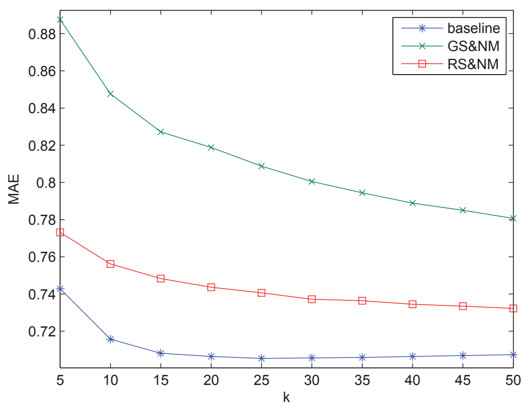
(d) COS-user

Figure 3.2: *Global Sensitivity* vs. *Recommendation-Aware Sensitivity* on *Netflix*

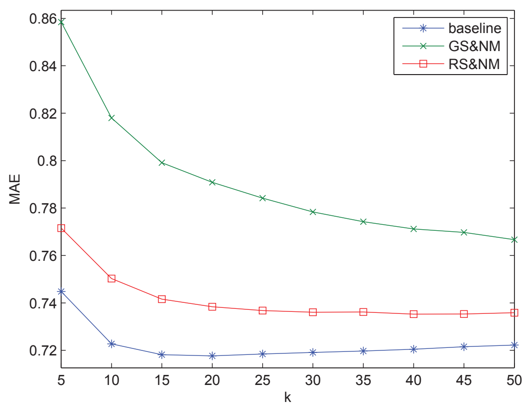
at 0.7312, outperforming the result of 0.8154 from *Global Sensitivity* by 10.33%. Moreover, it is clear the performance of *Recommendation-Aware Sensitivity* is very close to the *non-private* baseline, which focuses on accuracy but disregards the issue of privacy. This indicates that *Recommendation-Aware Sensitivity* can achieve the privacy preserving objective while retaining a high accuracy of recommendations. Similar trends can be observed on COS and *user-based* as shown in Fig. 3.2b, 3.2c, and 3.2d, respectively. This indicates *Recommendation-Aware Sensitivity* is independent of the applied similarity metrics, and works from both a user and item perspective in the context of *neighborhood-based* CF methods.

Fig. 3.3 shows the results on the *MovieLens* dataset. We observed that *Recommendation-Aware Sensitivity* also significantly outperformed the traditional *Global Sensitivity* in all configurations, thus confirming the effectiveness of *Recommendation-Aware Sensitivity* for recommendation purposes. Specifically,when $k = 40$ in an *item-based* manner with the PCC metric, *Recommendation-Aware Sensitivity* achieves a MAE of 0.7345. This outperforms the result of 0.7888 from *Global Sensitivity* by 6.88%.

On the other side, we also check the impact of k on the performance of compared algorithms. As shown in Fig. 3.2 and 3.3, we observed *Global Sensitivity* achieving better performance as k increased. However, at the same time, *Recommendation-Aware Sensitivity* can always obtain a better MAE than *Global Sensitivity* across all k values. Moreover, *Recommendation-Aware Sensitivity* also performs closely to the *non-private* baseline. This confirms Lemma 1: *Recommendation-Aware Sensitivity* is only related to the value of rating and the number of co-rated items/users. This also demonstrates that *Recommendation-Aware Sensitivity* is insensitive to k and is independent to the applied similarity metric.



(a) PCC-item



(b) COS-item

Figure 3.3: *Global Sensitivity* vs. *Recommendation-Aware Sensitivity* on *Movielens*

Table 3.3: Paired-t-test for RS&NM vs. GS&NM on *Netflix*

		df	*t*	*p*-value
Item-based	PCC	9	22.7078	< 0.0001
	COS	9	11.1771	< 0.0001
User-based	PCC	9	17.0451	< 0.0001
	COS	9	10.9179	< 0.0001

To show the statistical effectiveness of *Recommendation-aware Sensitivity*, we applied a paired t test (with a 95% confidence level) to examine the difference in performances of *Recommendation-Aware Sensitivity* and *Global Sensitivity.* The statistics for results on *Netflix* are shown in Table 3.3, and it was observed that differences between them in terms of MAE are statistically significant.

3.5.4 Performance of the ***Private Neighbor Selection*** Mechanism

To preserve the privacy from the perspective of neighbors, the mechanism for neighbor selection will be the determinant factor. In this chapter, we introduce the *Private Neighbor Selection* mechanism, and thoroughly examined it by comparing it with the state-of-the-art Naive Exponential Mechanism (NM). Specifically, to clearly demonstrate the effectiveness of the *Private Neighbor Selection* mechanism, we apply *Recommendation-Aware Sensitivity* (RS) to both *PNCF* and NM (this is termed RS&NM). Once again, we conducted the experiment on both a user-based and item-based manner with two popular metrics, PCC and COS, on two benchmark datasets, *Netflix* and *MovieLens*.

Fig. 3.4 shows the results on the *Netflix* dataset. It is clear that *PNCF* outperformed RS&NM in all configurations. Specifically, in an *item-based* manner with the PCC metric, and when $k = 40$ as shown in Fig. 3.4a, *PNCF* achieved a MAE of 0.7178 which was significantly better than 0.7312 from RS&NM. Moreover, the results from the *MovieLens* dataset are shown in Fig. 3.5. It was also observed that *PNCF* performs better than RS&NM. This is due to the *Private Neighbor Selection* using the truncated similarity to enhance the quality of selected neighbors as analysed in

Section 3.4.2.

The results on both datasets demonstrate that the proposed *Private Neighbor Selection* mechanism in *PNCF* algorithm is superior to the previous naive exponential mechanism, and is also independent of the applied similarity metrics.

3.5.5 Effect of Privacy Budget

In the context of *differential privacy*, privacy budget ϵ serves as a key parameter to determine the privacy preserving level. A smaller ϵ results in a higher privacy level, which makes the algorithm less vulnerable from *KNN attack*. According to the literature [22], $\epsilon = 1$ or less would be suitable for privacy preserving purposes, and we follow this in our experiments. However, to achieve a comprehensive examination of *PNCF*, we evaluated its performance under various privacy preserving levels by varying ϵ from 0.1 to 1 with a 0.1 step on two benchmark datasets from both the *user-based* and *item-based* manner. Due to space limitation, we report the results on PCC metric when $k = 35$, as shown in Fig. 3.6. It was observed that as ϵ increases, the MAE performance improves, which means the lower the privacy preserving level, the larger the utility level. Similar trends were also observed when k takes other values. Moreover, we observe that MAE decreases much faster when ϵ decreases from 0.1 to 0.4, compared to when ϵ decreases from 0.4 to 1. This indicates that a big utility cost is needed when preserving a higher privacy level ($\epsilon = 0.1$). On the other hand, we also observe that, for the *PNCF* algorithm, its performance is stable when $\epsilon \geq 0.7$. This confirms that *PNCF* is capable of retaining the utility for the recommendation purpose, while satisfying the privacy preserving purposes as analysed in Section 3.4.2.

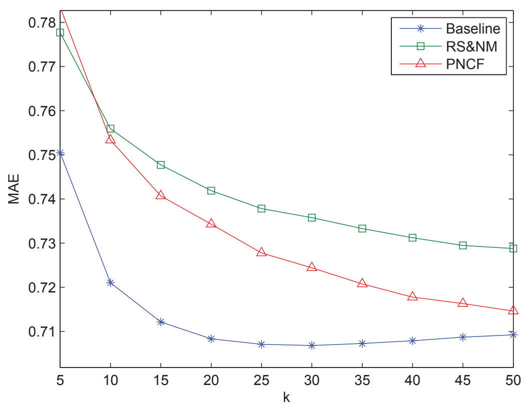
3.6 Summary

This chapter proposes an effective privacy preserving method for *neighborhood-based* collaborative filtering and makes the following contributions:

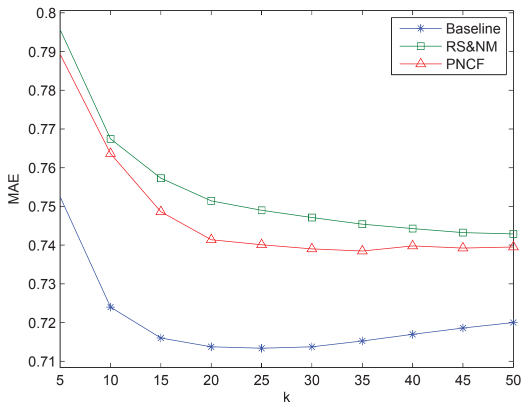
- A novel *Recommendation-Aware Sensitivity* is proposed to reduce the large magnitude of noise. We apply the notion of *local sensitivity* to formulate the *Recommendation-Aware Sensitivity*, which is used for recommendation purposes.
- We propose *Private Neighbor Selection* to protect neighbors. Private selection aims to resist a *KNN attack* by preventing an attacker from inferring a user's rating history from the predictions. In addition, a successive perturbation operation is applied to hide the rating of a particular neighbor.
- A re-designed *exponential mechanism* is proposed to enhance the performance of *neighborhood-based* CF methods. This mechanism can select neighbors with a higher quality than the naive exponential mechanism, and we also provide theoretical analysis and experimental results to demonstrate its effectiveness.

These three contributions provide a practical way to apply a rigid privacy notion to collaborative filtering without much accuracy loss. The experimental results also show the robustness and effectiveness of the proposed *PNCF* algorithm in various similarity metrics. Most notably, the proposed algorithm can be extended to other scenarios that need to perform top k private selection from candidates, such as top k queries. It has been proven that our algorithm can guarantee a better quality of selected neighbors than the naive exponential mechanism.

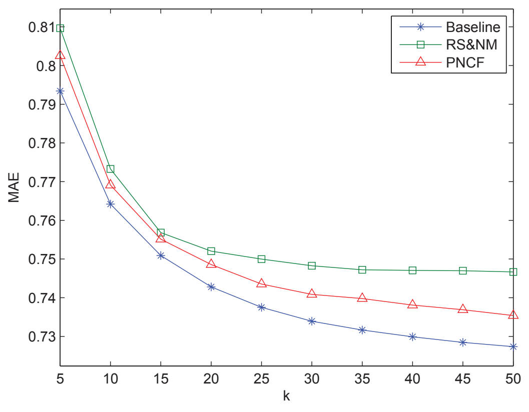
This chapter concentrates on the differential privacy application on the *neighborhood-based* CF. Other recommendation techniques, such as the tagging recommender system, still suffer from violations of privacy. Our next chapter will explore the privacy issue in a tagging recommender system and meeting the challenge of shrinking the randomization domain.



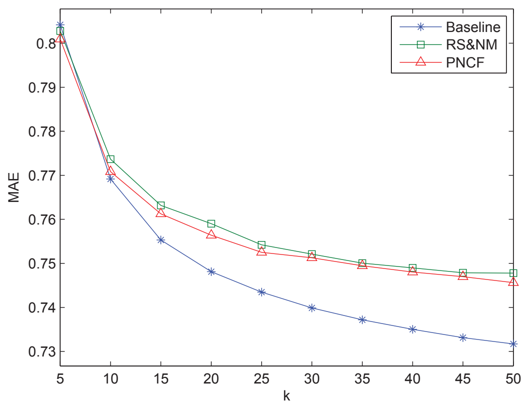
(a) PCC-item



(b) COS-item

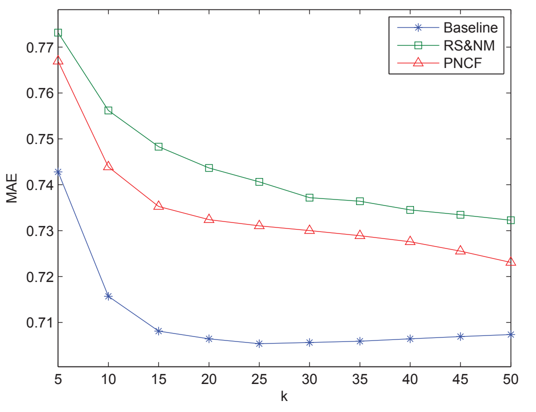


(c) PCC-user

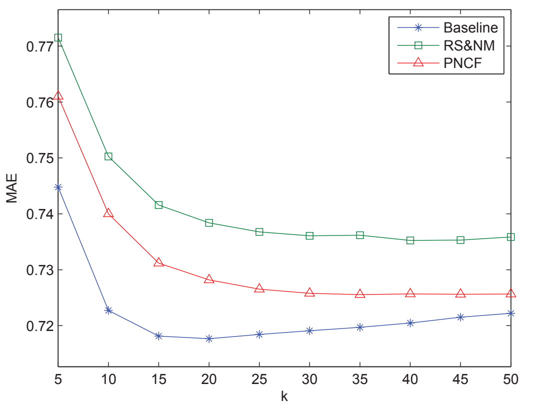


(d) COS-user

Figure 3.4: Comparison between the *Exponential Mechanism* and *PNCF* on *Netflix*

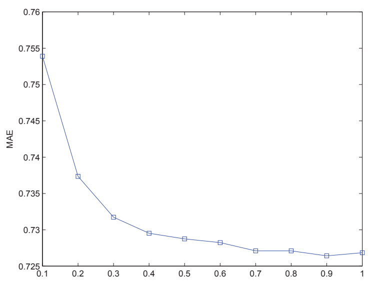


(a) PCC-item

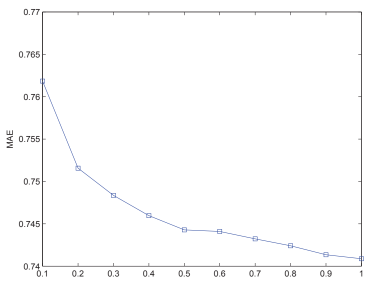


(b) COS-item

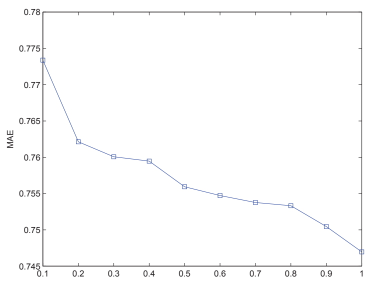
Figure 3.5: Comparison between *Exponential Mechanism* and *PNCF* on *MovieLens*



(a) PCC-item on *Netflix*



(b) PCC-user on *Netflix*



(c) PCC-user on *MovieLens*

Figure 3.6: Impact of ϵ on *PNCF* when $k = 35$

Chapter 4

Deferentially Private Tagging Recommender System

4.1 Introduction

This chapter focus on the randomization domain shrinking problem in the context of a *tagging recommender system*. The widespread success of social network web sites, such as *Del.icio.us* and *Bibsonomy*, introduces a new concept called the *tagging recommender system* [38]. These social network web sites usually allow users to annotate resources with customized tags, which in turn facilitates the recommendation of resources. Over the last few years, a large collection of data has been generated, but the issue of privacy in the recommender process has generally been overlooked [71]. An adversary with background information may re-identify a particular user in a tagging dataset and obtain the user's historical tagging records [29]. Moreover, in comparison with traditional recommender systems, tagging recommender systems involve more semantic information that directly discloses users' preferences. Hence, the privacy violation involved is more serious than traditional violations [71]. Consequently, how to preserve privacy in tagging recommender systems is an emerging issue that needs to be addressed.

Privacy preserving approaches on traditional recommender systems can hardly be applied to tagging recommender systems due to the semantic property of tags. Specifically, *cryptography* completely erases the semantic means of tags, while *perturbation* and *obfuscation* can only be applied to numerical values instead of words. These deficiencies render these approaches impractical in tagging recommendation [72]. To overcome these deficiencies, the *tag suppression* method has recently been proposed to protect a user's privacy by modeling users' profiles and eliminating selected sensitive tags [71]. However, this method only releases an incomplete dataset that significantly affects the recommendation performance. Moreover, the traditional privacy preserving methods can hardly provide a provable privacy guarantee for the recommender system. Accordingly, a more effective and rigourous privacy mechanism adapted to tagging recommender systems is required.

As the recent advanced research on the differential privacy, it can provide a rigorous privacy guarantee that suitable for tagging recommender systems. This chapter introduces *differential privacy* into tagging recommender systems, with the aim of preventing re-identification of users and avoiding the association of sensitive tags (e.g., healthcare tags) with a particular user. When applying differential privacy in tagging recommendation, there remain two research challenges:

- The naive *differential privacy* mechanism only focuses on releasing statistical information that can barely retain the structure of the tagging dataset. For example, this naive mechanism lists all the tags, counts the number and adds noise to the statistical output, but ignores the relationship among users, resources and tags. This simple statistical information is inadequate for recommendations.

- Differential privacy utilizes the randomized mechanism to preserve privacy, and usually introduces a large amount of noise due to the sparsity of the tagging dataset. For a dataset with millions of tags, the randomized mechanism will result in a large magnitude of noise.

Both challenges imply the naive *differential privacy* mechanism can not be simply applied in a tagging recommender system, and a novel differentially private mechanism is needed. To overcome the first challenge, we generate a synthetic dataset that retains the relationship among tags, resources and users rather than releasing simple statistical information. The second challenge can be addressed by shrinking the randomized domain, because the noise can decrease when the randomized range is limited. The topic model method is a possible way to structure tags into groups and limit the randomized domain within each topic. Based on these observations, we propose a tailored *differential privacy* mechanism that optimizes the performance of recommendation with a fixed level of privacy.

The contributions of this chapter can be summarized as follows:

- The main contribution maintains an acceptable utility of the tagging dataset by designing a practical private tagging release algorithm, *PriTop*, with a rigid privacy guarantee. To the best of our knowledge, this is the first work that adopts *differential privacy* for the tagging recommender system.

- In *PriTop*, a novel private topic model-based method is proposed to structure the tags and to shrink the randomized domain. The effectiveness of the proposed method is verified by extensive experiments on real-world datasets.
- With *differential privacy* composition properties, a theoretical privacy and utility analysis confirms an improved trade-off between privacy and utility.

The rest of this chapter is organized as follows. We present the background of the tagging recommender system in Section 4.2, and propose the *Private Tagging Release* algorithm in Section 4.3, together with the theoretical privacy and utility analysis of the algorithm in Section 4.4. Section 4.5 presents the results from experiments, followed by the summary in Section 4.6.

4.2 Fundamentals of Tagging Recommender System

4.2.1 Notations

In a tagging recommender system, G is a tagging dataset consisting of users, resources and tags. Let $U = \{u_1, u_2, \cdots\}$ be a set of users, $R = \{r_1, r_2, \cdots\}$ be a set of resources, and $T = \{t_1, t_2, \cdots\}$ be the set of all tags. For a particular user $u_a \in U$ and a resource $r_b \in R$, we use $T(u_a, r_b)$ to represent all tags flagged by the u_a on r_b, and use $T(u_a)$ to denote all tags utilized by user u_a. The recommended tags for u_a on a given resource $r \in R$ are represented by $T^p(u_a, r)$,

Tagging dataset D can be structured by a set of users' profiles $\mathbf{P} = \{P(u_1), ..., P(u_{|U|})\}$. A user u_a's profile $P(u_a) =< T(u_a), W(u_a) >$ is usually modeled by his tagging records, including a tag's names $T(u_a) = \{t_1, ..., t_{|T(u_a)|}\}$ and weights $W(u_a) = \{w_1, ..., w_{|T(u_a)|}\}$.

4.2.2 Tagging Recommender Systems

A considerable amount of literature has explored various techniques for tagging recommendations that offers users the possibility to annotate resources with personalized tags and to ease the process of finding suitable tags for a resource [82]. Sigurbjornsson et. al. provided a typical recommender strategy [83]. Given a resource with user-defined tags, an ordered candidate list of candidate tags is derived for each user-defined tag based on tag co-occurrence. After aggregating and ranking in the candidate list, the system provides top-N ranked tags.

Another well-known study is *FolkRank* [38], which adapts the *PageRank* method into the tagging recommender system. The key idea of *FolkRank* is that a resource flagged with important tags by important users becomes important itself. The importance is measured by weight $\overrightarrow{\omega}$, which is computed iteratively as follows.

$$\overrightarrow{\omega} \leftarrow \lambda A \overrightarrow{\omega} + (1-\lambda)\overrightarrow{\rho} \tag{4.2.1}$$

where A is the *adjacency matrix* of folksonomy, $\overrightarrow{\rho}$ is the preference vector and $\lambda \in [0, 1]$ is the damping factor measuring the influence of $\overrightarrow{\rho}$.

There are other methods for tagging recommender systems, such as the clustering based method [82], the tensor decomposition [88], and the topic-model method [46].

4.2.3 Related Work

Compared to general recommender systems, the privacy problem in tagging recommendation systems is more complicated due to its unique structure and semantic content. The most relevant paper was from Parra-Arnau et al. [71], who made the first contribution towards the development of a privacy preserving tagging system by proposing the *tag suppression* approach. They first modeled the user's profile using a tagging histogram and eliminated sensitive tags from the profile. To retain utility, they applied a clustering method to structure all tags and to suppress the less represented ones. Finally, they analyzed the effectiveness of their approach by discussing the *semantic loss* of users. However, there are several limitations on *tag suppression*. It only releases an incomplete dataset, with parts of the sensitive tags deleted. Sensitive tags are subjective, and therefore, have no quantity measurement. Furthermore, if the dataset is publicly shared, users can be identified because the remaining tags still have the potential to reveal a user's identity. The privacy issue in tagging recommender systems remains largely unexplored, and we attempt to fill this void in this chapter.

Hence, we propose a *Private Tagging Release* algorithm, with the aim of preserving comprehensive privacy for individuals and maximizing the utility of the released dataset. Specifically, we attempt to address the following issues:

- How is the unique structure of a tagging dataset retained? As mentioned earlier, the tagging dataset has a unique structure, and the relationship among users, resources and tags should be preserved within the privacy mechanism, otherwise, the utility will significantly decrease. Unfortunately, a naive differentially privacy mechanism will lose its structure. An effective way to solve this problem is to provide a synthetic dataset instead of simple statistical information.

- How can decrease the large magnitude of noise decrease when using *differential privacy*? In the tagging dataset, the key method to decrease noise is to shrink the scale of the randomization mechanism. Previous work focuses on methods that eliminate the number of tags but this results in high utility loss.

In this chapter, we retain the structure by using an improved *differential privacy* mechanism and shrink the randomized domain to limit the noise. Both issues will be investigated in the following sections.

4.3 Private Tagging Release

In tagging dataset D, user's profiles $\mathbf{P} = \{P(u_1), ..., P(u_{|U|})\}$, may disclose the user's privacy. If an adversary has part of the information on $P(u_a)$, he/she may re-identify a particular user in a tagging dataset by simply searching the known tags. More background information results in a higher probability of re-identifing a user. Traditional privacy approaches hardly provide sufficient protection to users due to the difficulty of modeling background information [29].

Differential privacy assumes all tags in the dataset have some probabilities to appear in $P(u_a)$. Specifically, tags in $P(u_a)$ are represented by $T(u) = \{t_1, ..., t_{|T|}\}$ and weights are denoted as $W(u_a) = \{w_1, ..., w_{|T|}\}$, where $w_i = 0$ indicates that t_i is unused. *Differential privacy* then utilizes the randomized mechanism that adds noise to the weight $W(u_a)$ and releases a noisy profile $\widehat{P}(u_a) =< T(u_a), \widehat{W}(u_a) >$. In this case, $W(u_a)$ is a sparse vector because a user tends to flag limited tags. When applying the randomized mechanism, $\widehat{W}(u_a)$ will contain a large amount of noise because lots of weights in $W(u_a)$ will change from zero to a positive value.

One way to reduce the noise is to shrink the randomized domain, which refers to the diminished number of zero weights in the profile. To achieve this objective, we structure the tags into K *topics* and each user is represented by a *topic-based* profile $P_z(u_a) =< T_z(u_a), W_z(u_a) >$, where $T_z(u_a) = \{T_{z_1}(u_a), ..., T_{z_K}(u_a)\}$ represents tags in each topic and $W_z(u_a) = w_{z_1}(u_a), ..., w_{z_K}(u_a)$ is the frequency of tags. Compared to $W(u_a)$, $W_z(u_a)$ is less sparse. Because the noise added to each $w_{z_i} \in W_z(u_a)$ is equal to $w_i \in W(u_a)$, the total noise added to $W_z(u_a)$ will significantly diminish.

In this section, we propose a ***Pri**vate **Top**ic-based Tagging Release* (PriTop) algorithm to address the privacy issues in tagging recommender systems. We first present an overview of the algorithm, then provide details of its operations.

4.3.1 Private Tagging Release Algorithm Overview

The *PriTop* algorithm aims to publish all users' profiles by masking their exact tags and weights under the notion of *differential privacy*. Three private operations are introduced to ensure each user in the releasing dataset cannot be re-identified.

- Private Topic Model Generation: This creates multiple topics according to the resources and tags by masking the topic distribution on tags. From the output, the adversary cannot infer to which topic a tag belongs.
- Topic Weight Perturbation: This operation masks the weights of tags in a user's profile to prevent an adversary from inferring how many tags a user has annotated on a certain topic.
- Private Tag Selection: Some privately selected tags replace the original tags

Algorithm 3 *Private Topic-based Tagging Release (PriTop)* Algorithm

```
Require: $D$, privacy parameter $\epsilon$, $K$.
Ensure: $\widehat{D}$
    1. Divide privacy budget into $\epsilon/2$, $\epsilon/4$ and $\epsilon/4$;
    2. Private Topic Generation: create topic-based user profiles $P(u_a)$ based on the
    private topic model with $\epsilon/2$ privacy budget;
    for each user $u_a$ do
        3. Topic Weight Perturbation: add Laplace noise to the topic weights with $\epsilon/4$
        privacy budget;
            $\widehat{W}(u_a) = W(u_a) + Laplace(\frac{4}{\epsilon})^K$
        for each topic $z_k$ in $P(u_a)$ do
            4. Private Tag Selection: Select tags according to the $\widehat{W}(u_a)$ in $\epsilon/4$ privacy
            budget;
        end for
    end for
    5. Output $\widehat{D}$ for tagging recommendations;
```

On the basis of these private operations, the proposed *PriTop* algorithm generates a new tagging dataset for recommendations. Its pseudocode is provided in Algorithm 3. Firstly, Step 1 divides the privacy budget into three parts for three private operations. Step 2 groups all tags into K topics and in Step 3, the weight for each topic is perturbed by *Laplace* noise. After privately selecting the new tags to replace the original ones in Step 4, the sanitized dataset $\widehat{D}$ is finally released for recommendation purposes in the last step.

These three private operations simultaneously guarantee fixed ϵ-*differential privacy* and retain the acceptable recommendation performance. Details for the *Private Topic Model Generation* operation is presented in subsection 4.3.2, followed by the *Topic Weight Perturbation* in subsection 4.3.3 and *Private Tag Selection* in subsection 4.3.4.

4.3.2 Private Topic Model Generation

This operation categorizes unstructured tags into topics to eliminate the randomization domain. The method is based upon the idea of a topic model, which considers a document as a mixture of topics and a topic is a probability distribution over words [8]. In our case, we apply one of the most popular topic models, *Latent Dirichlet Allocation* (LDA) [9], as the baseline model and introduce *differential privacy* to generate a private topic model. In this model, a resource is considered as a document and a tag is interpreted as a word. After discovering a set of topics expressed by tags, the *Laplace mechanism* ensures an adversary cannot infer which topic a tag belongs to.

We conceptualize *Private Topic Model Generation* in three steps:

- LDA Model Construction: Generating a LDA model using the *Gibbs Sampling* approach [31].
- Private Model Generation: Adding *Laplace* noise to the LDA model to create a private model.
- Topic-based Profile Generation: Creating a user's profile according to the private LDA model.

LDA Model Construction

The first step constructs the LDA model by *Gibbs Sampling*. LDA makes the assumption there are K topics associated with a given set of documents, and that each resource in this collection is composed of a weighted distribution of these topics. Let $Z = \{z_1, ...z_K\}$ be a group of topics, with the following equation representing a standard LDA model to specify the distribution over tag t.

$$Pr(t|r) = \sum_{l=1}^{K} Pr(t|z_l)Pr(z_l|r) \qquad (4.3.1)$$

where $Pr(t|z_l)$ is the probability of tag t under a topic z_l and $Pr(z_l|r)$ is the probability of sampling a tag from topic z in resource r.

In the model, the main variables of interest are the topic-tag distribution $Pr(t|z)$ and the resource-topic distribution $Pr(z|r)$. *Gibbs sampling* is a relatively efficient method to estimate these variables in the model by extracting a set of topics from a large document collection [31]. It iterates multiple times over each tag t of resource r and samples the new topic z for the tag based on the posterior probability $Pr(z|t_i, r, Z_{-i})$ by Eq. 4.3.2 until the model converges.

$$Pr(z|t_i, r, Z_{-i}) \propto \frac{C^{TK}_{tK} + \beta}{\sum_{t_i}^{|T|} C^{TK}_{t_iK} + |T|\beta} \frac{C^{RK}_{rk} + \alpha}{\sum_{k=1}^{K} C^{RK}_{r_ik} + K\alpha} \qquad (4.3.2)$$

where C^{TK} maintains a count of all topic-tag assignments and C^{RK} counts the resource-topic assignments. Z_{-i} represents all topic-tag assignments and resource-topic assignments except the current z for t_i. α and β are hyperparameters for the Dirichlet priors, which can be interpreted as the prior observation for the counts.

Evaluation on the $Pr(t|z)$ and $Pr(z|r)$ is formulated as follows:

$$Pr(t|z) = \frac{C^{TK}_{tk} + \beta}{\sum_{t_i}^{|T|} C^{TK}_{t_iK} + |T|\beta} \qquad (4.3.3)$$

$$Pr(z|r) = \frac{C_{rK}^{RK} + \alpha}{\sum_{k=1}^{K} C_{r_i k}^{RK} + K\alpha} \tag{4.3.4}$$

After converging, the LDA model is generated by estimating $Pr(z|t,r)$, $P(t|z)$ and $P(z|r)$.

Private Model Generation

The second step introduces *differential privacy* by adding *Laplace* noise to the final counts in the LDA model. There are four difference counts in Eq. 4.3.2: C_{tK}^{TK}, $\sum_{t_i}^{|T|} C_{t_i K}^{TK}$, C_{rK}^{RK} and $\sum_{K=1}^{K} C_{r_i K}^{RK}$. If we changed the topic assignment on current t_i, the C_{tK}^{TK} will decrease by 1 and $\sum_{t_i}^{|T|} C_{t_i K}^{TK}$ will increase by one. Similarly, if the C_{rK}^{RK} decreases by 1, the $\sum_{K=1}^{K} C_{r_i K}^{RK}$ will increase by 1 accordingly. Based on this observation, we sample two groups of independent random noise from *Laplace* distribution and add them to four count parameters. The new $\widehat{Pr}(z|t,r)$ is evaluated as follows:

$$\widehat{Pr}(z|t,r) \propto \frac{C_{tK}^{TK} + \eta_1 + \beta}{\sum_{t_i}^{|T|} C_{t_i K}^{TK} - \eta_1 + |T|\beta} \frac{C_{rK}^{RK} + \eta_2 + \alpha}{\sum_{k=1}^{K} C_{r_i k}^{RK} - \eta_2 + K\alpha} \tag{4.3.5}$$

where η_1 and η_2 are both sampled from *Laplace* distribution *Laplace*$(\frac{2}{\epsilon})$ with the *sensitivity* as 1.

Topic-based Profile Generation

The third step creates topic-based user profiles. For each user with tags $T(u_a) = \{t_1, ..., t_{|T(u_a)|}\}$ and related resources $R(u_a) = \{r_1, ..., r_{|R(u_a)|}\}$, each tag can be assigned to a particular topic $z_l \in Z$ according to the $\widehat{Pr}(z|t,r)$. This means the user profile can be represented by a topic-based $P_z(u_a) =< T_z(u_a), W_z(u_a) >$ with the weight $W_z(u_a) = \{w_1(u_a), ..., w_K u_a\}$.

Generally, a *Private Topic Model Generation* operation constructs a private LDA model to create the topic-based user profile and retains the structure between the tag and resource. Details of this operation are shown in Algorithm 4.

For resources, the resulting topics represent a collaborative view of the resource, and tags of topics reflect the vocabulary to describe the resource. For users, the resulting topics indicate the preference of a user.

4.3.3 Topic Weight Perturbation

After generating the topic-based user profile $P_z(u_a)$, *Laplace* noise will be added to mask the counts of tags in each topic.

$$\widehat{W}_z(u_a) = W_z(u_a) + Laplace(\frac{4}{\epsilon})^K \tag{4.3.6}$$

Noise added on the weight $W_z(u_a)$ implies the revision of the tag list $T_z(u_a)$. Positive noise indicates that new tags are added to the $T_z(u_a)$, while negative noise indicates some tags have been deleted from the list. For positive noise in the topic z_l, the operation will choose the tags with the highest probability in the current topic z_j according to the $Pr(t|z)$. For negative noise, the operation will delete the tag with the lowest probability in the current topic z_j.

$$\widetilde{T}_{z_l}(u_a) = T_{z_l}(u_a) + t_{new} \tag{4.3.7}$$

where $t_{new} = \max_{i=1}^{|T|} Pr(t_i|z_l)$.

$$\widetilde{T}_{z_l}(u_a) = T_{z_l}(u_a) - t_{delete} \tag{4.3.8}$$

where $t_{delete} = \min_{i=1}^{|T|} Pr(t_i|z_l)$.

Algorithm 4 *Private Topic Model Generation*

```
Require: $D$, privacy budget $\frac{\epsilon}{2}$, numbers of topics $K$
Ensure: $\mathbf{P_z} = \{P_z(u_1), ...P_z(u_{|U|})\}$
  for each tag $t_i$ do
    Randomly initial the topic assignment $Pr(z = z_j|t_i, r, Z_{-i})$;
  end for
  for each $r_j$ do
    for each $t_i$ do
      repeat
        estimate $Pr(t|z) = \frac{C_{tk}^{TK}+\beta}{\sum_{t_i}^{|T|} C_{t_ij}^{TK}+|T|\beta}$;
        estimate $Pr(z|r) = \frac{C_{rk}^{RK}+\alpha}{\sum_{k=1}^{K} C_{r_ik}^{RK}+K\alpha}$;
        resign $t_i$ a new topic $z$ according to
          $$Pr(z|t_i, r, Z_{-i}) \propto \frac{C_{tk}^{TK}+\beta}{\sum_{t_i}^{|T|} C_{t_iK}^{TK}+|T|\beta} \frac{C_{rk}^{RK}+\alpha}{\sum_{k=1}^{K} C_{r_ik}^{RK}+K\alpha}$$
      until converge
    end for
  end for
  for each $r_j$ do
    for each $t_i$ do
      Sample noise from the Laplace distribution:
        $$\eta_1 \sim Laplace(\frac{2}{\epsilon})$$
        $$\eta_2 \sim Laplace(\frac{2}{\epsilon})$$
      Estimate
        $$\widehat{Pr}(z|t, r) \propto \frac{C_{tK}^{TK}+\eta_1+\beta}{\sum_{t_i}^{|T|} C_{t_iK}^{TK}-\eta_1+|T|\beta} \frac{C_{rK}^{RK}+\eta_2+\alpha}{\sum_{k=1}^{K} C_{r_iK}^{RK}-\eta_2+K\alpha}$$
    end for
```

After perturbation, we use $\widetilde{P}_z(u_a) =< \widetilde{T}_z(u_a), \widehat{W}_z(u_a) >$ to represent the noisy topic-based user profile. However, the $\widetilde{P}_z(u_a)$ still has a high probability of being re-identified because it retains a major part of the original tags. The next operation will replace all tags in $\widetilde{T}(u_a)$ to preserve privacy.

4.3.4 Private Tag Selection

The *Private Tag Selection* operation manages to replace original tags with selected new tags. The challenge is how to select suitable new tags in related topics. For a tag $t_i \in \widetilde{T}_{z_l}(u_a)$, uniformly random tag selection within $\hat{T}_{z_l}(u_a)$ is unacceptable due to significant detriment to the utility. The intuitive solution to retain the utility is to use the most similar tag to replace the original one. However, this approach is also dangerous because an adversary can easily figure out the tag most similar using simple statistical analysis. Consequently, *Private Tag Selection* needs to: 1) retain the utility of tags, and 2) mask the similarities between tags.

To achieve both of these, *Private Tag Selection* adopts the *Exponential* mechanism to privately select tags from a list of candidates. Specifically, for a particular tag t_i, the operation first locates the topic z_l to which it belongs, and all tags in $\hat{T}_{z_l}(u_a)$ are then included in a candidate list I. Each tag in I is associated with a probability based on a *score function* and the *sensitivity* of the function. The selection of tags is performed based on the allocated probabilities.

The *score function* is defined by the distance between tags. In the LDA model, the distance between tags is measured by the extent of identical shared topics [86]. Using a probabilistic approach, the distance between two tags t_1 and t_2 is computed based on the *Jensen-shannon divergence* (*JS* divergence) between $Pr(z|t_i = t_1)$ and

$Pr(z|t_i = t_2)$. *JS* divergence is a symmetrized and smoothed version of the *Kullback-Leibler divergence* (*KL* divergence).

Pr_1 and Pr_2 is defined to be:

$$Dist_{KL}(Pr_1||Pr_2) = \sum_i \ln(\frac{Pr_1(i)}{Pr_2(i)})Pr_1(i) \tag{4.3.9}$$

$$Dist_{JS}(Pr_1||Pr_2) = \frac{1}{2}Dist_{KL}(Pr_1||M) + \frac{1}{2}Dist_{KL}(Pr_2||M) \tag{4.3.10}$$

where $M = \frac{1}{2}(Pr_1 + Pr_1)$.

Because JS divergence is bounded by 1 when using the base 2 logarithm [54], we define the score function q for a target tag t_i as follows:

$$q_i(I, t_j) = (1 - Dist_{JS}(Pr_i||Pr_j)), \tag{4.3.11}$$

where I is tag t_i's candidate list, and $t_j \in I$ are the candidate tags for replacement. Each tag t_j has a score according to Eq. 4.3.11.

The *sensitivity* for score function q is measured by the maximal change in the distance of two tags when removing a topic shared by both t_i and t_j. Let $Dist'_{JS}(Pr_i||Pr_j)$ denote the new distance between t_i and t_j after deleting a topic, and the maximal difference between $Dist'_{JS}(Pr_i||Pr_j)$ and $Dist_{JS}(Pr_i||Pr_j)$ is bounded by 1. Therefore, the sensitivity of the score function is 1.

On the basis of the *score function* and *sensitivity*, the probability arranged to each tags t_j is computed by Eq. 4.3.12 with the privacy budget $\frac{\epsilon}{4}$. The pseudocode of *Private Tag Selection* is presented in Algorithm 5:

$$Pr_{t_j \in I}(t_j) = \frac{\exp\left(\frac{\epsilon \cdot q_i(I,t_j)}{8}\right)}{\sum_{j \in z_l} \exp\left(\frac{\epsilon \cdot q_i(I,t_j)}{8}\right)}. \tag{4.3.12}$$

where z_l is the topic in which t_j belongs to.

Algorithm 5 Private Tag Selection

```
Require: $\frac{\epsilon}{4}$, $\widetilde{T}_z(u_a)$, $Pr(z|t)$
Ensure: $\widehat{T}_z(u_a)$
  for each tags $t_i$ in $T_z(u_a)$ do
    1. located the $t_i$ in topic $z_l$;
    for each tags $t_j$ in $z_l$ do
      2. Allocate probability as:
         $$\frac{\exp\left(\frac{\epsilon \cdot q_i(I,t_j)}{8}\right)}{\sum_{j \in z_l} \exp\left(\frac{\epsilon \cdot q_i(I,t_j)}{8}\right)}.$$
    end for
    3. Select a tag $t_j$ from $z_l$ without replacement according to the probability;
  end for
  4. Output $\widehat{T}_z(u_a)$
```

Table 4.1: Privacy Budget Allocation in the *PriTop* Algorithm

Operations	Privacy Budget
Private Topic Model Generation	$\epsilon/2$
Topic Weight Perturbation	$\epsilon/4$
Private Tag Selection	$\epsilon/4$

4.4 Privacy and Utility Analysis

The proposed *PriTop* algorithm aims to obtain acceptable utility with a fixed ϵ differentially privacy level. This section first proves the algorithm is satisfied with ϵ-differential privacy, and then analyzes the utility cost.

4.4.1 Privacy Analysis

To analyze the privacy guarantee, we apply two composite properties of the privacy budget: the *sequential* and the *parallel composition* in Definitions 2 and 3, respectively. The *sequential composition* accumulates privacy budget ϵ of each step when a series of private analysis is performed *sequentially* on a dataset. The *parallel composition* corresponds to the case that each private step is applied on disjointed subsets of the dataset. The ultimate privacy guarantee depends on the step with the maximal ϵ.

The *PriTop* algorithm contains three private operations: *Private Topic Model Generation*, *Topic Weight Perturbation* and *Private Tag Selection*. The privacy budget ϵ is consequently divided into three pieces, as illustrated in Table 4.1.

Based on the above Lemmas and privacy budget allocation in Table 4.1, we measure the privacy level of our algorithm as follows:

- The *Private Topic Model Generation* operation is performed on the whole dataset with the privacy budget $\frac{\epsilon}{2}$. According to Def. 2, this operation preserves $\frac{\epsilon}{2}$-*differential privacy*.
- The *Topic Weight Perturbation* applies the *Laplace* mechanism to the weights of topics. The noise is calibrated by $Lap\left(\frac{4}{|T(u)|\cdot\epsilon}\right)^K$ and preserves $\frac{\epsilon}{4}$−*differential privacy* for each user. Furthermore, as a user's profile is independent, replacing a user's tags has no effect on other user profiles. According to Def. 3, the *Private Tag Selection* preserves $\frac{\epsilon}{4}$-*differential privacy* as a whole.
- The *Private Tag Selection* processes the *Exponential* mechanism successively. For one user u, each tag in the profile is replaced by a privately selected tag until all tags are replaced. Each selection is performed on individual tags, therefore according to Def. 2, the selection for each user guarantees $\frac{\epsilon}{4}$-*differential privacy*. Similar to the previous operation, every user can be considered as a subset of the entire dataset. Thus, the *Private Tag Selection* guarantees $\frac{\epsilon}{4}$-*differential privacy*.

Consequently, the proposed *PriTop* algorithm preserves ϵ-*differential privacy*.

4.4.2 Utility Analysis

Given a target user u_a, the utility level of the proposed *PriTop* algorithm is determined by the accuracy of the tagging recommendation, which is highly dependent on the distance between $P(u_a)$ and $\widehat{P}(u_a)$ [71]. The distance between $P(u_a)$ and $\widehat{P}(u_a)$ is referred to as *semantic loss* [71].

$$SLoss = \frac{1}{|U|}\sum_{u\in U}(\frac{\sum_{t\in P(u_a)} d(t,\widehat{t})}{\max d \cdot |T(u)|}). \tag{4.4.1}$$

where $\widehat{t}$ is the new tag replacing the tag t.

If we consider each private step as query f, then the difference between $f(D)$ and $f(D')$ is the *sematic loss*. We then apply a widely used utility definition in *differential privacy* suggested by Blum et al. [11]:

Definition 10 ((α,δ)-usefulness). A mechanism $\mathcal{M}$ is (α,δ)-useful for a set of query F, if with probability $1-\delta$, for every query $f \in F$ and every dataset G, for $\widehat{D} = \mathcal{M}(D)$, we have

$$\max_{f\in F}|f(\widehat{D}) - f(D)|\leq \alpha, \tag{4.4.2}$$

where F is a group of queries.

Based on definition, we will demonstrate the *sematic loss* is bounded by a certain value α with a high probability.

All three private steps affect the *semantic loss*. However, the first step, private topic model generation, only affects the distance measurement between tags. Therefore, we only need to measure the $SLoss_1$ in the perturbation step and $SLoss_2$ in the selection step.

Theorem 4.4.1. *For any user $u \in U$, for all $\delta > 0$, with probability at least $1-\delta$, the $SLoss_1$ of the user in the perturbation is less than α. When*

$$|T(u)|\geq \frac{K \cdot \exp(\frac{-\epsilon\alpha_a}{4})}{\delta}$$

the perturbation operation is satisfied with (α,δ)-useful.

Proof. The perturbation adds *Laplace* noise with $\epsilon/4$ privacy budget to the weight of each topic in a user's profile. According to the property of *Laplace*(b):

$$Pr(|\gamma|> t) = Pr(\gamma > t) + Pr(\gamma < -t) = 2\int_t^\infty x\exp(-\frac{x}{b})dx \tag{4.4.3}$$

$$= \exp(-\frac{t}{b}) \tag{4.4.4}$$

We have

$$Pr(SLoss_1 > \alpha_a) = \frac{2K \cdot d(t_{ai}, \widehat{t_{ai}})}{\max d|T(u_a)|}\int_{\alpha_a}^0 \frac{\epsilon}{8}\exp(-\frac{\epsilon x}{4})dx \tag{4.4.5}$$

$$Pr(SLoss_1 > \alpha_a) = \frac{K \cdot d(t_{ai}, \widehat{t_{ai}})}{\max d|T(u_a)|}\exp(-\frac{\epsilon\alpha_a}{4}) \tag{4.4.6}$$

As the perturbation step adds new tags or delete tags, the $d(t_{ai}, \widehat{t_{ai}})$ will be less than the maximal value. When we use the *JS* divergence, the maximal $d(t_{ai}, \widehat{t_{ai}})$ is 1, so we obtain the evaluation on the $SLoss_1$ as

$$Pr(SLoss_1 < \alpha_a) \leq 1 - \frac{K \cdot \exp(-\frac{\epsilon\alpha_a}{4})}{|T(u_a)|} \tag{4.4.7}$$

Let

$$1 - \frac{K \cdot \exp(-\frac{\epsilon\alpha_a}{4})}{|T(u_a)|} \geq 1 - \delta$$

Thus

$$|T(u_a)|\geq \frac{K \cdot \exp(\frac{-\epsilon\alpha_a}{4})}{\delta} \tag{4.4.8}$$

The average semantic loss for all users is less than the maximal value, $\alpha = \max_{u_a \in U}\alpha_a$, we have

$$|T(u)|\geq \frac{K \cdot \exp(\frac{-\epsilon\alpha_a}{4})}{\delta} \tag{4.4.9}$$

□

Theorem 5.5.1 reveals the *semantic loss* of perturbation depends on the number of tags a user has. More tags results in lower *semantic loss.*

Theorem 4.4.2. *For any user $u \in U$, for all $\delta > 0$, with probability at least $1-\delta$, the $SLoss_2$ of the user in the private selection is less than α. When*

$$Q \leq \frac{\exp(\frac{\epsilon}{8})}{1-\delta\alpha},$$

where Q is the normalization factor that depends on the topic that $t \in T(u)$ belongs to, the private selection operation is satisfied with (α, δ)-useful.

Proof. According to Marlkov's inequality, we obtain

$$Pr(SLoss_2 > \alpha_a) \leq \frac{E(SLoss_2)}{\alpha_a} \tag{4.4.10}$$

For each tag t_{ai} in $\widehat{P}_a$, the probability of 'unchange' in the private selection is proportional to $\frac{\exp(\frac{\epsilon}{8})}{Q_i}$, where Q_i is the normalization factor depending on the topic t_{ai} belongs to. Therefore, we obtain

$$E(SLoss_2) = \sum_{t_i \in T(u_a)} \frac{d(t_{ai}, \widehat{t_{ai}})}{\max d|T(u_a)|}(1 - \frac{\exp(\frac{\epsilon}{8})}{Q_i})$$

According to 4.4.10, the evaluation of the $SLoss_2$ is

$$Pr(SLoss_2 > \alpha_a) \leq \frac{\sum_{t_i \in T(u_a)} d(t_{ai}, \widehat{t_{ai}})(1 - \frac{\exp(\frac{\epsilon}{8})}{Q_i})}{|T(u_a)|\alpha_a}$$

When we take the maximal $d(t_{ai}, \widehat{t_{ai}})$ and $Q = \max Q_i$, it can be simplified as

$$Pr(SLoss_2 \leq \alpha_a) \geq 1 - \frac{1 - \frac{1}{Q}\exp(\frac{\epsilon}{8})}{\alpha_a} \tag{4.4.11}$$

Let

$$1 - \frac{1 - \frac{1}{Q}\exp(\frac{\epsilon}{8})}{\alpha_a} \geq 1 - \delta$$

Thus

$$Q \leq \frac{\exp(\frac{\epsilon}{8})}{1 - \delta\alpha_a} \tag{4.4.12}$$

For all users, α is determined by the maximal value: $\alpha = \max_{u_a \in U} \alpha_a$.

Finally, we obtain

$$Q \leq \frac{\exp(\frac{\epsilon}{8})}{1 - \delta\alpha} \tag{4.4.13}$$

where $Q = \max Q_i$, and $Q_i = \sum_{j \in z_l} \exp\left(\frac{\epsilon \cdot d(t_i, t_j)}{8}\right)$

$\square$

The proof shows the *semantic loss* of private selection mainly depends on the privacy budget and the normalization factor Q_i, which is measured by the total distance inside topic z to which t_i belongs. The shorter distance leads to a smaller Q_i and less *semantic loss*.

Further analysis shows the total distance in a topic is determined by privacy budget ϵ in the *private topic model generation*. It can be concluded the privacy budget has a significant impact on the utility level of *PriTop*. The *semantic loss* will be evaluated in experimental Section 4.5.2.

4.5 Experiment and Analysis

This section evaluates the performance of the proposed *PriTop* algorithm by answering the following questions:

- *How does the PriTop algorithm affect the semantic loss?* In Section 4.4.2, we theoretically analyzed the *semantic loss*. Here, we empirically investigate it in the real-world datasets. Moreover, to illustrate its effectiveness, we compare the proposed *PriTop* algorithm with *tag suppression* [71] in terms of *semantic loss*.

- *How does the PriTop algorithm perform in a real tagging recommender system?* The purpose of releasing synthetic datasets is to provide recommendations with a fixed privacy level. In this part, we investigate the performance of *PriTop* in *FolkRank*, a state-of-the-art recommender system. In addition, we not only compare *PriTop* with *tag suppression*, but also with the non-private recommender results.

- *How does the privacy budget affect the performance of the algorithm?* In the context of privacy preserving, the privacy budget is a key parameter that controls the privacy level of algorithms. To show its impact, we examine the trade-off between the utility and the privacy of *PriTop* by varying the privacy budget over a wide range.

4.5.1 Datasets

To obtain a thorough comparsion, we conduct the experiment on four datasets: *Del.icio.us*, *Bibsonomy*, *MovieLens* and *Last.fm*. The statistics for all datasets are summarized in Table. 4.2.

All four datasets are structured in the form of triples (*user*, *resource*, *tag*), and filtered by automatically removing added tags like "imported", "public", etc.

`Del.icio.us` The *Del.icio.us* dataset is retrieved from the *Del.icio.us* web site by the *Distributed Artificial Intelligence Laboratory* (DAI-Labor)[1], and includes

[1]http://www.dai-labor.de/

Table 4.2: Characteristics of the datasets

Dataset	*Record*	$\|U\|$	$\|R\|$	$\|T\|$
Del.icio.us	$130,160$	3000	$34,212$	$12,183$
Bibsonomy	$163,510$	3000	$421,928$	$93,756$
Last.fm	$186,479$	1892	$12,523$	$9,749$
MovieLens	$47,957$	2113	$5,908$	$9,079$

around 132 million resources and $950,000$ users. We extracted a subset with $3,000$ users, $34,212$ bookmarks and $12,183$ tags.

`Bibsonomy` The *Bibsonomy* dataset is provided by *Discovery Challenge 2009 ECML/PKDD2009*[2]. The dataset contains $3,000$ individual users, $421,928$ resources and $93,756$ tags.

`MovieLens` **and** `Last.fm` Both datasets were obtained from *HetRec 2011*[3], which were generated by the *Information Retrieval Group* at *Universidad Autonoma de Madrid*.

4.5.2 Semantic Loss Analysis

To maintain consistency with previous research, we thoroughly compare *the semantic loss* of *PriTop* with *tag suppression* [71] on four datasets.

For the *PriTop* algorithm, we selected $\epsilon = 0.1, 0.3, 0.5, 0.7$ and 1.0 to represent different privacy levels and the number of topic K varies from 10 to 100 with a step of 10. In *tag suppression* [71], the average *semantic loss* exhibits a linear relationship

[2]http://www.kde.cs.uni-kassel.de/ws/dc09/
[3]http://ir.ii.uam.es/hetrec2011

with the *eliminate parameter* σ. When we choose the representative value $\sigma = 0.8$, the *sematic loss* is fixed to 0.2. Based on this configuration, the following experiment compares the *sematic loss* of the *PriTop* algorithm with the fixed suppression $SLoss = 0.2$.

Fig. 4.1 shows the results on the four datasets. It can be observed the *semantic loss* of the *PriTop* algorithm in a variety of datasets was less than 0.2 with different privacy budgets, which indicates that *PriTop* outperforms *tag suppression* on all datasets. Specifically, the *PriTop* algorithm obtains a considerably lower *semantic loss* when $\epsilon = 1$. For example, in Fig. 4.1a, when $K = 90$ and $\epsilon = 1$, the user *semantic loss* is 0.0767, which is 62% lower than *tag suppression* with $SLoss = 0.2$. Even in a higher privacy level ($\epsilon = 0.1$), the *semantic loss* is still lower than *tag suppression* by 23% when $K = 90$. This trend is retained when K equals other values, thus illustrating the effectiveness of *PriTop* in terms of semantic information retained for recommendations.

Similar trends can also be observed in Fig. 4.1b, 4.1c and 4.1d. All figures show that *PriTop* obtains a stable *semantic loss* at a lower level. These figures also indicate the *PriTop* algorithm retains more utility than *tag suppression*, because *PriTop* retains the relationship between tags and resources. Consequently, the semantic information can be well preserved and the tags will still make the profiles of users meaningful. The results on all datasets demonstrate the proposed *PriTop* outperforms *tag suppression* in terms of semantic retaining.

A further analysis is conducted to investigate the impact on the number of topic K on the *semantic loss*. Generally, a larger K induces less *semantic loss*, but the decreasing speed varies on different datasets. For example, Fig. 4.1b shows that in the *MovieLens* dataset, the *semantic loss* decreases faster when $K < 60$ and tends to decrease slightly when $K \geq 60$. On other datasets, as shown in Fig. 4.1a, Fig. 4.1c and Fig. 4.1d, the *semantic loss* keeps decreasing with a similar slope. However, even we obtain lower *semantic loss* when selection has a larger value of K, however on a dataset with limited tags, the larger K will over-reduce the randomization scope in the *private selection* operation, which may impact the privacy of the entire algorithm. Consequently, we set $K = 100$ in the following experiments to obtain a reasonable result.

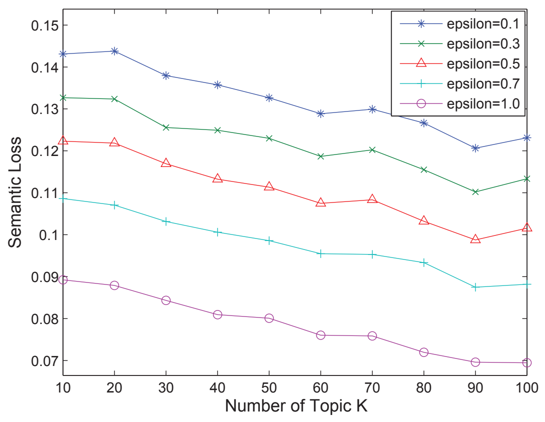
4.5.3 Performance of Tagging Recommendation

This section investigates the effectiveness of *PriTop* in the context of tagging recommendations and compares it with *tag suppression*. We apply a state-of-the-art tagging recommender system, *FolkRank* [38], to measure the degradation of tag recommendations with privacy preserving.

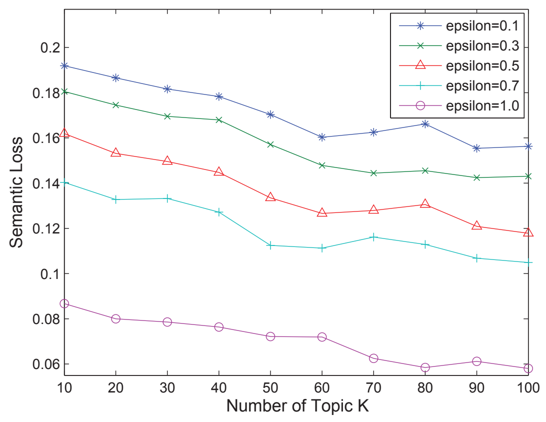
In the *FolkRank* configuration, we set $\lambda = 0.7$, $\overrightarrow{\rho} = 1$, and the preference weights are set to $1 + |U|$ and $1 + |R|$, respectively. The computation repeats for 10 iterations or stops when the distance between two consecutive weight vectors is less than 10^{-6}.

We apply the *Leave-One-Out* measurement strategy, which is a popular configuration in evaluating tag recommendations [59]. To begin with, we randomly select one resource of each user, and predict a list of N (top-N list) tags using all remaining tags in the dataset. *Precision* and *recall* are used to quantify the performance. A larger value of *precision* or *recall* means better performance.

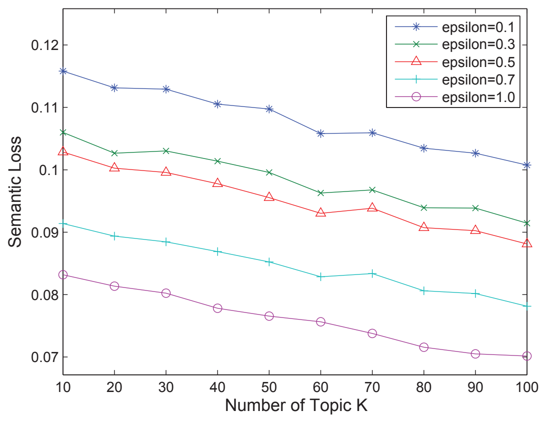
$$precision(T(u,r), T^p(u,r)) = \frac{T(u,r) \cap \widetilde{T}(u,r)}{|T^p(u,r)|}. \qquad (4.5.1)$$



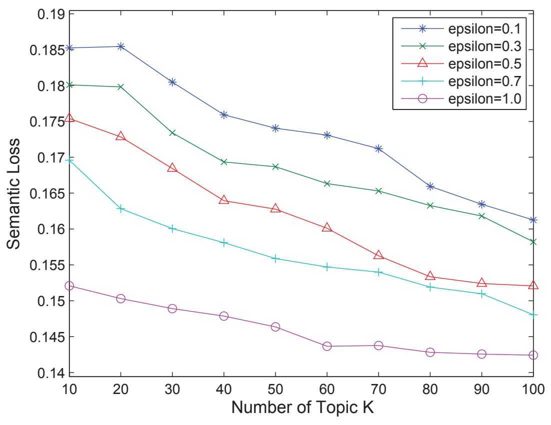
(a) *Semantic Loss* in *Del.icio.us*



(b) *Semantic Loss* in *MovieLens*

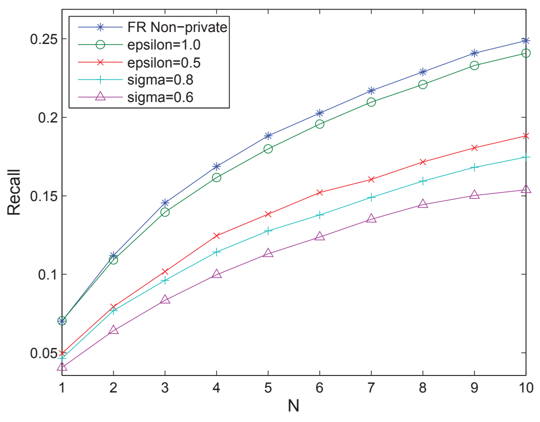


(c) *Semantic Loss* in *Last.fm*

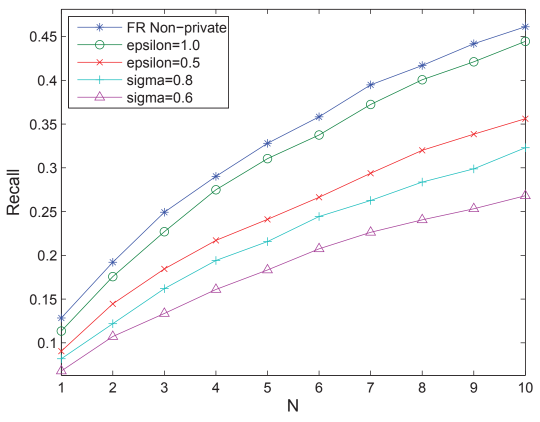


(d) *Semantic Loss* in *Bibsonomy*

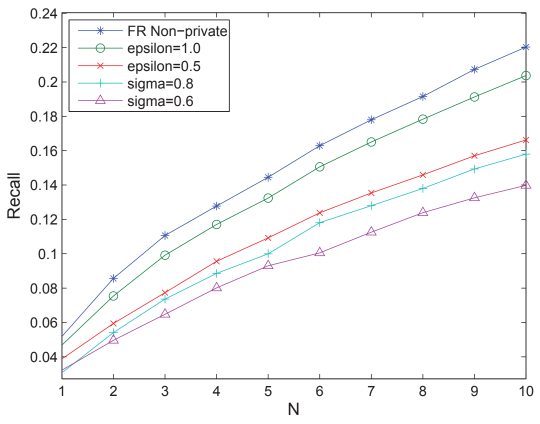
Figure 4.1: *Semantic Loss* on Different Datasets



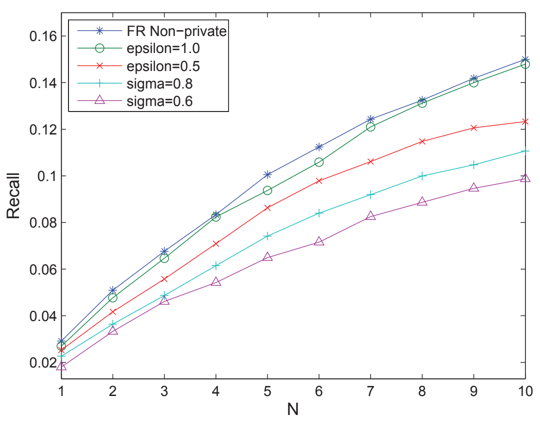
(a) *Del.icio.us*



(b) *MovieLens*



(c) *Last.fm*



(d) *Bibsonomy*

Figure 4.2: *FolkRank* Recall Result

$$recall(T(u,r), T^p(u,r)) = \frac{T(u,r) \cap T^p(u,r)}{|T(u,r)|}. \tag{4.5.2}$$

The following experiments compare *PriTop* with *tag suppression* when N varies from 1 to 10. For *PriTop*, we chose the number of topic $K = 100$, and test the performance when $\epsilon = 1$ and $\epsilon = 0.5$. For *tag suppression*, we fix the *eliminate parameter* to $\sigma = 0.8$ and $\sigma = 0.6$, which corresponds to the suppression rates of 0.2 and 0.4, respectively.

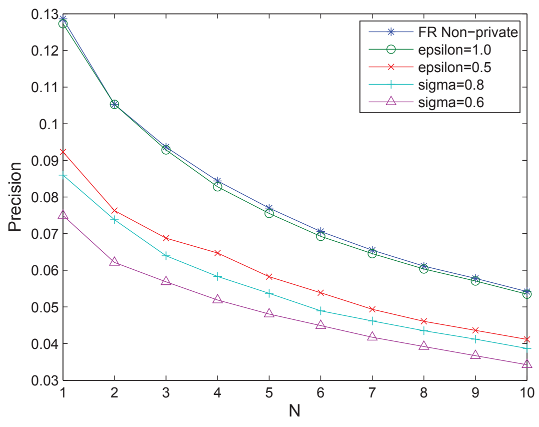
Fig. 4.2 presents the recall of recommendation results. It is observed the proposed *PriTop* algorithm significantly outperforms the *tag suppression* method on both privacy budgets. Specifically, as shown in Fig. 4.2a, when $N = 1$, *PriTop* achieves a *recall* at 0.0704 with the $\epsilon = 1$, which outperforms the result from *tag suppression* with $\sigma = 0.6$, 0.0407 by 42.19%. This trend is retained as N increases. For example, when $N = 5$, *PriTop* achieves a *recall* of 0.1799 with the $\epsilon = 1$, which outperforms the result from the *tag suppression* by 37.19% when $\sigma = 0.6$, 0.113. When N reaches 10, the *PriTop* still retains 36.09% higher on *recall* than *tag suppression*. Even we choose the lower privacy budget with $\epsilon = 0.5$ and a higher *eliminate* parameter $sigma = 0.8$, the improvement of *PriTop* is still significant. *PriTop* has a *recall* of 0.1382, which is also 7.67% higher than *tag suppression* with a *recall* of 0.1276. Moreover, the improvement of *PriTop* is more obvious when $N = 10$. It achieves *recalls* of 0.1882 and 0.2408 when $\epsilon = 1$ and $\epsilon = 0.5$, respectively. However, *tag suppression* only achieves *recalls* of 0.1538 and 0.1881 with $\sigma = 0.6$ and $\sigma = 0.8$. Similar trends can also be observed in Fig. 4.2b, 4.2c and 4.2d. For example, in the *MovieLens* dataset, when $N = 10$ and $\epsilon = 1.0$, the recall of *PriTop* is 0.4445, which is 27.33% higher than *tag suppression* with $\sigma = 0.8$. With the same configuration, *PriTop* is 22.43% and 25.22%

higher than *tag suppression* in the *Last.fm* and *Bibsonomy* datasets. The experimental results show the *PriTop* algorithm outperforms *tag suppression* in a variety of N, which implies *PriTop* can retain more useful information for recommendations than simply deleting tags.

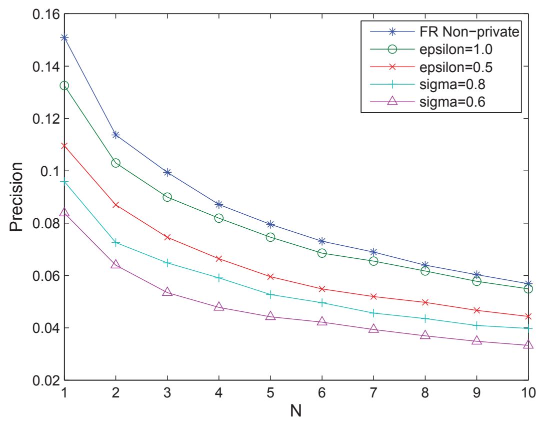
Moreover, it is clear the performance of *PriTop* is very close to the *non-private* baseline. For example, in Fig. 4.2a, when $\epsilon = 1$, the *recall* of the *De.licio.us* dataset is 0.2408, which is only 3.00% lower than the non-private recommender result. Other datasets show the same trend. As shown in Fig. 4.2b, 4.2c and 4.2d, with the same configuration, the *PriTop* result is 3.62% lower than the non-private result in the *MovieLens* dataset, 7.58% lower in the *Last.fm* dataset and 1.4% lower in the *Bibsonomy* dataset. The results indicate the *PriTop* algorithm can achieve the privacy preserving objective while retaining a high accuracy of recommendations.

Fig. 4.3 supports the above claims by plotting the precision results, which also shows the improvement of *PriTop* compared to *tag suppression* and the high recommender accuracy results of *PriTop*. However, curves in Fig. 4.2d are not as smooth as others. This may be caused by the statistical property of the *Bibsonomy* dataset, which contains a large number of tags that appeare only once. When the test samples include a large proportion of these tags, the *precision* fluctuates.

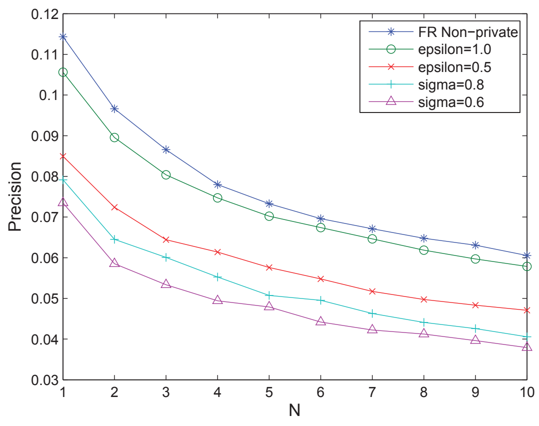
To show the statistical effectiveness of *PriTop*, we apply a paired t test (with a 95% confidence level) to examine the difference on the performance of *PriTop* with $\epsilon = 1.0$ and *tag suppression* with $\sigma = 0.2$. The statistics for results on the four datasets are shown in Table 4.3. All t values are greater than 6 and all p values are less than 0.0001, thus indicating both improvements on *recall* and *precision* are statistically significant.



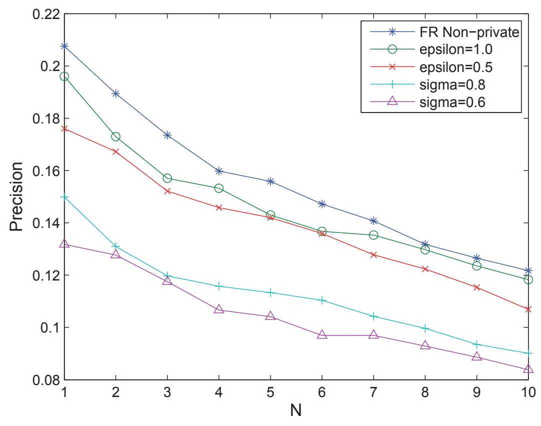
(a) *Del.icio.us*



(b) *MovieLens*



(c) *Last.fm*



(d) *Bibsonomy*

Figure 4.3: *FolkRank* Precision Result

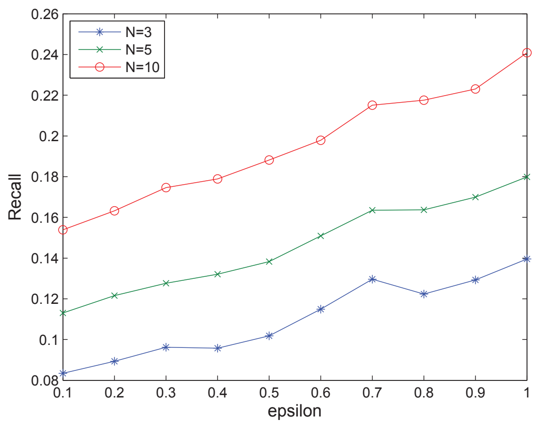
Table 4.3: Paired-t-test for *PriTop* vs. *Tag Suppression*

		df	*t*	*p*-value
De.licio.us	Recall	9	11.3276	< 0.0001
	Precision	9	8.8598	< 0.0001
MovieLens	Recall	9	9.0957	< 0.0001
	Precision	9	10.7693	< 0.0001
Last.fm	Recall	9	10.7546	< 0.0001
	Precision	9	11.1514	< 0.0001
Bibsonomy	Recall	9	6.7718	< 0.0001
	Precision	9	16.3477	< 0.0001

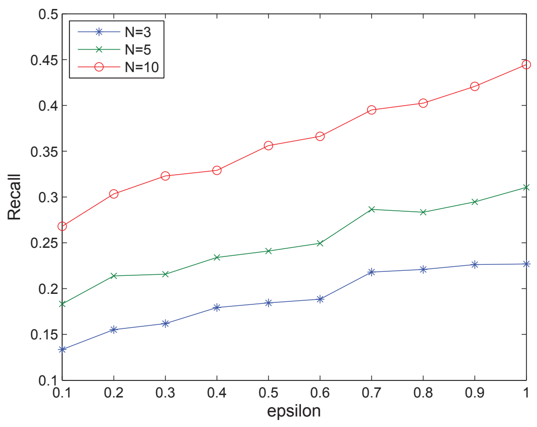
4.5.4 Impact of Privacy Budget

In the context of *differential privacy*, privacy budget ϵ serves as a key parameter in determining the privacy level. According to the literature [22], the lower ϵ represents a higher privacy level. The budget $\epsilon \leq 1$ would be suitable for privacy preserving purposes. To achieve a comprehensive examination of *PriTop*, we evaluate the performance of recommendation under diverse privacy levels on a variety of datasets.

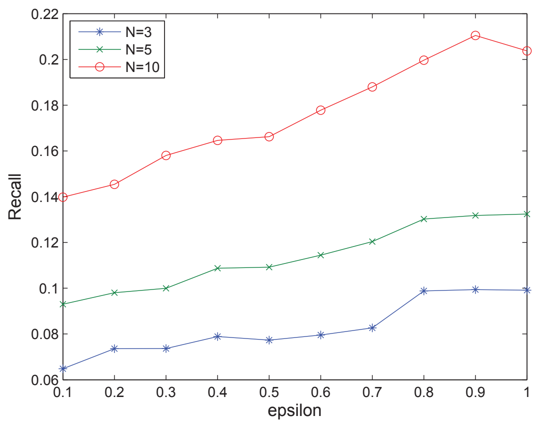
Fig. 4.4 shows the *recall* on the four datasets. It presents the recommendation performance achieved by *PriTop* in different N when the privacy budget ϵ varies from 0.1 to 1 with a step of 0.1. It is clear the *recall* of tag recommendations is significantly affected by the required privacy budget. The *recall* increases as ϵ increases. For example, as plotted in Fig. 4.4a on the *Del.icio.us* dataset, when $N = 10$, *PriTop*



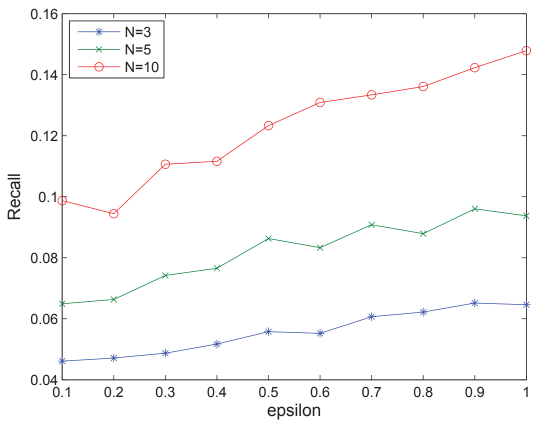
(a) *Del.icio.us*



(b) *MovieLens*

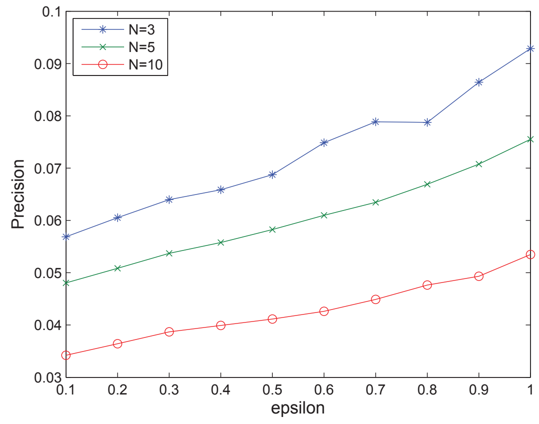


(c) *Last.fm*

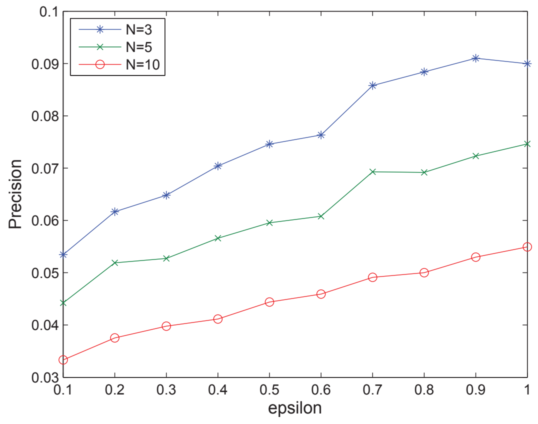


(d) *Bibsonomy*

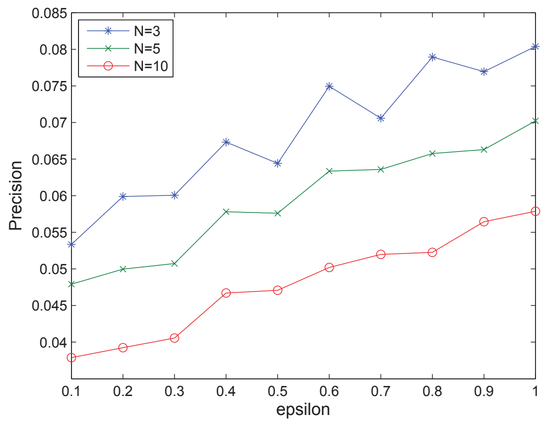
Figure 4.4: *FolkRank* Recall Result



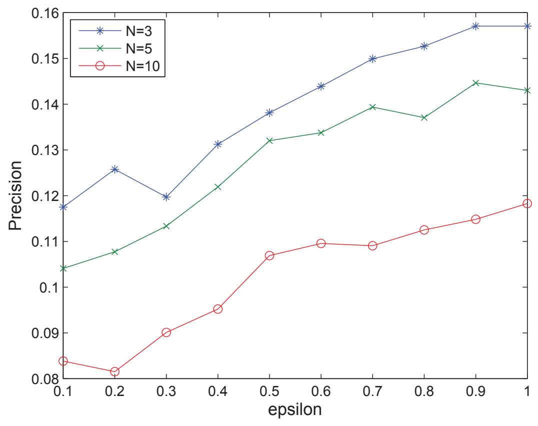
(a) *Del.icio.us*



(b) *MovieLens*



(c) *Last.fm*



(d) *Bibsonomy*

Figure 4.5: *FolkRank* Precision Result

achieves a *recall* at 0.1538 with $\epsilon = 0.1$ and 0.2408 with $\epsilon = 1$. In the *Movielens* dataset, as shown in Fig. 4.4b, the *recall* increases to 40.90% when ϵ increases from 0.1 to 1. Similar trends are observed when N is fixed to other values on other datasets, as shown in Fig. 4.4c and 4.4d. The reason is the privacy and utility issues are two opposite components of the datasets. We have to sacrifice the utility to obtain the privacy, therefore our purpose is to obtain optimal utility when fixing the privacy to an acceptable level.

PriTop serves as an effective method to obtain an acceptable trade-off between utility and privacy. For example, when $\epsilon = 0.7$, *PriTop* obtains a good *recall* performance on the *MovieLens* and *Bibsonomy* datasets, while *De.licio.us* and *Last.fm* obtains a good performance when $\epsilon = 0.8$. The results show *PriTop* can retain an acceptable utility of the dataset while achieving the fixed privacy level. Moreover, Fig. 4.5 shows precision results. Similarly, *PriTop* achieves better performance when ϵ takes larger values, thus confirming the proposed algorithm can obtain a trade-off between utility and privacy.

In general, these results on a real tagging recommender system confirms the practical effectiveness of the proposed *PriTop* algorithm on tag recommendations.

4.6 Summary

This chapter proposes an effective privacy tagging release algorithm for recommendation purposes and makes the following contributions:

- A private tagging release algorithm is proposed to protect users from being re-identified in a tagging dataset by adversaries.

- A private topic model is designed to reduce the magnitude of noise by shrinking the randomization domain. In addition, two successive private operations, *Private Weight Perturbation* and *Private Tag Selection*, are applied to achieve the privacy purpose.

- A better trade-off between privacy and utility is obtained by taking advantage of the differentially private composition properties. We provide theoretical analysis and experimental results to demonstrate the effectiveness.

These contributions provide a practical way to apply a rigid privacy notion to a tagging recommender system without high utility costs. The experimental results also show the robustness and effectiveness of the proposed *PriTop* algorithm.

Chapter 5

Coupled Differential Privacy for Non-IID Datasets

5.1 Introduction

With advances in social networks, data collection and storage technology, research topics such as *Privacy Preserving Data Mining* (PPDM) and *Privacy Preserving Data Release* (PPDR) have attracted significant attention. While the early definition of privacy is vague and difficult to theoretically prove, the notion of *differential privacy* has drawn considerable research interest [23]. The motivation of *differential privacy* is to release aggregate information without compromising the privacy of individual respondents. Over the years, *differential privacy* has become an important privacy model that can help to construct diverse PPDM and PPDR frameworks [23].

Although *differential privacy* has been widely accepted in the PPDM and PPDR research community, previous work has mainly focused on independent datasets which assumes all records were sampled from a universe independently. Despite this, a real-world dataset often exhibits strong coupling relations: some records are often coupled with each other, and this may disclose more information than expected. For example, differential privacy ensures that deleting a user will not affect the aggregated result. However, in a social network dataset, users are always interacting with other users, and this kind of relationship may provide helpful information for identifying those deleted users. Another example assumes members in the same family may have a high probability of catching the same infectious disease. If an adversary knows one person gets the flu, he has a high probability of inferring the health of this person's family. We refer to this relationship as *coupled information*, and the involved records related to each other are *coupled records*. An adversary with knowledge on the coupled information will have a higher chance of obtaining private information [44], and violating the definition of differential privacy. Hence, how to preserve rigorous differential privacy in a coupled dataset is an emerging issue that needs to be addressed.

Over the last decade, limited research has been concerned with coupled differential privacy. A pioneer study by Kifer et al. [44], confirmed that if coupled records are ignored, the released data will have a lower than expected privacy guarantee. Their successive paper proposed a new privacy definition named *Pufferfish* [45], which takes the coupled records into consideration, but it does not meet the requirement of differential privacy. Chen et al. [18] dealt with the coupled problem in social networks by multiplying the original sensitivity with the number of coupled records. This straightforward method was not optimal because it introduced a large amount of noise into the output, that overwhelmed the true answer and demolished the utility of the dataset. Hence, a major research barrier in coupled differential privacy is that the coupled dataset can provide extra information to the adversary, which can not be modeled by the traditional mechanism. In such a situation, satisfying the definition of differential privacy is a more complicated task.

As advances in coupled data analysis are made, especially with recent developments in the research of *non-iid* data [14], it is now possible to overcome the research barrier mentioned above. The coupled information can be modeled by functions or parameters that can be further defined as background information in the differential privacy mechanism. For example, Cao et al. [14] utilized the time interval and correlation analysis to identify coupled records and model coupled information by inter-behavior functions. This solution can help tackle the research barrier by incorporating the modeled information to the differential privacy mechanism. However, there are still three main challenges with this approach:

- The first challenge is how to identify and represent coupled records. Records are often correlated in terms of certain relationships that are not obvious. A deep exploration of the relationship is necessary to understand which records are correlated and how they interact with others.

- The second challenge lies in the fact that if we just increase the sensitivity by multiplying it with the number of coupled records, the new sensitivity will be large, especially when lots of records couple with each other. To achieve a rigorous privacy guarantee, a large magnitude of noise has to be added, and this will significantly decrease the utility of the coupled dataset.

- The third challenge occurs when answering a large number of queries. When the number of queries is large, the privacy budget has to be divided into many small parts, which increases the noise for each query. This problem is more serious in a coupled dataset because the more queries that need to be answered, the more coupled records that will be involved, and the larger the amount of noise that will be introduced.

All these challenges imply coupled information should not be incorporated into differential privacy in a straight forward manner, and a novel mechanism is in high demand. For the first challenge, we observe most records are only partially coupled. In other words, deleting a record may have a different impact on other records. We define the impact as the *coupled degree* and take advantage of diverse coupled degrees to calibrate noise. Based on this, we propose the notion of *coupled sensitivity*, which introduces less noise than the traditional *global sensitivity*. The second challenge can be alleviated by saving the privacy budgets in the data releasing mechanism. If some queries can be answered by random or median values, the privacy budget can be saved and the total noise will decrease significantly. Based on these observations, we propose the solution to coupled differential privacy. The contributions in this thesis can be summarized as follows:

- First, the problem of coupled differential privacy is theoretically analyzed, together with its applications and challenges. This provides a clear problem definition, and explains the significance of exploring differential privacy for coupled datasets.
- Second, we propose the notion of *coupled sensitivity* based on the coupled degrees among records that can help decrease the noise magnitude in differential privacy.

- Third, we design a coupled iteration mechanism to answer a large group of queries. This mechanism constructs a dataset sequence to answer all queries by iteratively updating the datasets. The major advantage is to save privacy budgets and decrease the noise for each query.

The rest of this chapter is organized as follows. We present related work in Section 5.1.1, and provide the problem statement in Section 5.2. Section 4 discusses the coupled dataset analysis together with the coupled iteration mechanism in Section 5. The theoretical privacy and utility analysis of the mechanism is proposed in Section 5.5. Section 5.6 presents experimental results, followed by the conclusion in Section 5.7.

5.1.1 Related Work

Coupled Differential Privacy

In the past decade, a growing body of literature has been published on differential privacy from the PPDR and PPDM community. Most existing work typically assumes the dataset consists of independent records. However, in real world applications, records are often coupled with each other. Coupled differential privacy has emerged as a crucial issue to be tackled. Currently, only limited effort appears to have been made along these lines. Kifer et al. [44] was the first to argue that differential privacy without considering correlation between records will decrease the privacy guarantee on the coupled dataset. For example, suppose a record r has influence on a set of other records, and this set of records will provide evidence on r even though record r is deleted from the dataset. In this scenario, traditional differential privacy fails to provide sufficient privacy as it claims. To deal with this problem, their successive paper proposed a new privacy framework, *Pufferfish* [45], which allows application domain experts to add extra data relationships to develop a customized privacy definition. However, the *Pufferfish* framework does not satisfy differential privacy, and the privacy guarantee for coupled datasets still calls for further investigation.

Chen et al. [18] considered the social network as a coupled dataset and dealt with the coupled problem by multiplying the *global sensitivity* with the number of coupled records. They defined the maximal number of coupled records as $k = 25$, however, this naive method is not optimal because it introduces a large amount of noise into the output that overwhelms the true answer and demolishes the utility of the dataset.

Consequently, when addressing the privacy issue in a coupled dataset, the essential problem is how to model the extra background information introduced by coupled records. This problems can be addressed with the development of coupled data analysis.

Coupled Data Analysis

Several studies on coupled data analysis have attempted to identify and model coupled information. Coupled information can be identified by coupled analysis including *time interval analysis*, *attribute analysis*, or *similarity analysis*. Cao et al. [15] presented a coupled *Hidden Markov* detection model to detect abnormal group-based trading behaviors. They defined a time interval and assumed behaviors falling into the same interval as coupled behaviors. Song et al. [84] proposed a hybrid coupling framework, which applied some particular attributes to identify relationships among records. Zhang et al. [99] identified the network traffic coupled record using an IP address. They presented a correlated network traffic classification algorithm.

Coupled information can be modeled in varies ways. Cao et al. [15] modeled coupled information using the inter-couple and intra-couple behavior functions. These functions were adopted in the coupled framework to represent the coupled degree between behaviors. Zhou et al. [100] mapped the coupled records to an undirected graph and proposed the multi-instance learning algorithm.

These approaches help to model background information for differential privacy. An advanced differential privacy releasing mechanism will be proposed with the aim of guaranteeing a sufficient privacy level as well as decreasing extra noise.

Summary

Even current research on privacy preserving fails to provide practical solutions to coupled differential privacy due to a lack of exploitation on background information modeling. However, with the development of coupled dataset analysis, this chapter aims to fill this void by incorporating coupled analysis with an advanced differential privacy releasing mechanism that aims to preserve comprehensive privacy for coupled datasets. More specifically, we attempt to address the following research issues:

- How to identify coupled records in a dataset?
- How to calibrate the sensitivity for coupled records?
- How to design a coupled data releasing mechanism?

The solution will be investigated in the following sections.

Table 5.1: Frequency Dataset D

Attribute	Count
A	2
B	100
C	200

5.2 Problem Statement

In this section, we will illustrate the coupling problem using a simplified example and then formalize the problem statement.

5.2.1 An Example: Coupled Records in a Dataset

Suppose we want to publish a dataset D with n records. To simplify the example, we assume there is only one attribute in D and its values are A, B and C. Dataset D can then be easily transfered to frequency dataset x in Table 5.2, where the *Attribute* column stores the attribute values and the *Count* column represents the number of records with each value. The target of privacy preserving is to hide the true count in x.

To preserve ϵ-differential privacy, the randomization mechanism $\mathcal{M}$ will add independent noise to the count. Since deleting a record will impact the count number at most by 1, the sensitivity of the count query is 1, and the independent noise will be sampled from the *Laplace* distribution $Laplace(\frac{1}{\epsilon})$.

This process works well when records are sampled independently from domain $\mathcal{X}$. However, if some records are coupled with each other, traditional differential privacy may under estimate the privacy risk [44]. For example, let the frequency dataset x in Table. 5.2 represent a medical report in which *Attribute* represents the address and *Count* denotes the number of patients who have the Flu. Suppose a patient named *Alice* and her 9 immediate family members are living at the same address B. When *Alice* contracts the Flu, the entire family will also be infected. In this case, deleting the record of *Alice* in address B will impact 9 other records, and the count of address B will change to 90 (*Alice* got the Flu) or remain 100 (*Alice* is healthy). Suppose the noisy count returns 99 and the noise is sampled from $Laplace(\frac{1}{\epsilon})$. This means there is high probability *Alice* is healthy because the query answer is close to 100. Specifically, the answer 99 is $e^{10\epsilon}$ times more likely than the probability of *Alice* to get the Flu. Compared to the independent records with privacy bounded in e^{ϵ}, coupled records have a probability of 10 times more likely to be disclosed. In this instance, traditional differential privacy seriously mismatches the reality for coupled records.

We define the problem as a *coupled differential privacy* problem. To deal with this, one possible solution is to design a new sensitivity measurement based on the relationship between records. A naive way to measure the sensitivity is to multiply global sensitivity with the number of coupled records. In the above mentioned example, while deleting *Alice* will impact at most 10 records, the sensitivity is re-measured as 1×10, and the noise will be sampled from $Laplace(\frac{10}{\epsilon})$.

This naive sensitivity measurement can be extended to differential scenarios. For instance, if A, B, C in Table 5.2 are linear dependent, that is $A + B = C$, deleting a record in A will eliminate 1 count in A and 1 count in C at the most, and sensitivity will be measured to 2. If we have $A * B = C$, deleting a record in A will at most change the count of 100 in C, so the sensitivity is measured as $\max(count(A), count(B))$. It is obvious that in some cases, sensitivities will be very high, leading to considerable redundance noise. This naive solution is not optimal in coupled differential privacy. How to define new sensitivity in a proper way is a problem of critical importance.

In summary, a major disadvantage of traditional differential privacy is overlooking the relationship between records, which means the query result leaks more information than is allowed. If we deal with the problem by simply multiplying the number of coupled records to the sensitivity, the query result will contain redundant noise and damages the utility of the dataset. Consequently, a sophisticated solution to the coupled differential privacy problem is urgently needed.

5.2.2 Coupled Differential Privacy Problem

Before dealing with the coupled differential privacy problem, we define related notions and terminology. For simplicity, *coupled*, *relationship* and *correlation* are interchangeable in this chapter.

If a record $r_i \in D$ is correlated to other $k-1$ records, we call this group of k ($k \leq |D|$) records *coupled records*, which is denoted by a set $\mathbf{r_i} = \{r_j \in D | \text{all } r_j \text{ are correlated to } r_i\}$. Dataset D is then referred to as a *coupled dataset*. The i.i.d dataset is a special case of a coupled dataset, in which $k = 1$. k varies from different datasets and is independent to queries.

If a coupled dataset D contains d attributes, it is more convenient to map D to a histogram x over domain $\mathcal{X}$. Each bin b represents the combination of attributes, and the number of bins is denoted by N. The frequencies in a histogram is the fraction of the count of bins, which are denoted by $x(b_i), (i \in N)$. For example, Table 5.2 is actually a histogram with bins A, B and C, whose frequency $x(A) = 0.0066$,

$x(B) = 0.3311$ and $x(C) = 0.6623$, respectively. Formally, we define the histogram representation as follows:

Definition 11 (Histogram Representation). A dataset D can be represented by a histogram x in a domain $\mathcal{X}$: $x \in N^{|\mathcal{X}|}$, Two datasets D and D' are defined as *neighboring datasets* if and only if their corresponding histograms x and x' satisfy $||x - x'||_1 \leq 1$.

Another important notion is *coupled degree*. The naive multiple method assumes deleting a record will definitely change other records in a coupled dataset. However, most records are only partially coupled, and deleting a record may have a different impacts on other records. We define these impacts as the *coupled degree* of records.

Definition 12 (Coupled Degree). Suppose two records r_i and r_j are coupled to each other. This means the relationship between them is represented by the *coupled degree* $\delta_{ij} \in [-1, 1]$ and $|\delta_{ij}| \geq \delta_0$, where δ_0 is the threshold of the coupled degree.

Corollary 1. If $\delta_{ij} < 0$, r_i and r_j have a *negative coupling*; if $\delta_{ij} > 0$, they have a *positive coupling*; $\delta = 0$ indicates no relationship. If $|\delta_{ij}| = 1$, we say record r_i and r_j are *fully coupled* with each other.

The coupled degree represents the impact of a record on another record. The smaller absolute value of δ_{ij} illustrates a *weak* coupling, and indicates that deleting r_i will have a low possibility of impacting r_j. When δ_{ij} is closed to 1 or -1, the coupling is strong, and deleting r_i will greatly impact r_j. However, we should note that in real world applications, few records are fully coupled, and this observation can be useful in our proposed method.

From the perspective coupled data analysis, it is possible to list all relationships between records and maintain a *coupled degree matrix* Δ, in which $\delta \in \Delta$.

$$\Delta = \begin{pmatrix} \delta_{11} & \delta_{12} & ... & \delta_{1n} \\ \delta_{21} & \delta_{22} & ... & \delta_{2n} \\ ... & ... & ... & ... \\ \delta_{n1} & \delta_{n2} & ... & \delta_{nn} \end{pmatrix} \tag{5.2.1}$$

Here are four properties of Δ: 1) It is *symmetrical* with $\delta_{ij} = \delta_{ji}$, which indicates the relationship between two records is irrelevant to their sequence; 2) Elements on the diagonal are equal to 1, which implies every record is fully coupled with itself; 3) A threshold δ_0 is defined to filter the weak coupled degree. In Δ, $|\delta_{ij}| \geq \delta_0$. If $|\delta_{ij}| < \delta_0$, δ_{ij} is set to zero; 4) It is *sparse*. Only parts of records are coupled with each other.

The above terms and coupled degree matrix will help solve the coupled differential privacy problem. In the following sections, we will propose our solution and discuss its privacy and utility.

5.2.3 Research Issues and Challenges

Privacy preserving on a coupled dataset is challenging because of its special dataset structure and corresponding privacy requirement. Introducing differential privacy to a coupled dataset, brings three major challenges.

- *How to identify coupled records in a dateset*?

  It is often hard to identify coupled records and coupled degree δ. Different types of datasets may have various ways to couple with their records. Moreover, several records may mix together and have exponential possible relationships, thus making coupled analysis very complex. In sub-section 5.3.1, we will categorize possible coupled record analysis methods. These methods will be applied to several datasets in the experiment section.

- *How to calibrate sensitivity for coupled records*?

  Traditional *global sensitivity* may not be suitable for coupled datasets due to large noise. In our previous Flu example, *global sensitivity* introduces 10 times larger noise to the count output in a coupled dataset. In addition, we cannot use *local sensitivity* because it still only relates to an individual record without considering coupling information. In sub-section 5.3.2, we will define *coupled sensitivity*, which retains relationships between records while decreasing noise.

- *How to re-design the differential privacy mechanism*?

  Even *coupled sensitivity* can significantly decrease noise compared to large noise when answering a large set of queries for *global sensitivity.* When dealing with the coupled dataset, the traditional mechanism may not be suitable for a coupled dataset. A new mechanism is expected to satisfy differential privacy, as well as retain sufficient utility for future applications. In sub-section 5.4, we propose an iteration based data releasing mechanism to deal with this problem.

5.3 Coupled Dataset Analysis

This section presents the definition of *coupled sensitivity.* Specifically, we identify coupled records, and then use the coupled degree to develop the *couple sensitivity* to guarantee ϵ-differential privacy.

5.3.1 Coupled Analysis

Coupled analysis is carried out to generate the coupled degree matrix Δ for a coupled dataset. This can be done in various ways depending on the background knowledge of the curator or the characteristics of the dataset. Typical methods can be conceptualized into two categories.

The first type of coupled analysis assumes the curator or the attacker obtained the background knowledge in advance. The coupled degree matrix Δ is pre-defined as background knowledge. Take Table 5.2 as an example. The curator or attacker discover there are full coupling relationships among A, B and C, e.g. $A + B = C$ or $A * B = C$. Δ can then be created according to the background information. Identifying a full coupling relationship among records is relatively easy. But for some weak couplings, they needs further domain knowledge or determination by an expert.

Another type of coupled analysis can be carried out without any direct background knowledge. The coupled degree will be defined in various ways.

1. *Attribute analysis.* This utilizes certain particular attributes to discover the relationships among records. When the values of these attributes are the same or similar to each other, records with those values are considered as coupled records. For example, the `address` attribute can be used to determine family members in a survey dataset. In a network traffic dataset, the `IP-address` attribute can help identify traffic coming from the same host. Moreover, the similarity in attribute values can be adopted to measure the coupled degree; a high similarity implies a strong coupling. This method can identify coupled records effectively and accurately. However, it can hardly be implemented when no attribute is available to disclose the relationship.

2. *Time interval analysis.* This method pre-defines the size of a time interval to identify the correlation in the stream dataset. Records falling into the same interval are considered as coupled records. For instance, Cao et al. [15] aggregated the behaviours within time intervals and modeled the coupling between these activities. This method can figure out the multiple records mixed together but is only suitable for a time related dataset.

3. *Pearson Correlation analysis.* If the dataset contains no proper attribute or time information to identify the coupled information, the *Pearson Correlation Coefficient* is an efficient way to discover coupled records. It extracts all or parts of an attribute in a coupled dataset and calculates the *Pearson Correlation Coefficient* between records. By defining the threshold δ_0, the coupled degree matrix can be generated according to the correlation coefficient. Other correlation or distance measurements can also be applied. For example, Song et al. [84] applied KL divergence to measure the coupled degree between records. However, this type of method can only identify the linear correlation between records.

Other strategies also exist for coupled analysis. However, no matter what methods are applied, the target is to define the coupled degree matrix Δ, which plays an essential role in coupled differential privacy.

5.3.2 Coupled Sensitivity

Before proposing the definition of *coupled sensitivity*, we first analyze the source of redundant noise from *global sensitivity*. Traditional *global sensitivity* will result in redundant noise derived from both records and queries. For the record, as analyzed earlier, the traditional method assumes records are fully coupled with each other, and therefore, it just multiplies the *global sensitivity* with the maximal number of coupled records leading to large noise. For a query, the traditional method uses a fixed *global sensitive* without considering the prosperity of different queries. In actual fact, only some of the responding records are coupled with others, and we only need to consider the coupled information within these responding records. Hence, sensitivity should be adaptive for both the coupled record and the query.

Based on this observation, we define *coupled sensitivity*, and take both record and query into consideration. We firstly introduce the notion of *record sensitivity* relating to the coupled degree of each record. Based on this notion, we then propose the *coupled sensitivity* associated with the query.

Definition 13 (Record Sensitivity)*.* For a given Δ and a query Q, the record sensitivity of r_i is

$$CS_i = \sum_{j=0}^{n} |\delta_{ij}|(||Q(D^j) - Q(D^{-j})||_1) \tag{5.3.1}$$

where $\delta_{ij} \in \Delta$.

The *record sensitivity* measures the effect on all records in D when deleting a record r_i. $\delta_{ij} \in \Delta$ estimates the coupled degree between records r_i and $r_j \in D$. This notion combines the number of coupled records and the coupled degree together. If D is an independent dataset, CS_i is equal to the global sensitivity.

Definition 14 (Coupled Sensitivity)*.* For a query Q, *coupled sensitivity* is determined by the maximal record sensitivity,

$$CS_q = \max_{i \in q}(CS_i), \tag{5.3.2}$$

where q is a record set of all records responding to a query Q.

Coupled sensitivity is related to a query Q. It lists all the records q responding to Q and selects the maximal *record sensitivity* as the *coupled sensitivity.* The advantage of the measurement is that when a query only covers the independent or weak coupled record, coupled sensitivity will not bring extra noise.

After defining *coupled sensitivity* for each query Q, the noisy answer will eventually be calibrated by the following equation:

$$\hat{Q}(D) = Q(D) + Laplace(\frac{CS_q}{\epsilon}). \tag{5.3.3}$$

We can observe *coupled sensitivity* CS_q will be smaller than *global sensitivity* GS and *local sensitivity* LS. Both assume each record is fully coupled with each other and the coupled degree is also ignored.

Lemma 3. For a query Q, *coupled sensitivity* is equal to or less than the *global sensitivity* GS and the *local sensitivity* LS.

Proof. Suppose there are at most k coupled records in a dataset D, then we have $GS = k \cdot \max_{D,D'}(||Q(D) - Q(D')||_1)$, and $CS_i = \sum_{j=1}^{n} \delta_{ij}(||Q(D^j) - Q(D^{-j})||_1)$. Because at most k records are coupled, we have $\sum_{j=1}^{n} \delta_{ij} = \sum_{j=1}^{k} \delta_{ij} \leq k$. As $||Q(D^j) - Q(D^{-j})||_1) \leq \max_{D,D'}(||Q(D) - Q(D')||_1)$, we have $CS_i \leq GS$. As any CS_i are less or equal to GS, for a query Q, we have $CS_q \leq GS$

For the *local sensitivity* $LS = k \cdot \max_{D'}(||Q(D) - Q(D')||_1)$, we also have $||Q(D^j) - Q(D^{-j})||_1) \leq \max_{D'}(||Q(D) - Q(D')||_1)$ and $\sum_{j=1}^{n} \delta_{ij} = \sum_{j=1}^{k} \delta_{ij} \leq k$, then we have $CS_q \leq LS$ $\square$

Coupled sensitivity CS can be used in various types of data releasing mechanisms. If records in the dataset are independent, the CS will be equal to the global sensitivity, while for the coupled dataset, the CS will introduce less noise than GS and LS.

5.4 Coupled Iteration Mechanism

Even though *coupled sensitivity* decreases the noise compared with *global sensitivity*, when dealing with a large number of queries, the answers still have high noise because the privacy budget has to be divided into several small parts. This is especially so when the records are strongly coupled with others and the noise is significantly higher than the independent dataset. To tackle the problem, an iterative-based mechanism will be adopted to limit the noise in the query answer.

The iterative-based mechanism was first proposed by Hardt et al. [34] who constructed a dataset sequence to answer all queries by iteratively updating the datasets. When a given query witnesses a significant difference between the current dataset and the true dataset, the mechanism updates the current dataset in the next iteration [34]. The main advantage of this mechanism is that it can save the privacy budget and decrease the noise when confronting lots of queries. Hence, it will be suitable for data releasing in the coupled dataset.

In this section, we propose a **C**oupled **I**teration **M**echanism (*CIM*) to answer a set of queries on the coupled dataset. We first present an overview of the algorithm and then provide details of its operations.

5.4.1 Overview of Coupled Iteration Mechanism

The *CIM* aims to release the results of a set of queries by iteratively updating the dataset under the notion of *differential privacy*. In this procedure, a dataset is represented by a histogram x with length N. Let t be the round index, and the histogram be represented by x_t at the end of round t. We are given a query set $\mathcal{Q}$ and select a Q_t in each round t. We recognize that a_t denotes the true answer and $\hat{a}_t$ denotes the noisy answer:

$$a_t = Q_t(x). \tag{5.4.1}$$

$$\hat{a}_t = Q_t(x) + Laplace(\frac{CS_{q_t}}{\epsilon}). \tag{5.4.2}$$

We use $\hat{d}_t$ to denote the difference between the true answer given by x_{t-1} and the noisy answer from x_t:

$$\hat{d}_t = Q_t(x_{t-1}) - \hat{a}_t. \tag{5.4.3}$$

This is utilized to control the update round in each iteration. At a high level, *CIM* maintains a sequence of histogram x_0, x_1,..., x_t, which gives increasing approximation to the original dataset x.

The mechanism is shown in Algorithm. 6. Firstly, the privacy budget is divided into several parts and the histogram is initialized as the uniform distribution x_0. In each round t, we select a query $Q_t \in \mathcal{Q}$, using x_t to generate the answer $a_t = Q_t(x_t)$ and the noise answer $\hat{a}_t$. The distance $\hat{d}_t$ between the query Q_t on x_{t-1} and the noisy answer $\hat{a}_t$ is computed. If $|\hat{d}_t|$ is less than a threshold T, the x_{t-1} is considered to be a good approximation of x on query Q_t. We will release the $Q_t(x_{t-1})$ directly and put the x_{t-1} into the next iteration. If the distance is larger than the threshold, the histogram x_{t-1} will be improved in this round. We will release $\hat{a}_t$ and use an coupled updating function U to generate the new histogram x_t.

The *CIM* aims to answer a large group of queries with limited privacy budgets on a coupled dataset. In summary, this mechanism has the following features:

- First, it takes the relationship between records into consideration. It applies not only *coupled sensitivity*, but more importantly, it develops a coupled update function to improve the histogram in each iteration.

- Second, it decreases the total amount of noise. The *CIM* maintains a histogram sequence to answer a set of queries $\mathcal{Q}$, rather than using a single histogram to answer all queries. One histogram in the sequence roughly corresponds to one query in $\mathcal{Q}$. This way, each histogram can approximate the close answer to the true answer.

- Finally, more queries can be answered than the traditional mechanism with the same privacy budget. Only the update steps will consume the privacy budget. Alg. 6 indicates that even for a very large set of queries, the number of update rounds is still less than the total number of queries.

5.4.2 Coupled Update Function

This section defines a coupled update function U in the histogram context. For a histogram x_{t-1}, the function U firstly identifies all responded records $r \in q_t$. For each record in q_t, all coupled records are listed and denoted as superset $\mathbf{q_t}$. The update function U then identifies a set of bins $\mathbf{b}$ that contain $\mathbf{q_t}$ and re-arranges the frequency of each bin in $\mathbf{b}$ according to Eq. 4. The final frequency of the x_t will be normalized so they sum to 1.

Definition 15 (Coupled Update Function). Let x_0, x_2, ..., x_t be a histogram sequence, and function U is defined as a coupled update function if it satisfies $x_t = U(x_{t-1})$. The U is defined as:

$$x_t(b_i) = x_{t-1}(b_i) \cdot \exp(-\eta \cdot \delta_{q_t} \cdot y_t(x_{t-1})), \tag{5.4.4}$$

where $y_t(x_{t-1}) = Q_t(x_{t-1})$ if $\hat{d} > 0$ and otherwise, $y_t(x_{t-1}) = 1 - Q_t(x_{t-1})$. η is an update parameter associated with the number of maximal update rounds.

Algorithm 6 Coupled Iteration Mechanism

```
Require: $x$, $\epsilon$, $\mathcal{Q} = Q_1, ..., Q_L$, $L$, $\Delta$, $T$.
Ensure: $\mathcal{Q}(x)$
    1. $\epsilon_0 = \frac{\epsilon \eta^2 \delta_0^2}{\log N}$;
    for $i = 1, ..., N$ do
        2. $x_0(b_i) = 1/N$;
    end for
    for each round $t \leftarrow 1...L$ do
        3. select a query $Q_t$;
        4. sample $\lambda_t$ from $Laplace(CS_{q_t}/\epsilon_0)$;
        5. compute the noise answer $\hat{a}_t = Q_t(x) + \lambda_t$;
        6. compute $\hat{d}_t = Q_t(x_{t-1}) - \hat{a}_t$;
        if $|\hat{d}_t| \leq T$ then
            7. $x_t = x_{t-1}$, output $Q_t(x_{t-1})$;
            8. continue;
        else
            9. $x_t = U(x_{t-1})$, output $\hat{a}_t$;
        end if
    end for
```

Algorithm 7 Coupled Update Function

```
Require: $x_{t-1}$,$\hat{d}$, $Q_t$, $\Delta$, $\eta$.
Ensure: $x_t$.
    1. Identifying $q_t$;
    2. Identifying the coupled record set $\mathbf{q_t}$;
    3. Identifying the bin set $\mathbf{b}$ contains $\mathbf{q_t}$;
    for For each bin $b_i \in \mathbf{b}$ do
        4. Update the frequency of $x(b_i)$;
    end for
    5. Normalization of $x_t$
```

The coupled update function is based on the intuition that if the answer derived from x_{t-1} is too small compared with the true answer, the frequency of the relative bins will be enhanced. Otherwise, we will decrease the frequency if the answer is too large,.

5.4.3 Parameters Discussion

This section discuss the estimation of parameters in *CIM*. As mentioned earlier, only the update round consumes the privacy budgets. To measure the parameters T and η, we need to estimate the maximal number of update rounds u_{max} and the possible number of update rounds $u_{\mathcal{Q}}$. The u_{max} helps determine the privacy budgets in each iteration. In addition, the $u_{\mathcal{Q}}$ for $\mathcal{Q}$ is related to the accuracy.

Firstly, we measure the maximal number of update rounds u_{max}. Given a dataset x, the u_{max} can be measured based on the following lemma.

Lemma 4. Given a histogram x with length N, the u_{max} for coupled update function

U, defined by Eq. is

$$u_{max} = \frac{\log N}{\eta^2 \cdot \delta_0^2} \tag{5.4.5}$$

PROOF. Give the original histogram x and the initial update histogram x_0, the *CIM* will update the x_0 in each round t until $x_t = x$. The u_{max} depends on how many steps that x_0 can be transferred to x. We follow the update strategy of Hardt el al. [34], who define the distance between x_0 and x in terms of *relative potential*:

$$\phi_t = RE(x||x_t) = \sum x(b) \log(\frac{x(b)}{x_t(b)}) \tag{5.4.6}$$

Based on Eq. 5.4.6, we have $\phi_0 \leq \log N$. When ϕ_t drops to zero, the update will be terminated and x_t=x. According to Hardt et al. [34], the potential drops in each round is at least $\eta^2\delta_0^2$, therefore there are at most $\frac{\log N}{\eta^2 \cdot \delta_0^2}$ rounds in the *CIM*. □

The u_{max} is utilized to determine the privacy budget ϵ_0 in each round. We have Eq. 5.4.7:

$$\epsilon_0 = \frac{\eta^2\delta^2\epsilon}{\log N} \tag{5.4.7}$$

Compared to the traditional data releasing mechanism, which divides the privacy budget ϵ according to the number of queries, we can easily demonstrate that $\epsilon_0 \geq \epsilon/|\mathcal{Q}|$.

Lemma 4 also indicates u_{max} is associated with parameter η and the couple parameter threshold δ_0. If we want to successfully answer more queries, we can choose a smaller η to allow more rounds. However, this will lead to larger noise in each query answer because the privacy budget ϵ_0 in each round will also be diminished.

Secondly, we estimate the possible number of update rounds $u_{\mathcal{Q}}$ for a query set $\mathcal{Q}$. Let the probability of updating be ρ_1 and the probability of non-updating be ρ_2. We will then have the follow Lemma:

Lemma 5. When both the privacy budget ϵ_0 in each round and the parameter T are fixed, the probability of the update will be

$$\rho_1 = \exp(\frac{-\epsilon_0|T-\alpha|}{CS_q}), \tag{5.4.8}$$

and the probability of the non-update will be

$$\rho_2 = 1 - \exp(\frac{-\epsilon_0|T-\alpha|}{CS_q}). \tag{5.4.9}$$

where α bounds the accuracy of the *CIM*.

Corollary 2. Given a query set $\mathcal{Q}$, the $u_{\mathcal{Q}}$ will satisfy Eq. 5.4.10

$$u_{\mathcal{Q}} = |Q|\exp(\frac{-\epsilon_0|T-\alpha|}{CS_q}) \tag{5.4.10}$$

Proof. Suppose we have $|\mathcal{Q}|$ queries, and altogether $|\mathcal{Q}|$ rounds. The probability of the update is $Pr(|\hat{d}_t|> T)$. We have

$$Pr(|\hat{d}_t|> T) = Pr(|Q_t(x) + \lambda_t - Q_t(x_{t-1})|> T)$$

. Let $|Q_t(x) - Q_t(x_{t-1})| \leq \alpha$, and λ_t be sampled from *Laplace*(ϵ_0/CS_q) according to the property of *Laplace* distribution:

$$Pr(|\gamma|> t) = Pr(\gamma > t) + Pr(\gamma < t) = 2\int_t^\infty x\exp(-\frac{x}{\sigma})dx \quad = \tag{5.4.11}$$

$$= exp(-\frac{t}{\sigma}) \tag{5.4.12}$$

Because $\sigma = \frac{CS_q}{\epsilon_0}$, we have

$$Pr(|\hat{d}_t|> T) \leq \exp(\frac{\epsilon_0(\alpha - T)}{CS_q})$$

. If there are $|\mathcal{Q}|$ queries, the algorithm will update at most $|\mathcal{Q}|\exp(\frac{\epsilon_0(\alpha-T)}{CS_q}CS_q)$ rounds. $\square$

Lemma 5 shows the probability of the update is related to T and α. If parameter T is much smaller than α, the update probability will be very high and the noise will increase simultaneously which will affect the accuracy of the answer. However, if T is very large, even though we decrease the number of update rounds, but the output answer is always far away from the true answer, which also decreases the accuracy, we can conclude the accuracy of *CIM* is related to T. In Section 5.6, we will use the experiment to demonstrate the trade-off between T and the accuracy of *CIM*.

5.5 Mechanism Analysis

The proposed *CIM* aims to obtain an acceptable utility with a fixed ϵ differentially privacy budget. In this section, we will first prove the algorithm is satisfied with ϵ-differential privacy, and then analyze the utility cost.

5.5.1 Privacy Analysis

To prove *CIM* is satisfied with *differential privacy*, we first analyze which steps in *CIM* will consume the privacy budget. According to Algorithm 6, we access the histogram and generate a noisy answer in each round. However, the noisy answer is only used to check whether the current histogram is accurate enough to answer the query. In most rounds, we do not release the noisy answer, and therefore we consume no privacy budget. The noisy answer is only released in the update round when the current histogram is not accurate enough. Consequently, the privacy budget is only consumed in the update step and the privacy analysis can be easily limited in the coupled update functions.

We apply useful composite properties of the privacy budget to analyze the privacy guarantee: the *sequential composition* [63]. The *sequential composition* accumulates privacy budget ϵ for each step when a series of private analyses is performed *sequentially* on the dataset. The properties are shown in Lemma 6.

Lemma 6. *Sequential Composition* [63]: Suppose a set of privacy steps $\mathcal{M} = \{\mathcal{M}_1, ...\mathcal{M}_m\}$ are sequentially performed on a dataset, and each $\mathcal{M}_i$ provides ϵ privacy guarantee, the $\mathcal{M}$ will provide $m \cdot \epsilon$-*differential privacy.*

Lemma 6 can be used directly to analyze the privacy guarantee of *CIM*. As mentioned earlier, given a x, we have $u_{max} = \eta^{-2}\delta^{-2}\log N$. The privacy budget ϵ_0 allocated to each round is $\epsilon_0 = \frac{\epsilon\eta^2\delta_0^2}{\log N}$. According to the sequential composition, the released answers for the query set $\mathcal{Q}$ will consume the $\epsilon_0 * u_{\mathcal{Q}}$ privacy budget. Because $u_{\mathcal{Q}} \leq u_{max}$, we have $\epsilon_0 * u_{\mathcal{Q}} \leq \epsilon$. Consequently, the proposed *CIM* algorithm preserves ϵ-*differential privacy.*

5.5.2 Utility Analysis

For the utility analysis, we apply a widely used utility definition in *differential privacy* suggested by Blum et al. [11]:

Definition 16 ((α,β)-accuracy). A mechanism $\mathcal{M}$ is (α,β)-accuracy for a set of queries $\mathcal{Q}$, if: with probability $1-\beta$, for every query $Q \in \mathcal{Q}$ and every histogram x, we have

$$\max_{Q\in\mathcal{Q}} |\hat{Q}(x) - Q(x)| \leq \alpha. \tag{5.5.1}$$

In the *CIM*, the utility is measured by a set of released query answers. Accordingly, we will measure the utility by the maximal distance between the original and the noisy answer.

Definition 17 ((α,β)-accuracy for *CIM*). The *CIM* is (α,β)-accuracy for a set of query $\mathcal{Q}$, if: with probability $1-\beta$, for every query $Q \in \mathcal{Q}$ and every x, we have

$$\max_{Q\in\mathcal{Q},t\in L} |CIM_t(x_t) - Q_t(x)| \leq \alpha, \tag{5.5.2}$$

where $CIM_t(x_t)$ is the output of *CIM* in round t.

Based on the definition, we will demonstrate the *CIM* mechanism is bounded by a certain value α with a high probability.

Theorem 5.5.1. *For any query $Q \in \mathcal{Q}$, for all $\beta > 0$, with probability at least $1-\beta$, the error of* CIM *output is less than α. When*

$$\alpha \geq \frac{CS_q}{2\epsilon_0}(\log\frac{\rho_1\rho_2 L}{\beta}) + \frac{T}{2}$$

the CIM *is satisfied with (α,β)-accuracy.*

Proof. The value of $CIM_t(x_t)$ is determined by the $\hat{d}$, which results in the non-update round or the update round. Both scenarios will be considered in the utility measurement. Let $error_{non-update}$ represents the error introduced by non-update rounds and $error_{update}$ denotes the error introduced by update rounds. According to the union bound, we have

$$Pr(\max_{Q\in\mathcal{Q},t\in L}|CIM_t(x_t) - Q_t(x)| > \alpha) \leq \rho_1 * Pr(error_{non-update} > \alpha) + \rho_2 * Pr(error_{update} > \alpha)$$

If $\hat{d} \leq T$, it will be a non-update round and *CIM* will output $Q_t(x_{t-1})$.

$$error_{non-update} = |CIM(x_t) - Q_t(x)| = |Q_t(x_{t-1}) - Q_t(x)|$$

Because

$$|\hat{d}_t|= |Q_t(x_{t-1}) - \hat{a}_t|\leq T$$

, we have

$$|Q_t(x_{t-1}) - Q_t(x)|\leq T + \lambda_t$$

, where $\lambda_t \sim Laplace(\frac{CS_{q_t}}{\epsilon'})$. According to the property of *Laplace* distribution $Laplace(b)$, we have

$$Pr(error_{non-update} > \alpha) = Pr(\max_{t\in L}|T+\lambda|> \alpha) \leq L * Pr(|T+\lambda|> \alpha)$$

$$\leq L\exp(\frac{-|\alpha - T|\epsilon_0}{CS_q})$$

, where $CS_q = \max_{t\in 1,\ldots,m} CS_{q_t}$.

If $\hat{d} > T$, it will be an update round and *CIM* will output $\hat{a}_t$. We have

$$error_{update} = |CIM(x_t) - Q_t(x)|= |Q_t(x) + \lambda_t - Q_t(x)|= |\lambda_t|$$

. Then we have

$$Pr(error_{update} > \alpha) = Pr(\max_{t\in L}|\lambda|> \alpha) \leq L * Pr(|\lambda|> \alpha)$$

$$\leq L\exp(\frac{-\alpha\epsilon_0}{CS_q})$$

Accordingly,

$$Pr(\max_{Q\in\mathcal{Q},t\in L}|CIM_t(x_t) - Q_t(x)|> \alpha) \leq L\rho_1\exp(\frac{-|\alpha - T|\epsilon_0}{CS_q}) + L\rho_2\exp(\frac{-\alpha\epsilon_0}{CS_q})$$

Let

$$L\rho_1\exp(\frac{-|\alpha - T|\epsilon_0}{CS_q}) + L\rho_2\exp(\frac{-\alpha\epsilon_0}{CS_q}) \leq \beta$$

, we have

$$\rho_1\exp(\frac{-|\alpha - T|\epsilon_0}{CS_q}) + \rho_2\exp(\frac{-\alpha\epsilon_0}{CS_q}) \leq \frac{\beta}{L}$$

$$\Rightarrow \log \rho_1\rho_2 + \frac{(T-2\alpha)\epsilon_0}{CS_q} \leq \log \frac{\beta}{L}$$
$$\Rightarrow \alpha \geq \frac{CS_q}{2\epsilon_0}(\log \frac{\rho_1\rho_2 L}{\beta}) + \frac{T}{2}$$

□

5.6 Experiment and Analysis

This section evaluates the performance of the proposed coupled data releasing mechanism *CIM* and the coupled sensitivity by answering the following questions:

- *How does the CIM mechanism retain the utility with a limited privacy budget?*

  The proposed *CIM* aims to effectively answer a set of queries. In this experiment, we investigate the performance of *CIM* on a set of queries in terms of *Mean Square Error* (MSE) and compare it with the traditional *Laplace* mechanism [25].

- *How does the coupled sensitivity impact on the performance of the data releasing mechanism?*

  In Section 5.3, we proposed *coupled sensitivity* and proved it was better than traditional sensitivities on coupled datasets. Here, we empirically investigated its impact on the utility of the released query answers through comparing the *coupled sensitivity* with *global sensitivity*.

- *How do the parameters impact on the performance of CIM?*

  CIM contains two essential parameters T and η. Parameter T controls the number of update rounds, which affects the performance of the *CIM*. η determines the maximal update rounds, which affects the number of queries *CIM* can answer. In Section 5.5, we theoretically analyze these impacts, and in the following sub-section we use experiments to confirm them.

5.6.1 Datasets and Configuration

The experiments involve six datasets:

`Adult` The `Adult` dataset from the *UCI Machine Learning repository*[1] originally had 48, 842 records and 14 attributes. After deleting the missing records and filtering the attributes, we eventually had 30, 162 records with 15 dimensions.

`IDS` This dataset was collected by *The Third International Knowledge Discovery and Data Mining Tools Competition* that aimed to build an *Intrusion Detection System* (IDS). We sampled a subset with 40, 124 records and 65 dimensions.

`NLTCS` The *National Long-Term Case Study* (`NLTCS`) dataset was derived from *StatLib*[2]. It contained 21, 574 records with 16 binary attributes, which corresponded to 16 functional disability measures. These are 6 activities of daily living and 10 instrumental activities of daily living.

Three other datasets were derived from Hay's work [36], which have been widely used in the differentially private histogram release test.

`Search Logs` This synthetic data set was generated by interpolating *Google Trends* data and *America Online* search logs. It contained 32, 768 records, each of which stored the frequency of searches with the keyword *Obama* within a 90 minute interval between *Jan. 1, 2004* and *Aug. 9, 2009*.

[1]http://archive.ics.uci.edu/ml/

[2]http://lib.stat.cmu.edu/

`NetTrace` This contained the IP-level network trace at a border gateway of a university. Each record reported the number of external hosts connected to an internal host. There were $65,536$ records with connection numbers ranging from 1 to 1423.

`Social Network` This records the friendship relations among $11,000$ students from the same institution, sampled from an online social network web site. Each record contains the number of friends of certain students. There were $32,768$ students, each of who had at most 1678 friends.

All datasets contained no pre-defined coupled information. We used the *Pearson Correlation Coefficient* to generate the coupled degree matrix Δ with the threshold $\delta_0 = 0.6$. Approximately all datasets had a quarter of their records coupled with some other records, and the maximal size of the coupled group was around 10. For each dataset, we generated a query set $\mathcal{Q}$ with $10,000$ random linear queries and each answer fell into $[0,1]$. The accuracy of results was measured by *Mean Square Error* (MSE).

$$MSE = \frac{1}{|\mathcal{Q}|} \sum_{Q_i \in \mathcal{Q}}^{\mathcal{Q}} (\hat{Q}_i(x) - Q_i(x))^2 \tag{5.6.1}$$

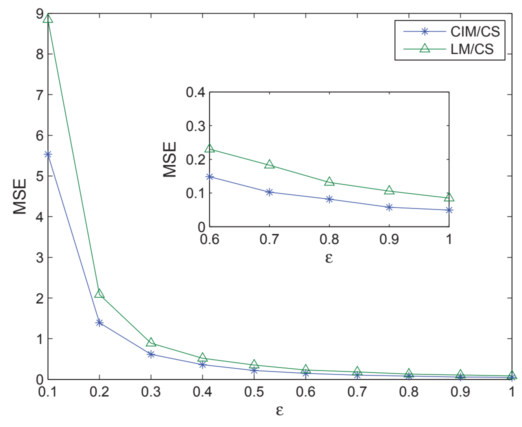
A lower MSE implies a better utility for the corresponding mechanism.

5.6.2 The Performance of *CIM*

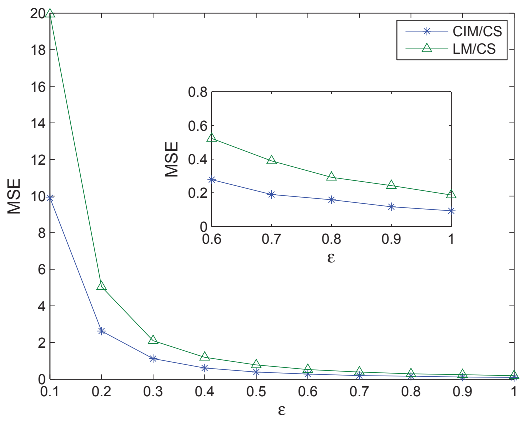
The performance of the *CIM* mechanism was examined in this section through comparison with the state-of-the-art naive *Laplace Mechanism* (LM) [22]. To show its effectiveness, we used the *couple sensitivity* (CS) in both *CIM* and *LM*, and denoted the results as CIM&CS and LM&CS, respectively. The experiments were conducted on all datasets and the privacy budgets varied from 0.1 to 1. Two parameters, T and η, were set to 0.3000 and 7.0000, respectively.

As shown in Fig. 5.1, we observe *CIM* has lower MSE than *LM* on all datasets. Specifically, for the `Adult` dataset in Fig. 5.1a, when $\epsilon = 0.4$, the *CIM* achieves a MSE of 0.3593 while *LM* achieves 0.5171. Thus, *CIM* outperfroms *LM* by 43.84%. When $\epsilon = 1$, *CIM* achieves a MSE of 0.0491 which outperformed *LM* by 73.12%. These results illustrate that in coupled datasets, *CIM* outperforms *LM* when answering a large set of queries. The improvement by *CIM* can also be observed in Fig. 5.1b, 5.1c, 5.1d, 5.1e and 5.1f. The proposed *CIM* has better performance because it only consumes the privacy budget in the update rounds, which is less than the total number of queries $|\mathcal{Q}|$. While the traditional *LM* mechanism consumes the privacy budget when answering every query, this actually leads to inaccurate answers. The experimental results show the effectiveness of *CIM* when answering a large set of queries.

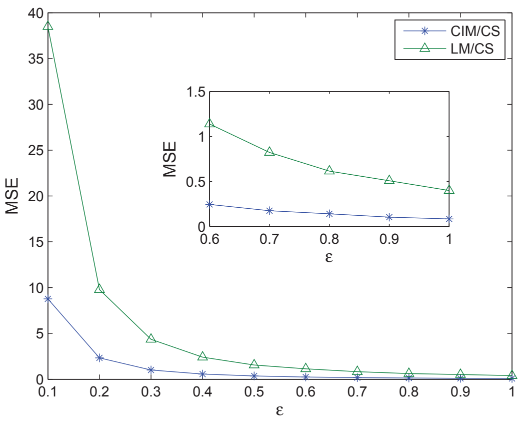
In the context of *differential privacy*, the privacy budget ϵ serves as a key parameter to determine privacy. From Fig. 5.1, we can also check the impact of ϵ on the performance of *CIM*. According to Dwork [22], $\epsilon = 1$ or less would be suitable for privacy preserving purposes, and we follow this heuristic in our experiments. For a comprehensive investigation, we evaluate *CIM*'s performance under various privacy preserving levels by varying the privacy budget ϵ from 0.1 to 1 with a 0.1 step on six datasets. It was observed that as ϵ increases, the MSE evaluation becomes better, which means the lower the privacy preserving level, the larger the utility. In Fig. 5.1a, the MSE of *CIM* is 5.5350 when $\epsilon = 0.1$. Even though it preserves a strict privacy guarantee, the query answer is quite inaccurate. When $\epsilon = 0.7$, the MSE drops to



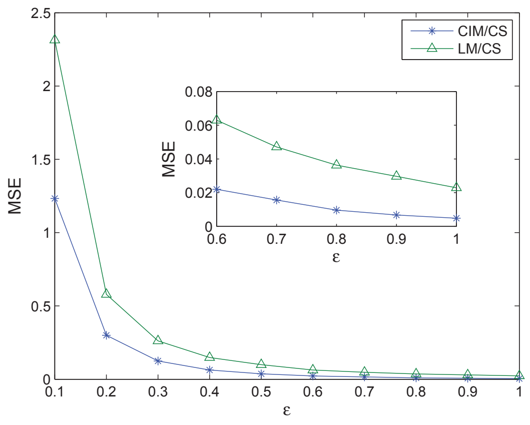
(a) `Adult`



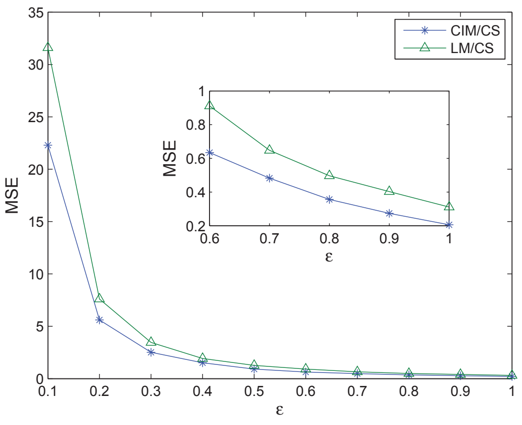
(b) `NLTCS`



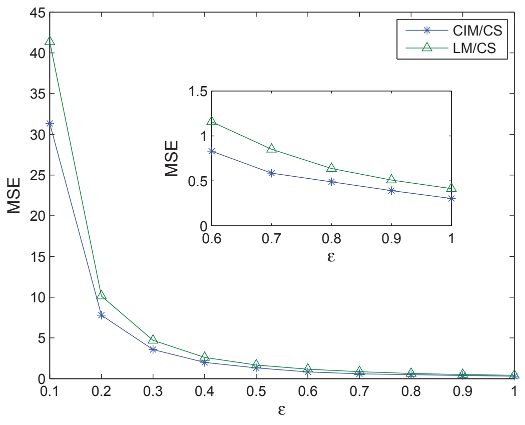
(c) `IDS`



(d) `Search Log`



(e) `NetTrace`



(f) `Social Network`

Figure 5.1: Effectiveness of *CIM*

0.1025, retaining an acceptable utility of the result. The same trend can be observed on other datasets. For example, when $\epsilon = 0.7$, the MSE is 0.1894 in Fig. 5.1b, and is 0.1733 in Fig. 5.1c. These results confirm the utility will be enhanced as the privacy budget increases.

Moreover, we observed the MSE decreased much faster when ϵ ascends from 0.1 to 0.4 compared to when ϵ ascends from 0.4 to 1. This indicates a larger utility cost is needed to achieve a higher privacy level ($\epsilon = 0.1$). We also observed the performance for both the *CIM* and *LM* mechanism was stable when $\epsilon \geq 0.7$. This indicates the *CIM* was capable of retaining the utility for data releasing while satisfying a suitable privacy preserving requirement.

In addition, we compared the MSE of *CIM* with the non-privacy release. If we answer all queries without any privacy guarantee, the MSE is 0. Fig.5.1 shows the MSE of *CIM* was very close to 0 when $\epsilon \geq 0.7$. This was because *CIM* applied iterative steps and *coupled sensitivity* to reduce the magnitude of introduced noise. This result confirms coupled differential privacy can ensure rigorous privacy with an acceptable utility loss.

The evaluation shows the effectiveness of *CIM* on several aspects.

1. The proposed *CIM* can retain a higher utility of released data compared with the *LM*.

2. As the privacy budget increased, the performance of *CIM* was significantly enhanced. We can select a suitable privacy guarantee to achieve a better tradeoff.

3. When we have a sufficient privacy budget, the utility loss of released data is small.

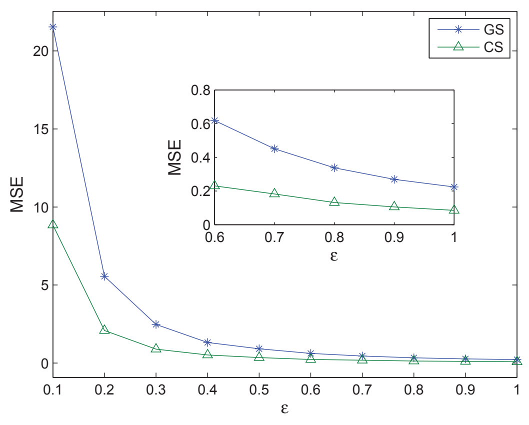
Table 5.2: The Maximal Size of Coupled Groups

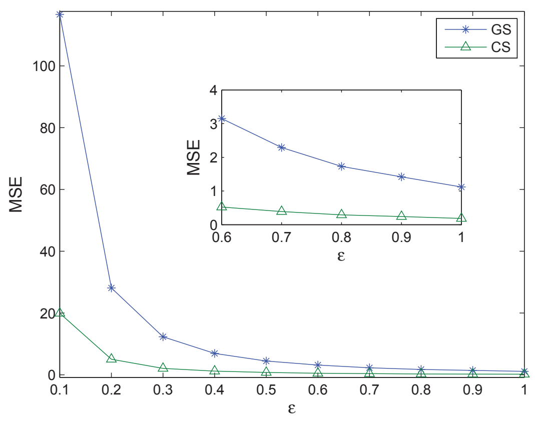
Datasets	k
`Adult`	15
`NLTCS`	10
`IDS`	10
`Search Log`	10
`Net Trace`	10
`Social Network`	20

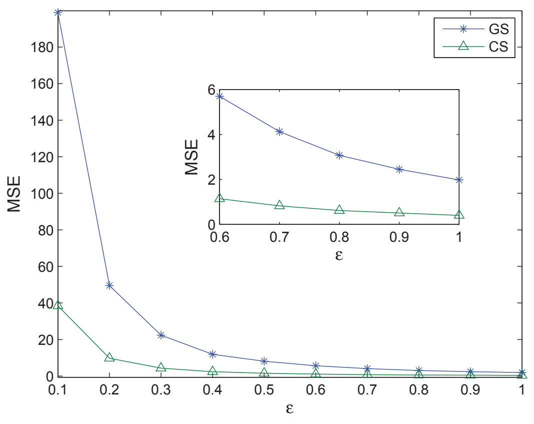
5.6.3 ***Coupled Sensitivity* vs. *Global Sensitivity***

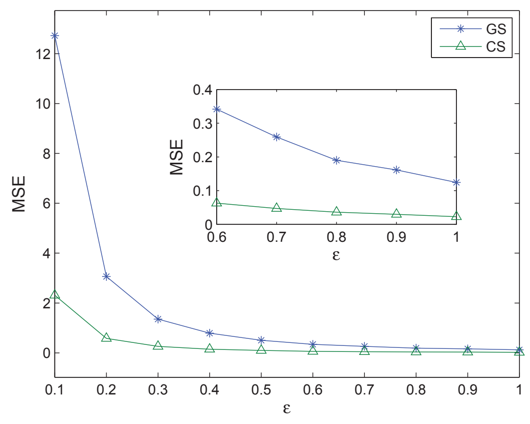
In this subsection, we examine the performance of the *coupled sensitivity* on the data releasing mechanism. In order to show the effectiveness of *coupled sensitivity*, we select the *LM* to answer the query set $\mathcal{Q}$ and compare its performance of *coupled sensitivity* with *global sensitivity* (GS). To deal with the coupled dataset, *global sensitivity* was multiplied by k, the maximal number of records in a coupled group. Table 5.2 lists the size k of different datasets.

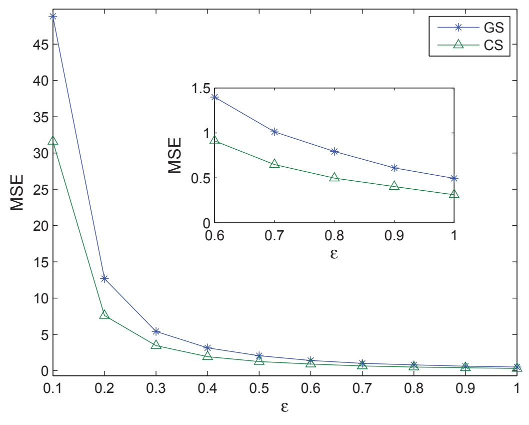
Fig. 5.2 shows the results in which the sensitivities are termed CS and GS, with the privacy budget varying from 0.1 to 1.0. It can be observed that all MSE measures of *coupled sensitivity* on all six datasets were less than MSE of *global sensitivity* with different privacy budgets. These results imply *coupled sensitivity* leads to less error than *global sensitivity* in the context of coupled datesets. Specifically, as shown in Fig. 5.2a, when $\epsilon = 1$, the *LM* with CS achieves an MSE at 0.0850. This outperforms the *LM* result with a GS of 0.2293. The performance of improvement is more


(a) `Adult`


(b) `NLTCS`


(c) `IDS`


(d) `Search Log`


(e) `NetTrace`

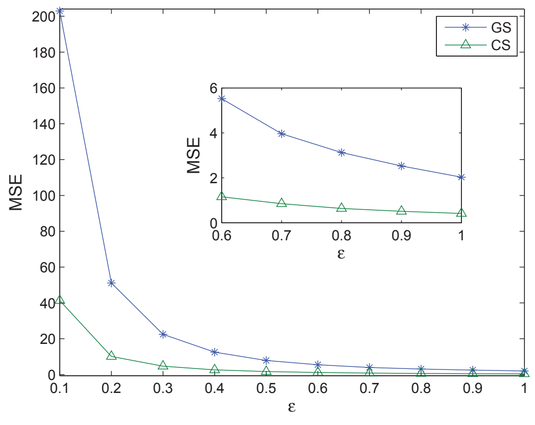

(f) `Social Network`

Figure 5.2: Effectiveness of *Coupled Sensitivity*

significant as the privacy budget decreases. When $\epsilon = 0.3$, the MSE of CS is 0.8906, which is much lower than MSE of the *LM* with a GS of 2.4785.

Moreover, it is clear MSE of *coupled sensitivity* is close to 0, which indicates *coupled sensitivity* can achieve the privacy preserving purpose while retaining a high utility of query answers.

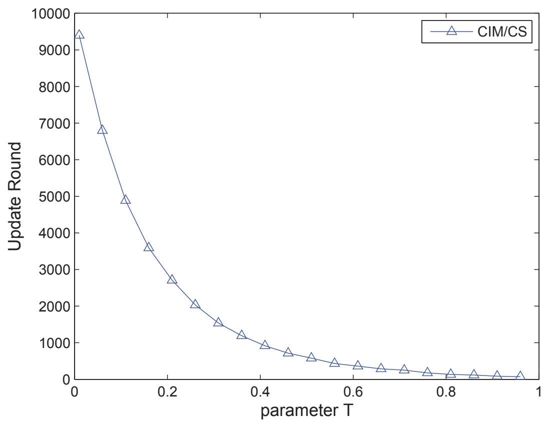
Similar trends can be observed on other datasets as shown in Fig. 5.2b, 5.2c, 5.2d, 5.2e and 5.2f. For example, in Fig. 5.2b, the *LM* with CS leads to an MSE of 0.5222, which is much smaller than 3.1534 from the *LM* with GS. These results illustrate CS is independent to the characteristics of datasets.

The experimental results illustrate that even with the naive mechanism, *coupled sensitivity* can lead to a better performance than *global sensitivity*. It confirms the conclusion in Lemma. 3, which proves the magnitude of *coupled sensitivity* is equal to or less than the *global sensitivity*.

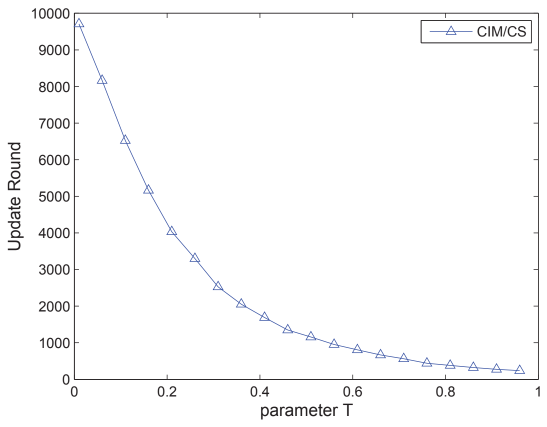
5.6.4 Impact of Parameter T

In *CIM*, T is a threshold that bounds the difference between noisy query answers, which controls the number of update rounds in *CIM*. According to Section 5.4.3, the number of update rounds will have a direct impact on the utility of the query result. To achieve a comprehensive investigation, we investigated the impact of T on the number of update rounds and then estimated the impact on the utility. The parameter T varied from 0.01 to 1 with a step of 0.5, and the privacy budget ϵ was fixed at 1.

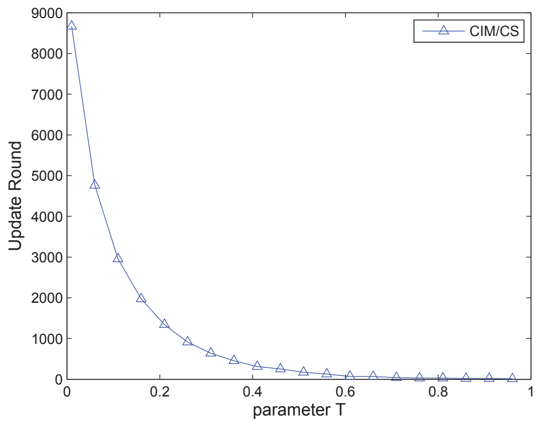
Fig. 5.3 shows the number of update rounds decreases smoothly as T increases. When T is small, nearly all iterations in *CIM* will update the histogram. The number



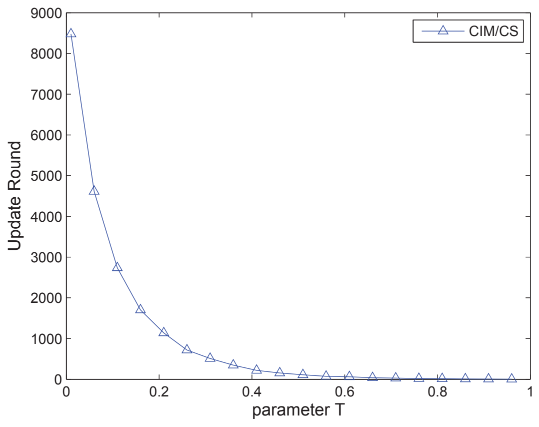
(a) Adult



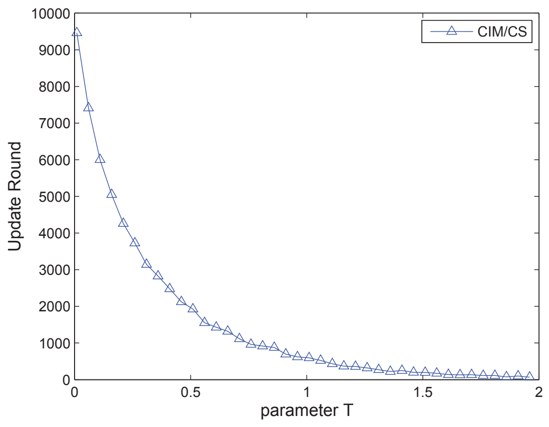
(b) NLTCS



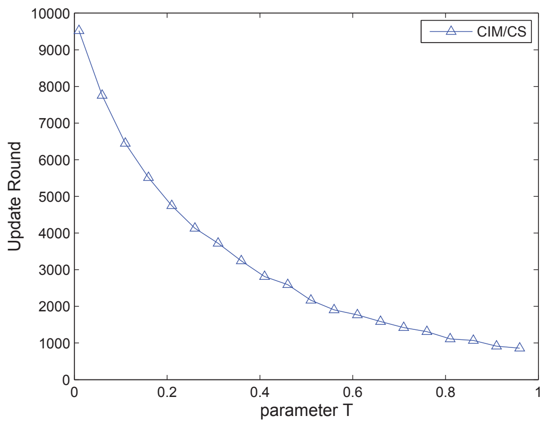
(c) IDS



(d) Search Log



(e) NetTrace

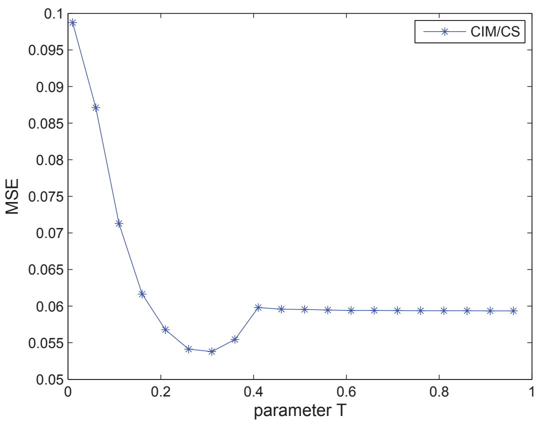


(f) Social Network

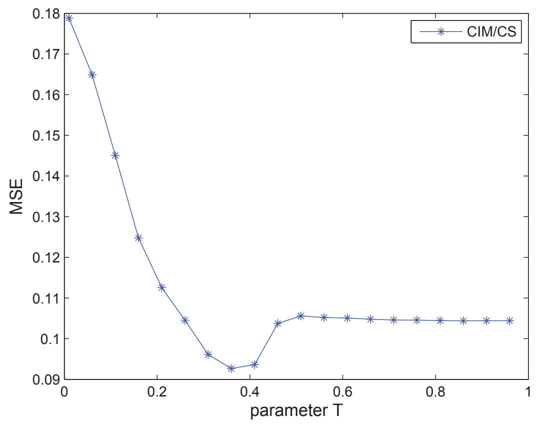
Figure 5.3: Impact of T on Update Round

of update rounds begins to decrease as T increases. For example, in Fig. 5.3a, the number of update rounds is 9398 when $T = 0.01$, and then drops as T increases. When $T = 1$, it drops to 72, which is only 0.72% of all queries. Other datasets show similar results. For example, for the `IDS` dataset in Fig 5.3c, the number of update rounds drops to 10 when $T = 1$. These results represent the relationship between T and the number of update rounds, with a large T resulting in a small number of update rounds. In *CIM*, only the update round will consume privacy budgets, therefore the number of update rounds will affect the performance of *CIM*. However, even when a large value of T decreases the number of update rounds and saves the privacy budget, the performance of *CIM* will not improve when T is larger than the threshold. Because a small number of update rounds results in a less accurate histogram x_t, this means inaccurate query answers are found in the *CIM*. This is confirmed by results in Fig. 5.4.

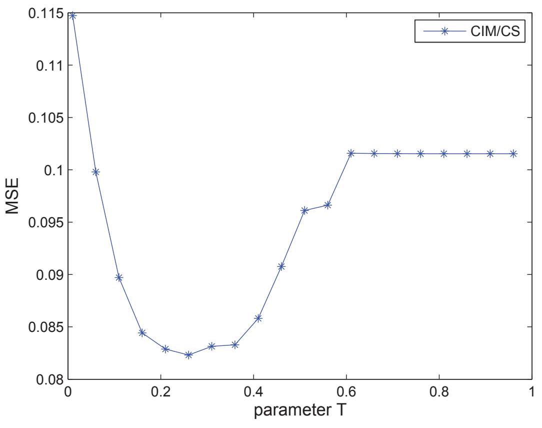
Fig. 5.4 displays the impact of T on the performance of *CIM*. At the beginning, it is apparent with an increasing of T, MSE drops quickly. But when T achieves a threshold, MSE reaches it minimum and keeps increasing until it achieves a stable value. As shown in Fig. 5.5a, MSE keeps decreasing until $T = 0.3100$, with $MSE = 0.0537$ at its lowest point. After this, as T increases, MSE keeps rising until $T = 0.4$, but then remains stable at 0.0594. This trend can be observed in other data sets. For example, in Fig. 5.5b, the MSE reaches its minimum when $T = 0.3600$ and remains stable when $T \geq 0.7100$. Fig. 5.5c, 5.5d, 5.5e and 5.5f all show the same trend. This confirms the theoretical analysis in Section 5.4.3 which shows that even a large value of T can decrease the number of update rounds to save the privacy budget in *CIM*. A T that is too large will lead to an inaccurate histogram and less utility.



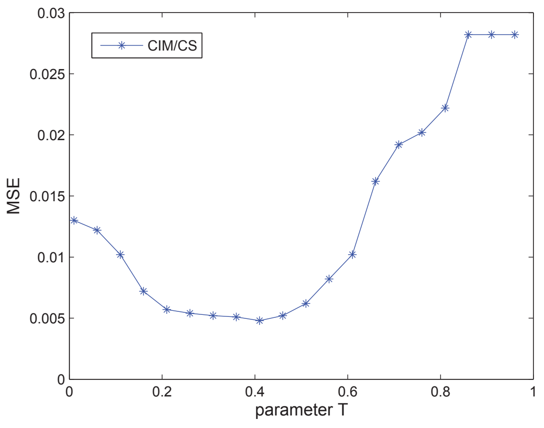
(a) `Adult`



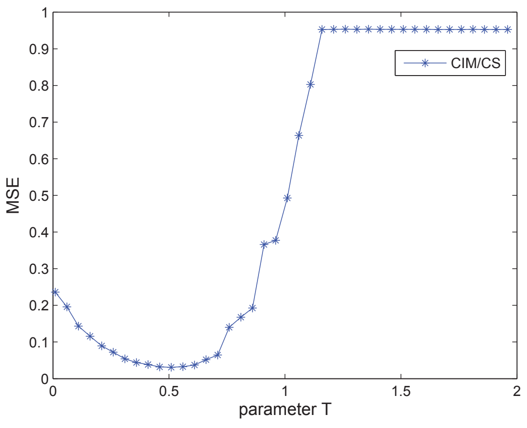
(b) `NLTCS`



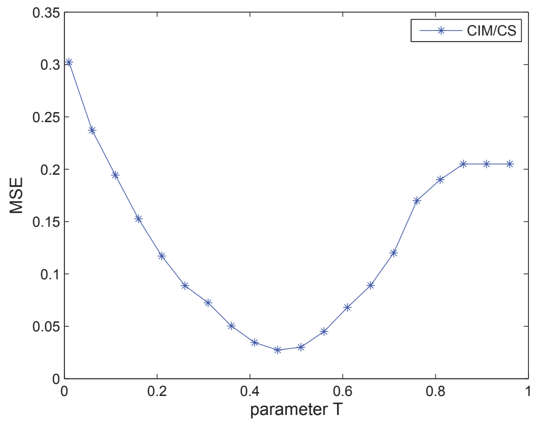
(c) `IDS`



(d) `Search Log`



(e) `NetTrace`



(f) `Social Network`

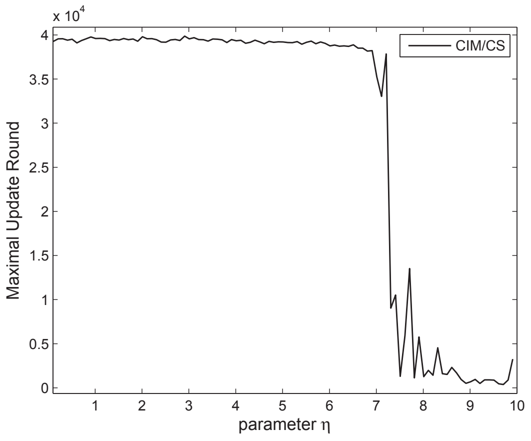
Figure 5.4: Impact of T on *CIM* Performance

This evaluation illustrates the relationship between the parameter T and the number of update rounds, which affects the utility of the releasing data. This relationship can help us control the update round and to allocate the privacy budget in *CIM*.

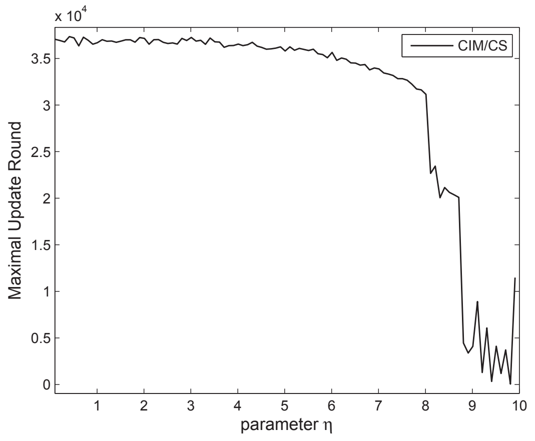
5.6.5 Impact of Parameter η

The parameter η controls the size of the step in update function U, which affects the maximal number of update rounds u_{max} in *CIM*. Please note the u_{max} is only related to η and independent to the query set. According to Eq. 5.4.7, the u_{max} determines the privacy budgets ϵ_0 allocated in each round and the maximal number of queries the *CIM* can answer. To evaluate the impact of η on the u_{max}, we set $100,000$ iterations on *CIM* and vary η from 0.1 to 10.

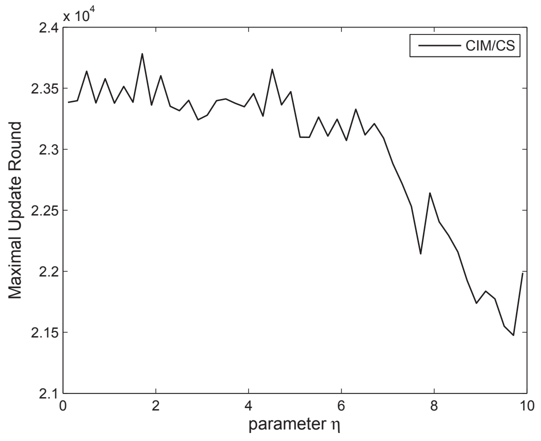
Fig. 5.5 illustrates that as η increases, the maximal number of update rounds drops. It is clear a smaller η indicates a larger u_{max} because the *CIM* will have more update rounds to achieve the original histogram, while larger η results in less u_{max}. The `Adult` dataset in Fig. 5.5a shows that when η is less then 7.0000, the maximal number of update rounds is stable around $38,000$. This means *CIM* will stop after answering $38,000$ queries. Fig. 5.5a shows that when η increases to 8.0000, the maximal number of update rounds drops to 1280. However, even when a smaller η leads to a large u_{max}, the privacy budget ϵ_0 will be small and incur a larger noise. Hence, $\eta = 7.0000$ is considered an inflexion point that can obtain a tradeoff between u_{max} and the ϵ_0. For the `NLTCS` dataset, the inflexion point is also $\eta = 8.0000$. However, for the `IDS` dataset, the maximal number of update rounds decreases smoothly as η increases. This is because the high relative potential of `IDS` leads to a large number of update rounds even though η is relatively high. In this case, the inflexion point can



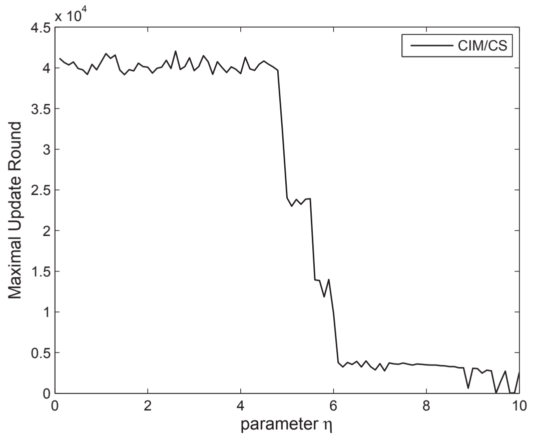
(a) `Adult`



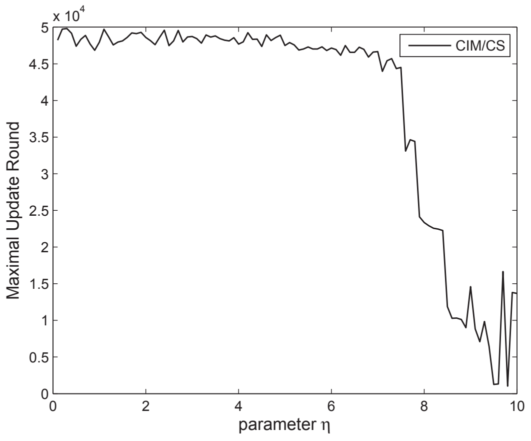
(b) `NLTCS`



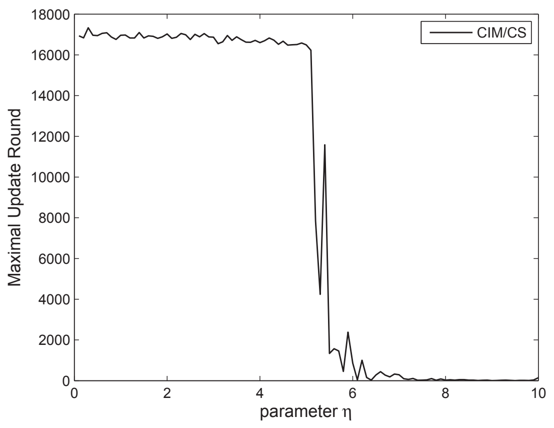
(c) `IDS`



(d) `Search Log`



(e) `NetTrace`



(f) `Social Network`

Figure 5.5: Impact of η on the Maximal Update Round

be chosen according to the number of queries. For the `Search Log` dataset, the maximal number of update rounds begins to drop when η is close to 5.0000, while for the `NetTrace` dataset, the inflexion point of η is around 7.0000. Compared with others, the `Social Network` dataset has a smaller maximal number of update rounds. Even when we set $\eta = 0.1$, the maximal number of update rounds was only $16,934$. This decreases quickly when η is over 6.0000. When $\eta = 8.0000$, the number of update rounds is around 20. This is because for the `Social Network` dataset, the original histogram is close to the uniform distribution. On the other hand, the relative potential is low. When the initial histogram is uniform, the relative potential will quickly drop to 0 after a few update rounds.

The parameter η does not directly affect the performance of *CIM*. This determines the privacy budgets ϵ_0 in each round. A larger η results in a bigger privacy budget, which can produce more accurate answers for *CIM*. However, an overly large η fails to answer a large number of queries, which is impractical in real-world applications. Alternatively, a smaller η leads to a larger number of update rounds but poorer accuracy for answers as the privacy budgets allocated to each round is very small. Thus, to balance the capability of answering the number of queries and the utility of *CIM*, an optimal value for η is the inflexion point in each plot. We use these as the default values in our experiments.

5.7 Conclusions

Differential privacy is one of the most successful notions in the privacy preserving community as it has been proven harder for an adversary to infer the presence or absence of any individual in a dataset. Users are more likely to participate in a system with a privacy warranty. However, traditional differential privacy mainly focuses on independent datasets, failing to offer sufficient privacy guarantees for coupled datasets. This chapter proposes a solution to coupled differential privacy with the following contributions:

- The problem of coupled differential privacy is identified. The coupled problem can be extended to more complex applications, such as a coupled private recommender system or coupled location privacy.
- Novel *coupled sensitivity* is proposed to deal with coupled records. Compared to *global sensitivity*, it guarantees more rigorous privacy while retaining an acceptable utility.
- A novel data releasing mechanism is proposed to enhance performance when answering a large group of queries. This mechanism has proven to ensure better query results than the naive *Laplace Mechanism*.

These provide a practical way to apply a coupled privacy notion to differential privacy with less utility loss. The experimental results also show the robustness and effectiveness of the proposed solution.

Chapter 6

Conclusion

The research presented in this thesis consists of three parts: the first part focuses on the application-aware sensitivity issue in the context of a recommender system; the second part focuses on shrinking the randomized domain for the tagging recommender system; and the last part concentrates on the coupled differential privacy solution for a coupled dataset. The proposed methods aim to improve the data utility with guaranteed privacy by proposing differential privacy solutions on application-aware sensitivity, shrinking the randomized domain and the coupled differentially private solution. This chapter summarizes the research results and the main contributions of this thesis. Several open issues in differential privacy and future directions for research have also been identified.

6.1 Contributions

Theoretical and experimental results have led to the conclusions and main contributions of this thesis. These are:

- We developed an effective differentially private *neighborhood-based* collaborative filtering algorithm by defining novel *Recommendation-Aware Sensitivity* and proposing the *Private Neighbor Selection* method. The large amount of noise in differential privacy is derived from queries with high sensitivity in recommender systems. It can be reduced by the proposed *Recommendation-Aware Sensitivity*. Traditional sensitivity determined only by the queries calibrates the maximal changes when deleting one record in a dataset. If we take each record into consideration, the maximal change will not always occur. *Recommendation-Aware Sensitivity* is determined by the change of each record instead of using the maximal change. Through this novel strategy, we significantly decrease the sensitivity of queries. Theoretical analysis proves the proposed application-aware *sensitivity* can retain the utility of applications while satisfying the requirement of *differential privacy*.

- We proposed a differentially private tagging release algorithm to protect users from being re-identified in a tagging dataset by adversaries. We enhanced the performance of the algorithm by tackling the problem of shrinking the randomized domain. We adapt the private topic model to structure records into groups and limit the randomized domain within each group. After this, a better trade-off between *privacy* and *utility* is obtained by taking the advantage of the differentially private composition properties. Theoretical analysis and experimental results indicate the effectiveness of the proposed method.

- We defined the problem of coupled differential privacy by exploring a novel *coupled sensitivity* and a novel data releasing mechanism. We measured the sensitivity to capture the relationship among records. As most coupled records are only partially coupled, we take advantage of the diverse coupled level between records and propose the notion of *coupled sensitivity*, which is less than traditional global sensitivity. We also design an iteration based coupled releasing mechanism to save the privacy budgets. Based on novel sensitivity and mechanism, we propose the solution of coupled differential privacy to answer a large number of queries for a coupled dataset.

6.2 Future Work

Although the proposed methods have extended differential privacy to various applications, there are still problems that need to be settled.

For the study of *Application-Aware sensitivity*, it only concentrates on the privacy of *neighborhood-based* CF with the proposed *Recommender-Aware sensitivity*. However, other recommendation techniques, such as *Matrix Factorization*, still suffer from violations of privacy. Therefore, future work should consider the privacy issue for other recommendation techniques and propose their related *application-aware sensitivities*.

For exploration of randomized domain shrinking, the proposed private topic model method is only tested within tagging recommender systems. Its application in other scenarios or systems needs further investigation.

There are still further topics that need to be considered in differential privacy. Below we consider the following directions that are worthy of future attention:

Differential privacy for continuous releasing Web sites such as social networks and online stores may need to release their aggregate information to enhance their social and economic utilities. How can these web sites or their curator continually release updated statistics, and simultaneously preserve each individual users privacy is a new problem that has emerged. The continual release is different from the traditional one-time release that differential privacy investigated. In continual release, the input dataset is dynamic and evolves over time, and the data release mechanism must update the output periodically as new data items arrive. Therefore, traditional static differentially private mechanisms either fail to apply directly for continual release, or result in an unsatisfactory loss in terms of utility or privacy if applied naively. The designing of private streaming algorithms that continually output a dataset is a new direction for future research.

Distributed Differential Privacy Most existing work is concerned with the centralized model of differential privacy, in which a trusted data curator holds the entire private dataset, and computes it in a differentially private way. But if a dataset is divided among multiple curators who are mutually untrusting, how can they compute differentially private messages for communication between themselves? Mironov et al. [64] explored a two-party scenario by showing a lower bound for the problem of computing the hamming distance between two datasets. But the solution is unlikely to be valid when the number of curator's lies is more than 2. How to preserve distributed differential privacy within mulitple parties is a good topic for open-ended theoretical exploration.

Synthetic Dataset Releasing for Arbitrary Mining Purposes Synthetic Dataset releasing is an attractive issue for data miners because they can easily construct their models without considering the limitation on the number of queries. However, there have been a series of negative results concerning synthetic database releasing [24]. The most serious one lies in the fact that a synthetic dataset can only perform a fixed data mining task instead of an arbitrary one. Blum et al. obtained a synthetic database that remarkably maintained the approximately correct fractional counts for all concepts in the learning model simultaneously [11], while ensuring the *differential privacy* guarantee. However, they only solved part of the problem because the synthetic dataset can only be used for a certain data mining task, rather than arbitrary data mining tasks. How to efficiently release a synthetic dataset with more data mining flexibility remains a challenge.

Privacy and Game Theory This direction involves with the purchase and sale of private data. This is a scenario in which a data analyst wishes to buy information from a population to estimate some statistical information, while the owners of the private data experience some cost for their loss of privacy. Agents between the seller and buyer wish to maximize their profit, so the goal is to design a truthful auction mechanism while preserving the privacy of the dataset. McSherry and Talwar first proposed designing auction mechanisms using differentially private mechanisms as a building block. This private mechanism is only approximately truthful. Nissim, Smorodinsky, and Tennenholtz show how to convert differentially private mechanisms into exactly truthful mechanisms. However, the mechanism loses its privacy properties. How to design mechanisms that are both truthful and private is a problem that needs to be further explored.

While this thesis provides a thorough report on our research in *extending the theory of differential privacy to enable its effective applicability in real world application.* many interesting and promising issues remain unexplored. The development of social networks and *Big Data* provides great opportunities in research on privacy preserving but also a challenge in how to effectively utilize the large volume of data. Algorithms developed in this thesis can be a starting point for these new challenges because they precisely demonstrate how differential privacy can be extended in real-world scenarios.

Bibliography

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6):734–749, June 2005.

[2] C. Aggarwal and P. Yu. *Privacy-preserving data mining: models and algorithms*. Advances in database systems. Springer-Verlag New York Inc, 2008.

[3] S. Banerjee, N. Hegde, and L. Massoulié. The price of privacy in untrusted recommendation engines. *CoRR*, abs/1207.3269, 2012.

[4] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar. Privacy, accuracy, and consistency too: A holistic solution to contingency table release. pages 273–282, 2007. cited By (since 1996) 32.

[5] A. Beimel, S. Kasiviswanathan, and K. Nissim. Bounds on the sample complexity for private learning and private data release. *Theory of Cryptography*, pages 437–454, 2010.

[6] S. Berkovsky, Y. Eytani, T. Kuflik, and F. Ricci. Enhancing privacy and preserving accuracy of a distributed collaborative filtering. In *Proceedings of the 2007 ACM conference on Recommender systems*, RecSys '07, pages 9–16, New York, NY, USA, 2007. ACM.

[7] R. Bhaskar, S. Laxman, A. Smith, and A. Thakurta. Discovering frequent patterns in sensitive data. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 503–512, New York, NY, USA, 2010. ACM.

[8] D. M. Blei. Probabilistic topic models. *Commun. ACM*, 55(4):77–84, Apr. 2012.

[9] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, Mar. 2003.

[10] A. Blum, C. Dwork, F. McSherry, and K. Nissim. Practical privacy: The sulq framework. pages 128–138, 2005.

[11] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy.

[12] J. A. Calandrino, A. Kilzer, A. Narayanan, E. W. Felten, and V. Shmatikov. "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, 2011. IEEE Computer Society.

[13] J. Canny. Collaborative filtering with privacy via factor analysis. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '02, pages 238–245, New York, NY, USA, 2002. ACM.

[14] L. Cao. Non-iidness learning in behavioral and social data. *The Computer Journal*, 2013.

[15] L. Cao, Y. Ou, and P. S. Yu. Coupled behavior analysis with applications. *IEEE Transactions on Knowledge and Data Engineering*, 24(8):1378–1392, 2012.

[16] K. Chaudhuri and C. Monteleoni. Privacy-preserving logistic regression. In *Neural Information Processing Systems*, pages 289–296, 2008.

[17] S. Chawla, C. Dwork, F. McSherry, A. Smith, and H. Wee. Toward privacy in public databases. In *TCC'05: Proceedings of the Second international conference on Theory of Cryptography*, pages 363–385, Berlin, Heidelberg, 2005. Springer-Verlag.

[18] R. Chen, B. Fung, P. Yu, and B. Desai. Correlated network data publication via differential privacy. *The VLDB Journal*, pages 1–24, 2013.

[19] R. Chen, B. C. Fung, B. C. Desai, and N. M. Sossou. Differentially private transit data publication: a case study on the montreal transportation system. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '12, pages 213–221, New York, NY, USA, 2012. ACM.

[20] G. Cormode, D. Srivastava, N. Li, and T. Li. Minimizing minimality and maximizing utility: analyzing method-based attacks on anonymized data. *Proc. VLDB Endow.*, 3:1045–1056, September 2010.

[21] C. Dwork. Differential privacy. In *ICALP'06: Proceedings of the 33rd international conference on Automata, Languages and Programming*, pages 1–12, Berlin, Heidelberg, 2006. Springer-Verlag.

[22] C. Dwork. Differential privacy: a survey of results. In *TAMC'08: Proceedings of the 5th international conference on Theory and applications of models of computation*, pages 1–19, Berlin, Heidelberg, 2008. Springer-Verlag.

[23] C. Dwork. A firm foundation for private data analysis. *Commun. ACM*, 54(1):86–95, 2011.

[24] C. Dwork and J. Lei. Differential privacy and robust statistics. In *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 371–380, New York, NY, USA, 2009. ACM.

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

[26] C. Dwork, M. Naor, O. Reingold, G. N. Rothblum, and S. Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 381–390, New York, NY, USA, 2009. ACM.

[27] S. Fienberg, A. Rinaldo, and X. Yang. Differential privacy and the risk-utility tradeoff for multi-dimensional contingency tables. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6344 LNCS:187–199, 2010. cited By (since 1996) 0.

[28] A. Friedman and A. Schuster. Data mining with differential privacy. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 493–502, New York, NY, USA, 2010. ACM.

[29] B. C. M. Fung, K. Wang, R. Chen, and P. S. Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Comput. Surv.*, 42(4), 2010.

[30] M. Götz, A. Machanavajjhala, G. Wang, X. Xiao, and J. Gehrke. Publishing search logs: A comparative study of privacy guarantees. *IEEE Transactions on Knowledge and Data Engineering*, 24(3):520, 2012.

[31] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 101(Suppl 1):5228–5235, 2004.

[32] A. Gupta, A. Roth, and J. Ullman. Iterative constructions and private data release. In *Proceedings of the 9th international conference on Theory of Cryptography*, TCC'12, pages 339–356, Berlin, Heidelberg, 2012. Springer-Verlag.

[33] A. Haeberlen, B. Pierce, and A. Narayan. Differential privacy under fire. In *Proceedings of the 20th USENIX conference on Security*, pages 33–33. USENIX Association, 2011.

[34] M. Hardt and G. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. pages 61–70, 2010. cited By (since 1996) 0.

[35] M. Hardt, G. N. Rothblum, and R. A. Servedio. Private data release via learning thresholds. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 168–187. SIAM, 2012.

[36] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially private histograms through consistency. *Proceedings of the VLDB Endowment*, 3(1-2):1021–1032, 2010.

[37] G. Jagannathan, K. Pillaipakkamnatt, and R. Wright. A practical differentially private random decision tree classifier. pages 114–121, 2009. cited By (since 1996) 3.

[38] R. Jäschke, L. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme. Tag recommendations in folksonomies. In *Proceedings of the 11th European conference on Principles and Practice of Knowledge Discovery in Databases*, PKDD 2007, pages 506–514, Berlin, Heidelberg, 2007. Springer-Verlag.

[39] A. Jha, C. DesRoches, E. Campbell, K. Donelan, S. Rao, T. Ferris, A. Shields, S. Rosenbaum, and D. Blumenthal. Use of electronic health records in us hospitals. *New England Journal of Medicine*, 360(16):1628–1638, 2009.

[40] H. Jiawei and M. Kamber. Data mining: concepts and techniques. *San Francisco, CA, itd: Morgan Kaufmann*, 5, 2001.

[41] X. Jin, N. Zhang, and G. Das. Algorithm-safe privacy-preserving data publishing. In *Proceedings of the 13th International Conference on Extending Database Technology*, EDBT '10, pages 633–644, New York, NY, USA, 2010. ACM.

[42] S. Kasiviswanathan, H. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? pages 531–540, 2008. cited By (since 1996) 23.

[43] S. Kasiviswanathan, M. Rudelson, A. Smith, and J. Ullman. The price of privately releasing contingency tables and the spectra of random matrices with correlated rows. pages 775–784, 2010. cited By (since 1996) 1.

[44] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, SIGMOD '11, pages 193–204, New York, NY, USA, 2011. ACM.

[45] D. Kifer and A. Machanavajjhala. Pufferfish: A framework for mathematical privacy definitions. *ACM Trans. Database Syst.*, 39(1):3:1–3:36, Jan. 2014.

[46] R. Krestel, P. Fankhauser, and W. Nejdl. Latent dirichlet allocation for tag recommendation. In *Proceedings of the third ACM conference on Recommender systems*, RecSys '09, pages 61–68, New York, NY, USA, 2009. ACM.

[47] S. Le Blond, C. Zhang, A. Legout, K. Ross, and W. Dabbous. I know where you are and what you are sharing: exploiting p2p communications to invade users' privacy. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet*

measurement conference, IMC '11, pages 45–60, New York, NY, USA, 2011. ACM.

[48] C. Li, M. Hay, V. Rastogi, G. Miklau, and A. McGregor. Optimizing linear counting queries under differential privacy. pages 123–134, 2010. cited By (since 1996) 9.

[49] C. Li and G. Miklau. An adaptive mechanism for accurate query answering under differential privacy. *Proc. VLDB Endow.*, 5(6):514–525, Feb. 2012.

[50] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 106 –115, april 2007.

[51] N. Li, T. Li, and S. Venkatasubramanian. Closeness: A new privacy measure for data publishing. *Knowledge and Data Engineering, IEEE Transactions on*, 22(7):943 –956, july 2010.

[52] N. Li, W. Qardaji, D. Su, and J. Cao. Privbasis: Frequent itemset mining with differential privacy. *Proc. VLDB Endow.*, 5(11):1340–1351, July 2012.

[53] N. Li, W. H. Qardaji, and D. Su. Provably private data anonymization: Or, k-anonymity meets differential privacy. *CoRR*, abs/1101.2604, 2011.

[54] J. Lin. Divergence measures based on the shannon entropy. *Information Theory, IEEE Transactions on*, 37(1):145–151, 1991.

[55] Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Advances in Cryptology in CRYPTO 2000*, pages 36–54. Springer, 2000.

[56] L. Lu, M. Medo, C. H. Yeung, Y.-C. Zhang, Z.-K. Zhang, and T. Zhou. Recommender systems. *Physics Reports*, 519(1):1 – 49, 2012.

[57] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1), Mar. 2007.

[58] A. Machanavajjhala, A. Korolova, and A. Sarma. Personalized social recommendations: accurate or private. *Proceedings of the VLDB Endowment*, 4(7):440–450, 2011.

[59] L. Marinho, A. Hotho, R. Jschke, A. Nanopoulos, S. Rendle, L. Schmidt-Thieme, G. Stumme, and P. Symeonidis. In *Recommender Systems for Social Tagging Systems*, SpringerBriefs in Electrical and Computer Engineering, pages 75–80. Springer US, 2012.

[60] F. McSherry. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. *Communications of the ACM*, 53(9):89–97, 2010. cited By (since 1996) 1.

[61] F. McSherry and R. Mahajan. Differentially-private network trace analysis. volume 40, pages 123–134, New York, NY, USA, Aug. 2010. ACM.

[62] F. McSherry and I. Mironov. 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, 2009. ACM.

[63] F. McSherry and K. Talwar. 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, 2007. IEEE Computer Society.

[64] I. Mironov, O. Pandey, O. Reingold, and S. Vadhan. Computational differential privacy. In S. Halevi, editor, *Advances in Cryptology - CRYPTO 2009*, volume

5677 of *Lecture Notes in Computer Science*, pages 126–142. Springer Berlin Heidelberg, 2009.

[65] N. Mohammed, R. Chen, B. Fung, and P. Yu. Differentially private data release for data mining. pages 493–501, 2011. cited By (since 1996) 0.

[66] A. Narayanan and V. Shmatikov. How to break anonymity of the netflix prize dataset. *CoRR*, abs/cs/0610105, 2006.

[67] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, SP '08, pages 111–125, Washington, DC, USA, 2008. IEEE Computer Society.

[68] M. E. Nergiz, M. Atzori, and C. Clifton. Hiding the presence of individuals from shared databases. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, SIGMOD '07, pages 665–676, New York, NY, USA, 2007. ACM.

[69] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, STOC '07, pages 75–84, New York, NY, USA, 2007. ACM.

[70] R. Parameswaran and D. Blough. Privacy preserving collaborative filtering using data obfuscation. In *Granular Computing, 2007. GRC 2007. IEEE International Conference on Granular Computing*, page 380, nov. 2007.

[71] J. Parra-Arnau, A. Perego, E. Ferrari, J. Forne, and D. Rebollo-Monedero. Privacy-preserving enhanced collaborative tagging. *IEEE Transactions on Knowledge and Data Engineering*, 99(PrePrints):1, 2013.

[72] J. Parra-Arnau, D. Rebollo-Monedero, and J. Forne. Measuring the privacy of user profiles in personalized information systems. *Future Generation Computer Systems*, (0):–, 2013.

[73] H. Polat and W. Du. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pages 625 – 628, nov. 2003.

[74] H. Polat and W. Du. Achieving private recommendations using randomized response techniques. In *Proceedings of the 10th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining*, PAKDD'06, pages 637–646, Berlin, Heidelberg, 2006. Springer-Verlag.

[75] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986. 10.1007/BF00116251.

[76] N. Ramakrishnan, B. J. Keller, B. J. Mirza, A. Y. Grama, and G. Karypis. Privacy risks in recommender systems. *IEEE Internet Computing*, 5(6):54–62, Nov. 2001.

[77] Y. Ren, G. Li, J. Zhang, and W. Zhou. The efficient imputation method for neighborhood-based collaborative filtering. In *CIKM*, pages 684–693, 2012.

[78] Y. Ren, G. Li, and W. Zhou. Learning rating patterns for top-n recommendations. In *ASONAM*, pages 472–479, 2012.

[79] Y. Ren, G. Li, and W. Zhou. A learning method for top-n recommendations with incomplete data. *Social Network Analysis and Mining*, pages 1–14, 2013.

[80] A. Roth and T. Roughgarden. Interactive privacy via the median mechanism. pages 765–774, 2010. cited By (since 1996) 4.

[81] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information. page 188, 1998. cited By (since 1996) 101.

[82] A. Shepitsen, J. Gemmell, B. Mobasher, and R. Burke. Personalized recommendation in social tagging systems using hierarchical clustering. In *Proceedings of the 2008 ACM conference on Recommender systems*, RecSys '08, pages 259–266, New York, NY, USA, 2008. ACM.

[83] B. Sigurbjörnsson and R. van Zwol. Flickr tag recommendation based on collective knowledge. In *Proceedings of the 17th international conference on World Wide Web*, WWW '08, pages 327–336, New York, NY, USA, 2008. ACM.

[84] Y. Song, L. Cao, X. Wu, G. Wei, W. Ye, and W. Ding. Coupled behavior analysis for capturing coupling relationships in group-based market manipulations.

[85] M. Srivatsa and M. Hicks. Deanonymizing mobility traces: using social network as a side-channel. In *Proceedings of the 2012 ACM conference on Computer and communications security*, CCS '12, pages 628–637, New York, NY, USA, 2012. ACM.

[86] M. Steyvers and T. Griffiths. Probabilistic topic models. *Handbook of latent semantic analysis*, 427(7):424–440, 2007.

[87] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowlege-Based Systems*, 10(5):557–570, 2002. cited By (since 1996) 959.

[88] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos. Tag recommendations based on tensor dimensionality reduction. In *Proceedings of the 2008 ACM conference on Recommender systems*, RecSys '08, pages 43–50, New York, NY, USA, 2008. ACM.

[89] V. Verykios, E. Bertino, I. Fovino, L. Provenza, Y. Saygin, and Y. Theodoridis. State-of-the-art in privacy preserving data mining. *ACM Sigmod Record*, 33(1):50–57, 2004.

[90] R. C.-W. Wong, A. W.-C. Fu, K. Wang, and J. Pei. Minimality attack in privacy preserving data publishing. In *Proceedings of the 33rd international conference on Very large data bases*, VLDB '07, pages 543–554. VLDB Endowment, 2007.

[91] R. C.-W. Wong, A. W.-C. Fu, K. Wang, P. S. Yu, and J. Pei. Can the utility of anonymized data be used for privacy breaches? *ACM Trans. Knowl. Discov. Data*, 5(3):16:1–16:24, Aug. 2011.

[92] X. Xiao, Y. Tao, and N. Koudas. Transparent anonymization: Thwarting adversaries who know the algorithm. *ACM Trans. Database Syst.*, 35(2):8:1–8:48, May 2010.

[93] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. *IEEE Trans. on Knowl. and Data Eng.*, 23(8):1200–1214, Aug. 2011.

[94] Y. Xiao, L. Xiong, L. Fan, and S. Goryczka. Dpcube: Differentially private histogram release through multidimensional partitioning. *Arxiv preprint arXiv:1202.5358*, 2012.

[95] Y. Xiao, L. Xiong, and C. Yuan. Differentially private data release through multidimensional partitioning. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6358 LNCS:150–168, 2010. cited By (since 1996) 0.

[96] J. Xu, Z. Zhang, X. Xiao, Y. Yang, and G. Yu. Differentially private histogram publication. ICDE '2012, 2012.

[97] Y. Xu, K. Wang, A. Fu, and P. Yu. Anonymizing transaction databases for publication. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 767–775. ACM, 2008.

[98] J. Zhan, C.-L. Hsieh, I.-C. Wang, T. sheng Hsu, C.-J. Liau, and D.-W. Wang. Privacy-preserving collaborative recommender systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 40(4):472 –476, july 2010.

[99] J. Zhang, Y. Xiang, Y. Wang, W. Zhou, Y. Xiang, and Y. Guan. Network traffic classification using correlation information. *Parallel and Distributed Systems, IEEE Transactions on*, 24(1):104–117, Jan 2013.

[100] Z.-H. Zhou, Y.-Y. Sun, and Y.-F. Li. Multi-instance learning by treating instances as non-i.i.d. samples. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 1249–1256, New York, NY, USA, 2009. ACM.