




2023

PERSONALIZED POINT OF INTEREST RECOMMENDATIONS WITH PRIVACY-PRESERVING TECHNIQUES

Longyin Cui

University of Kentucky, lcu225@uky.edu

Author ORCID Identifier:

 <https://orcid.org/0000-0003-4314-8542>

Digital Object Identifier: <https://doi.org/10.13023/etd.2023.270>

[Right click to open a feedback form in a new tab to let us know how this document benefits you.](#)

Recommended Citation

Cui, Longyin, "PERSONALIZED POINT OF INTEREST RECOMMENDATIONS WITH PRIVACY-PRESERVING TECHNIQUES" (2023). *Theses and Dissertations--Computer Science*. 136.
https://uknowledge.uky.edu/cs_etds/136

This Doctoral Dissertation is brought to you for free and open access by the Computer Science at UKnowledge. It has been accepted for inclusion in Theses and Dissertations--Computer Science by an authorized administrator of UKnowledge. For more information, please contact UKnowledge@sv.uky.edu.

STUDENT AGREEMENT:

I represent that my thesis or dissertation and abstract are my original work. Proper attribution has been given to all outside sources. I understand that I am solely responsible for obtaining any needed copyright permissions. I have obtained needed written permission statement(s) from the owner(s) of each third-party copyrighted matter to be included in my work, allowing electronic distribution (if such use is not permitted by the fair use doctrine) which will be submitted to UKnowledge as Additional File.

I hereby grant to The University of Kentucky and its agents the irrevocable, non-exclusive, and royalty-free license to archive and make accessible my work in whole or in part in all forms of media, now or hereafter known. I agree that the document mentioned above may be made available immediately for worldwide access unless an embargo applies.

I retain all other ownership rights to the copyright of my work. I also retain the right to use in future works (such as articles or books) all or part of my work. I understand that I am free to register the copyright to my work.

REVIEW, APPROVAL AND ACCEPTANCE

The document mentioned above has been reviewed and accepted by the student's advisor, on behalf of the advisory committee, and by the Director of Graduate Studies (DGS), on behalf of the program; we verify that this is the final, approved version of the student's thesis including all changes required by the advisory committee. The undersigned agree to abide by the statements above.

Longyin Cui, Student

Dr. Zongming Fei, Major Professor

Dr. Simone Silvestri, Director of Graduate Studies

PERSONALIZED POINT OF INTEREST RECOMMENDATIONS
WITH PRIVACY-PRESERVING TECHNIQUES

DISSERTATION

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy in the
College of Engineering
at the University of Kentucky

By
Longyin Cui
Lexington, Kentucky

Director: Dr. Dr. Zongming Fei, Professor of Computer Science
Lexington, Kentucky 2023

Copyright© Longyin Cui
2023

ABSTRACT OF DISSERTATION

PERSONALIZED POINT OF INTEREST RECOMMENDATIONS WITH PRIVACY-PRESERVING TECHNIQUES

Location-based services (LBS) have become increasingly popular, with millions of people using mobile devices to access information about nearby points of interest (POIs). Personalized POI recommender systems have been developed to assist users in discovering and navigating these POIs. However, these systems typically require large amounts of user data, including location history and preferences, to provide personalized recommendations.

The collection and use of such data can pose significant privacy concerns. This dissertation proposes a privacy-preserving approach to POI recommendations that address these privacy concerns. The proposed approach uses clustering, tabular generative adversarial networks, and differential privacy to generate synthetic user data, allowing for personalized recommendations without revealing individual user data. Specifically, the approach clusters users based on their fuzzy locations, generates synthetic user data using a tabular generative adversarial network and perturbs user data with differential privacy before it is used for recommendation.

The proposed approaches achieve well-balanced trade-offs between accuracy and privacy preservation and can be applied to different recommender systems. The approach is evaluated through extensive experiments on real-world POI datasets, demonstrating that it is effective in providing personalized recommendations while preserving user privacy. The results show that the proposed approach achieves comparable accuracy to traditional POI recommender systems that do not consider privacy while providing significant privacy guarantees for users.

The research's contribution is twofold: it compares different methods for synthesizing user data specifically for POI recommender systems and offers a general privacy-preserving framework for different recommender systems. The proposed approach provides a novel solution to the privacy concerns of POI recommender systems, contributes to the development of more trustworthy and user-friendly LBS applications, and can enhance the trust of users in these systems.

KEYWORDS: recommender system, location-based services, privacy-preserving, data synthesis, differential privacy

Author's Name: _____ Longyin Cui

Date: _____ June 8, 2023

PERSONALIZED POINT OF INTEREST RECOMMENDATIONS
WITH PRIVACY-PRESERVING TECHNIQUES

By
Longyin Cui

Director of Dissertation: Dr. Zongming Fei

Director of Graduate Studies: Dr. Simone Silvestri

Date: June 8, 2023

ACKNOWLEDGMENTS

Upon the completion of my dissertation, I am compelled to convey my profound appreciation to the individuals who have provided unwavering encouragement, assistance, inspiration, and concern throughout my doctoral journey at the University of Kentucky. The invaluable mentorship of my advisor and esteemed committee members, the camaraderie and assistance of my peers, and the steadfast support of my family have been instrumental in enabling me to bring my dissertation to fruition.

First and foremost, I would like to extend my heartfelt gratitude to my esteemed academic advisor, Dr. Zongming Fei, for his exceptional guidance, patience, encouragement, and attentiveness. Dr. Fei has been instrumental in helping me navigate challenging periods during my research and academic pursuits. His expert support and astute insights have enabled me to conduct my research with heightened efficiency, contributing significantly to my notable academic achievements in recent years. As a mentor, Dr. Fei has fostered a welcoming environment for me in the United States. I am deeply honored to have had the opportunity to work and study under his supervision.

Special thanks to Dr. Jun Zhang for his advice and for leading me to the field of privacy-preserving recommender systems. Dr. Zhang inspired me to read papers and start my research in the area of recommender systems. Without his help, I would have never been able to continue my degree and begin such an exciting and meaningful research topic.

Moreover, I would like to express my gratitude to my external advisor, Dr. Xiwei Wang, at Northeastern Illinois University, for affording me the opportunity to participate in numerous summer projects. Dr. Wang not only introduced me to the role of a paper reviewer but also mentored me in honing my skills in this capacity.

Throughout my work on multiple publications, Dr. Wang imparted valuable writing techniques and stimulated my intellectual curiosity through innovative ideas and discussions. His guidance during the initial years of my studies has significantly enhanced my understanding of both programming and algorithm design. Furthermore, working with numerous undergraduate and graduate students under his tutelage has been a source of motivation and inspiration.

Subsequently, I extend my appreciation to the remaining members of my Advisory Committee: Dr. Qiang Ye, Dr. Corey Baker, Dr. Jerzy Jaromczyk, and Dr. Daniela Claudia Moga, for their unwavering support and guidance throughout the development of my dissertation. I would like to convey special gratitude to my supervisor, Dr. Yi Pike, for her invaluable assistance not only in professional matters but also in providing thoughtful suggestions regarding academic research and career prospects. Additionally, I am grateful to Dr. Debby Keen and Dr. Chi Shen, Chair of the Computer Science Department at Kentucky State University, for their compelling recommendation letters, invaluable advice, and words of encouragement during my pursuit of professional opportunities. The collective wisdom and support of these esteemed individuals have been instrumental in my growth and success.

I would also like to extend my heartfelt appreciation to the entire Department for granting me the Teaching Assistantship throughout my Ph.D. program. A special acknowledgment goes to Dr. Mirosław Truszczyński, the previous Director of Graduate Studies, and Dr. Simone Silvestri, the current Director of Graduate Studies, for their unwavering support and guidance during my academic journey. I am deeply grateful to Ms. Kathy Ice-Wedding, the Student Affairs Officer, for her genuine assistance and care, which have been instrumental in my progress. My sincere gratitude goes to the department as a whole for fostering a supportive and enriching environment that has greatly contributed to my personal and professional growth.

Lastly, but of paramount importance, I wish to convey my profound gratitude and

love to my family. I am indebted to them for bestowing upon me the gift of life and for their unwavering support, encouragement, and boundless love that has been the foundation of my personal and academic growth.

Table of Contents

Acknowledgments	iii
List of Figures	viii
List of Tables	xi
Chapter 1 Introduction	1
1.1 General Classifications	2
1.2 Preliminaries	7
1.3 Point-of-Interest Recommender Systems	10
1.4 General Evaluation Methods and Metrics	12
1.5 Dissertation Organization	13
Chapter 2 Initial Study on Bringing Clustering Technique to Point-of-Interest Recommender System to Preserve Privacy	15
2.1 Motivation and Problem Description	15
2.2 Methodology	15
2.3 The Challenger Model	17
2.4 Results and Discussion	19
2.5 Summary	21
Chapter 3 A Distributed Framework with Vendor-Based Third Party for Privacy- Preserving Point of Interest Recommender Systems	23
3.1 Motivation and Research Goals	23
3.2 Centralized and De-centralized Recommender Systems	24
3.3 Model and Methodology	25
3.4 Experiments	29
3.5 Summary	34
Chapter 4 Leveraging User Text Feedback for Improved Prediction and Privacy	35
4.1 Motivation and Research Goals	35
4.2 Exploration on Related Work	36
4.3 Proposed Model and Methodology	40
4.4 Datasets and Experiment Results	45
4.5 Summary	53
Chapter 5 Towards Privacy-Preserving POI Recommendations: a Generic Data Collection Approach	54
5.1 Motivation and Problem Description	54
5.2 Relative Techniques and Background	56
5.3 Framework Description	57
5.4 Notations and Expression Explanations	60

5.5	Experiments and Discussion	70
5.6	Summary	76
Chapter 6	Privacy-Preserving User Data Synthesis Scheme	78
6.1	The Final Motivation and Problem Description	78
6.2	Utilized Techniques and Important Information	79
6.3	Notifications and Formulas in Study	83
6.4	Experiment Design	85
6.5	Results and Explanations	90
6.6	Summary	92
Chapter 7	Conclusions and Future Work	94
7.1	Research Accomplishments	94
7.2	Potential Future Work	96
	Bibliography	101
	Vita	111

LIST OF FIGURES

1.1	An Example of Recommendations from Amazon	1
1.2	A Generalized Classification of Recommender Systems	2
1.3	The Different General Methods for Building Hybrid Recommender Systems	6
1.4	Basic Components of Location-based Social Network	11
1.5	Stages for Evaluating a Recommender System	12
2.1	An Illustration of the Proposed Model	16
2.2	Rating Distribution of the Champaign-Urbana Dataset	19
2.3	The Impact of the Rank in Performance	20
2.4	The Trend of Errors Over Time	21
2.5	The Impact of Removing Old Rows in the Rating Matrix	21
3.1	Yelp Dataset User Visiting Behavior Analysis	27
3.2	Plotting of the User Visiting Locations (Las Vegas)	29
3.3	Local RSs Accuracy Results (Urbana-Champaign). Four Models' RMSE Results for Each Local RS in the Urbana-Champaign Area.	31
3.4	Local RSs Accuracy Results (Las Vegas). Four Models' RMSE Results for Each Local RS in the Las Vegas Area.	31
3.5	Rating Distributions. (Red: Urbana-Champaign, Blue: Las Vegas). The Ratings Are Ordered from Old to New.	33
4.1	The Framework Of Traditional Centralized Point-of-Interest Recommender Systems	38
4.2	The Overall Model Structure, the Tasks of Each Component, and the Data Flow.	41
4.3	The User Rating Distribution (Phoenix). (The X axis represents the numerical user IDs starting from 1. For example, if we have 10 users, they Will be numbered from 1 to 10. The Y axis is the number of ratings they have left, and it is on a log scale to show a clearer distribution.)	46
4.4	The User Visiting Locations (Phoenix). (Abscissa: the latitude; Ordinate: the longitude)	47
4.5	The RMSE Comparisons among the Four Models for Each Training-Test Dataset Pair (Champaign-Urbana). (Abscissa: the value of RMSE; ordinate: index of folds).	49
4.6	The MAE Comparisons among the Four Models for Each Training-Test Dataset Pair (Champaign-Urbana). (Abscissa: the value of MAE; Ordinate: index of folds).	50
4.7	The RMSE Comparisons among the Four Models for Each Training-Test Dataset Pair (City of Phoenix). (Abscissa: the value of RMSE; ordinate: index of folds).	50

4.8	The MAE Comparisons among the Four Models for Each Training-Test Dataset Pair (City of Phoenix). (Abscissa: the value of MAE; ordinate: index of folds).	51
4.9	The Impact on the Average RMSE Value from Changes on α' in Equation (4.9) (Champaign-Urbana). (The α' indicates the weight of geographical similarities among the users played in the affinity matrix S .)	52
5.1	The Overall Framework Structure and Data Flow of the system. (The diagram illustrates the various components of the framework and the way in which data flows between them.)	58
5.2	Schemes in STTP Collecting Data from User Devices	58
5.3	The Factorization of the Rating Matrix in the Collaborative Filtering Technique. (The diagram illustrates the process of decomposing the matrix into lower-rank matrices that represent the latent factors underlying user-item interactions.)	60
5.4	The Visualization of User Location Clustering (Phoenix). (The clustering algorithms shown in the picture include DBSCAN (upper-left), PAM (upper-right), k-mean (lower-left), and OPTICS (lower-right).)	66
5.5	The two plots show the relationship between privacy parameters Epsilon 1 (ϵ_1) and Epsilon 2 (ϵ_2) and the mean absolute error (MAE). The first plot changes Epsilon 1 (ϵ_1) with Epsilon 2 (ϵ_2) fixed, and the second changes Epsilon 2 (ϵ_2) with Epsilon 1 (ϵ_1) fixed. Both plots show the effect of privacy parameters on MAE.	72
5.6	Comparison of Perturbation Results for Varying Epsilon Values: Scatter Plot of Mean Absolute Error (MAE) Values for Different Combinations of Epsilon 1 (ϵ_1) and Epsilon 2 (ϵ_2) Values Obtained through Overall Perturbation Analysis. (The size and color of the markers correspond to the magnitude of the MAE values. Higher MAE values are observed for certain combinations of epsilon values, indicating that the perturbation has a larger impact on the model's predictions for those parameter values.)	72
5.7	Plotting the Within-Cluster Sum of Squares (WSS) for Different Numbers of Groups.	73
5.8	The Precision Comparisons among Four Models for Each Training-Testing Dataset Pair in the City of Phoenix. (The dataset is partitioned chronologically.)	75
6.1	The Basic Diagram of a Generative Adversarial Network	81
6.2	Example of the Conventional Collaborative Filtering Technique	83
6.3	The visualization of User Location Clustering (Phoenix). (The clustering algorithm shown in the picture is K-mean.)	87
6.4	Comparison of RMSE Values for Different Cut Rates Using Original Data (Blue), Clustered Data (orange), and CTGAN-Generated Data (grey) on the Dataset from CU.	90

6.5	Comparison of RMSE Values for Different Cut Rates Using Original data (blue), Clustered data (orange), and CTGAN-Generated Data (grey) on the Dataset from PH.	91
7.1	An Example of The Input Data Required for FMs [1]	98
7.2	An Example of The Secure Multi-party Computation Scheme [2]	99

LIST OF TABLES

1.1	Model Accuracy (RMSE) Comparisons [3]	10
3.1	Datasets Statistics	30
3.2	Datasets Statistics	32
4.1	Notation summary.	44
4.2	Datasets statistics.	46
4.3	Average Result Comparison (Champaign–Urbana).	49
5.1	Datasets Statistics (rounded)	71
5.2	Average Result Comparison	74
6.1	Datasets Statistics	86
6.2	RMSE Result Comparison for Different <i>mix_rate</i> from CU	91
6.3	RMSE Result Comparison for Different <i>mix_rate</i> from PH	92

Chapter 1 Introduction

Recommender systems (RS) serve a dual purpose: enabling users to extract pertinent data from an abundance of information and allowing service providers to tailor their offerings to individual users [4]. In a more comprehensive definition, an RS may be described as an application or methodology that generates recommendations for users. These suggestions are formulated by considering factors such as user information, item characteristics, historical purchases, and other relevant data.

Advancements in technology have undoubtedly improved the quality of life, yet they have also introduced new complexities. For instance, an individual seeking to purchase a toy helicopter on Amazon is confronted with numerous options, encompassing various brands, features, and prices, as illustrated in Figure 1.1. Online retailers can harness purchase history and algorithmic approaches to curate a personalized list of items a user will most likely buy. A system capable of producing such tailored suggestions may be classified as a Recommender System.

Moreover, Recommender Systems extend beyond the realm of E-commerce, finding applications in diverse fields such as e-business and e-learning, where they facilitate or support a wide array of decision-making processes. Their widespread adoption can be attributed to the vast quantities of data generated daily by electronic and automated devices. Over the past several years, researchers have endeavored to develop and innovate an extensive range of recommender systems to address the growing demand for such solutions.

Amidst the accelerated growth of recommender systems, Point-of-Interest (POI) Recommendations have increasingly gained prominence. Location-based social networks (LBSNs) such as *Yelp*, *Foursquare*, and *Google Local* accumulate vast amounts of daily data from smartphones, tablets, and portable PCs. Consequently, POI Rec-

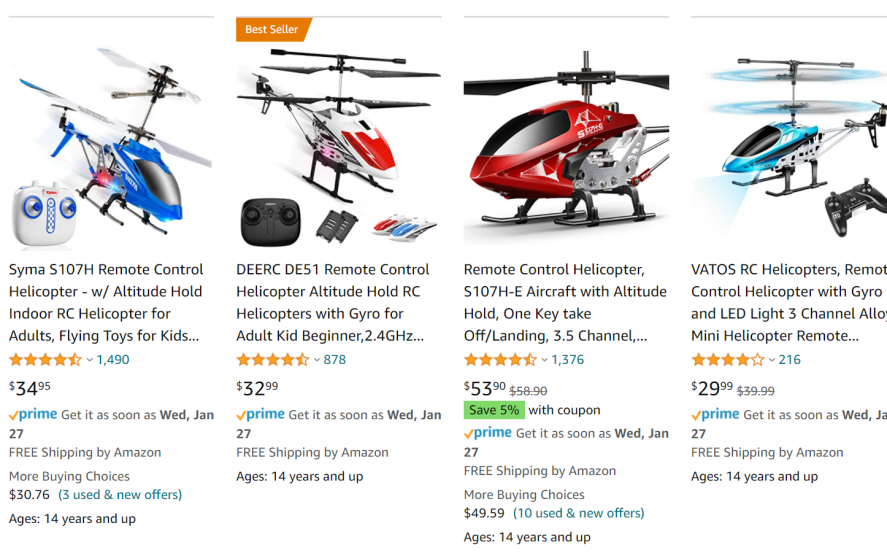


Figure 1.1: An Example of Recommendations from Amazon

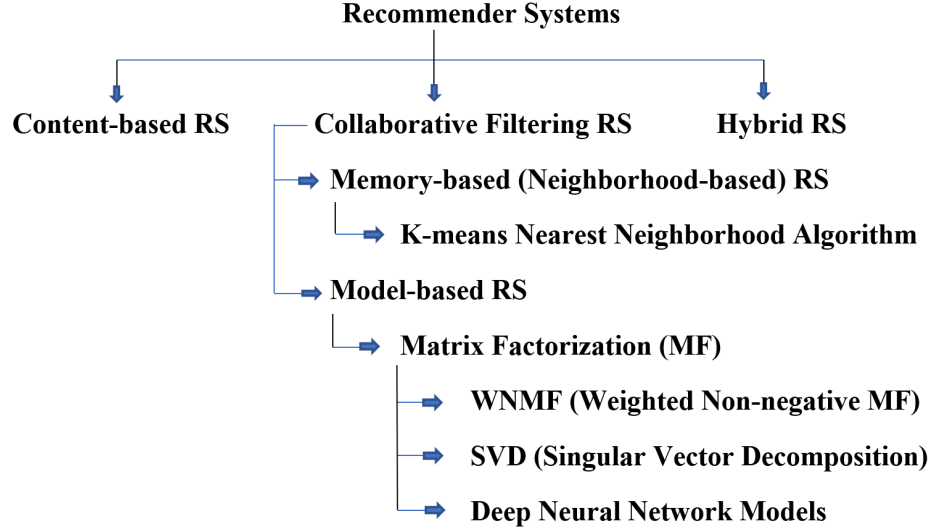


Figure 1.2: A Generalized Classification of Recommender Systems

ommender Systems have emerged as an optimal resource for organizations to harness pertinent information, including check-in records, venue details, and social connections, in order to generate the most precise predictions of POIs.

As a specialized branch of recommender systems, Point-of-Interest Recommender Systems can adopt ideas, structures, or algorithms from conventional Recommender Systems. Nevertheless, they also inherit the associated challenges and complexities.

1.1 General Classifications

As delineated in the introductory section, the scholarly investigation of producing high-quality recommendations has significantly advanced. A recommender system constitutes a software application or an ensemble of applications that harness data and implement algorithms to forecast users' predilections for purchases based on an analysis of their historical buying habits. Any methodology capable of fulfilling these functions may be categorized as a recommender system, ranging from content-based systems to collaborative filtering recommender systems (CFRS) and, ultimately, to hybrid recommender systems (refer to Figure 1.2 for a visual representation). While there exist more granular taxonomies for recommender systems, this report will focus solely on examining traditional models, thereby excluding approaches such as utility-based or demographic recommender systems.

Content-based Recommender Systems

Content-based recommender systems primarily focus on identifying similarities between items, deriving these similarities from the intrinsic attributes of the items or products rather than ratings. For instance, in the context of movie recommendations, such intrinsic attributes might include the film's genre, duration, and production year. In contrast to collaborative filtering, content-based recommender systems employ item

attributes to compute predictions. These systems do not emphasize the interactions between all users and items as collaborative filtering recommender systems (CFRS) do; instead, they rely on the target user’s ratings and the properties of the items favored by that user. Consequently, a key distinction of content-based recommender systems is that other users’ preferences hold minimal significance when generating recommendations for a specific user.

The ability to disregard other users’ preferences is advantageous for content-based recommender systems, as they can better address the cold start problem. The cold start problem pertains to two situations: 1) the absence of feedback from specific users, known as the new-user problem, and 2) the lack of general community feedback, referred to as the new-RS problem. Even if a recommender system is new and the majority of users’ ratings are temporarily inaccessible, content-based recommender systems can still produce relatively accurate recommendations if sufficient information about the target user is available. The lack of feedback is a common issue for new products, which require time to accumulate user feedback. In this case, content-based recommender systems can first extract critical attributes from the new product and then attempt to generate recommendations. However, content-based recommender systems are unable to provide high-quality recommendations when the user is new. Moreover, by overlooking other users’ ratings, the generated recommendations may exhibit poor performance in specific metrics such as novelty and diversity, which will be detailed in Section 6. Nevertheless, content-based recommender systems excel in text-rich domains, such as webpage and news recommendations, where users’ browsing and click history alone can generate recommendations.

Although content-based recommender systems vary significantly depending on the domain or case, they generally follow a three-step process to produce appropriate recommendations:

- Feature extraction: Content-based recommender systems can utilize various types of information across different domains. However, this information is not always readily available and often requires preprocessing and extraction from multiple sources to form a keyword-based, vector-space representation. The effectiveness of feature extraction directly influences the recommender system’s performance, although the specific extraction method may vary and be case-dependent.
- User profiling: This step is similar to conventional classification or regression modeling. As previously mentioned, this process is case-dependent and relies on whether the input rating is binary or numeric. Ratings and extracted features are combined to generate the dataset. Notably, while this step is termed “user profiling,” the subject could also be items. By transposing the input rating matrix, items can be treated as users and users as items. Additionally, rather than extracting features from items, features can be extracted from users, albeit this case is relatively uncommon given that users are typically not “text-rich.”
- Filtering and generating recommendations: The final step involves the trained model generating recommendations for each user. As most research is conducted

offline, computational efficiency is not a critical concern. However, for real-time online recommendations, this step must be highly efficient.

In summary, content-based recommender systems identify similarities between items based on their intrinsic attributes, with minimal consideration of other users' preferences. This approach is particularly advantageous in text-rich domains and when addressing the cold start problem. Content-based recommender systems generally follow a three-step process of feature extraction, user profiling, and filtering to generate recommendations.

Neighborhood-based Collaborative Filtering

There are two types of neighborhood-based algorithms: user-based collaborative filtering and item-based collaborative filtering. The user-based algorithm focuses on finding similar users for a specific user. Then, the RS will generate a list of recommendations based on the weighted average values of all the peer ratings for each item. The underlying idea or assumption is that because the current user is showing similar patterns as the group based on the feedback of purchased items, the user must also have similar feedback to a specific un-purchased item. In other words, similar users display similar rating behaviors. On the other hand, the item-based algorithm focuses on finding similar items. The underlying assumption is that similar items receive similar ratings.

It is essential to point out that different algorithms may have included various information about users. It is rather heuristic when calculating the similarities between user groups. Even in the simplest scenario where we only use users' explicit feedback to calculate similarities, the measurement of distances between users from different models can still vary entirely. The complexity exists because every user has different rating habits and rating scales. For example, user A could have a kind personality leading to giving high ratings to everything he or she reviews. At the same time, user B may have high standards giving comparatively negative feedback on everything he or she bought. In this case, for a rating scale from 1 to 5, user A's rating scope could be from 4.4 to 4.8, while user B's ratings can range from 1 to 3.5. In order to resolve the issue of calculating similarities between users, people usually need to define the similarity function between two users before implementing the model. In later chapters, it will introduce the equations and formulas to generate predictions using this model. Below are two examples of the similarity functions:

- Cosine Similarity: mathematically, this method is used to calculate the cosine value of two non-zero vectors by using the dot product formula:

$$A \cdot B = \|A\| \cdot \|B\| \cos(\theta) \quad (1.1)$$

So here, $\cos(\theta)$ is the similarity that we are looking for, and the two vectors containing ratings from two users are treated as A and B . However, only common users (or items) are taken into account.

The cosine similarity $Sim(u, v)$ is defined as:

$$Sim(u, v) = \frac{\sum_{i \in I_{uv}} r_{ui} \cdot r_{vi}}{\sqrt{\sum_{i \in I_{uv}} r_{ui}^2} \cdot \sqrt{\sum_{i \in I_{uv}} r_{vi}^2}} \quad (1.2)$$

Here, u and v are two different users. i is the index of each common item, and I_{uv} is the set of all the common items rated by both user u and v . r_{ui} is the rating given by user u to item i , and r_{vi} is the rating given by user v to item i .

- Mean Squared Difference (MSD): MSD is more straightforward compared to cosine similarity. For users u , and v , in a user-based model, the MSD is calculated according to the following formula:

$$msd(u, v) = \frac{1}{|I_{uv}|} \cdot \sum_{i \in I_{uv}} (r_{ui} - r_{vi})^2 \quad (1.3)$$

So, the similarity is defined as below:

$$Sim(u, v) = \frac{1}{msd(u, v) + 1} \quad (1.4)$$

Here, the same notations have the same meaning as in the cosine similarity. Also, the $+1$ used in the similarity function is used to avoid dividing by zero. In actual situations, it is prevalent that two different users have never given feedback on even the same item.

- Pearson Similarity: it is very similar to cosine similarity only that the vectors mentioned before A and B are standardized. In other words, statistically, the A and B here should have a mean that is 0 and a standard deviation of 1, and the Pearson correlation coefficient can be seen as mean-centered cosine similarity. However, unlike in statistics, the vectors will not be centered by being divided by standard deviation. Also, when calculating the mean, all ratings or feedback of a user will be used instead of using merely the shared items rated by both users. Below is how the Pearson similarity is defined:

$$Sim(u, v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \mu_u) \cdot (r_{vi} - \mu_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \mu_u)^2} \cdot \sqrt{\sum_{i \in I_{uv}} (r_{vi} - \mu_v)^2}} \quad (1.5)$$

Here, μ_u and μ_v are the average ratings for user u and user v . All the other notations have the same meaning as in the above cases.

Once the similarity matrix is available that records the similarity relationships between all users, we can move to the final step to generate predicted ratings. Suppose we choose the K -neighborhood approach. Then by checking the similarity matrix, we can quickly get the closest k users to the current predicting user and use their ratings and the similarities we just got to generate the predictions or recommendations.

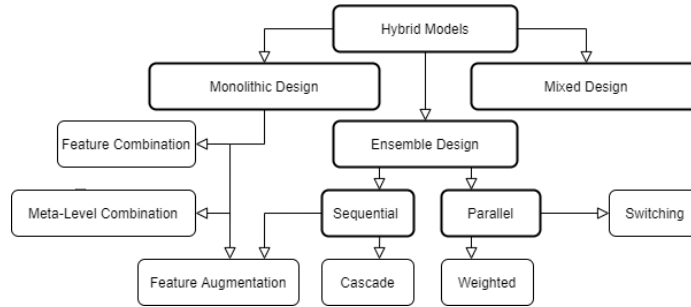


Figure 1.3: The Different General Methods for Building Hybrid Recommender Systems

Hybrid Recommender System

To predict or generate recommendations, Content-based RS and CFRS employ distinct strategies and utilize different types of data. For instance, Content-based RS makes use of a single user’s ratings and item descriptions, while CFRS considers all ratings from the interactions between users and items. As a result, each system and its variants possess their own advantages and disadvantages, such as CFRS being faster and Content-based RS being relatively better at handling cold start problems.

In fact, a hybrid model won the Netflix Prize competition that began in October 2006 [5]. The grand prize was awarded to a team that managed to improve accuracy from an RMSE of 0.9514 to below 0.8563 (over 10% improvement) [6]. Their final solution, “BellKor’s Pragmatic Chaos,” integrated numerous models, including the Neighborhood-based model (k-NN), Restricted Boltzmann Machines, and Matrix Factorization (MF) models, among others [6].

There is more than one way to combine different recommendation approaches. As shown in Figure 1.3, in some earlier surveys [7, 8], researchers identified seven distinct methods:

- **Weighted:** The scores of different components are combined numerically.
- **Switching:** The RS chooses components in real-time.
- **Mixed:** The recommendations generated by different recommenders are presented together.
- **Feature Combination:** Features derived from different sources are combined to feed a single recommendation algorithm.
- **Feature Augmentation:** Features generated from one recommendation technique are used as input for the following recommendation technique.
- **Cascade:** Recommenders are prioritized, breaking ties in the scoring of the higher recommended algorithms.
- **Meta-level:** The output of a recommendation technique is another recommendation model.

In [7], the author further organized these methods into three primary designs:

- Ensemble: Methods that combine results from different recommending techniques to form the final recommendations.
- Monolithic: The separation of different components is not clear, and the RS presents itself as an integrated system.
- Mixed System: Similar to Ensemble, but the recommendations generated are presented together instead of merging into one.

1.2 Preliminaries

Suppose we have m users (customers) and n items (locations). A rating r_{ui} represents the preference of user u for item i . When $r_{ui} \in [0, 1]$, it denotes an implicit rating, which can only be 0 or 1 (positive or negative). Thus, $r_{ui} = 0$ implies that user u has negative feedback for item i or dislikes the visiting experience of place i . Conversely, $r_{ui} = 1$ indicates the opposite. When $r_{ui} \in [1, 5]$, there are additional rating values between 1 and 5. $r_{ui} = 1$ signifies user u 's complete disappointment with item i , while $r_{ui} = 5$ demonstrates strong favor.

Let U and I represent the user and item sets, with m and n as their respective cardinalities. Consequently, we have $u \in U$ and $i \in I$. To distinguish between true and predicted ratings, let's use the notation \hat{r}_{ui} for predicted ratings and r_{ui} for true ratings.

Finally, let's define K such that $K = (u, i) \mid r_{ui} \text{ is known}$. For the Matrix Factorization (MF) models introduced later, unless specified, let's use k to represent the number of latent factors, such that $P \in \mathbb{R}^{m \times k}$ and $Q^T \in \mathbb{R}^{k \times n}$ are the user and item latent factor matrices, respectively. Let's further employ p_u and q_i as the column vectors of the matrices for the k -dimensional latent factors.

Most RS models learn user and item latent factors by regressing known user-item ratings from a preprocessed training dataset. An objective function, such as Equation 1.11, typically comprises three parts. First, the loss function calculates the distance between actual and predicted ratings. Second, the regularization part penalizes the magnitude of the latent parameters, denoted by the constants in the regularizing terms, λ . The third part, though not always necessary, consists of additional terms that penalize or reward based on other contextual information. The least-squares optimization problem is commonly solved using Stochastic Gradient Descent (SGD).

Simon Funk's SVD

The Singular Value Decomposition (SVD) model was initially introduced in the field of recommender systems as a feature reduction tool [9]. However, this term soon became conflated, as matrix factorization was inspired by the conventional SVD method [10]. In the context of recommender systems, SVD refers to Matrix Factorization (MF). Throughout this dissertation, any mention of SVD refers to the basic matrix factorization model.

Performing MF on a dense rating matrix results in three matrices:

$$R = P\Sigma Q^T \quad (1.6)$$

where $P \in \mathbb{R}^{m \times k}$ and $Q \in \mathbb{R}^{k \times n}$ have orthogonal columns, as R is a real matrix. The $\Sigma \in \mathbb{R}^{k \times k}$ is a diagonal matrix, which can be considered as a scalar. The columns of P and Q form the orthogonal basis spanning the column space and row space of the rating matrix R , respectively. By absorbing Σ , we can obtain the following:

$$R = PQ^T \quad (1.7)$$

To estimate the value of each entry in the rating matrix, the dot product of the row vector q_i^T and the column vector p_u can be employed:

$$r_{ui} = q_i^T \cdot p_u \quad (1.8)$$

If R is dense, P and Q can be quickly obtained by calculating the eigenvectors of $R^T R$ and RR^T , respectively. However, in the case of POI recommender systems, the rating matrix is typically sparse, rendering it impossible to compute either $R^T R$ or RR^T without imputation. Early studies on this topic, which included imputation with the global mean or zero, did not produce satisfactory results.

In [10], researchers found that three properties are not necessary when constructing a recommender system: (1) Equation 1.8 needs to be true for all u and i ; (2) all vectors for p_u are mutually orthogonal; (3) all vectors for q_i are mutually orthogonal.

Instead, we can find all such vectors by solving the following objective function:

$$\min_{p_u, q_i} \sum_{u, i \in K} (r_{ui} - q_i^T \cdot p_u)^2 \quad (1.9)$$

The authors in [10] contend that constraining the vectors does not lead to more accurate predictions. Furthermore, missing entries are disregarded during the training process. This algorithm has been widely adopted and examined in numerous studies, demonstrating its effectiveness.

Incorporating biases into the model is another important aspect of enhancing predictive accuracy. In later research, Koren addressed this issue by proposing the Model-based Collaborative Filtering (CF) Recommender System using a different approach [11]. The authors capture the biases and encapsulate the biasing effects independently, while the MF model captures most of the observed signal from the user-item interaction.

Subsequently, the SVD++ and timeSVD++ models were introduced, which integrated implicit contextual information and temporal effects, respectively [11, 3]. These models significantly improved the accuracy of predictions in the context of recommender systems, as demonstrated by their performance on large movie rating datasets such as Netflix.

In conclusion, the development of recommender system models from the basic SVD to the more advanced SVD++ and timeSVD++ models has led to a significant improvement in predictive accuracy. These models have been widely adopted in various research fields and have proven to be highly effective in delivering personalized recommendations to users.

The Baseline Integrated MF Model

User ratings are known to exhibit biases. To address this issue, Koren and his colleagues proposed a Model-based Collaborative Filtering (CF) Recommender System using a novel approach [11]. The authors capture biases and encapsulate their effects independently, while the MF model captures most of the observed signals from user-item interactions. The bias is defined as follows:

$$b_{ui} = \mu + b_u + b_i \quad (1.10)$$

Here, μ denotes the average rating of the entire training set; b_u represents the deviation of user u 's average rating from μ ; b_i indicates the deviation of item i 's average rating from μ . To estimate the values of b_u and b_i for each user and item, the following optimization functions can be adopted:

$$\min_{p_u, q_i} \sum_{u, i \in K} (r_{ui} - \hat{r}_{ui})^2 + \lambda(b_u^2 + b_i^2 + \|q_i\|^2 + \|p_u\|^2) \quad (1.11)$$

$$\hat{r}_{ui} = b_{ui} + p_u \cdot q_i^T \quad (1.12)$$

The first term in Equation 1.11 is designed to capture the bias present in the given ratings, while the second term serves as a regularization component. This approach effectively accounts for user and item biases, resulting in a more accurate and reliable recommender system.

The SVD++ and timeSVD++ Models

Time, space, and implicit feedback are considered vital contextual information which can be collected effortlessly. The SVD++ model enhances the SVD model by incorporating implicit contextual information, while the timeSVD++ model further refines the accuracy of predictions by considering temporal effects [11, 3].

Implicit feedback is utilized in the SVD++ model. This model recognizes that implicit feedback often coexists with explicit feedback. Actions such as renting a movie, purchasing an item, or visiting a place implicitly indicate a user's interest. A hidden matrix with implicit user ratings can be constructed. The updated objective function in Equation 1.11 becomes:

$$\hat{r}_{ui} = b_{ui} + (p_u + |R(u)|^{-\frac{1}{2}} \sum_{i \in R(u)} y_i) \cdot q_i^T \quad (1.13)$$

The only modification between Equation 1.11 and Equation 1.13 is the perception of user latent factor vectors. A new set of factor vectors $y_i \in \mathbb{R}^k$ is added. $R(u)$ is a set containing all items rated by user u . The term $|R(u)|^{-\frac{1}{2}} \sum_{i \in R(u)} y_i$ represents the perspective of implicit feedback.

Temporal effects are utilized in the timeSVD++ model. This type of data is commonly available in various databases such as Google Local, Foursquare, Pandora, and others. As a Time-aware Recommender System model, the timeSVD++ model still exhibits the best performance.

Table 1.1: Model Accuracy (RMSE) Comparisons [3]

Model Name	$k = 10$	$k = 20$	$k = 50$	$k = 100$	$k = 200$
SVD	0.914	0.9074	0.9046	0.9025	0.9009
SVD++	0.9131	0.9032	0.8952	0.8924	0.8911
timeSVD++	0.8971	0.8891	0.8824	0.8805	0.8799

This model makes two assumptions: (1) traits such as user biases $b_u(t)$, item biases $b_i(t)$, and user preferences $p_u(t)$ change over time; (2) item qualities q_i remain static, as the same products do not change with the passage of time.

The updated rating prediction function is shown below:

$$\hat{r}_{ui} = \mu + b_i(t) + b_u(t) + q_i^T \left(p_u(t) + |R(u)|^{-\frac{1}{2}} \sum_{i \in R(u)} y_i \right) \quad (1.14)$$

Biases $b_u(t)$, $b_i(t)$, and $p_u(t)$ are functions aiming to capture temporal effects. Various methods can model these three functions, such as time-linear models and spline-based models.

Table 1.1 compares the accuracy (lower is better) of the three models. The evaluation is performed on a large movie rating dataset provided by Netflix. The results table presents a general and intuitive overview of the improvements made by each model [3].

1.3 Point-of-Interest Recommender Systems

A POI (Point of Interest) recommender system is a type of recommender system that provides personalized recommendations of points of interest (such as restaurants, shops, museums, etc.) to users based on their preferences and past behavior. It is different from general recommender systems in that it takes into account the geographical location of the user and the POI, as well as other contextual information such as time of day, weather, and user mobility patterns.

The contextual information present in Location-based Social Networks (LBSNs) makes the POI recommendation task more challenging than the general RS task. For example, a user may have a preference for Italian cuisine but may only be interested in Italian restaurants within a certain radius of their current location. Additionally, the POI recommender system needs to be able to handle sparsity in the data, as it is common for users to visit only a tiny fraction of the available POIs. Furthermore, the LBSNs have brought new challenges to POI recommendations, such as the users' privacy concerns, which may affect the quality of the recommendations.

LBSN service is a network service based on real-world locations. Such locations include but are not limited to, trajectories, regions, and point locations [12]. Figure 1.4 is a visualization of an LBSN and its components.

To tackle these challenges, researchers have proposed various models, such as SVD and SVD++, which integrate implicit feedback and temporal effects into the

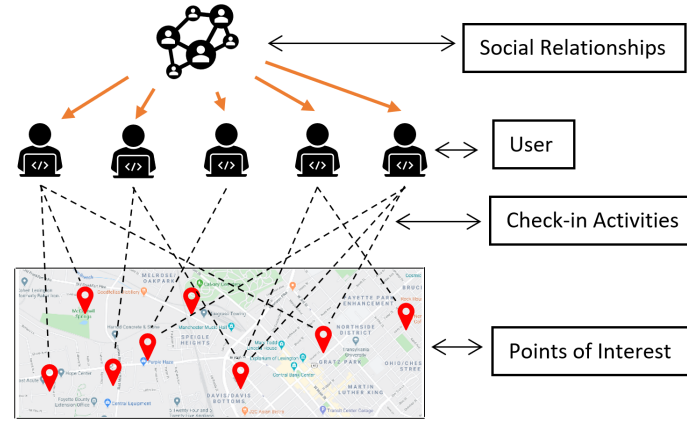


Figure 1.4: Basic Components of Location-based Social Network

recommendation process. These models have been shown to improve the accuracy of POI recommendations compared to traditional CF models. Nevertheless, POI recommendation remains an active area of research due to the unique features of LBSNs, and there is still much work to be done to develop practical recommendation algorithms for these networks.

Challenges of POI Recommendations

POI recommendations can borrow ideas from the conventional recommender system, but the challenges are inherited as well. Moreover, POI recommendations also have new challenges to face. Below are some of the typical challenges:

- **Cold start problem:** Inherited from the conventional RS models, it refers to the problem when the recommender system tries to generate recommendations for users when insufficient data is available. This problem is a common problem for all Recommender Systems, not limited to POI recommendations [13].
- **Data scarcity:** Inherited from the CF RS models. For the point-of-interest recommender system, the problem is worse. For instance, the density of the Netflix *MovieLens* dataset is 1.2%, whereas, for most POI recommendations, this rate could be lowered to 0.1% [5, 14].
- **Scalability:** The data volume is large, and the time complexity for a non-distributed recommender system is at least $O(n^2)$. If the system is distributed, two new problems will appear, communication costs and privacy concerns.
- **Physical constraints:** Unlike online shopping or movie rating systems, local stores and users interact differently in an LBSN. The users are only active in some geographic regions, and the stores or shops are only open for a certain amount of time.

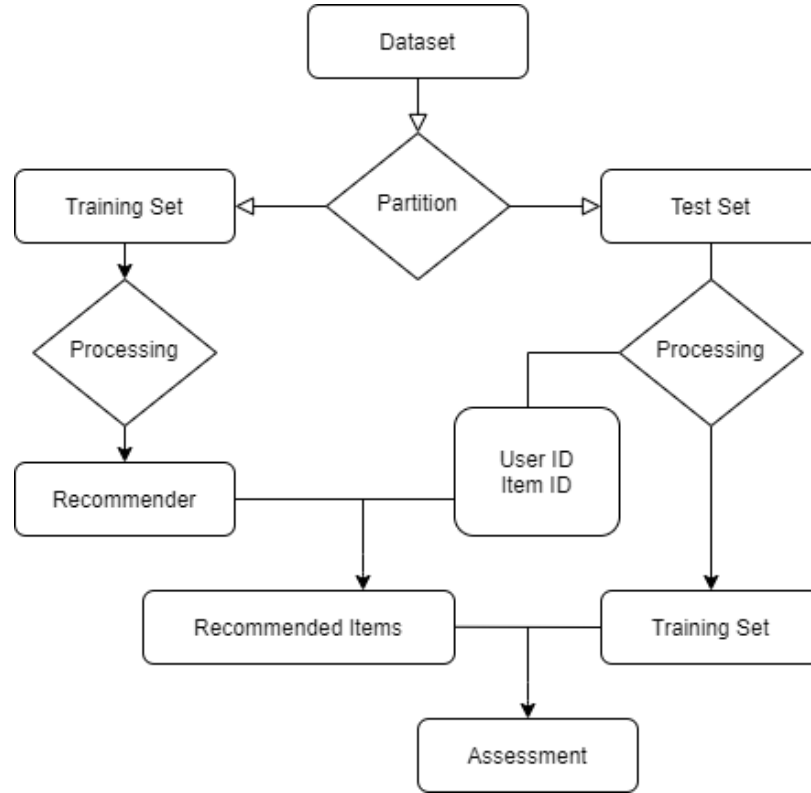


Figure 1.5: Stages for Evaluating a Recommender System

- **Privacy concerns:** At the expense of personal information, we can let the algorithms perform better. However, as Recommender Systems keep using more contextual information, the privacy concern only grows.

1.4 General Evaluation Methods and Metrics

Evaluation Methodologies

When evaluating a recommender system in an offline setting, the data is usually divided into two sets: the training set, which is used to build the system, and the test set, which is used to evaluate its performance. Figure 1.5 illustrates the typical steps involved in the offline evaluation.

After partitioning the dataset, the recommendation algorithm learns and optimizes its weight and biases using the training data. Once the model is trained and stabilized, the test set is used to evaluate the performance of the model by comparing its predictions to the actual data.

Evaluation Metrics

Evaluation of RS models typically involves partitioning the dataset into training and test sets and calculating metrics such as Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) on the test set. Research has shown that even small improvements in RMSE can lead to more accurate top recommendations [11, 6].

- **RMSE:** This evaluation metric was used as the standard metric for the Netflix Prize contest [5]. Let Te be the set of all users and items in the test set. The definition is shown below:

$$RMSE = \sqrt{\frac{\sum_{u,i \in Te} (r_{ui} - \hat{r}_{ui})^2}{m}} \quad (1.15)$$

- **MAE:** Unlike RMSE, MAE does not penalize large error. The equation for calculating MAE is:

$$MAE = \frac{\sum_{u,i \in Te} |r_{ui} - \hat{r}_{ui}|}{m} \quad (1.16)$$

The Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) are commonly used measures to evaluate the accuracy of recommendation models. While both metrics estimate the distance between predicted and actual ratings, RMSE is more sensitive to significant errors or outliers than MAE. This is due to the fact that RMSE penalizes significant errors more heavily than MAE.

In addition to RMSE and MAE, there are other metrics used to evaluate the performance of recommendation systems, depending on different evaluation considerations. These metrics and their definitions will be introduced in later chapters, providing more details on their usage and relevance.

1.5 Dissertation Organization

The dissertation is organized as follows: Chapter 2 presents an initial preliminary empirical study that examines the impact of incorporating clustering techniques into the recommendation system. The study focuses on the use of “super users” to anonymously collect and cluster data from nearby users, thus enabling the delivery of high-quality recommendations.

In Chapter 3, an enhanced framework is introduced, which formally incorporates the role of a trusted third party. This privacy-preserving scheme improves clustering performance and decentralizes the system by distributing central servers across multiple vendor-based servers. This approach offers significant benefits to small business owners by ensuring greater data security and privacy.

Chapter 4 proposes a further refined framework that safeguards not only users’ direct feedback but also their historical comments. This improved framework increases the accuracy of predictions made by the recommendation system, leading to more personalized and relevant suggestions for users.

Addressing the challenge of privacy quantification, Chapter 5 introduces Local Differential Privacy (LDP) and integrates it into the existing framework. Additionally, this chapter presents an upgraded clustering method that further strengthens the privacy-preserving capabilities of the system.

In Chapter 6, the concept of virtual users is formally acknowledged, and the potential of using CTGAN (Conditional Tabular Generative Adversarial Networks) for

synthesizing user data is investigated. This innovative approach enables the generation of realistic yet privacy-preserving user data for further enhancing the recommendation system.

Finally, Chapter 7 provides a discussion of future research directions and offers concluding remarks, summarizing the essential findings and contributions of the dissertation.

Chapter 2 Initial Study on Bringing Clustering Technique to Point-of-Interest Recommender System to Preserve Privacy

2.1 Motivation and Problem Description

As the popularity of location-based social networks (LBSNs) continues to grow, there is a rapidly increasing demand for practical recommendations [15]. The actions of governments and international organizations also fuel the motivation for this research. In May 2018, the European Union enacted the General Data Protection Regulation (GDPR) to enhance the protection of online privacy. This legislation empowers cybercitizens to control their own data, enabling them to determine what types of information may be stored and how service providers can use it.

Prior to this study, several privacy-preserving (PP) Point of Interest (POI) recommendation systems [16, 17, 18] have been proposed to deliver location-based recommendations while safeguarding users' privacy, such as their location and preferences. However, these systems either necessitate additional knowledge, such as friendship or trustworthiness information embedded in social networks or require the storage and maintenance of a topological graph of user geographic coordinates on the server.

Here the research proposes a group preference-based recommender system that employs matrix factorization, clustering, and anonymous ad hoc wireless peer-to-peer (P2P) communication to provide accurate POI recommendations without compromising users' privacy. The system assumes that both the central server and fellow users are untrustworthy. Users store their personal preferences and visit history on their own mobile devices and share their ratings anonymously with nearby users through an ad hoc P2P network via Wi-Fi Direct. The server conducts matrix factorization to learn the latent factors of the groups and locations. When a user device requires recommendations, it retrieves the latent factors from the server and reconstructs the global group preferences. Throughout this process, no individual preference data is shared with the server. After each one-time communication between users, the ad hoc P2P connection is terminated, making it impossible to deduce the identity of the sender.

2.2 Methodology

In the proposed model, each user u has a preference vector $\vec{u} \in \mathbb{R}^{1 \times n}$, which represents their preferences for locations they have visited. This vector is stored on the user's mobile device and updated as new places are visited. When the user's device detects nearby users via Wi-Fi Direct with a predefined SSID (e.g., a string beginning with "*RecSysDataExchange#*"), it counts the number of nearby users and prompts the user with this count, and the user can then decide whether to share their preferences. The exchange can safely occur only when a certain threshold of users (e.g., 10) is nearby. Notably, no other information, including user identity or media access control (MAC) address, is shared.

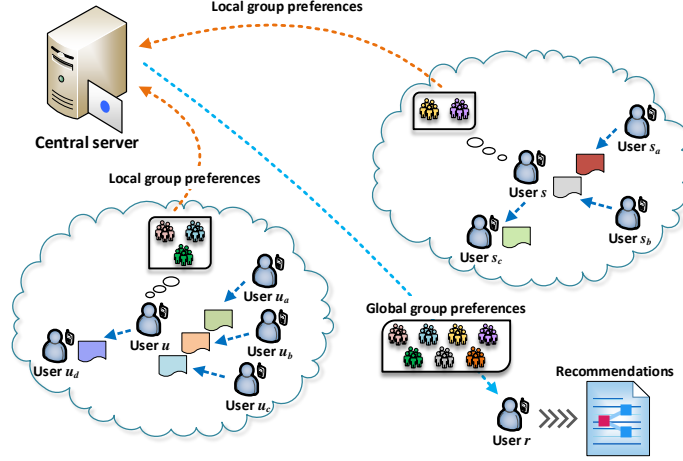


Figure 2.1: An Illustration of the Proposed Model

It is assumed that once a user shares their preferences with another user, the information will not be shared with anyone else. After receiving a sufficient amount of data from others, a user's device employs the k -means algorithm to cluster anonymous individuals into g groups and anonymously sends the group preferences to the server. The local group preferences are appended to the utility matrix R as a series of rows. As data accumulates, the server performs Non-negative Matrix Factorization (NMF) on the local group preferences collected from multiple users to obtain latent factors for global group preferences. The following objective function is used for matrix factorization:

$$\min_{G \geq 0, Q \geq 0} f(R, G, Q) = \|R - GQ^T\|_F^2 + \alpha(\|G\|_F^2 + \|Q\|_F^2), \quad (2.1)$$

where $G \in \mathbb{R}^{m \times k}$, $Q \in \mathbb{R}^{n \times k}$ are two orthogonal matrices, k is the rank, $\|\cdot\|_F$ is the Frobenius norm, and α is the regularization parameter. The stochastic gradient descent (SGD) technique is employed to find the update formulas for this objective function:

$$G_{ij} \leftarrow G_{ij} \cdot \frac{RQ_{ij}}{GQ^TQ + \alpha G_{ij}} \quad (2.2)$$

$$Q_{ij} \leftarrow Q_{ij} \cdot \frac{R^T G_{ij}}{QG^TG + \alpha Q_{ij}} \quad (2.3)$$

To make recommendations for user u_i , their mobile device first retrieves the latent factor matrices G and Q from the server and reconstructs the utility matrix R . Due to the lower dimensions, transferring the factor matrices requires significantly fewer communication resources than transferring R . For visited places q_j ($\forall j \in [1..n] \wedge u_{ij} \neq 0$), the latent factor vectors, i.e., the rows representing q_j in Q , are retrieved to construct

2.3 The Challenger Model

The Decentralized Matrix Factorization (DMF) [17] model is chosen as the challenger model for three reasons. This model is chosen to compare further to the proposed model.

- it is a distributed model for a POI Recommender System;
- it utilizes the users' active location information;
- it has a privacy-preserving framework.

Aside from the three reasons, this model was improved on centralized CF, which was perfect for showing the continuation of the development of Recommender Systems.

Decentralization can resolve the scalability problem by distributing the computing task to its users and by moving the large storage of user action data from the central server to millions of mobile devices. Decentralization will usually cause two new problems, which are privacy concerns and high communication costs. As well this model solved these two problems by proposing a new data exchange mechanism.

To solve the privacy problem, this model introduced a new method that decomposes item latent factors into two parts, the regional preferences and local preferences, and only the regional item latent factors can be exchanged. Furthermore, instead of exchanging the original rating, which can give away the user's preference on a particular item [19], the gradient of the loss is changed. This approach can prevent the DMF model from risking data leakage, which has been proven successful in other research as well [20, 19]. The model replaces the original item latent factor p_u with a new latent factor z_i^u such that:

$$z_i^u = q_i'^u + q_i''^u, \quad (2.4)$$

where q_i' represents the regional item latent factor and q_i'' represents the personal item latent factor. Each of them is a three-dimensional matrix, with each entry representing the intersection of the data holder, the current user, and the current item.

To solve the communication problem, instead of allowing all users to communicate with each other at all times, the model pre-calculated the similarities based on users' location information, such as GPS, to generate an adjacency matrix W , so only users within certain adjacency value can exchange data. This way, once a new check-in activity appears, only targeted devices will receive the updated information, and it also greatly reduces the number of communications. If we use $d_{u,v}$ to indicate the distance between user u and v , each entry in the matrix W is defined as below:

$$w_{u,v} = C^{u,v} \cdot f(d_{u,v}) \quad (2.5)$$

where $w_{u,v} \in [0, 1]$, and C is a Boolean value that equals 1 when u and v are in the same city and 0 otherwise. f is a mapping function whose value increases when the similarity between u and v becomes closer and decreases when the $d_{u,v}$ grows [8].

With the above defined, the updated predicting function in the DMF model can be formulated as:

$$\hat{r}_{ui} = (z_i^u)^T p_u \quad (2.6)$$

Consequently, the DMF model can be formulated as below:

$$\sum_{(u,i) \in K} (r_{ui} - \hat{r}_{ui})^2 + \alpha \sum_{u \in K} \|p_u\|_F^2 + \beta \sum_{(u,i) \in K} \|q_i'^u\|_F^2 + \gamma \sum_{(u,i) \in K} \|q_i''^u\|_F^2 \quad (2.7)$$

This objective function can be solved using SGD by calculating the gradients of the objective function with respect to p_u , q' , and q'' . Similar to the SVD model, the DMF model iterates through each known rating, calculating the prediction, the error, and the gradients and then updating corresponding vectors. The only difference is that after updating the corresponding vectors p_u , q'_v , q''_v according to the gradient of the loss function, the rater, i.e., the current user u , needs to communicate with a number of neighbors sending gradients of the regional item preference q'_u and update those neighbors with each of their item preferences q'_v .

The users of them will hold an independent recommender system on their mobile devices, and the application on their devices will consistently generate POI recommendations in the background. For each local recommender system, it will keep receiving updated information from nearby users when check-in activities appear. If the carrier himself or herself had a check-in activity, the application would send updated information to their neighbors as well. The training process will not stop while there are new check-ins happening.

Although this model addresses many of the difficulties mentioned above, there are still some problems persisting. First, as the authors pointed out, the data volume is large. Each user needs to hold a large amount of data to store both the adjacency matrix and the latent factor matrices. Second, this model still needs to collect users' GPS information directly, which is also a privacy concern.



Figure 2.2: Rating Distribution of the Champaign-Urbana Dataset

2.4 Results and Discussion

The proposed model was examined on a subset of the Yelp business review dataset [21] in the Urbana-Champaign area. The dataset includes 11,917 users, 1,579 businesses, and 33,770 ratings (ranging from 1 to 5 stars) recorded from January 2007 to December 2017. Figure 2.2 shows the distribution of the average ratings left on the businesses. The dataset was preprocessed by sorting all the ratings in chronological order, from the earliest to the latest, and split it into 34 batches with 1,000 ratings each (the last batch has 770 ratings). Each user receives around 1,000 ratings from nearby anonymous users at various times and locations. These ratings can be represented by a user-location matrix. Each time a matrix was constructed, it was imputed before the groups could be generated by the k -means algorithm. The group preferences were represented as several rows of ratings, i.e., the cluster centroids. They were attached to the end of the group-location rating matrix as a series of rows. The NMF was re-run on the group-location matrix each time when it had new rows attached, producing the updated latent factor matrices G and Q . To evaluate the prediction accuracy, after every batch of 1,000 ratings was processed, the test users were selected such that they had ratings in the previously processed batches, as well as in the rest of the batches that had not yet been touched.

To study how this privacy-preserving model affects the prediction accuracy, a simple NMF (referred to as “MFRS”) model is implemented, which takes 1,000 ratings in each batch, imputes the raw ratings, and adds them to the end of the existing user-location matrix (rather than the group-location matrix), performs the NMF on it, and makes estimates of the same test ratings used in this proposed model.

The prediction accuracy was then computed and measured using the mean abso-

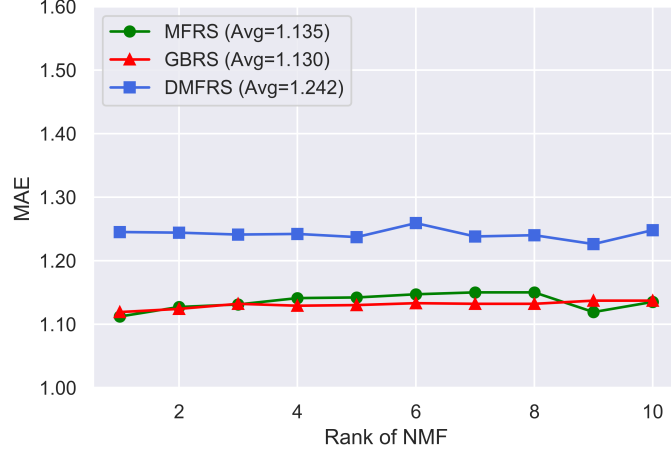


Figure 2.3: The Impact of the Rank in Performance

lute error (MAE). To evaluate the performance of the model against recent research, it is compared with the decentralized matrix factorization model proposed by Chen et al. in [17]. Let’s use “GBRS” to denote the proposed model, i.e., a group-based recommender system.

The batch group number is kept to 3 and the prediction group number to 10 when studying the impact of the rank k . The MAE was computed in each batch, and the average error was used for comparison. Figure 2.4 shows the performance of the three models with varying ranks (In the figure, MFRS was used to denote a centralized NMF and DMFRS to denote the previously mentioned DMF model). It is apparent that the number of latent factors had a trivial impact on all three models. In the following tests, the rank was set to 1 for DMF and GBRS and 9 for DMF for best performance.

When POI recommendations are expected, the user side program needs to identify his/her groups. The cosine similarity between the user preference vector and the group preference vectors is calculated. The program only uses the top 10 groups for prediction. For each location, the weighted average of the group preferences on it is considered the estimated preference for this user. Meanwhile, this research explored how many groups should be generated when a batch of ratings becomes available. The results show that the best MAEs were produced when the batch group number was set to 3 and the prediction group number to 10.

Figure 2.4 illustrates such a trend that although the DMF had overall decreasing errors over time, the average was noticeably higher than other methods. Comparing the GBRS with the two non-privacy models, i.e., baseline and MFRS, the increased portion of the errors is negligible. Inspired by this figure, the impact was studied for the old group preferences on the prediction accuracy in order to help the system determine when to get rid of the obsolete data and keep the knowledge on the server up to date. For this purpose, the experiment removed the rows in matrix R when they became a certain number of years old and examined the performance of the model. The MAEs plotted in Figure 2.5 indicate that by removing two-year-old data, the average prediction accuracy went up approximately 1.69% from the non-time

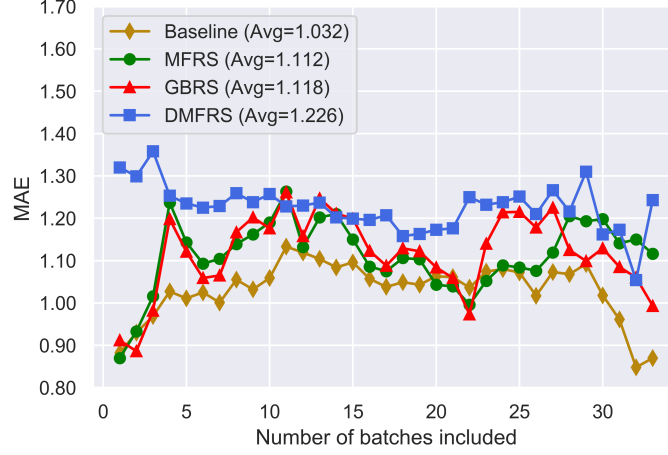


Figure 2.4: The Trend of Errors Over Time

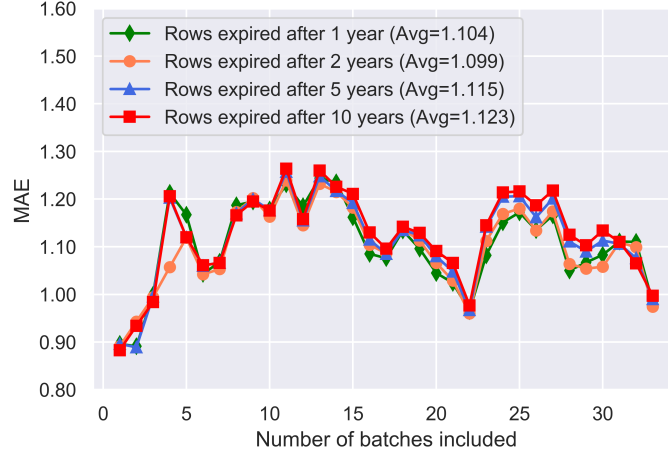


Figure 2.5: The Impact of Removing Old Rows in the Rating Matrix

sensitive version.

In summary, the proposed GBRS had a consistently better performance than the distributed nonnegative matrix factorization model. It is a technically sound framework that does not trade privacy for prediction accuracy, according to the comparison against the baseline and the simple NMF models.

2.5 Summary

The study aims to address the growing need for location-based social network (LBSN) recommendations while maintaining user privacy in accordance with regulations such as the General Data Protection Regulation (GDPR). Previous privacy-preserving (PP) Point of Interest (POI) recommendation systems had limitations, requiring additional information or involving significant server storage.

In this section, a group preference-based privacy-preserving point-of-interest recommender system was designed and proposed. It takes advantage of matrix factor-

ization, clustering, and anonymous ad hoc wireless peer-to-peer communication to provide real-time private location-based recommendations. This system prioritizes user privacy, with users storing their preferences on their devices and sharing ratings anonymously via Wi-Fi Direct in an ad hoc P2P network. The server conducts matrix factorization to learn about group preferences and locations, but individual preference data is not shared with it.

Each user’s preference vector is updated as they visit new locations and stored on their mobile device. Preferences are shared when a certain threshold of users is nearby, with no user identity or MAC address information disclosed. Received data is clustered into groups via the k-means algorithm, and the group preferences are anonymously sent to the server. The server then employs Non-negative Matrix Factorization (NMF) on the collected data to obtain latent factors for global group preferences. An objective function is used for the matrix factorization process, and stochastic gradient descent is employed for updates. To make recommendations for a user, their device retrieves the latent factor matrices from the server and reconstructs the utility matrix, facilitating an efficient and privacy-preserving way to offer POI recommendations.

The major contributions of this work are two folds: first, this model is distinctive in that it doesn’t require storing personal preferences on a central server, thus ensuring user privacy and conforming to regulations such as GDPR; second, by innovatively combining local preferences stored on individual devices and global group preferences deduced from server-side matrix factorization, the research provides a more comprehensive and accurate recommendation system. In other words, this approach is more resource-efficient, as it requires fewer communication resources due to lower dimensionality.

The results of the experiments demonstrate that it outperformed the recently proposed decentralized matrix factorization model in terms of accuracy and the elimination of the user geographic graph. While providing technically sound privacy protection, the model merely lost trivial accuracy compared to the non-privacy benchmark models.

Chapter 3 A Distributed Framework with Vendor-Based Third Party for Privacy-Preserving Point of Interest Recommender Systems

3.1 Motivation and Research Goals

The demand for Point of Interest (POI) recommendation services is proliferating. Location-based Social Networks (LBSN) providers, such as Yelp and Google Local, have effectively increased their market shares. According to Yelp’s Q4’19 report, there are 36 million unique mobile app users bringing in revenues of over one billion dollars in 2019. Moreover, the total number of user reviews it has collected since 2004 has surpassed 205 million [22]. From the Newzoo’s Global Mobile Market Report 2020: (1) there would be 3.5 billion smartphone users worldwide by the end of 2020; (2), in 2019, about 56% of the global website traffic was generated by mobile devices [23]. The primary structure and components of a typical LBSN are shown in Chapter 1 (Figure 1.4), in which the records of check-in activities are usually used to generate recommendations.

Conventionally, all the data collected is saved and stored in a central server to generate recommendations. However, once facing data breach issues, such centralized recommender systems are vulnerable. A recent Capital One data breach caused approximately 500 million dollars in financial damage on top of other indirect costs [24]. The case study on this incident also shows that, nowadays, companies worldwide are still not adequately adapted to securing their cloud computing environments [25].

Several privacy-preserving POI recommender system models [26, 27, 28, 29] were proposed using a decentralized approach, enabling users to process and generate recommendations among themselves. However, data itself is an essential resource. These frameworks require retrieving personal data, such as information from social networks. Consumers’ social relationships and personal data are usually absent and sensitive. Fetching such data is risky, even under the presence of a privacy disclaimer. In 2019, the Federal Trade Commission issued a 5 billion dollar fine on Facebook due to its violation of consumers’ privacy [30].

A vendor-based recommendation network scheme is proposed in this research. Instead of maintaining a central server or carrying out all computing activities on the user side, we can assume that the recommendation tasks are conducted on a local vendor for each small area. Nevertheless, to protect users’ privacy, data concerning users’ preferences or locations are untraceable, making it impossible for local vendors to retrieve enough data. Due to the lack of correlation in the high dimensional space, predictions directly made on this sparse matrix often suffer from low accuracy. With all users divided into smaller groups, the matrix becomes even sparser. We can resolve this issue by introducing “virtual users.”

This part of the work elucidates the benefits derived by both recommendation providers and consumers through the proposed scheme. Service providers experience enhanced user engagement, as users often place greater trust in local businesses compared to large corporations [31]. Simultaneously, users benefit from more efficient

data analysis while the risk of a comprehensive data breach is substantially reduced. Moreover, the selected vendors inherently render the system location-aware, as they correspond to the users’ active visiting areas.

The primary contributions of this research can be summarized as follows:

- The proposition of a localized point-of-interest (POI) recommender system framework, which redistributes the computational tasks and data storage responsibilities of a central server across smaller geographic areas.
- The introduction of the “virtual users” concept, a novel approach that enables localized recommender systems to collaborate with each other, effectively addressing the issue of data sparsity.
- The execution of experiments on real-world datasets demonstrates the importance of adhering to geographical constraints and validating the effectiveness of the proposed framework.

3.2 Centralized and De-centralized Recommender Systems

Centralized Recommendations

The significant difference between a centralized RS and a distributed or decentralized RS is how data is attained and processed. The data is stored on a single server for a centralized RS, where new recommendations are generated immediately after data pre-processing. There are many ways to implement centralized POI recommendation models. To investigate the trade-off between privacy preservation and recommendation accuracy, the study has selected several straightforward models to demonstrate the proposed framework. With that being said, the Recommender System Network (RSN) that is proposed here is compatible with various methods.

For example, for a classic Matrix Factorization (MF) model, a regression technique is realized to collaboratively learn the latent factors of users and items (i.e., POIs) [32]. While users’ feedback is reflected by their ratings, the latent factors indicate each user or item’s hidden characteristics. A general MF-based recommendation model can be represented by the following optimization problem as shown in Equations 3.1 and 3.2, where r_{ui} denotes the known rating given by user u to item i , and \hat{r}_{ui} denotes the predicted rating. Vectors p_u and q_i represent the user and item latent factors, respectively.

$$\min_{p_u, q_i} \sum_{r_{ui} \in R_{train}} (r_{ui} - \hat{r}_{ui})^2 + \lambda(\|q_i\|^2 + \|p_u\|^2) \quad (3.1)$$

$$\hat{r}_{ui} = q_i^T p_u \quad (3.2)$$

In the well-known biased MF model [32, 33], the predicted rating, however, is formalized as follows:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u \quad (3.3)$$

where μ , b_u , and b_i represent the global mean, the user bias, and the item bias, respectively. R_{train} is the set of observed ratings. Accordingly, the objective function is then updated as Equation 3.4. The notations and the parameters will be discussed in detail in later sections.

$$\min_{p_u, q_i} \sum_{r_{ui} \in R_{train}} (r_{ui} - \hat{r}_{ui})^2 + \lambda(b_u^2 + b_i^2 + \|q_i\|^2 + \|p_u\|^2) \quad (3.4)$$

We involve Biased MF heavily in this experiment due to its excellent combination of simplicity and reliability.

Decentralized Recommendations

To convert a centralized RS into a decentralized RS, service providers need to push the data storage and processing tasks to the users' end. Either the user data can be distributed efficiently, or a secure protocol such as a safe peer-to-peer structure is provided to allow information exchange. In a decentralized RS, every user keeps a fraction of the training data and is responsible for generating their own recommendations locally. Some researchers managed to shift the learning process to the users' end to resolve privacy concerns [28, 29, 34].

However, there are inevitable vulnerabilities in these models. For example, when users exchange ratings directly, a malicious user is able to gather other users' ratings by giving positive feedback to all locations. Alternatively, when only latent factors are exchanged, a malicious user can tell that another user visited a specific place if they share a similar latent factor associated with the exact location. Each of the researchers made breakthroughs and solved different problems, but many of them remain.

While distributed systems offer significant advantages in terms of privacy preservation, they also present specific challenges. One notable disadvantage is the increased complexity of managing and maintaining a distributed architecture, as it necessitates the synchronization and coordination of various nodes to ensure data consistency and system reliability. Additionally, this distribution of responsibilities across multiple nodes may inadvertently expose the system to potential security risks, as each node could become a potential target for cyberattacks or unauthorized access. Moreover, the need to communicate and share data between nodes in a distributed system may also compromise privacy, as the data in transit could be intercepted or altered by malicious actors. Consequently, it is crucial to implement robust security measures and encryption protocols to safeguard the privacy of the data in a distributed system while carefully balancing these precautions with the system's performance, scalability, and accessibility.

3.3 Model and Methodology

Involved Formula and Notifications

In a centralized or traditional recommender system, suppose we use u to denote a user (customer) and i an item (POI), then U and I are the user and item sets, where

we have $u \in U$ and $i \in I$. m and n represent the sizes of U and I , respectively. A rating r_{ui} indicates the preference of user u over item/POI i . In the dataset, each rating $r_{ui} \in [1, 5]$, where 1 indicates least favored and 5 most favored. As it was introduced in the previous section, we can use \hat{r}_{ui} for predicted ratings and r_{ui} for their observed counterparts. Aside from the objective function in Equation 3.4, if we denote the rating matrix by R , then we have the following formula:

$$R_{m \times n} \approx P_{m \times k} \cdot Q_{n \times k}^T \quad (3.5)$$

where k is the number of latent factors that are retained, p_u and q_i are column vectors of the two matrices, respectively. For the MF models, unless specified, let's use $P \in \mathbb{R}^{m \times k}$ to denote user latent factor matrix, and $Q \in \mathbb{R}^{n \times k}$ to denote the item latent factor matrix. Furthermore, let's define T_r as the training set and T_e as the test set. Typically, all MF methods require learning user and item's latent factors by regressing over the known user-item ratings from the pre-processed training dataset [35]. Because of this, both Equation 3.4 and Equation 3.5 aim to find the optimal P and Q that minimizes $\|R - P \times Q^T\|$. Finally, let's denote the constant in the regularizing terms in Equations such as Equation 3.1 and Equation 3.4 by λ . Both k and λ are adjusted and tuned using cross-validation. The Stochastic Gradient Descent (SGD) is utilized to solve the least squares optimization.

In the proposed RSN framework, the centralized RS is broken down into multiple local entities. Each area of the city has an independent RS, which maintains its own users and is considered as a local group, denoted by g_i ($g_1 \cup g_2 \cdots \cup g_n = U$). The set of all groups is represented by G . In addition to the physical user (real customers) set U , this research has introduced a virtual user (generated fake customers) set V . If we define the matrix that stores the virtual users' ratings as R_v and the real users R_r , then we have:

$$R_{train} = \begin{bmatrix} R_r \\ R_v \end{bmatrix} \quad (3.6)$$

Users choose a nearby vendor they trust to receive recommendations. In order to simulate the real scenario in the experiment, users in the dataset are clustered beforehand. The clustering is based on the Pearson Correlation Coefficient (PCC) of users' ratings. Specifically, for any pair of users a and b , the similarity between the two is defined by Equation 3.7.

$$Sim_{PCC}(u, v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \mu_u) \cdot (r_{vi} - \mu_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \mu_u)^2} \cdot \sqrt{\sum_{i \in I_{uv}} (r_{vi} - \mu_v)^2}} \quad (3.7)$$

where μ_a and μ_b are the average ratings of users a and b , and I_{ab} is the item set that a and b both rated. After the affinity matrix is constructed, we then perform the kernel k -means clustering to minimize their in-cluster variance. The PCC is chosen since it has the best performance with respect to mean absolute error (MAE) in neighborhood-based RS models [25]. Once all the users are clustered, the cluster centroids are treated as virtual users and sent to all the other RSs.

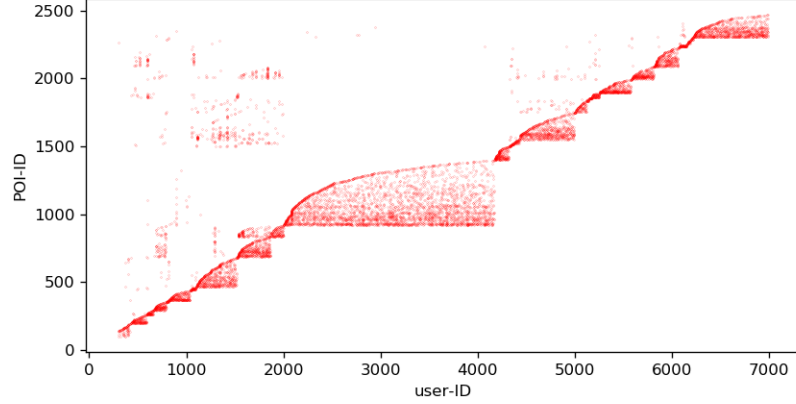


Figure 3.1: Yelp Dataset User Visiting Behavior Analysis

Problem Description

Two separate cities and their nearby districts are chosen to evaluate the model. Most users are only active in a particular town and remain in specific places. This phenomenon is usually referred to as “location aggregation” [28]. For example, Figure 3.1 shows the points are aggregated where each point is a visiting record. The x-axis and y-axis represent the user and POI IDs, respectively.

Users select local businesses they trust to share their data with before getting recommendation services. Accordingly, we can split all the users into different groups to simulate real user activities. This step in simulation is not required in practice since users choose their trusted vendors spontaneously. The research estimates a user’s active location (i.e., the latitude and longitude) by their previously visited POIs (Equations 3.8 and 3.9):

$$Lat_u = \frac{1}{|I_u|} \sum_{i \in I_u} Lat_i \quad (3.8)$$

$$Lon_u = \frac{1}{|I_u|} \sum_{i \in I_u} Lon_i \quad (3.9)$$

where I_u denotes all the POIs a user visited.

However, the side effect of such action is that it makes the already sparse POI rating matrix even sparser. To increase the number of ratings that can be used to train each model, we can provide each local RS with virtual users’ ratings. A local RS generates a certain number of virtual users by clustering the existing users. We can denote the cluster as c and the user set of that cluster as U_c .

$$r_{vi} = \frac{\sum_{i \in I_{uv}, u \in U_c} r_{ui}}{|U_c|} \quad (3.10)$$

Equation 3.10 shows how a virtual user rating is estimated. For a virtual user v , its rating toward location i is approximated by computing the mean value of the ratings left by users in the same cluster ($u \in U_c$) who visited the exact location ($i \in$

I_{uv}). The virtual users, whose ratings are shared by all RSs in an RSN, summarize the preferences of physical users.

Evaluation Algorithm

Since the simulation of users choosing trustworthy vendors and generating virtual users play an essential role in the experiment, the research organizes the work and shows it in Algorithm 1.

Algorithm 1: Preprocessing User Ratings

aInput: all ratings from R , all POIs' location information (longitude and latitude)

Output: n groups of processed training sets $\{Tr_1, Tr_2, \dots, Tr_n\}$ and test sets $\{Te_1, Te_2, \dots, Te_n\}$

```

1 for  $u = 1$  to  $U$  do
2   Calculate each user's longitude  $Long_u$  and latitude  $Lat_u$  according to (8)
   and (9)
3   Eliminate users outside the target city
4   Perform  $k$ -means clustering based on Euclidean distance among users
5 end
6 for  $g = 1$  to  $G$  do
7   Split  $R_g$  into training  $Tr_g$  set and test set  $Te_g$ 
8   for  $user\ u$  in  $Tr_g$  do
9     Calculate the similarities to all other users according to (7)
10  end
11  Complete the affinity matrix for the current group.
12  Perform the kernel  $k$ -means clustering to generate virtual user ratings
    $Rv_g$  according to (10)
13  Append  $Rv_g$  to  $Rv$ 
14 end
15 for  $g = 1$  to  $G$  do
16   Update  $Tr_g$  by appending  $Rv$  according to (6)
17 end
18 return  $Tr$  and  $Te$ 

```

Each vendor-based RS possesses a training set Tr_g in practice, learns the factorized matrices according to Equations 3.4 and 3.5, and then make the matrices P and Q ready for download for its users. The final recommendations are generated on every user's personal device, decreasing the workload for each vendor-based RS.

System Update and Maintenance

While the data is static in this simulation, the real-world users continuously move and change their active visiting areas. Furthermore, for privacy concerns, there should not exist a link between cluster centroids and physical users, which prevents the updating

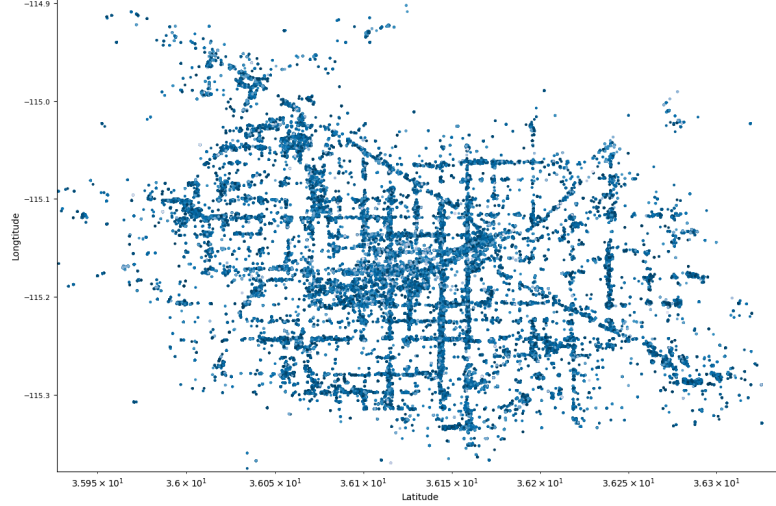


Figure 3.2: Plotting of the User Visiting Locations (Las Vegas)

process from using the same users. Therefore, each time a clustering is completed, its components, centroids, and the number of clusters will differ from the previous one. To make the cluster centroids better represent real users' personal preferences, two mechanisms are implemented:

- The clustering needs to be regularly performed to generate new virtual users.
- The old virtual users need to be turned inactive after the clustering becomes obsolete.

In real-world scenarios, each virtual user is attached to a timestamp. Once it reduces to zero, the virtual user expires and is then removed. When a virtual user is created, the local RS will broadcast it to all the RSs in the same network. The recipients will decide if the information is useful, depending on the overlap between virtual users' visited locations and the item set of the current RS.

3.4 Experiments

Datasets

We use two subsets of the Yelp business review dataset [21]. The first set was collected in the Urbana-Champaign area, and the second was from the city of Las Vegas and its surrounding areas. The rating type is an explicit rating (from 1 to 5 stars) collected by Yelp between January 2007 and December 2017. The research removed the users with too few ratings and repeated ratings.

All local RSs are in the same city or metropolitan area in the test. However, in a real-world scenario, if the RSs share identical items or overlapped item sets, communication can be established, and RSs in the same RNS can enhance each other.

During pre-processing, the research adopted multiple ways to test the appropriate number of RSs in a network. As discussed in previous sections, we need to group

Table 3.1: Datasets Statistics

Dataset	# of Users	# of Items	# of Ratings
UC	2737	1502	22654
Las Vegas	31540	30374	802900

the users based on their visiting locations to simulate the real-world scenario. Figure 3.2 shows the users’ visited POIs, almost reflecting the streets’ shape in Las Vegas and nearby areas. However, the results did not illustrate apparent segregation when attempting to guess users’ real locations by plotting their Euclidean centers of all the visited places. Clustering methods, including k -means, spectral, and density-based spatial, were all tested, and eventually, the k -means method was chosen for its simplicity and straightforwardness. When comparing different clustering methods, the research evaluates both the results of accuracy and the balance of user numbers in each area.

After pre-processing, the details of the datasets are listed in Table 3.1. We sort all the ratings in chronological order and split them by the ratio of 0.2 with the first 80% of ratings for training and 20% for testing.

Evaluation Metrics

We adopted two metrics to evaluate the model performance, the Root Mean Square Error (RMSE) and Mean Absolute Error (MAE). Although accuracy is not always the best metric to evaluate recommender systems [36], minor accuracy improvements, measured by RMSE or MAE, can still pose significant impacts on the quality of top-k recommendations [32, 37]. Equations (10) and (11) show the details of the definition:

$$RMSE = \sqrt{\frac{\sum_{u,i \in Te} (r_{ui} - \hat{r}_{ui})^2}{m}} \quad (3.11)$$

$$MAE = \frac{\sum_{u,i \in Te} |r_{ui} - \hat{r}_{ui}|}{m} \quad (3.12)$$

Results and Discussion

This experiment has compared the results with three existing models:

- The MF model, promoted by one of the Netflix winners, Simon Funk [38]. The research chooses the one that has integrated the baseline model proposed in [32]. The two latent matrices in Equation 3.5 are factorized and learned using the objective function in 3.4. The research used a well-built version to represent a typical centralized RS [39].
- The DMF model, a decentralized scheme that only allows users to exchange gradient loss among neighbors during training [28]. Like most decentralized RSs, there is no data stored on the server. All personal information is kept on users’ devices.

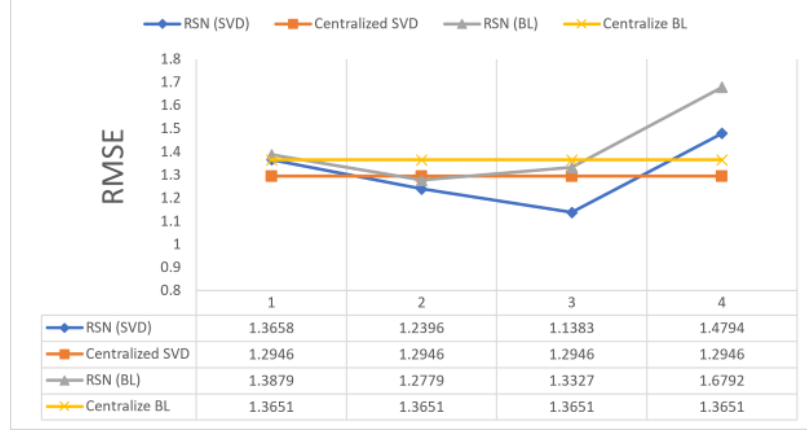


Figure 3.3: Local RSs Accuracy Results (Urbana-Champaign). Four Models' RMSE Results for Each Local RS in the Urbana-Champaign Area.

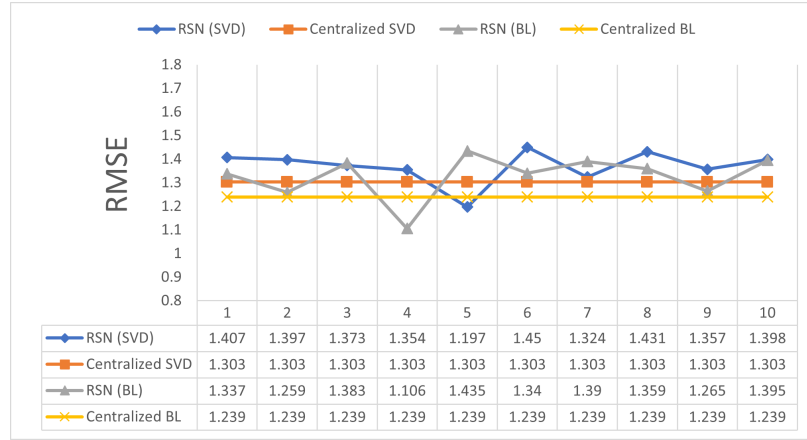


Figure 3.4: Local RSs Accuracy Results (Las Vegas). Four Models' RMSE Results for Each Local RS in the Las Vegas Area.

- The baseline model, of which the prediction function is defined by Equation 3.3 but without the last term. A predicted rating is merely calculated by adding the global mean, column bias, and row bias, and there are no iterative updates involved.

I have opted for straightforward recommendation methods over complex models. In this experiment, the core model can be replaced or combined with other schemes. Each local RS can use different algorithms to generate recommendations.

In the Urbana-Champaign dataset, users were divided into four sections based on their frequently visited locations. In contrast, the Las Vegas metropolitan area has been categorized into ten smaller regions based on the same criterion. The results of every local RS are shown in Figure 3.3 and Figure 3.4. The different number of groups is due to the different number of users and the cities' scale. On the one hand, if we keep the user size too small for an area, the number of users that can be clustered would be too small, leading to insufficient clusters. On the other hand, if this size

Table 3.2: Datasets Statistics

Urbana-Champaign					
Model	MF	RSN(MF)	Baseline	RSN(Baseline)	DMF
RMSE	1.2946	1.3121	1.3650	1.4279	1.4984
MAE	1.0307	1.0568	1.0469	1.0940	1.2018

Las Vegas					
Model	MF	RSN(MF)	Baseline	RSN(Baseline)	DMF
RMSE	1.3034	1.3704	1.2394	1.32996	1.4360
MAE	1.0012	1.1015	0.9452	1.04457	1.1241

is too large, each cluster will have too many users, causing the centroids to be too general to reflect physical users' preferences and interests.

In Figure 3.3 and Figure 3.4, for local RSs in RSN, their performance oscillates up and down on the curves formed by centralized RSs. In most cases, their accuracy is only slightly better than each local RS in an RSN. Occasionally, for some specific areas, such as areas 2 and 3 in Figure 3.3 or areas 4 and 5 in Figure 3.4, RSs in RSN produced higher accuracy than the centralized RSs. In practice, each local RS in an RSN can virtually work with any model and does not have to use the same method uniformly, so theoretically, the proposed model has the potential to outperform a centralized RS. This is similar to why a hybrid model performs consistently better than a pure model.

One thing to point out is how this research calculates the average RMSE and MAE. It is assumed each region has a local RS to generate its recommendations using actual and virtual ratings. It is necessary to estimate the performance of the RSN using all local RSs' average MSE and RMSE. For MAE, the average is the mean value of all the MAEs from every local RS. For the RMSE, however, the average RMSE is estimated by calculating the MSE first and then computing the average RMSE by taking the square root of the mean value of the MSE.

As far as hyper-parameters, in Figure 3.4, where every RS in the RSN uses biased MF as the default model, the number of latent factors $k(40)$ and learning rate $\lambda(0.1)$ were the same as the centralized MF model. The DMF model used the same value for k and set the regularizer to 0.01 and the learning rates to 0.05. The experiment probed each model with $k \in \{5, 40\}$, the learning rate $\lambda \in \{0.01, 0.5\}$, and the regularizer between $\{0.001, 10\}$.

Table 3.2 shows the results for different models. It is apparent that all MF models performed better on the Urbana-Champaign dataset. There could be two reasons. First, the Urbana-Champaign dataset is more compact, meaning a small area with relatively sufficient users and POIs to analyze their preferences. Although the Las Vegas dataset is from a densely populated city, it is still too sparse geographically. In fact, this dataset includes visiting records from the city of Las Vegas, North Las Vegas, Spring Valley, Paradise, and all small towns nearby. Second, as shown in Figure 3.5, the percentage of new businesses in the Las Vegas area is higher than that in Urbana-Champaign. As mentioned previously, the research ordered the rat-

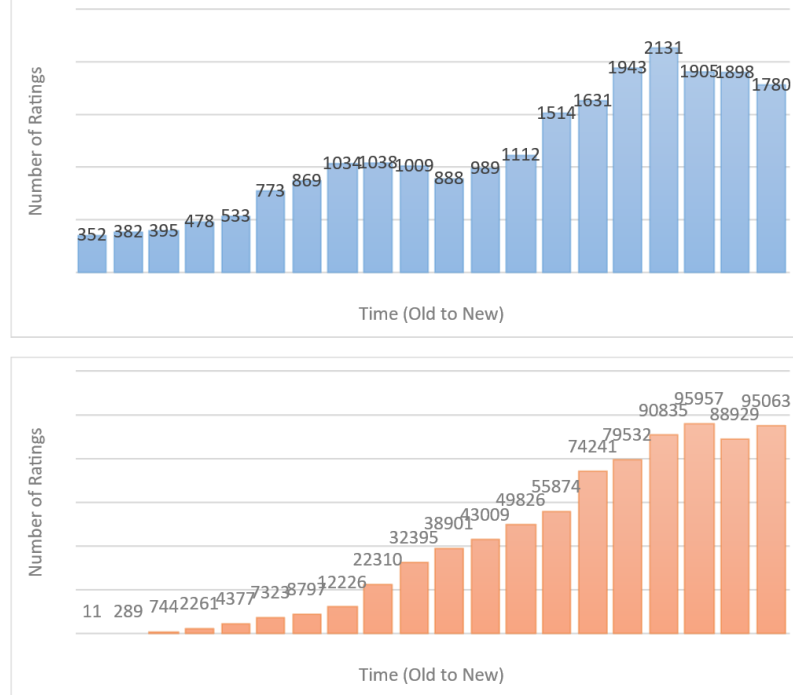


Figure 3.5: Rating Distributions. (Red: Urbana-Champaign, Blue: Las Vegas). The Ratings Are Ordered from Old to New.

ings chronologically, enabling the models to use old data to predict new ratings for simulating real-world scenarios.

In the first dataset, compared to the centralized Biased MF model, the accuracy tradeoff for the RSN model is very small, if not trivial (as low as 0.0175 in RMSE and 0.0261 in MAE). The tradeoff is more significant when the recommending method is changed from Biased MF to the Baseline, but it is still smaller than 0.1. This test result is under the circumstances that all local RSs in the RSN uniformly use the same method and much fewer ratings (1/4 of the total ratings in the Urbana-Champaign dataset, 1/10 of the total ratings in the Las Vegas dataset). With different recommendation methods implemented, the RSN can achieve better performance. Since each local RS maintains a much smaller set of users to generate recommendations, it reduces the training time. Also, with the help of “virtual users,” it scored similar accuracy as a centralized RS. In contrast to a completely decentralized RS such as DMF, the RSN sacrifices much less accuracy for privacy preservation.

This experiment estimated users’ preferred vendors who hold their personal histories using the locations of their most frequently visited stores. This, although not the most accurate way, is the best option in the assessment due to the limited information. One can confidently assume that, in practice, the overall performance of the proposed RSN framework will be better. In summary, the RSN can lower the privacy risk and boost user confidence with minimal loss of prediction accuracy. Each local RS has a light workload and fast speed of convergence. Moreover, it is easier to investigate and control the damage if there is any data breach.

3.5 Summary

Conventional centralized RSs face high risks in protecting users' privacy because they carry all personal information in the same system. On the other hand, even though decentralized models can eliminate the risks of data breaches, they almost give up all further data analysis or mining opportunities. However, the proposed framework decentralizes data storage and computation by redistributing responsibilities to trusted local vendors within smaller geographical areas. This localized approach aligns with the observed 'location aggregation' phenomenon, where users remain active within specific locations. The system simulates real user activities by dividing users into groups based on their trusted local vendors.

To address data sparsity—an inherent side effect of the localized approach—'virtual users' are introduced into the system. These virtual users are generated by clustering existing users and estimating their preferences based on the mean rating values of the clustered users. The scheme focuses on both the collaboration among users and among the RSs hosted by small local businesses that people trust.

The primary contributions of this research are three-fold. First, it proposes a localized POI recommender system that enhances user engagement, reduces the risk of comprehensive data breaches, and provides location-aware recommendations. Second, it introduces the concept of 'virtual users' to manage data sparsity, facilitating collaboration between different localized recommender systems. Finally, it validates the effectiveness of the proposed framework through real-world experiments, emphasizing the importance of geographical constraints in improving recommendation accuracy. Compared to previous research, this study not only further improved the prediction accuracy but also made the design logically more sound.

Also, this research leads to another work that includes integrating a distributed neural network into the RSN and exploring other ways to create virtual or synthetic users, i.e., users who are not physical but can reflect real users' interests and preferences.

Chapter 4 Leveraging User Text Feedback for Improved Prediction and Privacy

4.1 Motivation and Research Goals

The development of recommender systems accelerated during the past decade in different fields, such as online shopping, streaming services, and point-of-interest (POI) recommendations. People have cultivated a habit of surrendering various life details in exchange for the powerful tool, colloquially known as the recommender system, in coping with information overload. However, as the awareness of privacy issues continues to grow, more and more general data protection regulations and consumer privacy acts are being implemented worldwide [40, 41, 42].

Traditional recommendation methods, e.g., matrix factorization [43], infer users' preferences on items using only the implicit or explicit feedback data. Over the years, newer models have integrated more contextual information to generate better and more accurate predictions. From timeSVD++ [3], factorization machine [1], to the neural network with attention-based novel RS models [44], recommendation methods keep including more forms of sensitive user data.

Furthermore, privacy concerns have started to arise from the framework of traditional recommender systems. In the area of POI recommendations, users employ their end devices for restaurant and gas station recommendations, as well as for many other places that might interest them. While the users enjoy the convenience brought by a recommender system, it collects sensitive data, such as private user profiles or interaction records from personal devices, and stores them in a centralized server. Should there be any data breach or mismanagement, all private data may be at risk, which could lead to severe consequences.

Over the years, researchers have experimented with many different privacy-preserving methods to lower the risks of privacy breach issues. For example, decentralized RS aims to push off the computation entirely to the users' end [17], removing the need for saving any sensitive data. In contrast, the k-anonymity based-RS, which keeps the central server, hide user identities and broadens certain sensitive information [45]. Other methods, such as obfuscation, focus more on perturbations and noise injection [46]. However, each technique still faces unique challenges today. For example, K-anonymity-like algorithms are especially vulnerable to reverse engineering when platforms purposely collect any available contextual user information; decentralization requires more communication among users and more computing powers from each device; Obfuscation, although easy to implement, may cause the original data to be no longer available.

Previously a group-based RS was proposed where ratings are spread out among user devices to accumulate sufficient information for forming groups in order to enhance identity obfuscation [47]. The performance showed a reasonable trade-off compared to its centralized counterpart demonstrating the effectiveness of user groups in POI recommendation.

This research presents the cascade recommender system (CRS), a novel framework with a cascade structure that integrates and processes sensitive data differently at various levels while maintaining high prediction accuracy. Generally speaking, the data in the proposed model are processed on three different levels. Firstly, the research uses clustering methods to replace users with centroids based on users’ feedback on their visited POIs and estimated locations. At the same time, the research extracts user reviews from local POIs to calculate POI similarities using the Doc2Vec method [48]. Secondly, the research has conducted the recommendation algorithm using the centroids’ information to generate several data fragments for users to download in order to reconstruct an imputed centroids’ preference matrix. Finally, after the reconstruction, the users’ mobile devices will impute the users’ missing preferences to compare against their own, thus obtaining the final recommendations. Based on two large-scale location-based social networks (LBSN) datasets, the test results show a satisfying accuracy of POI recommendations under a secure framework.

the main contributions to this research are summarized as follows:

- The research has proposed a cascade recommendation framework for POI recommendation, whereby sensitive user information is processed at different levels to accommodate the gradually unsecured environment. The multiple aggregator servers added to the model ease the extra computational privacy cost on the central server and user devices.
- The research proposed to sever the connection between contextual information, such as text comments and user identity, while keeping it serviceable. The model in this research collects and uses such information, but when the processed information is fed to the central server, the highly processed information cannot be retraced to the customers.
- The research conducted experiments on real-world publicly available datasets, and the results demonstrate the effectiveness and a reasonable trade-off.

The remainder of this part’s research is organized as follows. The immediate next section discusses topics pertinent to the proposed research. “Proposed Model and Methodology” depicts the problem and the approach. In the section “Datasets and Experiment”, the datasets and results are presented. Finally, the summary will conclude the accomplishment of this era’s research.

4.2 Exploration on Related Work

This section presents some related background context and techniques which were utilized to develop the model, including conventional recommender systems, privacy-preserving recommender systems, user comments embedding, and clustering methods.

Conventional Recommender Systems

There are a variety of models and approaches to the traditional point-of-interest (POI) recommender system (RS), and various standards can categorize them differently.

For example, a broad categorization of the RSes can be content-based RSes [49], collaborative filtering (CF)-based RSes [50], and hybrid RSes that are a combination of multiple kinds. Collaborative filtering, one of the most well-known recommendation techniques, focuses on finding user preferences from similar user-item interactions. After reviewing enough feedback, these RS models carry out a set of predictions by filtering down the entire item set for a new user using the knowledge obtained from similar users. The models assume that users with similar behaviors tend to interact with similar items or locations. Within CF-based RS models, there are a large number of variants, including the matrix factorization (MF) models [43], regression-based latent factor models [51], Bayesian personalized ranking models [52] and deep MF models [53].

Due to the nature of POI recommendations, POI RS is heavily associated with LBSN. Figure 4.1 shows the diagram of data flow for a POI recommender system based on LBSN. Aside from collecting direct user feedback, more contextual information, including sensitive personal data, is also being stored.

In recent years, researchers increasingly focused on analyzing auxiliary information from users and POIs. Although RS models mainly use user feedback directly to predict future user actions, integrating auxiliary information vastly increases the models' performance. Thus, in another dimension, traditional recommender systems can also be categorized by the auxiliary information it focuses on to enhance the results. The earlier auxiliary information on which most researchers focused was the temporal effect, such as users' check-in time and visiting frequency [3]. This kind of study first focused on the temporal effect independently (time-aware RS) and later successively (sequence-aware RS) [54]. Meanwhile, the geographical information is an especially important trait for POI recommendations [55] and many attractive traits uniquely appear in this field. For example, users are usually active in only one city, and their visiting behaviors are often bound by where they live [17]. Moreover, the users' text comments also play an increasingly critical role in POI RSes due to the advancement in the field of natural language processing. Not only can researchers more accurately predict user behavior, but they are also able to make the machine decisions explainable [44]. Lastly, even though integrating user social relations is not as popular as previously mentioned auxiliary information, some researchers improved their results with the help of social network information [56]. Indirect methods such as group-based POI recommendations have also found ways to use the relationships among individuals and the group in which they are to boost prediction accuracy [57]. Recently, correlations have also been found between long-distance visits and social relations [58].

Privacy-Preserving Recommender Systems

Because of the ever-increasing privacy risks that recommender systems face, researchers have proposed different methods and models over the years. The first approach is through distribution. Two typical examples were entirely distributed recommender systems [17] and the federated recommender systems [59]. The former pushes the computation burden entirely to the user end, meaning that each user has

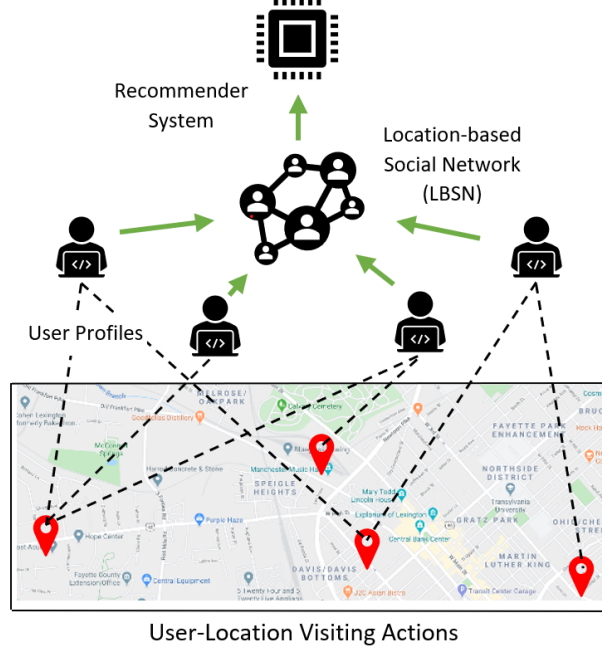


Figure 4.1: The Framework Of Traditional Centralized Point-of-Interest Recommender Systems

a separate private data and recommendation model. In contrast, the latter only requires the private data to be stored on each user’s device while a central server learns the neural network through gradient aggregation. The second most popular method is obfuscation, such as randomized perturbation. The idea is to protect users’ original private data by adding extra noise. As a traditional privacy-preservation technique, it has been frequently used in RS models [60, 46, 61, 62]. Last, the third type is by adding cryptographical protocols to secure users’ identities and data transmission [63, 64, 65, 66]. The advantage is that there is no or little sacrifice of the performance of the RS model, but the downside is it requires extra computation power or time.

As mentioned before, in the previous work [47], a two-layered system was proposed, employing clustering and grouping as a means for obfuscation. Instead of focusing on improving the prediction accuracy of individuals’ preferences as in [57], the grouping mechanism was implemented as a way for further concealing users’ identities. While the experiment showed a promising trade-off between the privacy-preservation feature and prediction accuracy, both of them have much room for improvement.

Despite their differences, privacy-preserving recommender systems (PPRS) share the same categorizations, compositions, and challenges since they are derived from traditional recommender systems. However, from the perspective of privacy preservation, the privacy of the utilization of auxiliary information has not been studied as thoroughly as direct user feedback, such as user ratings or approvals. This research share ways for protecting and safely collecting different contextual information in a POI recommender system.

Clustering Method

Data clustering is a popular data mining technique as well as a way to mask or obfuscate private user information. Since clustering is an integral part of the experiment, the research briefly mentions the differences among the grouping algorithms, including k -means, spectral [67], DBSCAN [68], and fuzzy c -means [69].

k -means, probably the most well-known clustering algorithm, groups n observations into k clusters with the nearest mean. On the other hand, spectral clustering treats the grouping task as a graph partition task. Both k -means and spectral clustering require a predefined number of groups. In addition, DBSCAN clustering is based on the number of observations in a certain radius, i.e., density. Compared to the former methods, it is better at eliminating outliers and, most importantly, does not require a predefined k . Lastly, the fuzzy c -means (FCM) algorithm is a soft clustering algorithm because it assigns observations to different clusters by percentage instead of assigning an observation entirely to one group.

It is worth noting that relying on customers to be physically close to each other while there is always Wi-Fi Direct can be challenging in real-world scenarios. Therefore, compared with the work in [47], the research adds a layer to use aggregator servers to aggregate user information actively. Consequently, to reflect this change, the research needs users' GPS information to establish the hypothetical locations of aggregator servers. Since this information is not readily available, especially in this dataset, the research uses the user-visited POIs' public GPS information to estimate their locations. However, the original k -means can no longer satisfy the current system due to how user affinities are constructed. To still perform the clustering, the research tested extensively on clustering methods, especially DBSCAN, which automatically decides the number of clusters but finally chose to use spectral clustering, which is more potent for locational classification.

The finished framework does not involve all of the clustering methods mentioned above, but the choice of picking is based on the comparison of each one's final results.

User Comments and Word Embedding

In most e-commerce and LBSN websites, users can write free-text reviews along with an explicit or implicit rating. The text comment feedback can be very informative in the sense that it contains the evaluation and subjective opinions toward a product or a place from multiple perspectives. The review information weighs separately in potential customers' minds when choosing the next place to visit. This is especially true in POI recommendations since there are often limited ratings for unpopular places.

Some studies, such as [70, 71], focus on preprocessing user comments, and these models aim to separate normal comments from "bad" comments (irrelevant comments, advertisements, or scams). Others, such as [72, 73, 74], integrate user comments to affect the latent factors for users and items while training. However, better results have been found for integrating speech recognition and natural language processing (NLP) [75]. In [44], Chen et al. proposed a neural attentional regression model

with review-level explanations (NARRE) to evaluate user reviews while generating accurate predictions.

In this model, word embedding is used to find the association among the POIs with similar comment sections. The term "word embedding" refers to the problems in NLP where we want similar vectors to represent similar texts. Specifically, the research used "Doc2Vec" [48], an unsupervised algorithm to generate vectors for a document to convert user comments to vectors. This algorithm is an extension of the algorithm "Word2Vec" [76] that generates vectors for a single word.

The motivation for integrating such a feature is because privacy-preserving POI recommender systems such as [47, 17, 59] tend to ignore such commonly existing LBSN resources. Additionally, the clustering processes in this framework narrow down available user records, rendering other contextual information more invaluable. However, there is a strong connection between user comments and their identities. Thus, "Doc2Vec" came into place to vectorize the text information, making it impossible to convert the vector back. Additionally, the evaluation method for such vectorization is challenging to establish, considering the already complicated system. Different text datasets were tested on the training of the "Doc2Vec" model, such as comments from the same city, general words, or nationwide POI comments. Eventually, it was decided to experiment using the non-private POIs' similarities based on vectors to regularize the private features in the loss function.

Subsequently, the system has many moving parts, and an optimal controller becomes immediately necessary, which will be discussed in the later sections.

4.3 Proposed Model and Methodology

This section introduces the model from its overall structure to the notations, as well as problem definitions. Meanwhile, the details of the framework are depicted as well, and the formulation is explained.

Overall Structure

As Figure 4.2 shows, physically, there are three significant components in this model. The first part is the users' mobile devices, where raw data are preprocessed, and final recommendations are generated. In each user's POI recommendation application, the device keeps records of their private information, such as the locations they visited and the comments they left. Specifically, each device needs to carry out three data processing tasks corresponding to each type of private data. (1) Each user's comments are vectorized using the embedding tool "Doc2Vec". (2) The mobile device uploads user feedback through secure ad hoc P2P Wi-Fi Direct to the nearby aggregator server. (3) The user's device uploads a package containing recent anonymized ratings and matching vectorized comments. In the first component, the environment is comparatively secure, and the data process level is low, meaning that it is closer to raw private user data.

Additionally, it is worth mentioning that mobile devices are also in charge of generating random user IDs when the connection is established. In other words, even

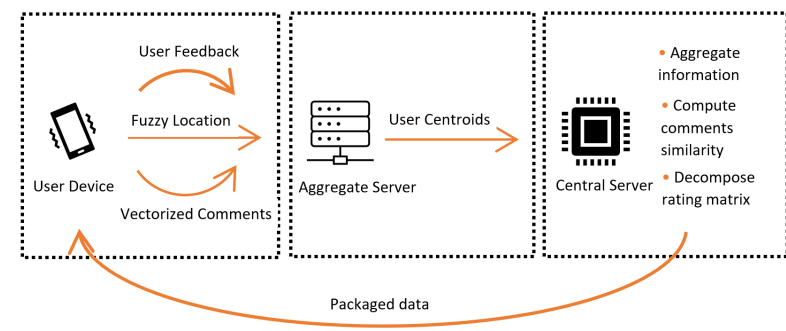


Figure 4.2: The Overall Model Structure, the Tasks of Each Component, and the Data Flow.

when multiple connections were initiated by the same user at the same place over a particular time, the user cannot be identified.

The next component’s primary task is user clustering on the aggregator server. The clustering is based on each user’s uploaded ratings and their estimated GPS locations using the locations of the POIs they visited. Note that the framework does not require the user’s address, nor does the framework require any information on the real-time user location. When an aggregator server collects enough ratings, e.g., 1,000, it then performs the clustering and transfers the results to the central server. Moreover, user ratings and their comments are detached and uploaded separately. In this component, the data are further processed to cope with the less secure environment.

In the third component, its task is similar to a conventional non-privacy preserving RS. The centroids collected from all the aggregator servers are treated as physical users, and the model is then trained with “user” ratings and comments. At this point, the trained model should have generated all the necessary fragments, such as lower-ranked decomposed matrices from the rating matrix factorization, to reconstruct the rating matrix. In order to receive recommendations, users need to download these fragments to reconstruct the imputed rating matrix on their devices. In the last component, the central server, sensitive data such as user ratings, locations, and reviews are substantially processed to minimize the potential data breach risks.

Notations and Problem Definitions

Suppose we use u to denote a user (customer) and i an item (POI); then U and I are the user and item sets, where $u \in U$ and $i \in I$. A rating r_{ui} indicates the preference of user u over item/POI i . In our datasets, each rating $r_{ui} \in [1, 5]$ where 1 indicates the least favored and 5 indicates the most favored. Furthermore, we can use \hat{r}_{ui} for predicted ratings, and r_{ui} for their observed counterparts. All the observed and unobserved r_{ui} together form the matrix R . Now, if we decompose it into two lower-ranked matrices, the column vectors p_u and q_i of each matrix represent the user and item latent factors, respectively.

A generic matrix factorization procedure can be regarded as an optimization problem, as shown below:

$$\min_{p_u, q_i} \sum_{r_{ui} \in R} (r_{ui} - \hat{r}_{ui})^2 + \lambda(\|q_i\|^2 + \|p_u\|^2) \quad (4.1)$$

In this framework, however, user comments are taken into consideration on the central server, and it is necessary to use centroid ratings instead of user ratings. The rationale behind these changes to the above objective function is that there may be potential patterns that each user follows when reading other users' comments to make their decisions to visit. Therefore, the research intends to reflect the weight of such effects by introducing POI similarities calculated from the vectorized user comments. If we define s_{ij} as the similarity between two POIs i and j based on the user impression of their comment section, we have the similarity matrix S_c , and the updated objective function is as follows:

$$\min_{p_u, q_i} \sum_{u, i \in R} (r_{ui} - \hat{r}_{ui})^2 + \alpha(\|b_u\|^2 + \|b_i\|^2 + \|p_u\|^2 + \|q_i\|^2) + \beta \sum_{j \in N(i)} s_{ij}(q_i - q_j)^2 \quad (4.2)$$

where $s_{ij} \in S_c$; b_u , b_i and $N(i)$ represent the user bias, item/POI bias and the set of item i 's neighbors, respectively; α and β are two different learning rates of the regularization terms. If we define $e_{ui} = r_{ui} - \hat{r}_{ui}$, then we have the following update functions. Suppose we let \mathcal{L} represent the loss function above; then, the gradients with respect to p_u and q_i can be calculated as follows:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial p_u} &= \frac{\partial}{\partial p_u} ((r_{ui} - p_u \cdot q_i)^2 + \alpha \|p_u\|^2) \\ &= -2q_i(r_{ui} - p_u \cdot q_i) + 2\alpha p_u \end{aligned} \quad (4.3)$$

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial q_i} &= \frac{\partial}{\partial q_i} ((r_{ui} - p_u \cdot q_i)^2 + \alpha \|q_i\|^2) + \beta \sum_{j \in N(i)} s_{ij}(q_i - q_j)^2 \\ &= -2p_u(r_{ui} - p_u \cdot q_i) + 2q_i(\alpha + \beta \left(\sum_{j \in N(i)} s_{ij} \right)) - 2\beta \sum_{j \in N(i)} s_{ij}q_j \end{aligned} \quad (4.4)$$

We use θ to globalize all terms' learning rate; then, the corresponding update protocols are as follows:

$$b_u \leftarrow b_u + \theta(e_{ui} - \alpha b_u) \quad (4.5)$$

$$b_i \leftarrow b_i + \theta(e_{ui} - \alpha b_i) \quad (4.6)$$

$$p_u \leftarrow p_u + \theta(e_{ui} \cdot q_i - \alpha p_u) \quad (4.7)$$

$$q_i \leftarrow q_i + \theta(e_{ui} \cdot p_u - (\alpha + \beta \left(\sum_{j \in N(i)} s_{ij} \right))q_i + \beta \sum_{j \in N(i)} s_{ij}q_j) \quad (4.8)$$

After settling the update protocol, we can use the classic stochastic gradient descent (SGD) to optimize the aforementioned objective function. In addition, it is essential to mention that even though the research uses centroids' ratings on the cen-

tral server, all centroids are still referred to as users to prevent excessive notations. However, in order to obtain all the centroids, a clustering task must be conducted on the aggregator server, as mentioned previously. Various clustering methods have been tested, and kernel spectral clustering was chosen in this research due to its better performance in terms of prediction accuracy.

Furthermore, in order to perform such clustering, the research needs to first construct the affinity matrix S by calculating the users' similarities. Suppose S_r is the user similarities based on user ratings, while S_l is based on estimated user locations. We then have the following equation:

$$S = \alpha' S_l + (1 - \alpha') S_r \quad (4.9)$$

where α' is the weight ratio of the two similarities and S_r is imputed based on the Pearson correlation coefficient (PCC) of users' ratings. Specifically, if we use μ to denote user mean rating, for any pair of users a and b , the similarity between the two is defined as:

$$s_{ab_PCC} = \frac{\sum_{i \in I_{ab}} (r_{ai} - \mu_a) \cdot (r_{bi} - \mu_b)}{\sqrt{\sum_{i \in I_{ab}} (r_{ai} - \mu_a)^2} \cdot \sqrt{\sum_{i \in I_{ab}} (r_{bi} - \mu_b)^2}} \quad (4.10)$$

On the other hand, to complete the construction of S_l , the users' locations need to be estimated using the GPS locations of their previously visited POIs. If we let Lat_u and Lon_u represent the latitude and longitude of a user, respectively, and I_u represents the set of all POIs that a user visited, then we have the following equations:

$$Lat_u = \frac{1}{|I_u|} \sum_{i \in I_u} Lat_i \quad (4.11)$$

$$Lon_u = \frac{1}{|I_u|} \sum_{i \in I_u} Lon_i \quad (4.12)$$

where I_u denotes all the POIs that the user u has visited.

Finally, after the clustering, the aggregator servers can upload the centroids to the central server to finish the previously mentioned optimization according to Equation (4.2). Afterward, the users can download b_u , b_i , p_u and q_i to their personal devices to reconstruct the complete centroid rating matrix \hat{R} . To receive personalized recommendations, a user's personal device needs to calculate the weighted average ratings of the top similar centroids by checking its private rating records against all centroids, thus applying the following equation:

$$\hat{r}_{\bar{u}i} = \sum_{v \in U} \bar{s}_{\bar{u}v} r_{vi} / \sum_{v \in U} \bar{s}_{\bar{u}v} \quad (4.13)$$

where \bar{u} is the target user and \bar{s} is the vector containing similarities between the target user and all centroids. The similarities were also calculated according to Equation (4.10).

Furthermore, let's use T_e and T_r to denote the test set and training set in later sections. For a complete summary of the notations, please refer to Table 4.1.

Table 4.1: Notation summary.

Notation	Description
U	User (centroid) set
I	Item (POI) set
R	Rating matrix
r	Observed rating
\hat{r}	Predicted rating
p_u	The column vector representing the latent factors of user u
q_i	The column vector representing the latent factors of item i
b_u	User (centroid) bias
b_i	Item (POI) bias
$N(i)$	Neighbor set of item i
\mathcal{L}	Loss function
e	The error between the observed rating and predicted rating
θ	Global learning rate
s_{ij}	Similarity between item (POI) i and item (POI) j
S_c	Similarity matrix based on comments
S_r	Similarity matrix based on ratings
S_l	Similarity matrix based on locations
μ_a	The mean rating of all ratings from user a
Lat_u	The estimated latitude of user u
Lon_u	The estimated longitude of user u
\bar{u}	A real user (cannot be considered a centroid)
\bar{s}	The similarity vector between a real user and all centroids
T_r	Training set
T_e	Test set

Implementation

In order to better illustrate the implementation of the theory mentioned at the beginning of this chapter, It is more suitable to compare hypothetical real-world scenarios with the conducted experiment.

First of all, the research needs to sort all datasets in temporal order from the perspective of time awareness. In reality, one cannot depend on interactions that happen in the future to find the affinities between current users to perform personalization. Thus, the research puts user records into a timely sorted “buckets” list, where each bucket represents a time window.

In the previously described theory, the users share anonymized personal information via a secure ad hoc P2P network with the nearest aggregator server, which naturally reflects the user’s active visiting area. Since there are no physical user devices in the experiment, this step is reproduced by estimating the users’ GPS lo-

cations using the geographical information of the visited POIs. At the same time, the research has vectorized the text comments left by device owners, removing any trace before leaving their host devices. In the implementation, “Doc2Vec” is used to convert user text comments into various vectors. It is worth noting that since the user IDs are anonymized, geographically or temporally separated data sharing will be considered from different users.

The proposed theory devotes a lot of effort to the clustering of users. This is not only to obfuscate the users’ identities further but also to find the “trend” of the users of the present location. To implement this part of the experiment, the research used users’ shared ratings in the current “bucket” and their estimated locations to calculate the similarities among the users. After the clustering, a set of centroids will be generated on the aggregator servers and sent to the central server. In the experiment, the research treats the centroids as new users with index-based IDs.

Then comes the matrix factorization on the central server, where there is no difference between the real-world scenario and the experiment. This step focuses on improving the imputation of the rating matrix and dimensionality reduction. It will be faster when users download the smaller-sized required “fragments” to rebuild a complete rating matrix.

Finally, the users can use their protected private information to compare against the imputed centroids rating matrix to see what “trends” they belong to more. In the experiment, the top similar centroids’ ratings will be extracted, and a weighted mean is taken to impute the current user’s rating vector.

4.4 Datasets and Experiment Results

This section first introduces the datasets used for training and testing the cascade recommender system (CRS) model. After the introduction of the metrics and evaluation methods, the proposed CRS, another two privacy models, as well as a non-privacy model are empirically compared by the same standards and procedures. Lastly, it will show the tuning methods and the effects of parameters on model performance.

Datasets

Two areas are chosen from Yelp’s real-world POI feedback dataset, Champaign–Urbana metropolitan area and the city of Phoenix [21] (<https://www.yelp.com/dataset>). Both sets fetched the information between January 2007 and December 2017. The Champaign–Urbana area was chosen for its small scale and concision. In contrast, Phoenix is one of the areas with the most enormous amount of feedback data in the entire Yelp business review dataset. However, both are significantly sparse compared to conventional RS datasets such as MovieLens [77]. The statistics of the two areas can be found in Table 4.2.

Moreover, there is no repeated feedback meaning that each user can only rate a place once. Figure 4.3 shows the distribution of the user rating numbers from Phoenix with a base 2 logarithmic scale on the y axis.

Table 4.2: Datasets statistics.

Area	User	POI	Rating	Density
CU	11,953	1,579	33,990	0.1802%
PH	204,887	17,213	576,700	0.0163%

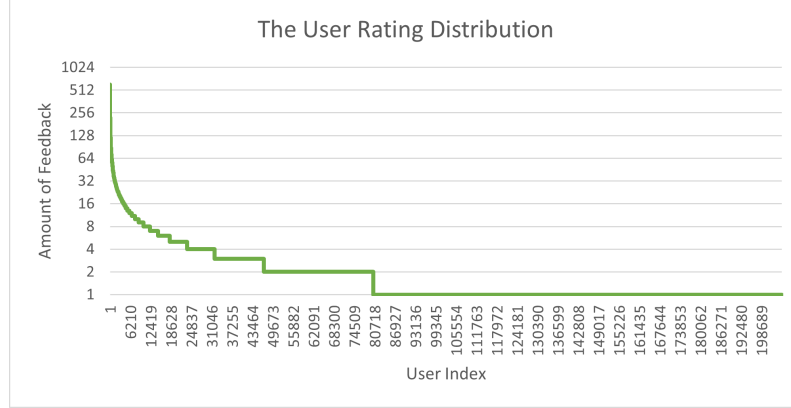


Figure 4.3: The User Rating Distribution (Phoenix). (The X axis represents the numerical user IDs starting from 1. For example, if we have 10 users, they Will be numbered from 1 to 10. The Y axis is the number of ratings they have left, and it is on a log scale to show a clearer distribution.)

The datasets contain features such as user IDs, POI IDs, explicit rating feedback, timestamps, user text comments, and POI GPS information. As previously noted, an explicit rating can only be $\{1, 2, 3, 4, 5\}$. Since Yelp does not collect users’ GPS information in real-time, we can only estimate their active visiting area. For privacy reasons, the estimation is necessary both in the experiment and in practice. Assuming that all users decided to disable GPS tracking, the location of POIs can be utilized instead. Since the GPS information of each POI is not private and publicly available, it is more than viable to be utilized as a source to estimate each user’s fuzzy position without privacy concerns. For example, if we determine that a user is only active in the downtown area, it is helpful to eliminate some unnecessary recommendations in suburban areas. Figure 4.4 shows the plot of the POI locations from which we can almost see the city’s shape.

Metrics and Evaluation Methods

We use both root mean square error (RMSE) and mean absolute error (MAE) to evaluate the model’s prediction accuracy to better study the trade-off between privacy preservation and the RS model’s accuracy. The following Equations (4.14) and 4.15 show the details of the definition:

$$RMSE = \sqrt{\frac{\sum_{u,i \in T_e} (r_{ui} - \hat{r}_{ui})^2}{n}} \quad (4.14)$$

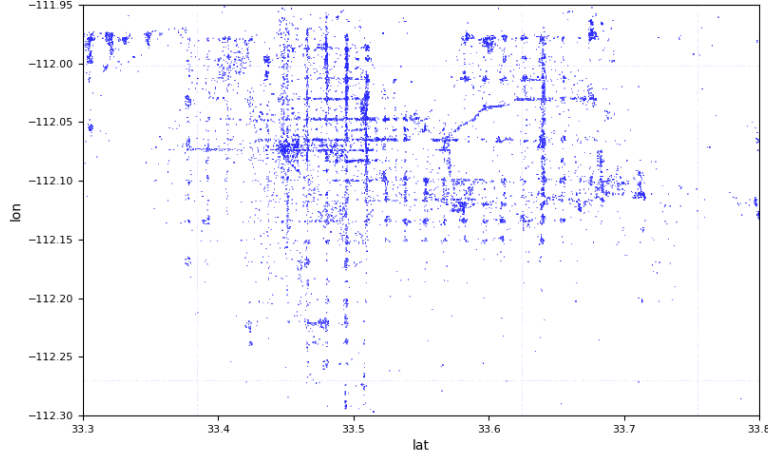


Figure 4.4: The User Visiting Locations (Phoenix). (Abscissa: the latitude; Ordinate: the longitude)

$$MAE = \frac{\sum_{u,i \in T_e} |r_{ui} - \hat{r}_{ui}|}{n} \quad (4.15)$$

where T_e is the entire test set and n is the number of ratings compared.

Meanwhile, the experiment was designed in a way to reflect real-world scenarios. First, a time-dependent cross-validation method is conducted, ensuring the time dependencies among all the training–test pairs. For any training set, the ratings given by the users are in an earlier time state than any ratings in the corresponding test set. Moreover, the window size of each training set is built through the number of ratings uploaded by the aggregator servers.

Second, the user’s GPS information is estimated by the locations of POIs that this user visited to avoid the need for private user information. Since the locations of POIs are not private, an estimation of a user’s active area using such information is sufficient, considering the correlation between a user’s physical location and their active visiting areas.

Third, all comments in the experiment can be collected beforehand due to its non-private nature, better assisting the “Doc2Vec” neural network’s training. Unlike the older centroids’ preference information, the information from comments can be accumulated due to their less significant temporal effect on semantics. In reality, even comments from different areas can be collected and utilized simultaneously to build a more robust word embedding model collaboratively. Algorithm 2 describes the procedure of model training and testing.

Results and Comparison

This section compares the framework with three other models: a baseline model, a purely decentralized model, and a federated RS model. Afterward, it briefly shows the overall improvement in contrast with preliminary work in [47].

Algorithm 2: CRS Model Training/Testing Algorithm

Input: Preprocessed training sets $\{Tr_1, Tr_2, \dots, Tr_n\}$, preprocessed test sets $\{Te_1, Te_2, \dots, Te_n\}$, the POI similarity matrix S_c , and the POIs' GPS information

Output: MAE, RMSE

```
1 for each training set  $Tr_i$  do
2   Calculate each user's longitude  $Lon_u$  and latitude  $Lat_u$  according to
   Equation (4.11) and Equation (4.12)
3   Construct affinity matrix  $S$  based on Equation (4.9) and Equation (4.10)
4   Perform clustering to generate the centroids' rating matrix  $R$ 
5   for each  $r_{ui}$  in  $R$  do
6     Calculate the gradients of  $\mathcal{L}$  with respect  $b_u, b_i, p_u$ , and  $q_i$ 
7     Update iteratively according to Equation (4.5), Equation (4.6),
     Equation (4.7), and Equation (4.8)
8   end
9   Reconstruct the centroids' rating matrix  $\hat{R}$ 
10  for each  $r_{\bar{u}i}$  in the corresponding  $Te_i$  do
11    Calculate the corresponding  $\hat{r}_{\bar{u}i}$  according to Equation (4.13)
12  end
13  Calculate  $RMSE_i$  and  $MAE_i$  according to Equation (4.14) and
    Equation (4.15) (We calculate  $RMSE$  for the convenient averaging)
14 end
15 Calculate the average  $RMSE$  and  $MAE$ 
16 return  $RMSE$  and  $MAE$ 
```

- It starts by choosing the well-known biased matrix factorization (MF) [43] as the baseline model. It is a non-privacy centralized RS model with simple structures and reliable performance, and this model was thoroughly studied and constantly compared.
- The federated recommender system, MetaMF [59], has a distinct advantage in terms of privacy protection and providing exceptionally accurate predictions. Despite requiring a global model on a central server, this method distributes its training to multiple local devices, reducing the risk of leaking private user data.
- The decentralized recommender system (DMF) [17] is a purely distributed privacy-preserving recommender system that relies on matrix factorization and gradient exchange. Due to its distributive framework, the only leakage risk of data is the loss function's gradient. Each user has a dynamic global and personal model combined, making the hacking even harder.

I have compared the CRS with the three aforementioned models with the classic biased MF as the baseline. For fairness, the data filtering methods and preprocessing are the same for all models involved. The overall results are presented in Table 4.3. It is worth noting that even though the scales of the two areas are different, the

Table 4.3: Average Result Comparison (Champaign–Urbana).

Champaign–Urbana				
Model Name	Biased MF	CRS	MetaMF	DMF
RMSE	1.1966	1.1685	1.3942	1.4984
MAE	0.9537	0.9494	1.1268	1.2018

Phoenix				
Model Name	Biased MF	CRS	MetaMF	DMF
RMSE	1.0463	1.0482	1.3647	1.4534
MAE	0.8247	0.8235	1.0647	1.0872

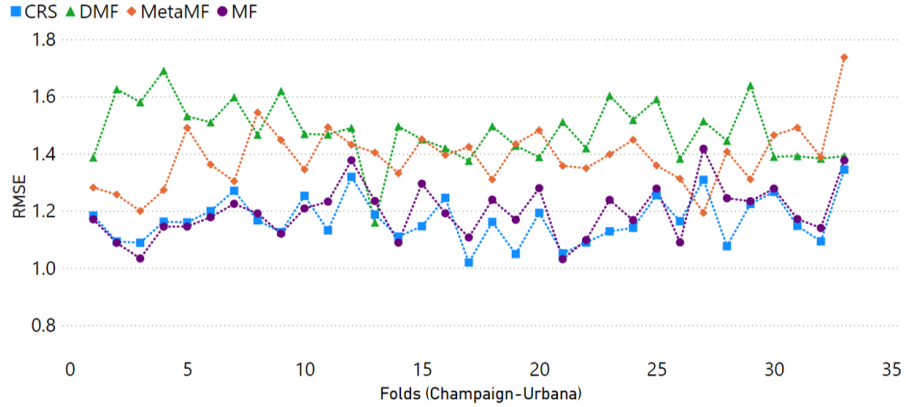


Figure 4.5: The RMSE Comparisons among the Four Models for Each Training-Test Dataset Pair (Champaign–Urbana). (Abscissa: the value of RMSE; ordinate: index of folds).

research strives to show the same detail in the results, presenting the results from each training–test pair. The fold size is 1,000 ratings for the dataset of Champaign–Urbana and 3,000 ratings for the city of Phoenix. The RMSE and MAE of each fold are shown in Figures 4.5–4.8.

Among the four compared models, the CRS has the best performance in terms of accuracy. The CRS is close to the baseline, the non-private centralized recommendation model. In specific folds, the performance of the CRS is even better. All the models use the same training and test datasets. The following subsection will discuss the tuning of models and hyper-parameters.

The testing procedure mimics real-world scenarios where recommendations are temporally given in succession. In other words, only old data can be used to predict newer data. The previous test set will be the training set in the subsequent round of testing. As the result figures have shown, each inflection point reflects a result from a training–test pair. To maintain a steady curve of accuracy, the research chooses to maintain the same number of ratings in all of the training and test sets instead of having the same time span from user ratings. This is also to ensure that we have

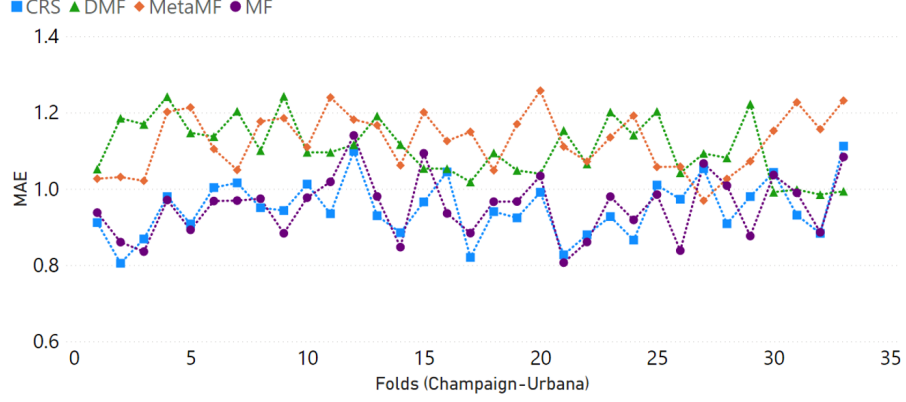


Figure 4.6: The MAE Comparisons among the Four Models for Each Training-Test Dataset Pair (Champaign-Urbana). (Abscissa: the value of MAE; Ordinate: index of folds).

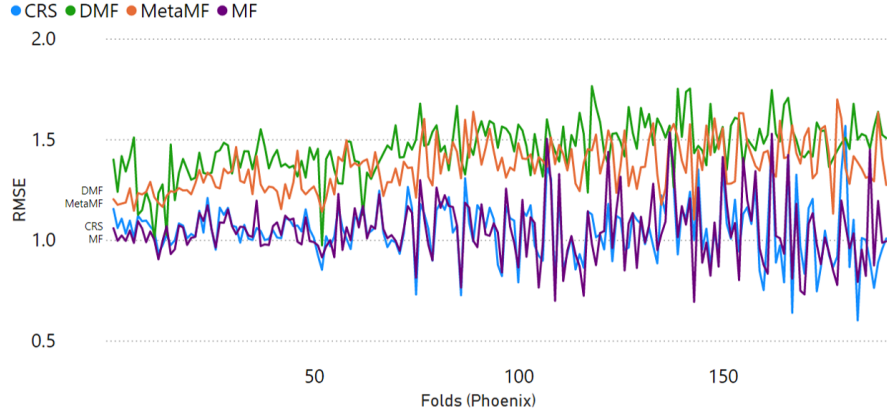


Figure 4.7: The RMSE Comparisons among the Four Models for Each Training-Test Dataset Pair (City of Phoenix). (Abscissa: the value of RMSE; ordinate: index of folds).

enough records for each time period for generating recommendations when simulating the real-world scenario.

For both RMSE and MAE, the CRS shows a very steady and good performance on both datasets in an environment that is very close to real-world settings. Compared to DMF, the CRS does not push all the computing burdens onto the users' end, ensuring an easier, more light-weighted way of generating recommendations. At the same time, the CRS's central server relies on the data that are already processed by the aggregator server, making the model's training process more accessible and faster. The CRS is also more modularized and detachable compared to the federated learning-based recommender systems. In comparison to its previous prototype [47], the average value of MAE falls from 1.118 to 0.949.

In terms of privacy preservation, unlike other privacy RS models, users in CRS anonymously and dynamically share private information with the aggregator server on an ad hoc P2P network via Wi-Fi Direct. In other words, each time a user

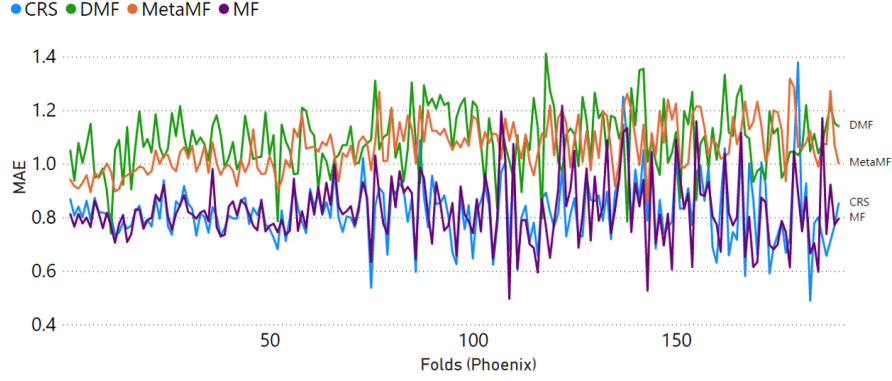


Figure 4.8: The MAE Comparisons among the Four Models for Each Training-Test Dataset Pair (City of Phoenix). (Abscissa: the value of MAE; ordinate: index of folds).

shares information, their IDs are temporary and different. Moreover, contextual information such as text comments and GPS information is protected and preserved. Other privacy-preserving models, such as DMF and MetaMF, require that users have a constant identity in order to maintain long-term communication.

Parameters and Tuning

The previous sections introduced the CRS model from three aspects: the users' mobile devices, aggregator servers, and the central server. Since various parameters can directly affect the result in CRS, we can connect the discussion of parameters with these three parts correspondingly for easier understanding.

- During the aggregation stage, there are two crucial parameters. First, the performing of the clustering is based on the affinity matrix according to Equation (4.9). In this equation, α' denotes the ratio or weight of either part in building the final affinity matrix. Second, the number of centroids, $n_clusters$, is required due to the clustering method we chose.
- On the central server, we have the number of latent factors k , the global learning rate θ , and each specific learning rate α and β of each term from the loss function. the research used n_epochs to denote the number of epochs used in SGD for each training process.
- On users' mobile devices, after users reconstruct the imputed rating matrix, they will compute their personalized predictions according to Equation (4.13). During the experiment, it was found that not including all centroids yields more accurate results. Here, let us use $n_centroids$ to denote the number of centroids involved.

Due to the number of parameters, broad intervals, and contentiousness, the research seeks an automatic approach to model tuning instead of brute-force searching or heuristic random guessing based on human experiences. In the field of optimal

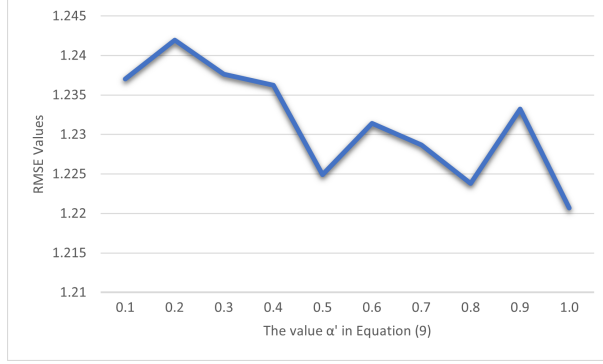


Figure 4.9: The Impact on the Average RMSE Value from Changes on α' in Equation (4.9) (Champaign-Urbana). (The α' indicates the weight of geographical similarities among the users played in the affinity matrix S .)

controls, researchers have made significant progress during the past decade on adaptive control schemes and solving nonlinear systems [78, 79]. However, since this is the initial attempt to adopt such methods, the strategy is to combine the use of a developed library with stable performance [80] and manual intervention. Therefore, the research adopted a Bayesian optimization framework wherein the model’s performance is treated as a sample from a Gaussian process. This approach, proposed by Snoek et al. [81], considers the entire training–test process as a continuous function.

According to the authors, the Bayesian optimization method reduces the heavy computation task of finding the close-to-optimal combinations of parameters by using proxy optimization. In the beginning, a function was constructed that wraps up the entire algorithm (Algorithm 2). In other words, instead of optimizing each part of the framework, the research includes all parts of the algorithm and puts them into one “function.” The number of observations keeps growing as the program runs, and a posterior distribution is gradually built. Each time a parameters-error pair is plotted, exploration strategies such as the upper confidence bound and expected improvement are used to determine the next set of parameters to try. The number of steps to find optimal parameters is claimed to be minimized compared to a brute-force strategy. Specifically, in the experiment, a method to estimate the range of optimal parameters was used in the tuning process and to perform manual fine-tuning. For example, the research would combine the observed parameters from different top-ranked results and increase or decrease specific parameters to see whether a better performance can be achieved.

In CRS, the model has set α' to 0 to maximize the effect of user location. However, this does not mean it is always a better choice to exclude all contributions from the similarities based on user feedback. For example, Figure 4.9 shows the comparison of the final RMSE by selecting different α' in 1.0, 0.9, 0.8 ... 0.0. $n_clusters$ is set to 6. It is worth noting that the best result was achieved when k is 1. This may be caused by the sparsity of the dataset and the fact that the data are clustered. For the coefficients of the loss function, θ , α and β are set to 0.12, 0.02 and 0.6, respectively. For each round of training, $n_centroids$ is as small as 20 to achieve convergence. Finally, $n_centroids$ is set to 9.

4.5 Summary

This research has proposed a privacy-preserving POI recommendation framework CRS that utilizes ad hoc wireless peer-to-peer communication and user clustering. The CRS processes sensitive user data at various levels in a gradually less secure environment, replacing users with centroids based on their feedback on visited POIs and estimated locations. This method severs the connection between contextual information, like text comments, and user identity while still keeping it useful for the system. By adopting a cascade structure, the CRS processes risky data under secure connections and uploads processed secure data when facing a potentially malicious environment.

The CRS model has three primary components: users’ mobile devices, aggregator servers, and a central server. Mobile devices preprocess raw data and generate final recommendations, and each device keeps records of users’ visited locations and their comments. User comments are vectorized using the “Doc2Vec” embedding tool, and user feedback is securely uploaded to a nearby aggregator server via P2P Wi-Fi Direct. Notably, each device generates random user IDs during connections to ensure user anonymity.

The aggregator server’s primary task is user clustering based on each user’s uploaded ratings and estimated GPS locations. When an aggregator server collects enough ratings, it performs the clustering and transfers the results to the central server. The central server treats the centroids collected from the aggregator servers as physical users, training the model with “user” ratings and comments. After the model training, the necessary fragments to reconstruct the rating matrix are generated and made available for users to download. Users can then compare these imputed preferences against their own to obtain final recommendations.

Moreover, because of the nature of clustering, CRS has excellent scalability as well as geolocational awareness. Furthermore, it mimics a real-world scenario by preparing the cross-validation folds temporally sequential. Under such a testing strategy alongside real-world datasets, the CRS shows a performance close to a centralized non-private model. This multi-level, cascade structure helps to protect user privacy while maintaining the functionality of the recommendation system.

Meanwhile, the CRS framework does face some difficulties when facing different datasets or uncertain real-world situations. Two significant challenges remain the need for ample input and its severance on the connections between the users and their historical information. The former is caused by the nature of CRS, where constant clustering is required. The latter one, even though protecting the user from being traced, will cap the personalization performance since personal historical records cannot be accumulated to improve the accuracy of the predictions further.

Chapter 5 Towards Privacy-Preserving POI Recommendations: a Generic Data Collection Approach

5.1 Motivation and Problem Description

As the research has constantly stressed, a recommender system is a technique used in data analysis to solve problems related to a surplus of choice. These choices may include an overwhelming number of movies, TV shows, and short videos available on online-streaming services [82]. For instance, customers of online streaming services can choose to share their private information and history records in exchange for a list of content they might enjoy watching, saving the time of going through endless content. On the other hand, corporations can use recommender systems to promote specific content or improve user experience, increasing user stickiness. Such techniques are designed to predict future interactions between users and products that the recommender system is promoting by analyzing information pertaining to the attributes, feedback, and contextual details.

The pace and changes in the development of recommender systems nowadays are fast, and the trend is toward integrating more types and volumes of information. During the short history of recommender systems development, more and more models, techniques, and approaches have been invented and added to improve the accuracy of predictions. From the conventional [43], which utilizes only explicit user feedback, to the xDeepFM [83] model, which uses every available source of information, recommender systems researchers have been trying to employ more collectible information to maximize user satisfaction. For example, POI recommender systems tend to geolocate user devices, track activities across all applications, and gather related social information such as one's contact list.

Consequently, there are increasing privacy concerns due to frequent data breach scandals, homogenous product competition, and international disputes over data ownership. Frequent data breach scandals undermine customers' trust in sharing personal data with private companies. Moreover, since data is invaluable and often tied to national security, certain countries are competing for the exclusive right to its access, putting international corporations in challenging positions. On top of that, regulations make information and data exist in the forms of an isolated island leading to a more unfriendly environment for the recommender systems to collect helpful information. Point-of-Interest (POI) recommender systems, which benefit from location-based services (LBS), naturally have access to rich and diverse data from a large number of mobile user devices. However, due to the growing privacy concerns of users, companies, and many other parties, there has been increased research on preserving privacy in POI recommender systems.

In recent years, a wide range of approaches has been carried out in research on privacy-preserving recommender systems to maintain good performance while minimizing the loss of prediction accuracy. With the approach of decentralization, a local RS can operate on its own, eliminating the need to store sensitive data on the cloud.

For another, anonymization and obfuscation-based recommender systems can protect sensitive information by perturbing or removing identifiable user details from critical user feedback. In addition, many non-private recommender systems use traditional cryptography methods to protect data from unauthorized access. However, many challenges still remain. Decentralized recommender systems require additional communication among users leading to more privacy considerations. More importantly, recommendation service providers lose the ability to gather the information that is vital for future research and usage. A common challenge for many of the aforementioned privacy-preserving models is that each requires a unique and strict framework, rendering them less compatible with newer inventions.

In this time’s research, a generic and flexible framework was proposed that collects user information through Local Differential Privacy (LDP) and location-based clustering to generate synthetic data for POI recommendations. First, the local Recommender System (RS) estimates the users’ location without knowing their true GPS information. An honest but curious (HOB) third party will then perform a clustering based on the estimated user locations while providing differential privacy guarantees for users’ explicit feedback. After aggregating user feedback, the information is no longer stored, and the centroids of historical user records are sent to the central RS as synthetic user data. Finally, users’ local RS will collaborate with the central RS to offer predictions to the users without sharing additional personal details.

The main contributions of this particular research are summarized as follows:

- The research proposes a generic user data collecting mechanism specifically for privacy-preserving POI RS that does not tie to a specific model structure. By indirectly synthesizing user feedback, the system circumvents conventional privacy issues in data storage and analysis. Furthermore, the proposed inclusive framework contains a semi-trusted (honest but curious) third-party, offering opportunities for utilizing more contextual information in future research.
- The framework offers a secure way of data communication. Based on estimated user location and Local Differential Privacy (LDP), the users are able to generate personalized recommendations on their local devices without jeopardizing the privacy of sensitive information. Throughout the paper, it will present the importance of relating users by their estimated activity center and providing differential privacy guarantees.
- The research was conducted on publicly available real-world datasets, demonstrating the effectiveness and a reasonable trade-off. The results also indicate the framework’s potential for supporting cross-platform collaborations.

The remainder of this piece of research is organized as follows. The section “Relative Techniques and Background” discusses related topics and background information for the proposed framework. “Framework Description” depicts the framework’s structure and the challenges the research aims to solve. Section “Notations and Expression Explanations” introduces the fundamentals, notations, and formulation of the proposed framework. Next, “Experiment and Discussion” introduces the datasets,

privacy analysis, and the results of this era’s work. Finally, a summary is presented in the final section.

5.2 Relative Techniques and Background

Privacy Concerns

Privacy concerns are driving increased regulations in the field of recommender systems [84, 85, 41, 86, 87, 88]. These regulations, which have been introduced in regions such as the EU, the US, and China, aim to prevent irresponsible or malicious data mining and analytics. Point-of-Interest (POI) recommender systems, which rely on large quantities of data and various data sources, are particularly affected by these regulations. Besides, the growing use of mobile devices, including personal cell phones, has led to the proliferation of location-based services (LBS) over the past decade, providing a wealth of data for POI recommender systems. That being said, it makes these systems vulnerable to malicious attacks and accidents.

Existing Privacy-Preserving Recommender System

In response to privacy concerns, a variety of privacy-preserving recommendation models have been proposed. These models can be broadly classified into centralized and decentralized approaches.

Decentralized approaches can further be divided into distributed and federated models. Distributed models, such as [89, 90, 17] using gradient exchange, involve local storage and processing of data, but lack the ability to collect or analyze data. Federated models such as [59], on the other hand, rely on a central server and involve the exchange of model weights between the server and individual users. After updating these weights according to each user’s local rating or feedback, the global model retrieves the updated weights in order to perform the training process. In practice, the process above repeats itself to keep the system dynamically updated. Although the users’ data is fully kept on their devices at any moment, the central system is comparatively vulnerable to malicious participants.

In centralized RS, frameworks revolve around conventional encryption protocols, k-anonymity, and perturbations. Encryption-based RS, such as [91, 92, 93], has better accuracy than its non-private counterpart but requires solid computational power and faces the same problem when collecting user data. K-anonymity in recommendation models has not changed its definition that privacy is achieved by severing the connection between sensitive or private information and user identities by data obfuscation or aggregation. Lastly, perturbation-based models add noise to raw data while preserving underlying statistics. However, these approaches face challenges when balancing privacy with other factors, such as data transfer or a large user base.

In addition to the traditional techniques mentioned above, Differential Privacy (DP) is a privacy-preserving standard that is getting growing attention. There are two types of DP: centralized DP (CDP), in which perturbation or randomization occurs at the curator level, and local or localized DP (LDP), in which perturbation occurs at the user level. CDP trusts the third party, while LDP assumes that the

third party is at risk of being compromised and therefore adds noise at the user level. It is worth noting that CDP is often referred to as DP in most cases.

Differential Privacy (DP) is a privacy preservation standard that does not require attack modeling. In the past, researchers had to define a potential attacker and details of attack scenarios in order to design a privacy-preserving framework. DP, on the other hand, claims to protect a user’s feedback and related contextual information even if the attacker is aware of particular users or uses them as infiltrators to gather information about other users. DP’s unique properties allow for the calculation and quantification of the trade-off between privacy and performance after data goes through a complex algorithm, even when multiple mechanisms are combined in a proposed framework.

In recommendation research, DP has been implemented to protect critical user information in recent years as well. For example, in [94] Arik Friedman, Arnaud Berlioz, et al. gave a thorough analysis on applying differential privacy to matrix factorization (MF) based recommender systems, which presents not only various ways to apply such privacy-preserving techniques but also shows particular optimal solutions. In comparison, H. Shin et al. applied Local DP (LDP), a newer and more localized version of DP, to the MF recommender systems to protect user feedback from central servers. Other researches such as [95] improved its remaining challenges giving deniability to not only rating values but also rating behaviors.

Data Clustering

Data clustering is a technique for grouping data objects into different clusters based on similarities or distances. There are many clustering algorithms, such as k-means, spectral, DBScan, and fuzzy c-means, that solve different clustering tasks. In this research, k-means clustering was selected because of its simplicity and suitability for addressing the problem at hand.

In the field of recommender systems (RS), clustering is widely used for two purposes. First, it is used to improve RS performance and address data scarcity problems, such as predicting how groups of people choose online streaming services or nearby restaurants [96, 97, 98]. Second, clustering has been used to protect user identities and mask sensitive user information in privacy-preserving schemes for RS and general data mining, helping to protect user feedback and certain contextual information [99, 47, 100].

5.3 Framework Description

This section introduces the proposed framework, the Generic User Synthesizer (GUS), offering a unique approach to generating personalized recommendations. The approach involves replacing real users with aggregated user information protecting user privacy while still generating meaningful recommendations. The research outlines the proposed framework by breaking it down into steps and discussing the challenges that each step addresses. Figure 5.1 shows the four main components of the multifaceted framework, with the last two hosted on the central server.

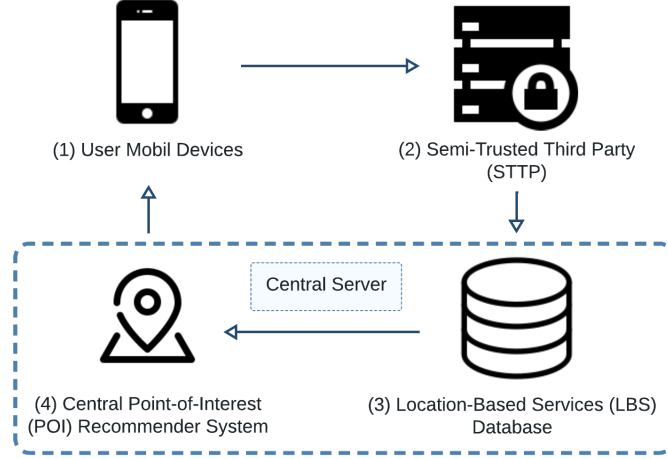


Figure 5.1: The Overall Framework Structure and Data Flow of the system. (The diagram illustrates the various components of the framework and the way in which data flows between them.)

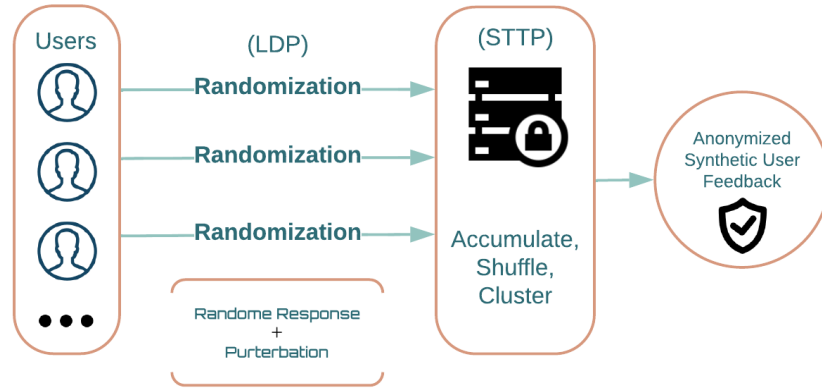


Figure 5.2: Schemes in STTP Collecting Data from User Devices

From Local Devices

To generate data-minable synthetic data, the proposed approach leverages a Semi-Trusted Third Party (STTP) that uses clustering algorithms to collect useful information from users. As Figure 5.2 shows, to protect user privacy, the transmission protocol between users' devices and the STTP is based on Local Differential Privacy (LDP), which leverages perturbation techniques to protect the value of user ratings, and randomized responses to protect user rating behaviors. This approach ensures that the data collection process is locally and differentially private, with perturbation and randomized responses occurring at the user level. The STTP is honest but curious (HBC), and individual data are aggregated in batches to prevent traceability to their original owners. Randomness is introduced from the beginning of the dataflow To provide privacy guarantees throughout the process. As a consequence of

the sequential property of LDP, which is further defined in later sections, all following computation is LDP-guaranteed.

On STTP

After receiving user information, the STTP employs a clustering algorithm that groups users based on their implicit location, which is estimated from previously visited Points of Interest (POIs) rather than using real-time GPS information. According to the previous research [101], this approach is likely more accurately reflects user visiting behaviors and is also more computationally efficient, as it greatly reduces the feature space. Intuitively speaking, users may be less motivated to visit a distant location even if it perfectly fits their preferences. In terms of clustering scalability, clustering thousands of users based on their feedback takes hours, while clustering based on implicit locations takes only seconds. The centroids of the clustering algorithm are treated as synthetic users, and their feedback is calculated by averaging the feedback from all corresponding group members.

To Central Servers

Upon receiving synthetic user feedback from the STTP, the central server performs several tasks. Firstly, it stores the synthetic data for future use and calculates the similarities among POIs(items) based on the synthetic user feedback. The resulting similarity matrix is then used in the randomized response mechanism to protect user rating behaviors and provide deniability for each visiting record. Additionally, the central recommender system prepares user/item embeddings and the similarity matrix for users to download. To minimize computation workload and downloadable package size, the central RS employs “pre-training” and feature reduction techniques. This approach helps prevent malicious users from accessing the exact synthetic user information, even though the synthetic users’ privacy is trivial.

Back to Local Devices

The local recommender system (RS) on a user device is capable of offering accurate recommendations by appending the owner’s personal rating record to the reconstructed rating matrix. In order to generate meaningful recommendations, the user device must first download assistive content, which is prepared by the central server. These preparations reduce the size of the downloading package and pre-train the model to help re-construct an imputed user rating matrix locally, like a “meal prep kit” for recommender systems. Once the rating matrix reflecting the user’s real visiting preferences is reconstructed, the local RS can predict future user preferences like any conventional RS. Unfortunately, simulating thousands of mobile devices applying adaptive training is unrealistic under experimental conditions. Due to computing power limitations in the experiment, a different approach was employed, using the most similar synthetic users to calculate a weighted average as the final prediction. A detailed definition of this approach will be provided in later sections.

5.4 Notations and Expression Explanations

This section will introduce the fundamental concepts of the different models, including the notations and problem definitions that are used, as well as the metrics and evaluation methods that the model employs.

Fundamentals of Matrix Factorization and Neural Collaborative Filtering

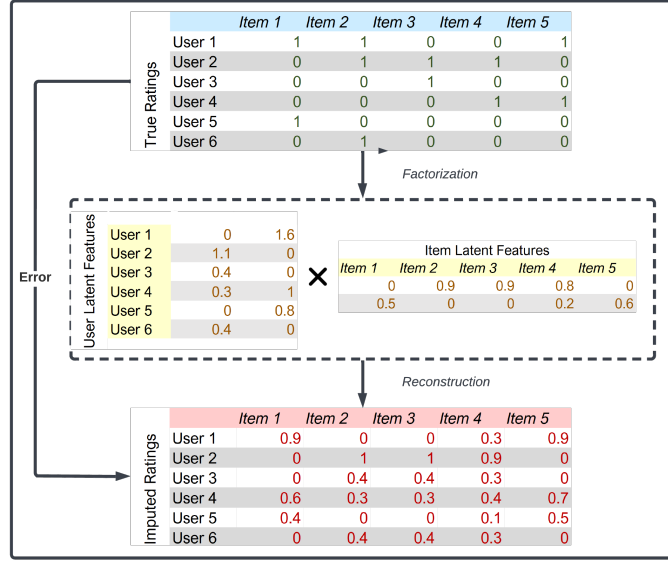


Figure 5.3: The Factorization of the Rating Matrix in the Collaborative Filtering Technique. (The diagram illustrates the process of decomposing the matrix into lower-rank matrices that represent the latent factors underlying user-item interactions.)

Neural Collaborative Filtering (NCF) [102] is a recommendation model based on Collaborative Filtering (CF) and Matrix Factorization (MF). MF decomposes the feedback matrix into two lower-rank matrices and iteratively minimizes the error between observed and predicted ratings. Missing entries in the feedback matrix are imputed to estimate users' potential preferences. The minimization problem is formulated as follows:

$$\min_{p_u, q_i} \sum_{r_{ui} \in R} (r_{ui} - \hat{r}_{ui})^2 + \lambda(\|q_i\|^2 + \|p_u\|^2) \quad (5.1)$$

where we use u to denote a user (customer) and i an item (POI). If we define U and I as the user and item sets, then $u \in U$ and $i \in I$. In the above equation, r_{ui} indicates an observed rating suggesting the preference of user u over item/POI i . Correspondingly, the variable \hat{r}_{ui} represents predicted ratings. The rating matrix R comprises all the observed and unobserved feedback or visiting records. Thus the prediction function for \hat{r}_{ui} can be written as below:

$$\hat{r}_{ui} = p_u^T \cdot q_i = \sum_k p_{uk} q_{ik} \quad (5.2)$$

where the column vectors p_u and q_i are from the decomposition result, two sub-matrices representing the user and item latent factors, respectively. Here, k is the dimension of the latent space.

Collaborative Filtering (CF) and Neural Collaborative Filtering (NCF) are two popular techniques used to model user-item interactions in recommendation systems. CF works by representing the rating matrix in a latent space using inner product optimization, while NCF uses a neural network to replace the lower-rank matrices used in CF. The advantage of NCF is that it can capture non-linear interactions and incorporate weighted user/item features, making it a more generalized approach that can handle complex interactions.

NCF consists of two parts: the Generalized Matrix Factorization (GMF) and the Multi-Layer Perceptron (MLP). The GMF is a linear model that learns user-item interactions, while the MLP uses a deep neural network to model nonlinear interactions between users and items. In short, the way GMF predicts future ratings is formulated as follows:

$$\hat{r}_{ui} = a_{out}(h^T(p_u^G \cdot q_i^G)) \quad (5.3)$$

where a_{out} is the activation function of the output layer, and h is the edge weight for the output layer. G indicates the user or item embedding is from GMF. On the other hand, MLP alters its prediction function pertaining to future ratings in the following way:

$$\hat{r}_{ui} = \sigma(h^T a_L(W_L^T(a_{L-1}(\dots a_2(W_2^T \begin{bmatrix} p_u^M \\ q_i^M \end{bmatrix} + b_2)\dots) + b_L)) \quad (5.4)$$

where σ is the sigmoid function and M indicates the user or item embedding is from MLP. Moreover, W_x , b_x , and a_x denote the weight matrix, bias vector, and activation function from the corresponding layer, i.e. x_{th} perception. Finally, a pre-trained NCF model combines the two trained models together to catch the interactions between users and items fully and thus forms the below prediction function:

$$\hat{r}_{ui} = \sigma(h^T \begin{bmatrix} \phi^{GMF} \\ \phi^{MLP} \end{bmatrix}) \quad (5.5)$$

where ϕ indicates all the above-mentioned function mappings up until the output layer of either GMF or MLP.

Other information or techniques applied in NCF, such as negative sampling, training strategies, and deduction process for the loss functions, are not discussed here due to the scope of the research.

In summary, Neural Collaborative Filtering (NCF) is a widely adopted neural network-based technique for modeling user-item interactions in recommendation systems. The research selected NCF as the baseline model because it has been shown to be reliable, is widely used in the literature, and has been extensively studied by peer researchers. Additionally, NCF's generic usage aligns well with the design of the framework, which aims to provide a flexible and scalable approach for recommendation tasks.

Local Differential Privacy (LDP)

In this section, it will introduce some essential notations and definitions of LDP.

Definitions and Theorems

In the previous sections, we have introduced differential privacy (DP) to give abstract ideas of its functioning and utilization. Thus, notations and definitions of LDP are included further in the current section for conducting proper demonstrations.

Formally, assuming we have an Algorithm A and a pair of neighboring datasets D and D' that differ of a single user record, A provides ϵ -differential privacy protection if all outputs of $A(D)$ and $A(D')$ satisfy the following condition:

$$Pr[A(D) \in S] \leq \exp(\epsilon) \cdot Pr[A(D') \in S] \quad (5.6)$$

where $S \subseteq \text{Range}(A)$ stands for any permitted output from Algorithm A and Pr is the probability over the randomness of A . The variable ϵ is called the budget of privacy protection. Larger values in ϵ indicate the data is “truer” since the level of randomization is low. In other words, the value of ϵ is aligned with data integrity but the opposite of privacy protection. The term “neighboring” may be defined differently in other research where the difference is all information of a single user. In contrast, some algorithms aim to guarantee per-item privacy where the protection extends to a single rating.

Definition 1 (ϵ -Local Differential Privacy). Suppose there is a randomized Algorithm A and a single dataset D , for any pair of different user feedback $r, r' \in D$ and for any outcome $S \subseteq \text{Range}(A)$, we have:

$$Pr[A(r) \in S] \leq \exp(\epsilon) \cdot Pr[A(r') \in S] \quad (5.7)$$

where randomization is applied to each user’s feedback independently. The LDP mechanism is based on Generalized Randomized Response (GRR) [103, 104]. The definition of *GRR* is given below:

Definition 2 (Generalized Randomized Response (GRR)). Given i being the true visited POI, let \bar{i} be the random response instead. If \bar{K} is the option space from which the random response is selected, i.e., $i \in \bar{K}$, then we have the following formulation:

$$Pr(\bar{i} = i) = \begin{cases} \frac{e^\epsilon}{e^\epsilon + |\bar{K}| - 1}, & \text{if } (i = \bar{i}) \\ \frac{1}{e^\epsilon + |\bar{K}| - 1}, & \text{if } (i \neq \bar{i}) \end{cases} \quad (5.8)$$

As the above formula shows, when $|\bar{K}| = 2$, which means the user can only choose one of two items/POIs, then it becomes the conventional true or false scenario where the traditional Randomized Response (RR) is implemented. Aside from using the GRR mechanism to protect the user rating behaviors, the feedback value is also guarded against the perturbation technique using Duchi et al.’s solution [105].

Definition 3 (Laplace Mechanism). Given a dataset D , and a function f where $f : D \rightarrow \mathbb{R}$ mapping over D , the Laplace mechanism is as followed:

$$F(D) = f(D) + \text{Laplace}\left(\frac{s}{\epsilon}\right) \quad (5.9)$$

where s is the sensitivity. The sensitivity is dependent on the dataset and the target algorithm. For example, if the dataset uses explicit rating feedback such as $\{1, 2, 3, 4, 5\}$, then $s = 4$.

Duchi et al.'s Solution. Given a tuple $r \in [-1, 1]$, a perturbed tuple r' that equals either $\frac{e^\epsilon+1}{e^\epsilon-1}$ or $-\frac{e^\epsilon+1}{e^\epsilon-1}$ is returned according to the following probability t :

$$Pr(r'_{ui} = t | r_{ui}) = \begin{cases} \frac{e^\epsilon-1}{2e^\epsilon+2} \cdot r_{ui} + \frac{1}{2}, & \text{if } (t = \frac{e^\epsilon+1}{e^\epsilon-1}) \\ \frac{1-e^\epsilon}{2e^\epsilon+2} \cdot r_{ui} + \frac{1}{2}, & \text{if } (t = -\frac{e^\epsilon+1}{e^\epsilon-1}) \end{cases} \quad (5.10)$$

where u is the user given ratings, and i is the target POI. The perturbed ratings r'_{ui} are unbiased estimators of the original ratings, according to Duchi et al.

Since LDP mechanisms in the framework include GRR and Duchi et al.'s Solution to protect user rating behavior, i.e., whether a user visited a POI, and POI preference, i.e., whether the user likes their visited place, respectfully, the research introduces the following properties.

Theorem 1 (Sequential Composition). If a mechanism G contains a series of n independent randomized functions $G = \{g_1, g_2, \dots, g_n\}$, and each function offer $\epsilon_i - LDP$ guarantee where $i \in n$, then the mechanism G provides $(\sum_{i=1}^n \epsilon_i) - LDP$.

Meanwhile, the LDP mechanisms are introduced at the very beginning, so the post-processing property is involved to guarantee the safety of the data manipulation and training process following the initial step.

Theorem 2 (Post-processing). Given a dataset D , and a function f that guarantees $\epsilon - LDP$ where $f : D \rightarrow \mathbb{R}$, for any randomized function $f' : \mathbb{R} \rightarrow \mathbb{R}'$ we have $f \circ f'$ also being $\epsilon - LDP$.

These theorems guarantee the privacy-preserving qualities for all the following computing and data processing procedures.

Rating Value Protection

In POI RS, the ratings are either explicit or implicit. The implicit ratings are either 0 or 1, where 1 indicates visited record and 0 indicating not visited. In the case of implicit user feedback, the user rating value does not require protection since the actual value only implies whether or not a user has visited a particular location instead of how much they enjoy visiting it. Explicit ratings, on the other hand, require such protection. Considering that the LDP-dependent algorithm in this research is to calculate centroids' ratings, which comes from averaging all user ratings in each group, the research chooses Duchi et al.'s solution. The algorithm is shown in Algorithm 3.

It is worth noting that the input of the following algorithms is a list of tuples that are in the format of $\{userID, itemID, rating\}$. Each tuple can be denoted as r_{ui} representing that user u possesses a rating r toward item i . Precisely, the collection of ratings thereby forms the rating list R_{list} . In Algorithm 3, each rating after normalization is perturbed according to the Equation (5.10). Based on the above Algorithm 3, the historical ratings stored in users' devices are firstly perturbed in value that reflects how much they enjoyed their previous visit.

Algorithm 3: Rating Perturbation Using Duchi et al.’s Solution

Input: list of rating tuples R_{list} , privacy parameter ϵ_1
Output: list of perturbed rating tuples R'_{list}

```
1 for each  $r_{ui}$  in  $R_{list}$  do
2   | Normalize  $r_{ui}$  such that  $r_{ui} \in [-1, 1]$ 
3   |  $r_{ui} = \frac{1}{2} \cdot (r_{ui} - 1) - 1$ 
4 end
5
6 for each  $r_{ui}$  in  $R_{list}$  do
7   | Sample a Bernoulli variable  $t$  where:
8   |  $Pr(t = 1) = \frac{e^{\epsilon_1} - 1}{2e^{\epsilon_1} + 2} \cdot r_{ui} + \frac{1}{2}$ 
9   | if  $t = 1$  then
10  |   |  $r'_{ui} = \frac{e^{\epsilon_1} + 1}{e^{\epsilon_1} - 1}$ 
11  | else
12  |   |  $r'_{ui} = \frac{e^{\epsilon_1} + 1}{1 - e^{\epsilon_1}}$ 
13  | end
14 end
15 for each  $r'_{ui}$  in  $R'_{list}$  do
16  | De-normalize  $r'_{ui}$  such that  $r'_{ui} \in [1, 5]$ 
17 end
18
19 return  $R'_{list}$ 
```

Rating Behavior Protection

Besides protecting the users’ rating values, the scheme also aims to protect the users’ behaviors. Among other research, such as [106], on applying DP or LDP to recommender systems, the studies show that guarding rating behaviors is very challenging since the column space is very large, considering the number of POIs. Perturbing each entry of the rating matrix is highly expensive in terms of privacy budget since the matrix is often exceedingly sparse. Thus in the framework, instead of camouflaging the actual ratings by perturbing all potential ratings, the research adds deniability only to existing ratings.

In the next section, it will introduce a dynamically pre-calculated similarity matrix on the central server. For now, let’s state the purpose that each *locationID* in the rating tuple in the form of $\{userID, locationID, rating\}$ can be perturbed using such a matrix. As a result, either the actual location remains or a similar POI takes its place. The advantage of having such a mechanism is to change the question from “whether or not the user visited the place” to “where the user visited, if at all.”

Algorithm 4 shows how the POIs are randomized. The input, the list of rating tuples R'_{list} , is the exact output from Algorithm 3. Variable k is the option space. For example, when $k = 2$, the *locationID* in $\{userID, locationID, rating\}$ can only be replaced by the most similar POI’s *locationID*. When $k = 3$, the *locationID* in $\{userID, locationID, rating\}$ is replaced by either one of the top two most similar

POIs' *locationIDs*. The larger k is, the higher the privacy budget. Eventually, the output R''_{list} is sent to the aggregating and clustering server.

Algorithm 4: Protecting Rating Presence

Input: list of rating tuples R'_{list} from Algorithm 3, privacy parameter ϵ_2 , similarity matrix S_I , number of options k

Output: list of perturbed rating tuples R''_{list}

```

1 for each  $r'_{ui}$  in  $R'_{list}$  do
2   Sample a Bernoulli variable  $t$  where:
3    $Pr(t = 1) = \frac{e^{\epsilon_2}}{e^{\epsilon_2} + k - 1}$ 
4   if  $t = 1$  then
5      $r''_{ui} = r'_{ui}$ 
6   else
7      $r''_{ui} = r'_{uS_I[i]}$ 
8   end
9 end
10 return  $R''_{list}$ 

```

Privacy Analysis

As previously mentioned, privacy protection is for user rating values and rating behaviors. The POI visiting history indicates not only the users' preference for selected POIs but also the POIs the users chose. To improve the privacy-preserving feature in this generic data collection framework, the honest but curious third party, although not designed to collaborate with the recommendation service providers, is separated from direct user rating information by LDP.

Theorem 1. *Algorithm 3 satisfies ϵ -LDP with respect to users' rating values.*

proof 1. *According to the definition of ϵ -LDP, one wants to prove that it is equally likely to generate the same output $r'_{list} = [r'_{ui}]_{i=1}^n$ for any two input $r^1_{list} = [r_{ui}]_{i=1}^n$ and $r^2_{list} = [r_{ui}]_{i=1}^n$ in Algorithm 3. Let x, X^1 , and X^2 be any values in $[-1, 1]$, then we have*

$$\begin{aligned}
\frac{Pr[r'_{ui} = x | r_{ui} = X^1]}{Pr[r'_{ui} = x | r_{ui} = X^2]} &\leq \frac{\max_{X^1}(Pr[r'_{ui} = x | r_{ui} = X^1])}{\min_{X^2}(Pr[r'_{ui} = x | r_{ui} = X^2])} \\
&= \frac{\max_{X^1}(\frac{e^{\epsilon_1-1}}{2e^{\epsilon_1}+2}X^1 + \frac{1}{2})}{\min_{X^2}(\frac{e^{\epsilon_1-1}}{2e^{\epsilon_1}+2}X^2 + \frac{1}{2})} = \frac{\frac{e^{\epsilon_1-1}}{2e^{\epsilon_1}+2}(1) + \frac{1}{2}}{\frac{e^{\epsilon_1-1}}{2e^{\epsilon_1}+2}(-1) + \frac{1}{2}} = e^{\epsilon_1}.
\end{aligned}$$

Thus, the perturbation of R_{list} in Algorithm 3 satisfies ϵ_1 -LDP and Algorithm 3 satisfies ϵ_1 -LDP with respect to users' rating values.

Theorem 2. *Algorithm 4 satisfies $(\epsilon_1 + \epsilon_2)$ -LDP for both users' rating values and rating behaviors.*

Proof. We start by proving that it is equally likely to generate the same output $r''_{list} = [r''_{ui}]_{i=1}^n$ for any two input $r^1_{list} = [r'_{ui}]_{i=1}^n$ and $r^2_{list} = [r'_{ui}]_{i=1}^n$ in Algorithm 4. Let y, Y^1 , and Y^2 be any values in $[-1, 1]$. According to Algorithm 4, we have

$$\frac{Pr[r''_{ui} = y | r'_{ui} = Y^1]}{Pr[r''_{ui} = y | r'_{ui} = Y^2]} \leq \frac{\max_{Y^1}(Pr[r''_{ui} = y | r'_{ui} = Y^1])}{\min_{Y^2}(Pr[r''_{ui} = y | r'_{ui} = Y^2])} = \frac{\frac{e^{\epsilon_2}}{e^{\epsilon_2} + k - 1}}{\frac{k - 1}{e^{\epsilon_2} + k - 1}} = \frac{e^{\epsilon_2}}{k - 1} \leq e^{\epsilon_2}.$$

Thus, the perturbation of R'_{list} in Algorithm 4 satisfies ϵ_2 -LDP and Algorithm 4 satisfies ϵ_2 -LDP with respect to users' rating behavior. Since Algorithm 4 protects the user's rating value by using Algorithm 3 and Algorithm 3 satisfies ϵ_1 -LDP, according to the sequential composition property, we can conclude that Algorithm 4 satisfies $(\epsilon_1 + \epsilon_2)$ -LDP. □

Notations and Methods for Clustering and POI Similarity Calculation

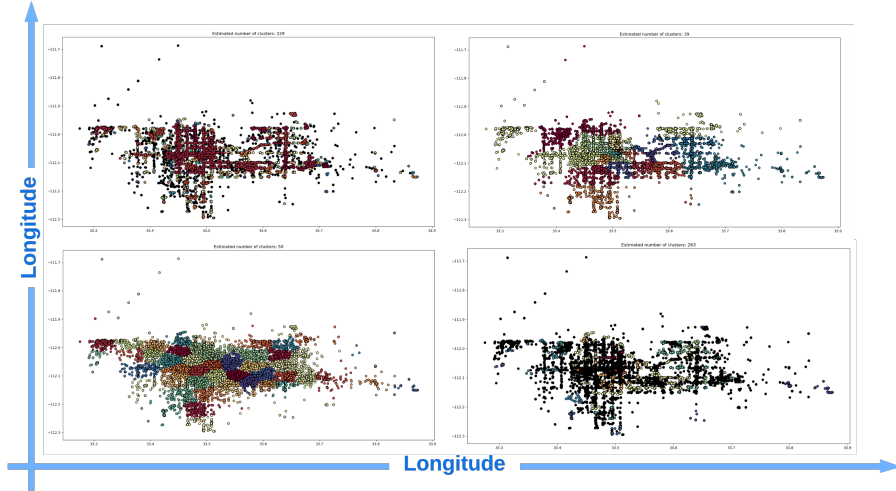


Figure 5.4: The Visualization of User Location Clustering (Phoenix). (The clustering algorithms shown in the picture include DBSCAN (upper-left), PAM (upper-right), k-mean (lower-left), and OPTICS (lower-right).)

In general geolocation analysis, it is common to use user-item interaction records to perform clustering. However, this approach is not effective for POI recommender systems, as users in datasets often only visit an extremely small portion of a city and do not have a significant number of interactions. In comparison, this research chose to use estimated user locations to determine the similarities among users' visiting behaviors. The user estimation locations $Loc_u[lat_u, long_u]$ is formulated below:

$$Lat_u = \frac{1}{|I_u|} \sum_{i \in I_u} Lat_i \quad (5.11)$$

$$Long_u = \frac{1}{|I_u|} \sum_{i \in I_u} Long_i \quad (5.12)$$

where I_u denotes all the POIs a user visited. It is worth mentioning that the calculation of the estimated location of a user is after the GRR data processing. Naturally, the clustering happens after the perturbation.

Various clustering methods have been investigated in the experiment. Initially, the approach started with DBSCAN [107], a standard geolocation clustering method that automatically decides the appropriate number of clusters. However, as Figure 5.4 shows (Upper-left), where the dataset is limited to a single city and user density is applied as the clustering criterion, this method leads to grouping users who are far from each other into one cluster. The red dots represent the users that are grouped into the first cluster. On the other hand, when testing OPTICS clustering [108], also a density-based clustering algorithm, the research shrank the maximum distance to avoid overly large groups. As a result, a large portion of the users is considered noise (black dots).

Moreover, the research has also considered hierarchical clustering methods such as the Partition Around Medoids (PAM) [109] clustering algorithm (upper-right). Unfortunately, it creates a lot of empty clusters despite the active tuning, and the result is stretched by users who are active farther from the metropolitan area. As a result, the K-mean method yields comparatively better results regarding the ability to include all users and generate balanced clusters. Spectral clustering [67] has also been tested as well by constructing the affinity matrix first. However, this method is too computationally costly. To address this, K-mean clustering is employed as a more efficient and competitive alternative.

After clustering, centroids from the output are utilized as virtual users or synthetic user information resulting from aggregation. The way to create ratings for the artificial users is by taking the average of all users in each cluster. Suppose the users are split into $|C|$ clusters where each cluster $c \in C$ and all users $u \in U_c$ belonging to that cluster, and we have the following equation to impute the ratings for each centroid from the clustering:

$$\hat{r}_{ci} = \sum_{u \in U_c} r_{ui} / |U_c \cap U_i| \quad (5.13)$$

where U_i denotes all users that have feedback for the item/POI i . The synthetic users with ratings can then be sent to the central server for model pre-training and POI similarity calculation.

As mentioned before, the central server actively calculates the similarities among POIs based on the clustered information. Suppose S_{ij} is the POI similarities between POI i and j based on user ratings, then S_{ij} is defined using Pearson Correlation Coefficient:

$$S_{ij_PCC} = \frac{\sum_{u \in U_{ij}} (r_{ui} - \mu_i) \cdot (r_{uj} - \mu_j)}{\sqrt{\sum_{u \in U_{ij}} (r_{ui} - \mu_i)^2} \cdot \sqrt{\sum_{u \in U_{ij}} (r_{uj} - \mu_j)^2}} \quad (5.14)$$

where U_{ij} is the set of users who have rated both POI i and j . The variable μ denotes the rating mean. For example, μ_i is the average rating for all ratings of POI i , and μ_j is the average rating for all ratings toward POI j .

Personalization

The central recommendation system (RS) in the approach serves two primary purposes. First, it can alleviate the computational burden on users' mobile devices by performing initial training as needed. Second, it can impute missing data in the rating matrix, which is often incredibly sparse in point-of-interest (POI) datasets. This is important because even the centroids may not have feedback for specific POIs in these datasets, unlike in datasets such as MovieLen.

The model uses synthetic data for training to generate the pre-trained models for users to download. It is essential to point out that, by design, the users need to perform adaptive training on the local RS. On each user's model device, the local RS finishes the recommendation by imputing the missing probabilities for the current user to visit each new location. However, iterating through all test users to simulate such a scenario during the experiment is exceptionally time-consuming and unrealistic. Thus, under experiment conditions, an alternative approach is taken where the real users rank the artificial users based on their rating (visiting probability) similarities and get personalized recommendations through the weighted average ratings (chance of visiting) of the top t similar centroids.

Suppose I_u represents all POIs visited by user u , and s_{uc} denotes the similarity score between physical user u and synthetic user c , the calculation is based on the following formula:

$$s_{uc} = \sum_{i \in I_u} Rank_{ci} \quad (5.15)$$

where $Rank_{ci} \leq k$ denotes the rank of the POI i in the top- k list of the synthetic user c 's preferred locations. Consequently, the users get their personalized prediction by applying the following equation:

$$\hat{r}_{ui} = \sum_{c \in C} s_{uc} r_{ci} / \sum_{c \in C} s_{uc} \quad (5.16)$$

where C is the set containing all synthetic users.

Evaluation and Metrics

Because of the non-linear feature of this system, it is challenging to optimize it as a whole. Therefore, the research has decided to optimize each of the three parts (perturbation, clustering, and final results) separately, as this approach allows us to tackle the heavy workload in a more efficient and effective way.

Perturbation

In order to evaluate the performance of the perturbation part, it was decided to use Mean Absolute Error (MAE) to measure the relationship between the LDP budget (ϵ_1, ϵ_2) and the resulting errors between real users and virtual users. By comparing the MAE values at different levels of the LDP budget, we can assess the impact of the

budget on the accuracy of following the clustering procedure. The formula is shown below:

$$MAE = \frac{1}{|R|} \sum_{i \in R} |r_i - \bar{r}_i| \quad (5.17)$$

where R is the set of all ratings. The variables r_i and \bar{r}_i are the corresponding ratings from real users and virtual users, respectively.

Clustering

Previously, we mentioned the reasoning behind selecting k-mean clustering as the aggregating method. Here the research states the metric for finding the optimal group number k : the within-cluster sum of squares (WSS). In order to avoid over-complex notations, the research needs to abuse the usage of variable k . WSS measures the sum of the squared distances between each data point and its assigned centroid in k-means clustering. By plotting the WSS for different values of k and looking for the “elbow” in the plot, it is found the desired range for k . The formula is listed below:

$$WSS = \sum_{i=1}^k \sum_{x_j \in C_i} \|x_j - c_i\|^2 \quad (5.18)$$

where k is the number of clusters, C_i is the i -th cluster, x_j is the j -th data point in the i -th cluster, c_i is the centroid of the i -th cluster, and $\|x_j - c_i\|^2$ represents the squared Euclidean distance in between.

Final Result

Regarding the assessment of the final result, the research uses precision at k and recall at k to study the trade-off from the framework output. If we let I_r and I_v denote the recommended POI set and visited POI set, then the following equations show the details of the definition:

$$Recall@k = \frac{|I_r \cap I_v|}{|I_v|} \quad (5.19)$$

$$Precision@k = \frac{|I_r \cap I_v|}{k} \quad (5.20)$$

where $Recall@k$ measures the portion of visited POIs in all visited POIs, and $Precision@k$ measures the portion of recommended POIs that are actually visited in the top- k POIs.

It is worth noting the unique aspects of this framework and experiment. Firstly, it uses time-dependent cross-validation with uniformly sized data partitions. Secondly, it employs on-device processing to exclude distant recommendations for enhanced accuracy. Finally, it assumes no disclosure of GPS information but can estimate user location for enhanced privacy.

For a comprehensive description of the evaluation process, the research refers the reader to the Algorithm 5.

Algorithm 5: Framework Evaluation Steps

Input: Preprocessed datasets $\{Train_1, Test_1, Train_2, Test_2, \dots, Train_n, Test_n\}$, number of recommendations k , POIs' GPS information

Output: Recall@k, Precision@k

```
1 for each training set  $Train_i$  do
2   —Estimate each user's longitude  $Long_u$  and latitude  $Lat_u$  according to
   (5.11) and (5.12)
3   —Perform rating value perturbation according to Algorithm 3
4   —Perform the general random response according to Algorithm 4
5   —Perform clustering and rating aggregation according to (5.13) to
   generate virtual users
6   —Feeding the virtual user data to the NCF model for pre-training
7   —Reconstruct the centroids' rating matrix  $\hat{R}$ 
8   —Calculate the POI similarity matrix  $S_I$  for future GRR
9   for each  $r_{ui}$  in the corresponding  $Test_i$  do
10    | Calculate the corresponding  $\hat{r}_{ui}$  according to (5.15) and (5.16)
11  end
12  Calculate  $Recall@k_i$  and  $Precision@k_i$  according to (5.19) and (5.20)
13 end
14 Calculate the average  $Recall@k$  and  $Precision@k$ 
15 return  $Recall@k$  and  $Precision@k$ 
```

5.5 Experiments and Discussion

The section begins by introducing the datasets in the experiment, followed by analyzing the privacy implications of data perturbation. Afterward, an overview of the parameter tuning and optimization strategy employed in the experiment is given before concluding with the final experimental results and their comparisons to prior research.

The Datasets

the research has experimented using user feedback from two areas: the Champaign–Urbana (CU) metropolitan area and Phoenix (PH) city area. The data were filtered from Yelp and Google Local datasets, respectively. Something new is that the research used four datasets: CU from Google, PH from Google, CU from Yelp, and PH from Yelp. PH was selected due to its large population and high volume of user feedback. At the same time, CU was chosen for comparison as it has fewer user records but with higher density. The research used both Yelp and Google Local datasets to account for different business models. As shown in Figure 5.4, user locations outline the city's shape.

The user feedback in this time's datasets has the following characteristics:

- The feedback is explicit, meaning the ratings are in $\{1, 2, 3, 4, 5\}$.

Table 5.1: Datasets Statistics (rounded)

Area	#Users	#POIs	#Feedback	Density
CU (Yelp)	11953	1579	33990	0.1802%
PH (Yelp)	204887	17213	576700	0.0163%
CU (Google)	1910	876	3310	0.1978%
PH (Google)	24899	7801	37245	0.0192%

- The user ratings are unique, meaning each user can only give a rating to the same place once.
- The sparsity of the datasets is comparatively much lower than conventional recommendation datasets, even with filtering. A straightforward comparison of statistics of the four areas can be found in Table 5.1.

Parameters and Tuning

In the previous sections, we discussed the GUS framework from the perspective of its three main components: the user end, the honest but curious STTP, and the central server that holds the RS model. The optimization is performed in each of the three parts:

- On each user’s mobile device, two optimization tasks must be performed: determining the appropriate privacy budget (ϵ_1 and ϵ_2) to control the level of perturbation and selecting the number of synthetic users (n_v) to use for prediction. In the experiments, it was found that using too many or too few synthetic users resulted in sub-optimal performance.
- On the STTP, the clustering is performed using the k-means method, in which we need to choose the number of clusters beforehand using the elbow method.
- On the central server, we have a number of hyperparameters that we can tune to optimize the performance of the NCF model. These include the number of latent factors $n_{factors}$, the learning rate r , the layer size $layer_sizes$ for the multi-layer perceptron component, the batch size $batch_size$, and the number of training epochs n_{epochs} . It is worth noting that the research has been training the NCF model from scratch for each dataset.

To streamline the hyperparameter tuning process for GUS, the research has developed a strategy that decomposes the tuning process into smaller, manageable sub-tasks that focus on optimizing specific sub-components. This approach enables me to fine-tune the parameters in an efficient and effective manner while reducing the computational burden of full hyperparameter tuning.

Specifically, the first part of the optimization process aims to strike a satisfactory balance between error values and privacy protection by adjusting the epsilon value. To achieve this, the research has used the MAE to calculate error values between

feedback generated from centroids, with and without the use of LDP. The objective is to identify the optimal trade-off point that offers both high accuracy and robust privacy protection. Considering the tolerable increase in MAE, the research has set both ϵ_1 and ϵ_2 to 0.6, as determined from the analysis in Figures 5.5 and 5.6.

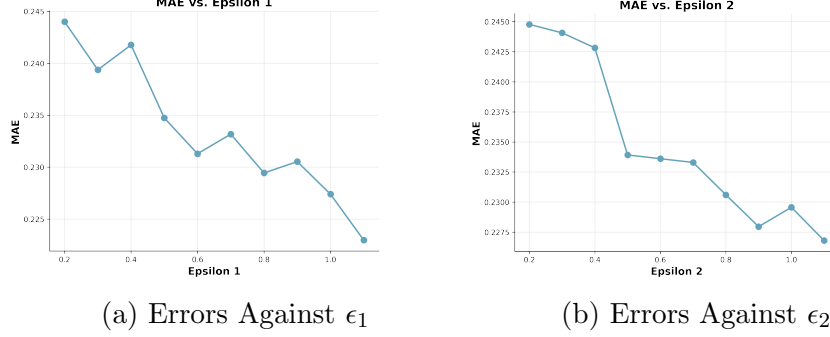


Figure 5.5: The two plots show the relationship between privacy parameters Epsilon 1 (ϵ_1) and Epsilon 2 (ϵ_2) and the mean absolute error (MAE). The first plot changes Epsilon 1 (ϵ_1) with Epsilon 2 (ϵ_2) fixed, and the second changes Epsilon 2 (ϵ_2) with Epsilon 1 (ϵ_1) fixed. Both plots show the effect of privacy parameters on MAE.

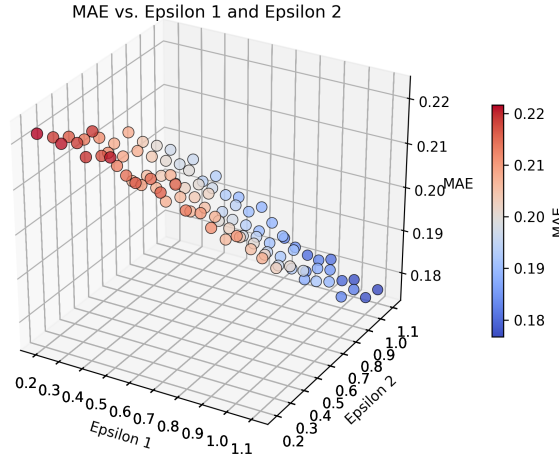


Figure 5.6: Comparison of Perturbation Results for Varying Epsilon Values: Scatter Plot of Mean Absolute Error (MAE) Values for Different Combinations of Epsilon 1 (ϵ_1) and Epsilon 2 (ϵ_2) Values Obtained through Overall Perturbation Analysis. (The size and color of the markers correspond to the magnitude of the MAE values. Higher MAE values are observed for certain combinations of epsilon values, indicating that the perturbation has a larger impact on the model's predictions for those parameter values.)

In the second stage of the optimization process, the research focuses on determining the optimal number of clusters for the clustering operation. Given the clustering results are highly related to the perturbation processing in the first part, it does not utilize perturbed data in this stage when determining the optimal number of clusters.

By considering the elbow method as shown in Figure 5.7 and computation cost, the number of clusters is set to 45.

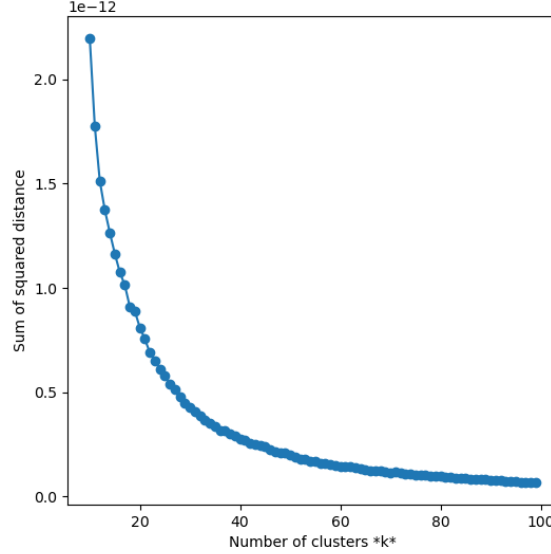


Figure 5.7: Plotting the Within-Cluster Sum of Squares (WSS) for Different Numbers of Groups.

In the last part, the research adopted an automatic approach for model tuning due to a large number of parameters, a wide range of possible values, and a lack of consensus on optimal settings. This approach [110] employs a Bayesian optimization framework in which the model’s overall performance is treated as a sample from a Gaussian process. By treating the training-testing process as a continuous function, this approach eliminates the need for brute-force searching or heuristic guesswork based on the human experience.

During the search process, a posterior distribution gradually develops as the number of observations increases. Exploration strategies such as upper confidence bound and expected improvement determine the next set of parameters to evaluate based on parameters-error pairs. This approach significantly reduces the number of steps required to find optimal parameters compared to brute-force search. the research used the method to estimate the approximate range of optimal parameters and then fine-tuned manually using the top-ranked results and adjusting specific parameters to achieve better performance in the experiment.

It was found when the number of latent factors k is 2, which is a reasonably low number, the experiment can have the best result. This may be caused by the sparsity of the dataset and the fact that the data is clustered. The structure of multi-layer perceptron in the NCF is set to be [128, 64, 32]. For each round of training and testing, The initial number of centroids for calculating the final prediction is set to be 20, used to generate the final user prediction is made to improve the final performance. Finally, the number can be reduced to 9 without affecting the final result. Comparatively speaking, the above parameters’ values yield the best average performance.

Table 5.2: Average Result Comparison

Champaign-Urbana								
Metric	P@5		R@5		P@10		R@10	
Data	Yelp	Google	Yelp	Google	Yelp	Google	Yelp	Google
NCF	0.014	0.081	0.051	0.004	0.014	0.008	0.070	0.0046
GUS	0.015	0.067	0.044	0.003	0.013	0.010	0.044	0.005
DMF	0.006	0.004	0.008	0.003	0.008	0.004	0.006	0.001
Meta	0.013	0.070	0.043	0.004	0.013	0.011	0.071	0.006

Phoenix								
Metric	P@5		R@5		P@10		R@10	
Data	Yelp	Google	Yelp	Google	Yelp	Google	Yelp	Google
NCF	0.015	0.090	0.066	0.006	0.017	0.010	0.070	0.007
GUS	0.015	0.077	0.053	0.004	0.015	0.008	0.044	0.004
DMF	0.012	0.048	0.042	0.002	0.077	0.004	0.007	0.0023
Meta	0.014	0.076	0.060	0.006	0.014	0.007	0.042	0.003

Results and Comparison

The research compared four models in this study: the Neural Collaborative Filtering (NCF) baseline model, the proposed framework (NCF+GUS), the Decentralized Matrix Factorization (DMF) model [17], and the federated recommender system (MetaMF) [59]. The research has selected these models for the following reasons:

- Given its prominence as a widely studied neural network-based collaborative filtering model, the NCF provides an ideal baseline for comparison in the research.
- As a decentralized approach that strongly emphasizes user privacy, DMF provides a valuable reference point for this research. This model can protect against malicious attacks and untrusted users by relying on gradient exchange and other security measures.
- MetaMF is a privacy-preserving RS model with a federated structure that strikes an excellent balance between privacy and prediction accuracy. Its semi-distributed structure separates central and local training processes.

All the models in this study, including the proposed framework, were evaluated on datasets that were filtered and pre-processed using the same methods. The overall average performance of each model is shown in Table 5.2. Each of the entries in Table 5.2 is the average performance of its corresponding model on all training-testing folds. If we expand each entry, we can get Figure 5.8. For example, Figure 5.8 displays precision comparisons for each training-testing dataset pair in the City of Phoenix, comparing the performance of the four models. The datasets are partitioned chronologically, with each point on the graph representing a distinct training-testing pair.

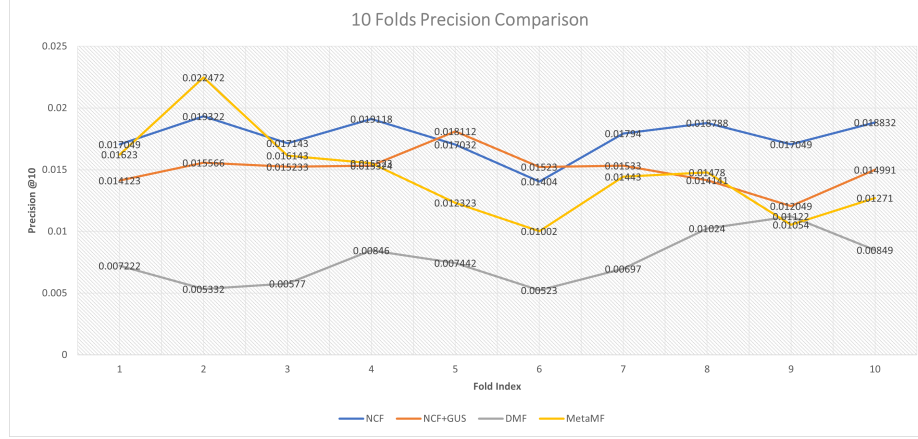


Figure 5.8: The Precision Comparisons among Four Models for Each Training-Testing Dataset Pair in the City of Phoenix. (The dataset is partitioned chronologically.)

The results from this experiment have lower precision and recall values compared to experiments using other popular datasets because of the sparsity nature of location-based services. Most users visit below three locations among thousands of POIs in the city. Nevertheless, comparatively speaking, of the four compared models, the non-private centralized recommendation model NCF had the highest precision since its trade-off from privacy concern is zero. However, in terms of the accuracy-privacy trade-off, the proposed framework, which includes the Generic User Synthesizer, has the best performance among the privacy models. It is worth noting that even though all models were trained and tested on the same datasets, their hyperparameters of choices are not necessarily the same.

As usual, the testing procedure simulates real-world scenarios where the recommendation system receives data sequentially and over time. During each round of cross-validation, the testing set becomes the training set, and the next fold is used as the testing set. The inflection points of the curves in Figure (5.8) represent the validation process results. To ensure that each fold has sufficient users, the research has adjusted the temporal window size of each fold so that each user has at least two ratings in the training set. This process is applied to all models being tested.

Integrating the Generic User Synthesizer (GUS) in the proposed framework shows a consistent and balanced trade-off in precision and recall on both datasets in a real-world setting. Possibly due to the lack or low density of data, DMF is not reaching its full potential. The GUS framework is a lightweight and efficient method for generating local recommendations, as it does not burden users with additional computational tasks. At the same time, the central server is trained on condensed data secured by the LDP standard and clustering, which not only makes the model's training process more efficient and fast but also preserves user privacy. The data-mineable synthetic data cannot be traced back to real users and can be transferred without compromising privacy.

Additionally, the GUS framework is more modularized and detachable than federated learning-based recommendation systems, allowing recommendation service providers to add it to their existing systems without significant changes easily. This

detachable feature is convenient for upgrading the recommendation system or using the desired model without sacrificing privacy. In contrast, other privacy-preserving models, such as the Decentralized Matrix Factorization (DMF) and the federated recommender system (MetaMF), require a complete switch to the privacy-preserving model and have limitations on the use of future data.

Lastly, an essential benefit of the GUS framework is that it generates synthetic user-item interaction data that can be used in future research and shared without concern for privacy liability. Because the synthetic users created by the framework are not associated with any real users and their identities cannot be traced back to the original data, there are no privacy concerns when transferring or sharing the data.

5.6 Summary

This study proposes a Privacy-Preserving Point-of-Interest Recommendation Framework that accurately estimates users' location while providing satisfactory prediction accuracy without disclosing GPS information. The framework utilizes a general user data collection approach based on LDP and data clustering, contributing to the framework's scalability and geolocation awareness.

The data uploading process is securely separated by integrating a third-party server that collects direct user feedback through LDP and obfuscates private data through clustering. Due to its sequential property, the data flow from the third-party server to the central server is still protected under the LDP standard.

This research introduces a flexible framework to resolve these challenges. It estimates users' location without accessing GPS data, utilizes an Honest but Curious (HOB) third party for clustering, and generates synthetic user data for POI recommendations. As a result, personalized recommendations are generated on local devices without compromising the privacy of sensitive information. The research further incorporates an automatic approach for model optimization using a Bayesian optimization framework, which reduces the number of steps needed to find optimal parameters.

To simulate a real-world scenario, the research has generated cross-validation folds in a temporally sequential manner. The proposed framework balances prediction accuracy and privacy protection through various processing techniques, strict testing conditions, and real-world datasets. The collected data is safe to store, transfer and ready for future usage without extensive liability. The results demonstrate a good level of user-item interaction information preserved in the virtual users, which is essential in training any recommender system.

The optimization process for the framework employs an automatic approach using a Bayesian optimization framework. The approach treats the model's overall performance as a sample from a Gaussian process, which significantly reduces the number of steps required to find optimal parameters compared to a brute-force strategy.

Future research will focus on enhancing the data-collecting mechanism with advanced contextual information processing tools and exploring user data synthesis techniques with generative adversarial networks and variational autoencoders. The study concludes that this novel framework is not only effective in balancing pre-

diction accuracy with privacy preservation, but it also has the potential to support cross-platform collaborations.

Chapter 6 Privacy-Preserving User Data Synthesis Scheme

6.1 The Final Motivation and Problem Description

Point-of-Interest (POI) recommender systems (RS) furnish users with personalized recommendations and promotions by filtering out extraneous information. Such services, exemplified by Yelp and Google Maps, facilitate the discovery of intriguing local venues, such as parks, restaurants, and museums, based on users' search histories and preferences. Despite the advantages POI RS offer in mitigating information overload, they also raise concerns regarding security, privacy, and control [111].

The proliferation of mobile devices and applications has been accompanied by mounting privacy apprehensions related to Point-of-Interest recommender systems. As privacy regulations struggle to adapt, the necessity for improved privacy safeguards is becoming more evident, as demonstrated by the European Union's General Data Protection Regulation (GDPR), the California Consumer Privacy Act (CCPA), and China's draft personal information protection law [88, 85, 41]. Both governments and corporations acknowledge the importance of addressing privacy concerns, leading to a shift toward privacy-preserving techniques instead of solely relying on personal data collection. Recent events, such as Google's \$392 million privacy lawsuit settlement across 40 U.S. states, illustrate this trend [112].

In light of escalating privacy concerns surrounding Point-of-Interest (POI) recommender systems (RSs), researchers have increasingly focused on developing models that strike a balance between privacy preservation and prediction accuracy [84]. The recommender systems community is examining various strategies to enhance this trade-off, including system decentralization to eliminate central servers that are often susceptible to malevolent hacking and data breaches [17, 90, 89]. Alternative approaches involve employing anonymity and obfuscation techniques, such as perturbing user feedback or concealing user identification information, to facilitate necessary data mining without disclosing user identities [113]. Additionally, incorporating traditional cryptography or differential privacy protocols into non-private recommender systems is frequently proposed as a means of bolstering privacy during information communication [94]. These endeavors underscore the ongoing demand for privacy-preserving features in recommender systems.

In the final study, an alternative solution was developed for preserving user privacy in Point-of-Interest (POI) recommender systems. The final approach entails generating synthetic users with a Conditional Tabular Generative Adversarial Network (CTGAN) and integrating them into the recommender system's dataset. The objective of this research is to ascertain whether deep neural networks, in conjunction with conventional methods such as clustering or differential privacy, can effectively synthesize virtual users while maintaining the recommender system's accuracy. The findings indicate that incorporating meticulously trained synthetic users into the dataset does not significantly affect the system's accuracy. This approach offers not only enhanced flexibility and generativity but also ensures secure storage and unrestricted transfer

of datasets. Furthermore, this method permits the evaluation of the impact of increasing the proportion of synthetic users in the original dataset. To validate this approach, experiments utilizing real-world datasets will be conducted to quantify the trade-off between privacy preservation and recommendation accuracy.

The main contributions to the communities and the overall research path are summarized as follows:

- Offering a comprehensive study on generating synthetic users and protecting users’ personal information in POI recommender systems using both CTGAN and clustering techniques.
- Investigation of the effectiveness of CTGAN-generated synthetic users compared to cluster-based synthetic users in terms of prediction accuracy.
- Evaluation of the trade-off between privacy preservation and recommendation accuracy by increasing the ratio of synthetic users in the original dataset, using real-world datasets, and providing a comparative analysis between CTGAN and clustering methods.

The structure of this chapter is as follows: The section “Utilized Techniques and Important Information” reviews the relevant techniques and provides background information that informs the proposed framework. The section “Notifications and Formulas in Study” outlines the foundation and notations related to the framework. Section “Experiment Design” presents the datasets the research has used, the methodology, the metrics the research employed to evaluate the framework’s performance, and the hyperparameters tuning strategy. The section “Results and Explanations” shows the results from using different datasets and various ratios of the insertion of synthetic data. Finally, the Section “Summary” concludes the research and outlines the following research directions.

6.2 Utilized Techniques and Important Information

This section of this research provides an overview of the relevant research, such as the newly updated clustering techniques in this research, data synthesizing for RS privacy preservation, and Generative Adversarial Networks (GANs) and Conditional Tabular Generative Adversarial Networks (CTGANs). The aim of this section is to provide convenient background information for the readers and to highlight how they contribute to the approach proposed in this paper.

POI Recommender Systems

Point-of-Interest recommender systems represent a technological advancement that offers personalized recommendations and promotions to customers, with the aim of streamlining the discovery of new locations [82]. These systems, as location-centric variants of recommender systems, primarily focus on recommending geographic locations or Points of Interest to users based on their preferences and historical location data. POI RSs can be accessed through many popular services, such as Yelp or Google

Maps, but they raise privacy and security concerns. Nonetheless, the popularity of POI RSs continues to grow due to the increasing number of mobile devices and their capacity to simplify and enhance the customer experience.

PP RSs can be categorized based on the underlying technology and the type of POI they recommend. For instance, collaborative filtering-based POI RSs generate recommendations according to the collective behavior of users, such as check-ins, ratings, and preferences. Conversely, content-based POI RSs offer recommendations based on POI attributes and users' historical preferences. Hybrid POI RSs employ both collaborative filtering and content-based methods to deliver more personalized recommendations. Well-known POI RSs encompass services such as Yelp, Google Maps, and Foursquare.

In this dissertation, the focus is placed on collaborative filtering-based Point-of-Interest Recommender Systems, which are widely acknowledged as the most prevalent and successful type of POI RSs. Specifically, the Neural Collaborative Filtering (NCF) model, which utilizes neural networks to enhance collaborative filtering, was chosen as the evaluation target due to its broad applicability and dominant position within the field of recommender systems [102]. By employing a generic model, the findings presented in this dissertation are comprehensive and applicable across various use cases.

GAN and CTGAN

Generative Adversarial Networks (GAN) [114] are a class of deep learning models that have gained significant attention in recent years for generating high-quality synthetic data that resemble real-world data [115]. GANs consist of two neural networks that compete with each other in a zero-sum game: a generator network that generates synthetic data and a discriminator network that tries to distinguish between natural and synthetic data. GANs have shown remarkable success in various applications, including image and speech synthesis, text-to-image translation, and data augmentation.

- The generator is trained to create realistic data, which subsequently serves as negative training examples for the discriminator.
- The discriminator, in turn, is tasked with differentiating between the generator's fabricated data and genuine data, penalizing the generator for generating implausible results

As shown in Figure 6.1, Both the generator and the discriminator are neural networks. The generator output is connected directly to the discriminator input. Through back-propagation, the discriminator's classification provides a signal that the generator uses to update its weights. At the onset of training, the generator initially produces evidently false data, enabling the discriminator to identify its artificial nature swiftly.

ctGAN (conditional tabular GAN) [116] is a variant of GAN that is specifically designed for generating synthetic tabular data. ctGAN extends GAN by incorporating conditional generation capabilities, allowing users to specify conditions on the

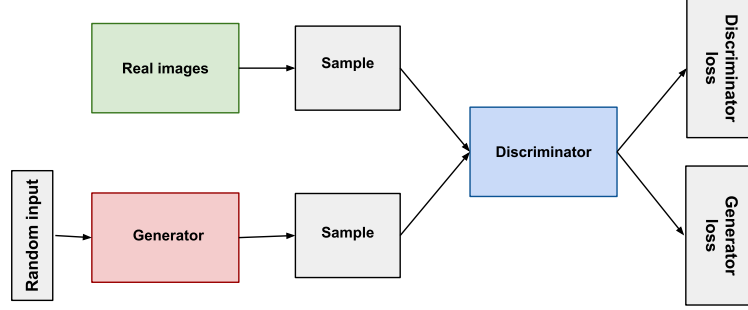


Figure 6.1: The Basic Diagram of a Generative Adversarial Network

generated data. Specifically, a variational Gaussian Mixture Model (VGM) is utilized for the numerical variables, and the Wasserstein GAN loss function is used for the gradient penalty. The ctGAN can be used to generate synthetic data that follow the identical statistical distributions as the original data, making it useful for tasks such as data augmentation, privacy-preserving data sharing, and synthetic data generation for model training.

The Wasserstein GAN loss function consists of two parts: the discriminator loss (L_D) and the generator loss (L_G).

Discriminator Loss (L_D):

$$L_D = \mathbb{E}_{x \sim p_{\text{data}}}[D(x)] - \mathbb{E}_{z \sim p_z(z)}[D(G(z))] \quad (6.1)$$

Generator Loss (L_G):

$$L_G = -\mathbb{E}_{z \sim p_z(z)}[D(G(z))] \quad (6.2)$$

Variables:

- L_D : Discriminator loss
- L_G : Generator loss
- x : Real data sample
- p_{data} : Data distribution
- $D(x)$: Discriminator's output for a real data sample x
- z : Random noise sample
- $p_z(z)$: Noise distribution
- $G(z)$: Generator's output for a noise sample z
- $D(G(z))$: Discriminator's output for a generated data sample $G(z)$
- \mathbb{E} : Expectation

Data Synthesizing in Privacy-Preserving Recommender Systems

Data synthesis refers to the process of generating artificial data that resembles real-world data with the aim of preserving the privacy of the original data while still enabling the use of the synthesized data for various purposes [117]. This technique has gained increasing attention in the field of privacy-preserving data analysis, where the goal is to maintain the privacy of individuals or organizations while still allowing researchers or practitioners to leverage the information contained in the data. Data synthesis can be achieved through various methods, including randomization, perturbation, and generative models such as GANs. These techniques aim to synthesize data that accurately represents the statistical properties of the original data while still ensuring the privacy of the original data.

Several models for synthesizing user data with the goal of preserving privacy have been proposed, including GANs, Variational Autoencoders (VAEs) [118], and Bayesian Networks [119]. However, empirical studies in the field of RS remain limited. The objective of these models is to generate synthetic data that closely resembles the original user data, with each model presenting its own unique challenges. In this research, GANs are utilized, precisely the CTGAN model, due to the advantage it offers in terms of differential privacy. As the generator in GANs does not have access to actual data during the training process, privacy protection can be achieved with ease. Additionally, GANs have been demonstrated to generate high-quality training models more readily when applied to large-scale real datasets.

Clustering Techniques

Clustering algorithms can be broadly classified as hierarchical and non-hierarchical methods, depending on the manner in which they group data points [120]. Clustering has numerous real-world applications, including market segmentation, anomaly detection, image segmentation, and recommendation systems like POI recommender systems. The resulting clusters are expected to exhibit homogeneity within themselves and heterogeneity with other clusters to ensure their effectiveness.

Moreover, clustering is a fundamental technique in data mining and machine learning that is used to group data points based on their similarities. In this research, clustering is employed to group similar users based on their estimated locations. Specifically, this research utilizes k-means, a popular non-hierarchical clustering method, to cluster POIs based on their geographic proximity and semantic similarity. In later sections, it will explain the reason behind the choice of K-means among all the other clustering techniques. K-means divides the data into k clusters, with each cluster represented by its centroid. The algorithm iteratively assigns each user to the nearest centroid and re-positions the centroid based on the mean position of all users in the cluster. The goal is to identify groups of users that share similar attributes and interests.

6.3 Notifications and Formulas in Study

This section introduces the different model fundamentals, the notations, problem definitions, and the metrics and evaluation methods.

Introduction to Neural Network Based Matrix Factorization

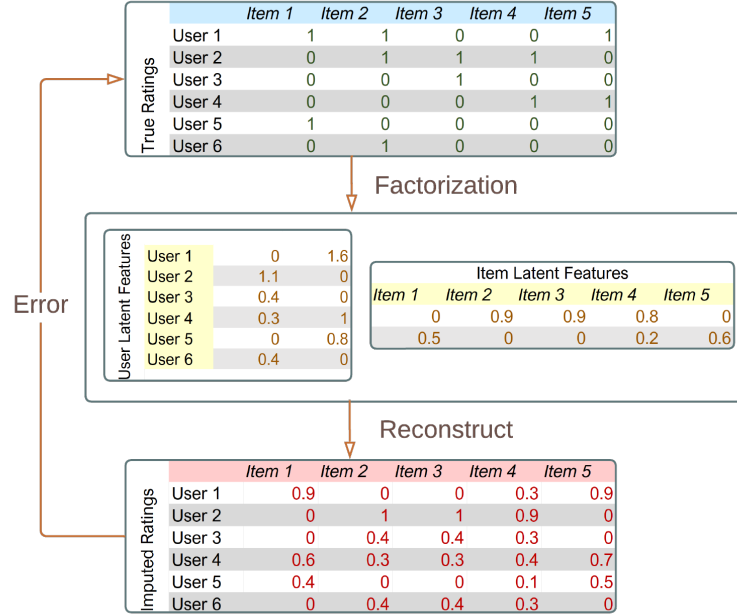


Figure 6.2: Example of the Conventional Collaborative Filtering Technique

Like most conventional RS, the core component is built upon Collaborative Filtering (CF) [43]. Iterative Matrix Factorization (MF) was the most used version of CF for recommendation models. It is worth mentioning that the MF decomposes the feedback matrix into two lower-rank matrices, as in figure 6.2. Iteratively, the algorithm stops when the total error between the observed true ratings and the predicted corresponding ratings is minimized. When reconstructing a feedback matrix or utility matrix as to which some may refer, the missing entries are imputed, giving people a vague idea of users' potential or future references. The minimization problem is formulated as below:

$$\min_{p_u, q_i} \sum_{r_{ui} \in R} (r_{ui} - \hat{r}_{ui})^2 + \lambda(\|q_i\|^2 + \|p_u\|^2) \quad (6.3)$$

where we use u to denote a user (customer) and i an item (POI). If we define U and I as the user and item sets, then $u \in U$ and $i \in I$. In the above equation, r_{ui} indicates an observed rating indicating the preference of user u over item/POI i . Correspondingly, the \hat{r}_{ui} represents predicted ratings. The rating matrix R is composed of all the observed and unobserved feedback or visiting records. Thus the prediction function for \hat{r}_{ui} can be written as below:

$$\hat{r}_{ui} = p_u^T \cdot q_i = \sum_k p_{uk} q_{ik} \quad (6.4)$$

where the column vectors p_u and q_i from the decomposition result, two sub matrices P and Q , represent the user and item latent factors, respectively. Here, k is the dimension of the latent space.

Collaborative Filtering and Neural Collaborative Filtering are two popular techniques used to model user-item interactions in recommendation systems. CF works by representing the rating matrix in a latent space using inner product optimization, while NCF uses a neural network to replace the lower-rank matrices used in CF. The advantage of NCF is that it can capture non-linear interactions and incorporate weighted user/item features, making it a more generalized approach that can handle complex interactions.

It is worth noting that NCF models, in general, can be designed to predict explicit ratings or implicit feedback (such as the probability of interaction). The specific design of the model, such as the choice of the last activation function or the loss function, will determine the final output of the model. In this dissertation, the last activation function was modified to be a linear function, which allows the model to output a continuous value, which can be interpreted as a rating. The choice of generating explicit feedback is to assist the comparisons of trade-offs better. In other research, a sigmoid or softmax activation function might be used at the end.

In summary, Neural Collaborative Filtering (NCF) is a widely adopted neural network-based technique for modeling user-item interactions in recommendation systems. The research has selected NCF as the baseline model because it has been shown to be reliable, is widely used in the literature, and has been extensively studied by peer researchers. Additionally, NCF's generic usage aligns well with the design of the framework, which aims to provide a flexible and scalable approach for recommendation tasks.

Notations for Clustering and User Similarity

The users' locations need to be estimated using the GPS locations of their previously visited POIs. If we let Lat_u represent the latitude of a user, Lon_u the longitude, and I_u the set of all POIs a user visited, then we have the following equations:

$$Lat_u = \frac{1}{|I_u|} \sum_{i \in I_u} Lat_i \quad (6.5)$$

$$Lon_u = \frac{1}{|I_u|} \sum_{i \in I_u} Lon_i \quad (6.6)$$

where I_u denotes all the POIs the user u has visited, and U_i denotes all users that have feedback for the item/POI i .

After clustering, centroids from the output are utilized as virtual users or synthetic users, even though their information results from aggregation. The way to create ratings for the artificial users is by taking the average of all users in each cluster.

Suppose the users are split into $|C|$ clusters where each cluster $c \in C$ and all users $u \in U_c$ belonging to that cluster. We have the following equation to impute the ratings for each centroid from the clustering:

$$\hat{r}_{ci} = \sum_{u \in U_c} r_{ui} / |U_c \cap U_i| \quad (6.7)$$

where U_i denotes all users that have feedback for the item/POI i . The synthetic users with ratings can then be sent to the central server for model pre-training and POI similarity calculation.

Evaluation Metrics

I am using both Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) to evaluate the model's prediction accuracy to better study the trade-off between privacy preservation and the RS model's accuracy. RMSE and MAE are famous evaluation metrics for regression tasks, and they measure the difference between the predicted ratings and the actual ratings. While both metrics measure prediction errors, they have different properties: RMSE gives more weight to more significant errors, making it more sensitive to outliers, whereas MAE treats all errors equally.

It is worth mentioning other metrics like precision, recall, mean average precision or normalized discounted cumulative gain are all famous evaluation metrics for modern recommender systems. These metrics value more on how well the system can help users make better decisions, so they are affected mainly by the top recommended items. Moreover, in many cases, it is indeed that users care only about the top few recommended items or POIs. However, the focus is more on the overall impact of integrating synthetic data into the system.

The following equations 6.8 and 6.9 show the details of the definition:

$$RMSE = \sqrt{\frac{\sum_{u,i \in T_e} (r_{ui} - \hat{r}_{ui})^2}{n}} \quad (6.8)$$

$$MAE = \frac{\sum_{u,i \in T_e} |r_{ui} - \hat{r}_{ui}|}{n} \quad (6.9)$$

where T_e is the entire test set, and n is the number of ratings compared. Lower values of RMSE and MAE indicate better prediction accuracy, with an RMSE or MAE of 0 representing perfect predictions.

6.4 Experiment Design

This section gives a description of the datasets and presents the experiment results in comparisons. Lastly, the parameters and their tuning are discussed.

Table 6.1: Datasets Statistics

Area	#Users	#POIs	#Feedback	Density
CU	11953	1579	33990	0.1802%
PH	168585	3784	453507	0.0711%

The Datasets

The Original Datasets

The selected datasets described user feedback on Yelp in two metropolitan areas: Champaign–Urbana (CU) and Phoenix (PH). This feedback took the explicit ratings in $\{1, 2, 3, 4, 5\}$, where user-item ordered pairs were unique. Thus, each item could only be rated once by any particular user. These two datasets were chosen for contrast in size. While CU contained fewer users in a smaller area, PH consisted of a larger population and rich user feedback. The sparsity of the datasets is comparatively much lower than conventional recommendation datasets. A straightforward comparison of statistics of the four areas can be found in Table 6.1.

The Synthetic Dataset from CTGAN

The synthetic dataset was obtained through training a conditional generative adversarial network (CTGAN). The research adopted a framework from the Synthetic Data Vault (SDV) for its compatibility with tabular data [116]. The GitHub documentation for the CTGAN library can be found at <https://github.com/sdv-dev/CTGAN>.

Before inputting the datasets into the CTGAN model, the list of tuples was transformed into a rating matrix in which rows represent users and columns denote items. This reformatting aimed to emphasize the interactions between users and locations better. Subsequently, the matrix was filtered to exclude users who had rated too few locations, such as low-quality users with only one rating. For testing purposes, the number of ratings required for users in the PH dataset is increased to 12, as demonstrated in the next section. It is important to note that during CTGAN training, any number of users can be removed as desired since the objective at this stage is not to simulate real-world situations but rather to extract the most valuable information from the training data.

In practice, retaining the original data format as a list of *userID*, *itemID*, *rating* tuples prevents the CTGAN model from understanding the significance of either *userID* or *itemID* unless set as the primary key. Even so, it leaves only a column of rating values for training, and feeding such data into the model would result in the generation of highly random and useless synthetic data. Instead, transforming the original data into a rating matrix more effectively captures the relationships between users and POIs. During training, the user index was manually added as a separate column, serving as the key.

The tuning of CTGAN hyperparameters in this implementation was based on the suggested values provided in the original research paper and some minor manual adjustments. Due to the extensive training times required for both CTGAN

and NCF, reliance on rough estimations of the synthetic data quality was necessary when adjusting the hyperparameters for CTGAN. To roughly estimate the quality of the generated synthetic data, a simple conventional collaborative filtering model with a five-fold cross-validation approach was employed. In other words, each time the training of the CTGAN model was completed, the synthetic data was sampled and the quality of the generated dataset was tested using a minimal CF model with cross-validation. For the PH dataset, manual tuning was halted once the mean absolute errors (MAEs) reached 0.6719, 0.672, 0.6732, 0.6718, 0.6721, a considerably good result. For the Champaign-Urbana dataset, the process was stopped at the value of 0.9676, 0.9666, 0.9658, 0.9580, 0.9712. Considering the complexity of the CTGAN hyperparameters and as the detailed analysis of each of them is beyond the scope of this investigation, their discussion has been omitted in this context.

The Synthetic Datasets from Clustering

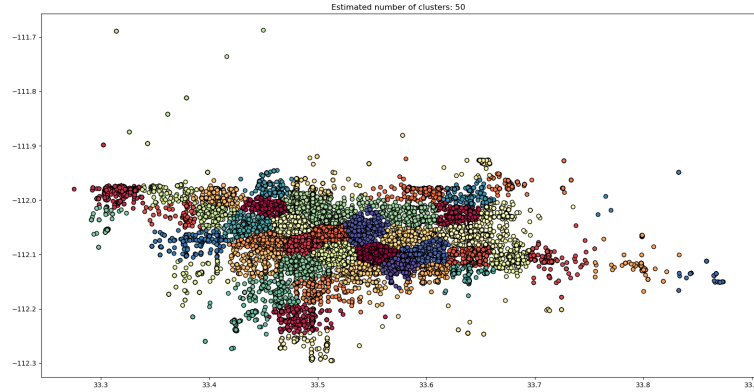


Figure 6.3: The visualization of User Location Clustering (Phoenix). (The clustering algorithm shown in the picture is K-mean.)

Various clustering methods were considered. Initially, DBSCAN [107], a widely-used geolocation clustering method that automatically determines the optimal number of clusters, was employed. However, this method resulted in grouping users who were distant from one another into a single cluster. Conversely, when implementing OPTICS clustering [108], another density-based clustering algorithm, the same problem was encountered. Moreover, reducing the maximum distance to prevent the formation of excessively large groups led to a considerable portion of users being classified as noise (not in any cluster).

Furthermore, hierarchical clustering methods such as Partition Around Medoids (PAM) [109] clustering algorithm were investigated. Regrettably, despite diligent tuning efforts, this method produced numerous empty clusters and was influenced by users active beyond the metropolitan area (noise). Consequently, the K-means algorithm demonstrated superior performance in terms of encompassing all users and generating balanced clusters, as shown in Figure 6.3. Spectral clustering [67] was

also evaluated by initially constructing the affinity matrix and then using the affinity matrix to perform clustering; however, this approach proved too computationally intensive, making it impossible to cluster large datasets within a reasonable time. To avoid these limitations, K-means clustering was ultimately selected as a more efficient and competitive alternative.

In both the CU and PH datasets, while clustering, a user filtering criterion was applied to exclude users who provided only one or two ratings. This was done to ensure that only users who could contribute meaningfully to collaborative filtering-based RS were included. It is intuitive that users with only one or two feedback records would have limited ability to suggest items to other users, making them less useful for recommendation purposes. Hence, excluding these users improves the quality of the clustered dataset.

Real-Synthetic Blend

Initially, the CU Yelp dataset was partitioned into base training and testing subsets using a 75-25 split by time. The combined training dataset used for training NCF consisted of a mix of real users from the base training subset and synthetic users generated by CTGAN or clustering. The proportion of real and synthetic users, *mix_rate*, was the varying factor in trials of ensemble subsets.

To distinguish between real and synthetic users in the ensemble dataset, the number of unique real users was added to every synthetic user identifier. Both real and synthetic tuple lists were pivoted into matrices of ratings. It is important to note that the real data refers to the training set of the original dataset, and the synthetic data refers to the entire sampled synthetic data pool. Rows of the original rating were then removed according to the *mix_rate*. For example, if the value of *mix_rate* is 0.5, it would mean that 50% of the original users are removed. To compensate for the removed data, the same amount of ratings from the synthetic data was added. It is essential to point out that real users were not directly replaced with synthetic users because the density of the synthetic data and the original data is drastically different. The combined rating matrix was unstacked to restore its original format, which is a list of tuples as *userID, itemID, rating*.

During the entire data processing stage, the correspondence relationships between users in the training set and test set were maintained. It is important to note that during the experiment, manipulating and slicing the training data frames must be consistent with the test set. Moreover, the indices of the remaining users or items, as well as the synthetic counterparts, had gaps, which is problematic for most recommender systems. Re-indexing solved this problem by iterating through the dataset and redefining user and item identifiers to be the least unoccupied integer. This effectively closed these gaps and sequentially filled the counting numbers from 0 to the number of respective users and items.

Evaluation

In an ideal setting, users would use their local RS to train on the synthetic data and provide optimal recommendations without the need to disclose private data. To sim-

ulate such a scenario, the experiment should be designed with the assumption that a central RS is responsible for pre-training the NCF model. Each user could then download the pre-trained model and finalize personalization by performing adaptive training using their historical record on their local device. However, implementing such a simulation is computationally expensive. With a thousand users, adaptive training would have to be performed a thousand times to complete only one instance of system evaluation. Consequently, an alternative approach was taken, which involved mixing the synthetic data with real users to evaluate the impact of integrating synthetic data into the original dataset.

Furthermore, a time-dependent dataset splitting method was conducted, ensuring time dependencies between the training-test pairs. The dataset was sorted temporally since using future ratings to predict past behaviors is not realistic. Temporal cross-validation was not implemented for two reasons. First, the sparsity of POI datasets is already low, and further dividing it into multiple folds temporally would increase sparsity. Second, as synthetic data does not contain timestamps, adding more strategies for injecting different synthetic data into each fold could make the evaluation process unnecessarily complex.

The evaluation procedure is straightforward. For a single-round evaluation algorithm, please refer to Algorithm 1. In the experiment, the value of *mix_rate* iterates the set 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9.

Algorithm 6: Data Mixing and Model Evaluation

Input: $D_{\text{orig}}, D_{\text{synth}}, \text{mix_rate}$

Output: RMSE, MAE

- 1: Split D_{orig} into D_{train} and D_{test} with a 75% and 25% ratio
 - 2: Randomly remove $(1 - \text{mix_rate})$ of the users in D_{train}
 - 3: Select the same amount of rating records from D_{synth}
 - 4: Create $D_{\text{mixed_train}}$ by combining the remaining original rating records with synthetic rating records.
 - 5: Train the NCF model on $D_{\text{mixed_train}}$
 - 6: Evaluate the model on D_{test}
 - 7: Return RMSE and MAE
-

In Algorithm 1, D_{orig} represents the original dataset; D_{synth} represents the synthetic dataset; D_{test} remains the test set, and $D_{\text{mixed_train}}$ is the prepared dataset.

Parameters and Tuning

Various parameters can directly impact the results; therefore, the approach taken in this study combines an automatic method for model tuning with subsequent manual adjustments, rather than relying solely on brute-force searching or heuristic random guessing based on human experience. Initially, a Bayesian optimization framework is adopted, treating the overall performance of the model as a sample from a Gaussian process. This approach [81] considers the entire training-testing process as a continuous function.

With this framework, a reasonable set of parameters is determined when the mix_{rate} is 0.5. Following the Bayesian optimization, manual tuning is employed to further refine the parameters, ensuring optimal performance for the NCF model. These parameters are then applied to all other cases.

During the NCF training process, the $gamma$ value is set to 0.9 and the $step_{size}$ to 10, which means the learning rate is multiplied by 0.9 every 10 epochs. The $hidden_{size}$, determining the size of the hidden layers in the neural network, is set to 16. It is worth noting that there is a structure of user and item embedding sub-networks within the NCF model. These sub-networks are designed as Multi-Layer Perceptrons (MLPs), a type of feedforward artificial neural network consisting of multiple layers of neurons. The structure, or the number of neurons in each layer, of the MLP is set to [16, 32, 64]. Finally, the batch size is set to 128. The number of training epochs is limited to 100, and the initial learning rate is set to 0.001.

6.5 Results and Explanations

The experimental methodology involved using two previously introduced datasets obtained from the Champaign-Urbana and Phoenix City metropolitan areas. For each dataset, three different dataset compositions were evaluated, namely the original dataset, the dataset partially replaced with virtual users generated from clustering, and virtual users generated from CTGAN. To assess the impact of different mix_{rate} values in integrating synthetic data, each composition was examined. It should be noted that the reason for evaluating by mixing the original data with synthetic data is due to the overwhelming computing time required to personalize for each user using only synthetic data.

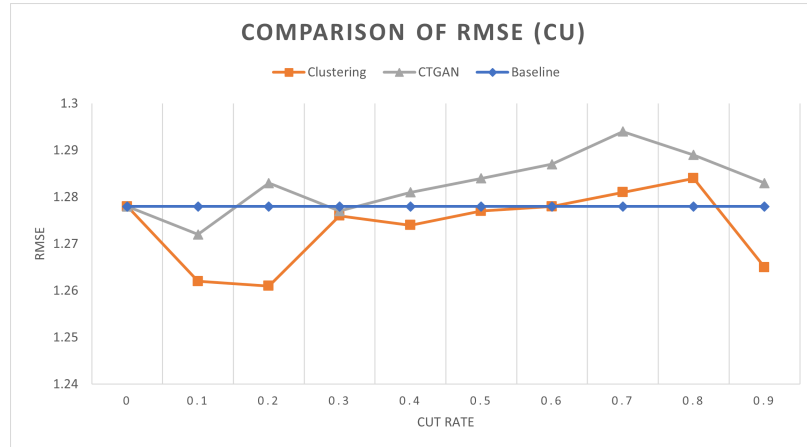


Figure 6.4: Comparison of RMSE Values for Different Cut Rates Using Original Data (Blue), Clustered Data (orange), and CTGAN-Generated Data (grey) on the Dataset from CU.

Figure 6.4 and 6.5 illustrate the results, where the $x - axis$ denotes the value of mix_{rate} , and the $y - axis$ represents the value of RMSE. The baseline (shown in blue) corresponds to the NCF model that operates solely on the original dataset

Table 6.2: RMSE Result Comparison for Different *mix_rate* from CU

<i>mix</i>	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Cluster	1.278	1.262	1.261	1.276	1.274	1.277	1.278	1.281	1.284	1.265
CTGAN	1.278	1.272	1.283	1.277	1.281	1.284	1.287	1.284	1.289	1.293

without any synthetic data injection. The baseline is represented as a straight line, indicating that it has not undergone any modifications. However, it is noteworthy that the RMSE value of the baseline is relatively high when compared to popular datasets like *MovieLen* [77], primarily due to the sparsity level of the POI datasets.

The remaining data curves correspond to datasets that were partially replaced with synthetic users generated from clustering (orange) or CTGAN (grey), respectively. For Figure 6.4, which represents the CU dataset, clustering-based mixed data yields better results than CGGAN-based augmentation, though the absolute differences between the three runs are not substantial. During the generation of synthetic data, a filter was applied to exclude users who rated fewer than 3 items. In other words, all users with only one or two ratings were excluded during the clustering and CTGAN training stages.

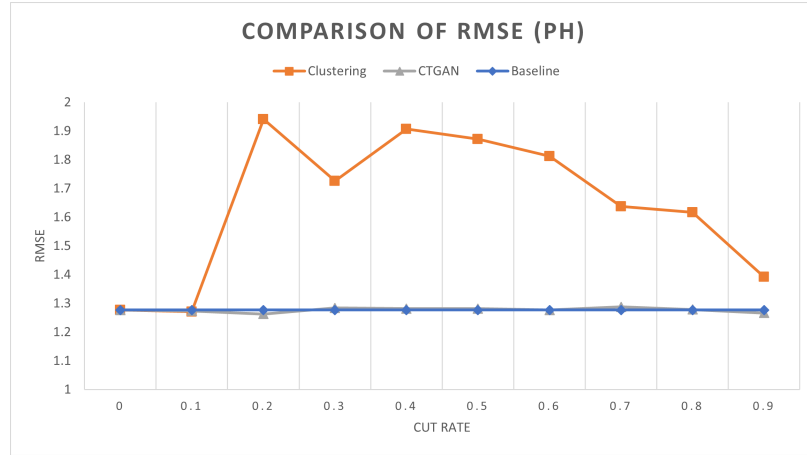


Figure 6.5: Comparison of RMSE Values for Different Cut Rates Using Original data (blue), Clustered data (orange), and CTGAN-Generated Data (grey) on the Dataset from PH.

By contrast, Figure 6.5 shows a different scenario where a meaningful change was made. During the training of CTGAN, the density of the dataset was significantly increased by excluding all users with fewer than 12 ratings, as opposed to 3. It is worth noting that no data or users were modified or filtered during the evaluation stage to simulate real-world conditions. Instead, the training of CTGAN was optimized to optimize the composition of the feeding dataset. As evident from the figure, the performance of the NCF model using the original dataset and the dataset augmented with synthetic users generated via CTGAN are almost indistinguishable. The clustered data was generated in the same manner as before.

Table 6.3: RMSE Result Comparison for Different *mix_rate* from PH

<i>mix</i>	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Cluster	1.277	1.271	1.941	1.726	1.907	1.872	1.813	1.637	1.617	1.392
CTGAN	1.277	1.274	1.263	1.284	1.282	1.282	1.277	1.288	1.279	1.266

6.6 Summary

The concluding chapter of this study presents a deep dive into the generation and application of synthetic users within the framework of Point-of-Interest (POI) recommender systems (RS). These systems serve to personalize recommendations and promotions by filtering extraneous information, as seen with services such as Yelp and Google Maps. They aid users in discovering local venues of interest based on their search histories and preferences. Yet, in mitigating information overload, these systems concurrently give rise to concerns related to security, privacy, and control.

With the surge of mobile devices and applications, privacy anxieties related to POI RS have become increasingly pronounced. As privacy regulations grapple with keeping pace, the call for reinforced privacy protections becomes more conspicuous. The culmination of this study provides an innovative solution for preserving user privacy in POI recommender systems, proposing the generation of synthetic users with a Conditional Tabular Generative Adversarial Network (CTGAN) and their subsequent integration into the recommender system’s dataset. This research seeks to determine whether deep neural networks, coupled with traditional methods such as clustering or differential privacy, can effectively synthesize virtual users while preserving the recommender system’s accuracy. The evidence suggests that the inclusion of meticulously trained synthetic users into the dataset does not significantly compromise the system’s accuracy. This method not only offers enhanced flexibility and generativity but also ensures secure storage and unrestricted transfer of datasets. Furthermore, it allows for the evaluation of the impact of increasing the proportion of synthetic users in the original dataset.

The significant contributions of this research to the communities and the overall research trajectory can be encapsulated in three parts: first, the provision of a holistic study on generating synthetic users and safeguarding users’ personal information in POI recommender systems, employing both CTGAN and clustering techniques; second, the investigation of the efficacy of CTGAN-generated synthetic users compared to cluster-based synthetic users in terms of prediction accuracy; and third, the evaluation of the trade-off between privacy preservation and recommendation accuracy by increasing the ratio of synthetic users in the original dataset, employing real-world datasets, and conducting a comparative analysis between CTGAN and clustering methodologies.

Looking forward, the plan is to expand the research by incorporating local differential privacy on top of CTGAN and exploring more advanced contextual information processing tools. It is also possible to aim to investigate other practical synthetic generation tools and delve deeper into parameter tuning for large non-linear systems. This chapter sets the stage for future work in enhancing privacy and performance in

POI recommendations by leveraging synthetic data and advanced techniques.

Chapter 7 Conclusions and Future Work

7.1 Research Accomplishments

In the past two decades, recommender systems have gained prominence as an indispensable component of personalized technology, profoundly impacting various sectors such as e-commerce, entertainment, and social media platforms. The growing significance of recommender systems can be attributed to their capacity to process large volumes of data and provide customized recommendations to users, ultimately enhancing user experience and satisfaction. Notable milestones in the evolution of recommender systems encompass the advent of collaborative filtering in the late 1990s, the influential Netflix Prize competition in 2006, and the emergence of deep learning-based recommendation models in recent years. These developments have considerably improved the precision, scalability, and versatility of recommendation algorithms. Nevertheless, the flourishing and widespread adoption of recommender systems has concurrently elicited significant privacy concerns. The extensive acquisition and analysis of user data for personalization purposes pose potential threats to individual privacy and even national security, underscoring the necessity for robust privacy-preserving mechanisms within the domain of recommender systems.

This dissertation primarily focuses on exploring potential solutions to the privacy-preserving challenges currently faced by recommender systems. The scope of the research is specifically narrowed down to the domain of POI recommender systems. Throughout the dissertation, the author recounts the investigative journey undertaken to develop privacy-preserving strategies.

Initially, an empirical study was conducted to examine the use of clustering techniques in concealing users' identities. Subsequently, the research introduced a trusted third party, eliminating the need for a superuser. As the study progressed, the protection and utilization of users' comments were also addressed. In response to the concerns raised by peers, the privacy budget was quantified by introducing differential privacy, leading to the formulation of a distinctive approach to safeguarding user privacy.

This unique perspective involved generating virtual user data through various methods, with the cutting-edge CTGAN technique proving to be particularly successful in synthesizing user data. The results demonstrated a seamless replacement of real user data, thereby highlighting the potential value of this approach in alleviating privacy concerns related to data storage and transfer in the field of recommender systems.

In summary, this dissertation can be concluded as five parts:

- Investigation on the use of clustering techniques to broaden the user's identity.
- Introducing a trusted and distributed third party.
- Utilize and protect the users' comment contextual information.

- Add quantification by introducing differential privacy.
- Using synthetic user data to replace real users' data.

An Empirical Study on Integrating Clustering Techniques into Recommender Systems

This approach, intuitively speaking, is to hide every single user's identity into a group. The study in chapter 2 presents a framework that uses "super users" to collect nearby user data indirectly instead of using the central server. The research demonstrated that by using a grouped preference, the users can still get personalized recommendations that are reasonably well-suited.

However, this research has remaining challenges. First, the clustering mechanism is sometimes not accurate since the users near the "super users" are not necessarily the users that are similar to them. It caused the group references not to be correctly grouped together, which affected the accuracy of the prediction heavily. Second, the number of ratings in each batch is too small to be usefully applied in practice. These challenges lead me to the subsequent research.

Introducing Trusted and Distributed Third Party

To resolve mentioned issues, such as preventing the risk of having a central server and decreasing the clustering errors, a recommender system network has been introduced. More importantly, this approach allows the integration of geolocation information. In the earlier research, the users' visiting records are clustered based on the assumption that similar users will gather together. However, in reality, this is not always true. By formally introducing a trusted third party, we can cluster users based on the entire user's history visit records and the fuzzy physical locations of the users.

Furthermore, not only did the research upgrade the clustering mechanism, but the datasets were also expanded to include the Chicago city area. The target recommender system model has also been improved from a plain NMF to a more advanced biased MF model. Furthermore, this is when the idea of using virtual users to replace real users began to emerge.

Protecting Contextual User Information

Subsequently, due to the emerging trends in the recommender system community, an increasing amount of contextual information has been leveraged to provide more accurate recommendations. This situation is particularly pronounced in POI recommender systems, as mobile devices are highly efficient in collecting various types of personal information. As a result, the focus shifted towards examining the natural process through which individuals select their POIs, with an emphasis on the role of user-generated comments.

By emulating real-world scenarios, the comment sections for each POI were aggregated and then subjected to an NLP technique known as "Doc2Vec" for vectorization. This allowed for the comparison of similarities, which could subsequently be utilized

in the updated objective function. Moreover, the clustering technique was further refined by incorporating both users' rating records and estimated GPS locations. Ultimately, the results demonstrated significant success, as the accuracy of the recommender system was notably enhanced, along with the added feature for protecting user comments.

Quantification of The Privacy Budget

Throughout the research, consistent inquiries from peer reviewers were received regarding the quantification of privacy-preserving features and the inclusion of an attack model. Consequently, it was decided to incorporate differential privacy, specifically local differential privacy (LDP), to develop a more robust and convincing framework.

During this stage of research, a variety of clustering techniques were explored, with users' estimated GPS information serving as the sole factor for calculating user similarities. The introduction of LDP protected both users' rating values and rating behaviors. Owing to the integration of LDP, the role of the trusted third party was relaxed to that of a semi-trusted third party. This framework, as a result, offers a versatile data collection mechanism for any recommender system model without necessitating alterations to the core components of the RS model.

Total Replacement of Real User Data

In the final segment of the dissertation, the research formally acknowledged the privacy-preserving technique for POI recommender systems, which involves generating synthetic user data. Much of the prior research focused on utilizing centroids as users to assist customers in personalizing their services. This concept led me to employ state-of-the-art techniques, such as GANs, to generate synthetic data using real or slightly perturbed user data. During this study, an aspect that was ignored previously was discovered, which is the ability to filter and optimize the training set as desired since it represents the mineable information the research sought to extract.

Ultimately, this approach proved successful, as CTGAN-generated synthetic users enabled customers to generate recommendations of nearly equivalent quality regarding RMSE and ME. At this juncture, synthetic user data can be generated by both classical clustering techniques and CTGAN, which can be combined with LDP. The research's success is evidenced by: 1) the development of multiple methods for generating synthetic user data; 2) the protection and utilization of contextual user information, such as location and comments; and 3) the ability of synthetic data to achieve similar performance without incurring privacy liabilities by using actual user data.

7.2 Potential Future Work

It would be interesting to investigate more details of the POI recommender system problem and the privacy issue in the future. In general, the following three topics will be studied:

- Synthesize user ratings and contextual user information: Previous research focused on only synthesizing the user ratings. Any other contextual information, such as timestamps, the user, or item location, cannot be incorporated into the current rating generation scheme.
- Secure Multi-Party Computation (SMPC): SMPC is a cryptographic technique that allows multiple parties to compute a function while keeping their inputs private jointly. Investigate using SMPC in recommender systems to enable collaborative filtering without revealing users' individual preferences and ratings.
- Evaluation Metrics and Benchmarks: Develop evaluation metrics and benchmarks to assess the privacy-utility trade-offs in recommender systems, enabling comparisons between different privacy-preserving methods and understanding their performance implications.

Contextual User Information Synthesis

Previously, the research has put much of the focus on making CF-based RS models to preserve user privacy. However, one notable limitation of collaborative filtering is its inability to incorporate contextual information effectively. Contextual information, such as time, location, and user preferences, can significantly influence users' choices and enhance the overall quality of recommendations. By not accounting for this context, collaborative filtering can lead to suboptimal recommendations that may fail to capture users' varying preferences across different situations. This limitation can be particularly detrimental in dynamic environments, where users' interests may change rapidly or depend heavily on contextual factors. Furthermore, the lack of contextual information can also exacerbate the cold-start problem, where little or no data is available about new users or items, leading to poor recommendations. In summary, while collaborative filtering has been successful in many scenarios, its inability to incorporate contextual information represents a significant drawback that may hinder its effectiveness in delivering highly relevant and personalized recommendations.

Factorization Machines (FMs) are advantageous in incorporating various auxiliary information. Instead of a rating matrix R , a feature matrix X composed of diverse features and a corresponding target vector Y are required. Figure 7.1 demonstrates an example of the necessary input data. In contrast to a rating matrix R , this feature matrix X can store everything by converting elements such as “movies rated” or “user ID” into binary indicators and combining them. The Y vector contains the final ratings of each user.

$$\hat{y} = w_0 + \sum_{i=1}^n w_i x_i + \sum_{i=1}^n \sum_{j=i+1}^n \langle v_i, v_j \rangle x_i x_j \quad (7.1)$$

$$\langle v_i, v_j \rangle = \sum_{f=1}^k v_{i,f} \cdot v_{j,f} \quad (7.2)$$

Equation 7.1 represents the FM model, where $x \in \mathbb{R}^n$ is a sample vector from the feature matrix X , $w_0 \in \mathbb{R}$ represents the global bias, $w_i \in \mathbb{R}$ denotes the weights

Feature vector \mathbf{x}															Target y							
$\mathbf{x}^{(1)}$	1	0	0	...	1	0	0	0	...	0.3	0.3	0.3	0	...	13	0	0	0	0	...	5	$y^{(1)}$
$\mathbf{x}^{(2)}$	1	0	0	...	0	1	0	0	...	0.3	0.3	0.3	0	...	14	1	0	0	0	...	3	$y^{(2)}$
$\mathbf{x}^{(3)}$	1	0	0	...	0	0	1	0	...	0.3	0.3	0.3	0	...	16	0	1	0	0	...	1	$y^{(2)}$
$\mathbf{x}^{(4)}$	0	1	0	...	0	0	1	0	...	0	0	0.5	0.5	...	5	0	0	0	0	...	4	$y^{(3)}$
$\mathbf{x}^{(5)}$	0	1	0	...	0	0	0	1	...	0	0	0.5	0.5	...	8	0	0	1	0	...	5	$y^{(4)}$
$\mathbf{x}^{(6)}$	0	0	1	...	1	0	0	0	...	0.5	0	0.5	0	...	9	0	0	0	0	...	1	$y^{(5)}$
$\mathbf{x}^{(7)}$	0	0	1	...	0	0	1	0	...	0.5	0	0.5	0	...	12	1	0	0	0	...	5	$y^{(6)}$
A B C ... User				TI NH SW ST ... Movie				TI NH SW ST ... Other Movies rated				Time	TI NH SW ST ... Last Movie rated									

Figure 7.1: An Example of The Input Data Required for FMs [1]

of the variable, and $V \in \mathbb{R}^{n \times k}$ is the feature embedding matrix. The first two terms form the linear regression model, and the last is a matrix factorization model. It is essential to note that this is a two-way FM, meaning that the last interaction term in the equation can be expanded to a higher order, similar to SVM with polynomial kernels. However, the equation's polynomial order is typically set to 2 in practice, and it will be set to 2 in the initial test. Therefore, the details of higher-order FMs are omitted here.

The most crucial aspect of the FM method is its time complexity. The research team reformulated the last term in Equation 7.1, resulting in a time complexity of $O(kn)$. In practice, only the known rating is calculated, so the algorithm's complexity is linear to the number of non-zero features. Due to the speed and efficiency of the algorithm, particularly with high-dimensional sparse inputs, FMs have had a significant impact for an extended period [121, 122, 123].

In the following research, the plan is to explore the potential of using Generative Adversarial Networks (GANs) to generate synthetic input data for Factorization Machines (FMs). GANs have demonstrated exceptional capabilities in generating realistic data samples across various domains, making them a promising candidates for data synthesis in recommender systems. By generating synthetic user-item interaction data, GANs can potentially address some of the common challenges recommender systems faces, such as data sparsity and privacy concerns. In this context, the GAN-generated data will be utilized as input for FMs, which are known for their effectiveness in capturing latent factors and higher-order feature interactions. By combining the power of GANs and FMs, future research aims to create a robust and privacy-preserving recommendation framework that can achieve high-quality recommendations while mitigating the risks of using real user data. Furthermore, this approach may offer additional benefits, such as improved generalization and reduced overfitting, by augmenting the training data with more diverse and realistic synthetic samples.

Secure Multi-Party Computation

Secure multi-party computation (SMC) is a technique for evaluating a function with multiple parties such that each party learns the output value but not each other's

inputs. There are various ways to implement secure MPC with different numbers of parties and security guarantees. Here, we concentrate on systems based on secret sharing.

Secret sharing, introduced by Blakley (1979) and Shamir (1979), is a method of splitting a secret value s into a number of shares s_1, s_2, \dots, s_n that are distributed among the parties (computing nodes). Depending on the type of scheme used, the original value can be reconstructed only by knowing all or a predefined number (threshold t) of these shares. Any group of t or more parties can combine their shares to reconstruct the original value. However, the result of combining fewer than t shares provides no information about the value they represent.

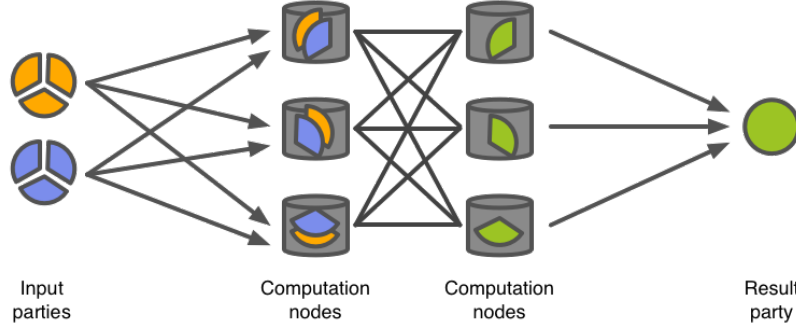


Figure 7.2: An Example of The Secure Multi-party Computation Scheme [2]

As shown in the above Figure 7.2, secure multi-party computation protocols can be used to process secret-shared data. These protocols take secret-shared values as inputs and output a secret-shared result that can be used in further computations. For example, let us have values u and v that are secret-shared and distributed among all the parties so that each computation node C_i gets the shares u_i and v_i . To evaluate $w = u \oplus v$ for some binary function \oplus , the computation nodes engage in a share computing protocol and output w in a shared form (node C_i holds w_i). During the computation, no computation node can recover the original values u or v , nor learn anything about the output value w .

In the following research, in an effort to enhance the privacy and security of the recommender system framework, the plan is to employ Secure Multi-party Computation techniques to eliminate the need for a third party. Previously, the framework necessitated a third party that mediates between users and the central server, responsible for clustering or generating synthetic users. By leveraging the capabilities of SMC, the aim is to develop a more decentralized approach where users can collaboratively compute the desired pre-processed data without disclosing their individual data to a third party. This would not only strengthen privacy protection but also increase the overall efficiency and robustness of the recommender system. By harnessing the power of SMC, the research endeavors to create a paradigm shift in the design and implementation of privacy-preserving recommender systems.

Developing Evaluation Metrics and Benchmarks

One of the challenges in evaluating privacy-preserving recommender systems lies in identifying an appropriate metric that can accurately measure the performance of various algorithms. The difficulty stems from the fact that different POI recommender systems may have distinct characteristics, making it hard to compare them using a single metric. For instance, considering RMSE as a performance indicator may not necessarily guarantee that a model with a lower RMSE value is better than another. This is because the model with a lower RMSE might not preserve as much privacy as the competing model, which could be a crucial aspect of the PPRS.

Moreover, although Differential Privacy is a useful privacy-preserving metric, not all algorithms employ this technique, further complicating the evaluation process. Without a universally accepted privacy metric, it becomes challenging to assess the level of privacy preservation in each model. Additionally, the datasets used by different researchers can also vary, making it even more challenging to perform a fair comparison of the privacy-preserving techniques employed in different recommender systems. This lack of a standardized evaluation metric and the inherent differences in datasets underscore the need for further research in developing comprehensive metrics and benchmarks tailored for privacy-preserving recommender systems.

Recognizing the challenges in evaluating privacy-preserving recommender systems due to the absence of a universal benchmark, the goal is to develop a comprehensive and standardized set of metrics that can be used to assess the performance of various PPRS. This will facilitate a more accurate and fair comparison of different algorithms while considering the recommender systems' diverse characteristics. A multi-faceted evaluation approach could involve considering a combination of metrics, such as accuracy (RMSE, MAE), privacy preservation (DP, k-anonymity, or other privacy measures), and utility (recommendation quality and coverage).

By promoting this unified set of metrics, researchers and practitioners in the field will be better equipped to compare their models, identify potential improvements, and foster a more collaborative environment for developing novel privacy-preserving techniques. Furthermore, considering multiple metrics as a whole will enable stakeholders to strike an optimal balance between privacy, utility, and accuracy, paving the way for more robust and efficient privacy-preserving recommender systems. Ultimately, establishing and promoting a universal benchmark will contribute significantly to advancing state of the art in PPRS and addressing the ongoing challenges related to privacy protection in the era of data-driven personalization.

Bibliography

- [1] S. Rendle, Factorization machines, in: 2010 IEEE International Conference on Data Mining, IEEE, Sydney, Australia, 2010, pp. 995–1000.
- [2] F. P. Schiavo, V. Sassone, L. Nicoletti, A. Margheri, Faas: Federation-as-a-service, arXiv preprint arXiv:1612.03937 (2016).
- [3] Y. Koren, Collaborative filtering with temporal dynamics, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, 2009, pp. 447–456.
- [4] F. Ricci, L. Rokach, B. Shapira, Recommender systems: introduction and challenges, Recommender systems handbook (2015) 1–34.
- [5] R. M. Bell, Y. Koren, Lessons from the NetFlix prize challenge, ACM SIGKDD Explorations Newsletter 9 (2) (2007) 75–79.
- [6] Y. Koren, The bellkor solution to the NetFlix grand prize, Netflix Prize Documentation 81 (2009) (2009) 1–10.
- [7] C. C. Aggarwal, C. C. Aggarwal, Machine learning with shallow neural networks, Neural Networks and Deep Learning: A Textbook (2018) 53–104.
- [8] S. Zhao, I. King, M. R. Lyu, A survey of point-of-interest recommendation in location-based social networks, arXiv preprint arXiv:1607.00647 (2016).
- [9] M. G. Vozalis, K. G. Margaritis, Applying SVD on item-based filtering, in: 5th International Conference on Intelligent Systems Design and Applications (ISDA’05), IEEE, Warsaw, Poland, 2005, pp. 464–469.
- [10] J. Bennett, S. Lanning, et al., The NetFlix Prize, in: Proceedings of KDD Cup and Workshop, Vol. 2007, Citeseer, 2007, p. 35.
- [11] Y. Koren, Factorization meets the neighborhood: A multifaceted collaborative filtering model, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, 2008, pp. 426–434.
- [12] Y. Zheng, Location-based social networks: Users, in: Computing with Spatial Trajectories, Springer, 2011, pp. 243–276.
- [13] J. Bobadilla, F. Ortega, A. Hernando, J. Bernal, A collaborative filtering approach to mitigate the new user cold start problem, Knowledge-Based Systems 26 (2012) 225–238.
- [14] B. M. Marlin, R. S. Zemel, Collaborative prediction and ranking with non-random missing data, in: Proceedings of the Third ACM Conference on Recommender Systems, New York, NY, USA, 2009, pp. 5–12.

- [15] J. Bao, Y. Zheng, D. Wilkie, M. Mokbel, Recommendations in Location-Based Social Networks: A Survey, *GeoInformatica* 19 (2015) 525–565.
- [16] S. Badsha, X. Yi, I. Khalil, D. Liu, S. Nepal, E. Bertino, Privacy preserving location recommendations, in: *International Conference on Web Information Systems Engineering*, Springer, Puschino, Russia, 2017, pp. 502–516.
- [17] C. Chen, Z. Liu, P. Zhao, J. Zhou, X. Li, Privacy preserving point-of-interest recommendation using decentralized matrix factorization, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32, 2018.
- [18] J. Bao, Y. Zheng, D. Wilkie, M. Mokbel, Recommendations in location-based social networks: a survey, *GeoInformatica* 19 (3) (2015) 525–565.
- [19] F. Yan, S. Sundaram, S. Vishwanathan, Y. Qi, Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties, *IEEE Transactions on Knowledge and Data Engineering* 25 (11) (2012) 2483–2493.
- [20] A. Nedic, A. Ozdaglar, Distributed subgradient methods for multi-agent optimization, *IEEE Transactions on Automatic Control* 54 (1) (2009) 48–61.
- [21] Yelp dataset.
URL <https://www.yelp.com/dataset>
- [22] Yelp - company - fast facts (2020).
URL <https://www.yelp-press.com/company/fast-facts/default.aspx>
- [23] Newzoo global mobile market report 2020 — free version.
URL <https://newzoo.com/insights/trend-reports/newzoo-global-mobile-market-report-2020-free-version/>
- [24] X. Luo, M. Zhou, Y. Xia, Q. Zhu, An efficient nonnegative matrix factorization-based approach to collaborative filtering for recommender systems, *IEEE Transactions on Industrial Informatics* 10 (2014) 1273–1284.
- [25] C. Desrosiers, G. Karypis, A Comprehensive Survey of Neighborhood-Based Recommendation Methods, *Recommender Systems Handbook*, Springer, 2011, pp. 107–144.
- [26] A. Nedic, A. Ozdaglar, Distributed sub-gradient methods for multi-agent optimization, *IEEE Transactions on Automatic Control* 54 (2009) 48–61.
- [27] H. Yun, H. Yu, C. Hsieh, S. Vishwanathan, I. Dhillon, Nomad: Nonlocking, stochastic multimachine algorithm for asynchronous and decentralized matrix completion, *arXiv preprint arXiv:1312.0193* (2013).
- [28] C. Chen, Z. Liu, P. Zhao, J. Zhou, X. Li, Privacy preserving point-of-interest recommendation using decentralized matrix factorization, *arXiv preprint arXiv:2003.05610* (2020).

- [29] X. Wang, M. Nguyen, J. Carr, L. Cui, K. Lim, A group preference-based privacy preserving poi recommender system, *ICT Express* (2020).
- [30] F. T. Commission, FTC imposes \$5 billion penalty and sweeping new privacy restrictions on facebook, Press release 24 (2019).
- [31] S. O. of Advocacy, Small business profile (2016).
URL https://www.sba.gov/sites/default/files/advocacy/United_States.pdf
- [32] Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, Factorization meets the neighborhood: a multi-faceted collaborative filtering model.
- [33] A. Mnih, R. R. Salakhutdinov, Probabilistic matrix factorization, *Advances in neural information processing systems* 20 (2007) 1257–1264.
- [34] F. Yan, S. Sundaram, S. Vishwanathan, Y. Qi, Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties, *IEEE Transactions on Knowledge and Data Engineering* 25 (2012) 2483–2493.
- [35] P. G. Campos, F. Díez, I. Cantador, Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols, *User Modeling and User-Adapted Interaction* 24 (1) (2014) 67–119.
- [36] S. M. McNee, J. Riedl, J. A. Konstan, Being accurate is not enough: how accuracy metrics have hurt recommender systems, in: *CHI’06 extended abstracts on Human factors in computing systems*, 2006, pp. 1097–1101.
- [37] Y. Koren, Factor in the neighbors: Scalable and accurate collaborative filtering, *ACM Transactions on Knowledge Discovery from Data (TKDD)* 4 (1) (2010) 1–24.
- [38] S. Funk, Netflix update: Try this at home (2006).
- [39] N. Hug, Surprise: A python library for recommender systems, *Journal of Open Source Software* 5 (2020) 2174. doi:10.21105/joss.02174.
- [40] P. Voigt, A. Von dem Bussche, The eu general data protection regulation (gdpr), *A Practical Guide*, 1st Ed., Cham: Springer International Publishing 10 (3152676) (2017) 10–5555.
- [41] E. Goldman, An introduction to the california consumer privacy act (ccpa), Santa Clara Univ. Legal Studies Research Paper (2020).
- [42] L. Determann, Z. J. Ruan, T. Gao, J. Tam, China’s draft personal information protection law, *Journal of Data Protection & Privacy* 4 (3) (2021) 235–259.
- [43] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, *Computer* 42 (8) (2009) 30–37.

- [44] C. Chen, M. Zhang, Y. Liu, S. Ma, Neural attentional rating regression with review-level explanations, in: Proceedings of the 2018 World Wide Web Conference, Lyon, France, 2018, pp. 1583–1592.
- [45] R. Wei, H. Tian, H. Shen, Improving k-anonymity based privacy preservation for collaborative filtering, *Computers & Electrical Engineering* 67 (2018) 509–519.
- [46] X. Wang, J. Zhang, Y. Wang, Trust-aware privacy-preserving recommender system, in: Proceedings of the 9th EAI International Conference on Mobile Multimedia Communications, 2016, pp. 107–115.
- [47] X. Wang, M. Nguyen, J. Carr, L. Cui, K. Lim, A group preference-based privacy-preserving poi recommender system, *ICT Express* 6 (3) (2020) 204–208.
- [48] J. H. Lau, T. Baldwin, An empirical evaluation of doc2vec with practical insights into document embedding generation, *arXiv preprint arXiv:1607.05368* (2016).
- [49] M. J. Pazzani, D. Billsus, Content-based recommendation systems, in: *The adaptive web*, Springer, 2007, pp. 325–341.
- [50] J. L. Herlocker, J. A. Konstan, L. G. Terveen, J. T. Riedl, Evaluating collaborative filtering recommender systems, *ACM Transactions on Information Systems (TOIS)* 22 (1) (2004) 5–53.
- [51] D. Agarwal, B.-C. Chen, Regression-based latent factor models, in: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, 2009, pp. 19–28.
- [52] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, Bpr: Bayesian personalized ranking from implicit feedback, *arXiv preprint arXiv:1205.2618* (2012).
- [53] H.-J. Xue, X. Dai, J. Zhang, S. Huang, J. Chen, Deep matrix factorization models for recommender systems., in: *IJCAI*, Vol. 17, Melbourne, Australia, 2017, pp. 3203–3209.
- [54] S. Raza, C. Ding, Progress in context-aware recommender systems—an overview, *Computer Science Review* 31 (2019) 84–97.
- [55] J. J. Levandoski, M. Sarwat, A. Eldawy, M. F. Mokbel, Lars: A location-aware recommender system, in: 2012 IEEE 28th international conference on data engineering, IEEE, 2012, pp. 450–461.
- [56] L. A. G. Camacho, S. N. Alves-Souza, Social network data to alleviate cold-start in recommender system: A systematic review, *Information Processing & Management* 54 (4) (2018) 529–544.

- [57] Z. B. Sojahrood, M. Taleai, A poi group recommendation method in location-based social networks based on user influence, *Expert Systems with Applications* 171 (2021) 114593.
- [58] C. Chen, J. Zhou, B. Wu, W. Fang, L. Wang, Y. Qi, X. Zheng, Practical privacy preserving poi recommendation, *ACM Transactions on Intelligent Systems and Technology (TIST)* 11 (5) (2020) 1–20.
- [59] Y. Lin, P. Ren, Z. Chen, Z. Ren, D. Yu, J. Ma, M. d. Rijke, X. Cheng, Meta matrix factorization for federated rating predictions, in: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 981–990.
- [60] R. Shokri, P. Pedarsani, G. Theodorakopoulos, J.-P. Hubaux, Preserving privacy in collaborative filtering through distributed aggregation of offline profiles, in: *Proceedings of the third ACM conference on Recommender systems*, 2009, pp. 157–164.
- [61] H. Shin, S. Kim, J. Shin, X. Xiao, Privacy enhanced matrix factorization for recommendation with local differential privacy, *IEEE Transactions on Knowledge and Data Engineering* 30 (9) (2018) 1770–1782.
- [62] L. Cui, X. Wang, J. Zhang, Vendor-based privacy-preserving poi recommendation network, in: *International Conference on Mobile Multimedia Communications*, Springer, 2021, pp. 477–490.
- [63] H. Kikuchi, H. Kizawa, M. Tada, Privacy-preserving collaborative filtering schemes, in: *2009 International Conference on Availability, Reliability and Security*, IEEE, 2009, pp. 911–916.
- [64] Z. Erkin, M. Beye, T. Veugen, R. Lagendijk, Privacy-preserving content-based recommendations through homomorphic encryption, in: *Information Theory in the Benelux and The 2nd Joint WIC/IEEE Symposium on Information Theory and Signal Processing in the Benelux*, 2012, p. 71.
- [65] S. Badsha, X. Yi, I. Khalil, A practical privacy-preserving recommender system, *Data Science and Engineering* 1 (3) (2016) 161–177.
- [66] L. Ravi, V. Subramaniaswamy, M. Devarajan, K. Ravichandran, S. Arunkumar, V. Indragandhi, V. Vijayakumar, Secrecsy: A secure framework for enhanced privacy-preserving location recommendations in cloud environment, *Wireless Personal Communications* 108 (3) (2019) 1869–1907.
- [67] A. Ng, M. Jordan, Y. Weiss, On spectral clustering: Analysis and an algorithm, *Advances in neural information processing systems* 14 (2001).
- [68] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al., A density-based algorithm for discovering clusters in large spatial databases with noise., in: *kdd*, Vol. 96, 1996, pp. 226–231.

- [69] J. C. Bezdek, R. Ehrlich, W. Full, Fcm: The fuzzy c-means clustering algorithm, *Computers & geosciences* 10 (2-3) (1984) 191–203.
- [70] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P. Kuksa, Natural language processing (almost) from scratch, *Journal of Machine Learning Research* 12 (ARTICLE) (2011) 2493–2537.
- [71] Y. Zhang, Y. Tan, M. Zhang, Y. Liu, T.-S. Chua, S. Ma, Catch the black sheep: unified framework for shilling attack detection based on fraudulent action propagation, in: *The Twenty-Fourth International Joint Conference on Artificial Intelligence*, Buenos Aires, Argentina, 2015.
- [72] J. McAuley, J. Leskovec, Hidden factors and hidden topics: understanding rating dimensions with review text, in: *Proceedings of the 7th ACM International Conference on Recommender Systems*, Hong Kong, China, 2013, pp. 165–172.
- [73] G. Ling, M. R. Lyu, I. King, Ratings meet reviews, a combined approach to recommend, in: *Proceedings of the 8th ACM Conference on Recommender Systems*, Foster City, Silicon Valley, California, USA, 2014, pp. 105–112.
- [74] Y. Zhang, G. Lai, M. Zhang, Y. Zhang, Y. Liu, S. Ma, Explicit factor models for explainable recommendation based on phrase-level sentiment analysis, in: *Proceedings of the 37th international ACM SIGIR Conference on Research & Development in Information Retrieval*, Gold Coast, Queensland, Australia, 2014, pp. 83–92.
- [75] I. Goodfellow, Y. Bengio, A. Courville, Y. Bengio, *Deep Learning*, Vol. 1, MIT press Cambridge, 2016.
- [76] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, *arXiv preprint arXiv:1301.3781* (2013).
- [77] F. M. Harper, J. A. Konstan, The movielens datasets: History and context, *Acm transactions on interactive intelligent systems (tiis)* 5 (4) (2015) 1–19.
- [78] R.-C. Roman, R.-E. Precup, E. M. Petriu, Hybrid data-driven fuzzy active disturbance rejection control for tower crane systems, *European Journal of Control* 58 (2021) 373–387.
- [79] Z. Zhu, Y. Pan, Q. Zhou, C. Lu, Event-triggered adaptive fuzzy control for stochastic nonlinear systems with unmeasured states and unknown backlash-like hysteresis, *IEEE Transactions on Fuzzy Systems* 29 (5) (2020) 1273–1283.
- [80] F. Nogueira, *Bayesian Optimization: Open source constrained global optimization tool for Python* (2014–).
URL <https://github.com/fmfn/BayesianOptimization>
- [81] J. Snoek, H. Larochelle, R. P. Adams, Practical bayesian optimization of machine learning algorithms, *Advances in neural information processing systems* 25 (2012).

- [82] D. H. Park, H. K. Kim, I. Y. Choi, J. K. Kim, A literature review and classification of recommender systems research, *Expert systems with applications* 39 (11) (2012) 10059–10072.
- [83] J. Lian, X. Zhou, F. Zhang, Z. Chen, X. Xie, G. Sun, xdeepfm: Combining explicit and implicit feature interactions for recommender systems, in: *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pp. 1754–1763.
- [84] D. Pramod, Privacy-preserving techniques in recommender systems: state-of-the-art review and future research agenda, *Data Technologies and Applications* n.d. (ahead-of-print) (2022).
- [85] G. D. P. Regulation, General data protection regulation (gdpr), Intersoft Consulting, Accessed in October 24 (1) (2018).
- [86] L. Ross, K. Zhou, China issues new cybersecurity review measures, *Journal of Investment Compliance* 22 (1) (2021) 47–52.
- [87] W. G. Voss, First the gdpr, now the proposed eprivacy regulation, *Journal of Internet Law* 21 (1) (2017) 3–11.
- [88] I. Calzada, Citizens’ data privacy in china: The state of the art of the personal information protection law (pipl), *Smart Cities* 5 (3) (2022) 1129–1150.
- [89] Q. Ling, Y. Xu, W. Yin, Z. Wen, Decentralized low-rank matrix completion, in: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2012, pp. 2925–2928.
- [90] H. Yun, H.-F. Yu, C.-J. Hsieh, S. Vishwanathan, I. Dhillon, Nomad: Non-locking, stochastic multi-machine algorithm for asynchronous and decentralized matrix completion, *arXiv preprint arXiv:1312.0193* (2013).
- [91] Z. Erkin, M. Beye, T. Veugen, R. L. Lagendijk, Privacy enhanced recommender system, in: *Thirty-first symposium on information theory in the Benelux*, 2010, pp. 35–42.
- [92] Z. Erkin, T. Veugen, T. Toft, R. L. Lagendijk, Generating private recommendations efficiently using homomorphic encryption and data packing, *IEEE transactions on information forensics and security* 7 (3) (2012) 1053–1066.
- [93] S. Kim, J. Kim, D. Koo, Y. Kim, H. Yoon, J. Shin, Efficient privacy-preserving matrix factorization via fully homomorphic encryption, in: *Proceedings of the 11th ACM on Asia conference on computer and communications security*, 2016, pp. 617–628.
- [94] A. Berlioz, A. Friedman, M. A. Kaafar, R. Boreli, S. Berkovsky, Applying differential privacy to matrix factorization, in: *Proceedings of the 9th ACM Conference on Recommender Systems*, 2015, pp. 107–114.

- [95] T. Bao, L. Xu, L. Zhu, L. Wang, R. Li, T. Li, Privacy-preserving collaborative filtering algorithm based on local differential privacy, *China Communications* 18 (11) (2021) 42–60.
- [96] R. Ahuja, A. Solanki, A. Nayyar, Movie recommender system using k-means clustering and k-nearest neighbor, in: 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), IEEE, 2019, pp. 263–268.
- [97] R. Logesh, V. Subramaniaswamy, Exploring hybrid recommender systems for personalized travel applications, in: *Cognitive Informatics and Soft Computing: Proceeding of CISC 2017*, Springer, 2019, pp. 535–544.
- [98] Z. Cui, X. Xu, X. Fei, X. Cai, Y. Cao, W. Zhang, J. Chen, Personalized recommendation system based on collaborative filtering for iot scenarios, *IEEE Transactions on Services Computing* 13 (4) (2020) 685–695.
- [99] M. Z. Islam, L. Brankovic, Privacy preserving data mining: A noise addition framework using a novel clustering technique, *Knowledge-Based Systems* 24 (8) (2011) 1214–1223.
- [100] S. Shaham, M. Ding, B. Liu, S. Dang, Z. Lin, J. Li, Privacy preserving location data publishing: A machine learning approach, *IEEE Transactions on Knowledge and Data Engineering* 33 (9) (2020) 3270–3283.
- [101] L. Cui, X. Wang, A cascade framework for privacy-preserving point-of-interest recommender system, *Electronics* 11 (7) (2022) 1153.
- [102] X. He, T.-S. Chua, Neural factorization machines for sparse predictive analytics, in: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Shinjuku, Tokyo, Japan, 2017, pp. 355–364.
- [103] P. Kairouz, K. Bonawitz, D. Ramage, Discrete distribution estimation under local privacy, in: *International Conference on Machine Learning*, PMLR, 2016, pp. 2436–2444.
- [104] S. Wang, L. Huang, P. Wang, H. Deng, H. Xu, W. Yang, Private weighted histogram aggregation in crowdsourcing, in: *Wireless Algorithms, Systems, and Applications: 11th International Conference, WASA 2016, Bozeman, MT, USA, August 8-10, 2016. Proceedings* 11, Springer, 2016, pp. 250–261.
- [105] J. C. Duchi, M. I. Jordan, M. J. Wainwright, Minimax optimal procedures for locally private estimation, *Journal of the American Statistical Association* 113 (521) (2018) 182–201.
- [106] A. Friedman, S. Berkovsky, M. A. Kaafar, A differential privacy framework for matrix factorization recommender systems, *User Modeling and User-Adapted Interaction* 26 (2016) 425–458.

- [107] I. Fiske, R. Chandler, Unmarked: an r package for fitting hierarchical models of wildlife occurrence and abundance, *Journal of statistical software* 43 (2011) 1–23.
- [108] M. Ankerst, M. M. Breunig, H.-P. Kriegel, J. Sander, Optics: Ordering points to identify the clustering structure, *ACM Sigmod record* 28 (2) (1999) 49–60.
- [109] L. Kaufman, P. J. Rousseeuw, Partitioning around medoids (program pam), *Finding groups in data: an introduction to cluster analysis* 344 (1990) 68–125.
- [110] F. Nogueira, et al., Bayesian optimization: Open source constrained global optimization tool for python, URL <https://github.com/fmfn/BayesianOptimization> (2014).
- [111] T. Zhou, The impact of privacy concern on user adoption of location-based services, *Industrial Management & Data Systems* (2011).
- [112] D. Wakabayashi, Google reaches privacy settlement with activist shareholder, *The New York Times* (Nov 2022).
URL <https://www.nytimes.com/2022/11/14/technology/google-privacy-settlement.html>
- [113] Y. Xiao, L. Xiao, X. Lu, H. Zhang, S. Yu, H. V. Poor, Deep-reinforcement-learning-based user profile perturbation for privacy-aware recommendation, *IEEE Internet of Things Journal* 8 (6) (2020) 4560–4568.
- [114] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, A. A. Bharath, Generative adversarial networks: An overview, *IEEE signal processing magazine* 35 (1) (2018) 53–65.
- [115] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial networks, *Communications of the ACM* 63 (11) (2020) 139–144.
- [116] L. Xu, M. Skoularidou, A. Cuesta-Infante, K. Veeramachaneni, Modeling tabular data using conditional gan, in: *Advances in Neural Information Processing Systems*, 2019.
- [117] S. Bourou, A. El Saer, T.-H. Velivassaki, A. Voulkidis, T. Zahariadis, A review of tabular data synthesis using gans on an ids dataset, *Information* 12 (09) (2021) 375.
- [118] D. P. Kingma, M. Welling, et al., An introduction to variational autoencoders, *Foundations and Trends® in Machine Learning* 12 (4) (2019) 307–392.
- [119] T. Koski, J. Noble, *Bayesian networks: an introduction*, John Wiley & Sons, 2011.
- [120] R. Xu, D. Wunsch, Survey of clustering algorithms, *IEEE Transactions on neural networks* 16 (3) (2005) 645–678.

- [121] I. Bayer, X. He, B. Kanagal, S. Rendle, A generic coordinate descent framework for learning from implicit feedback, in: Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 2017, pp. 1341–1350.
- [122] Y. Juan, Y. Zhuang, W.-S. Chin, C.-J. Lin, Field-aware factorization machines for ctr prediction, in: Proceedings of the 10th ACM Conference on Recommender Systems, Boston, Massachusetts, USA, 2016, pp. 43–50.
- [123] R. J. Oentaryo, E.-P. Lim, J.-W. Low, D. Lo, M. Finegold, Predicting response in mobile advertising with hierarchical importance-aware factorization machine, in: Proceedings of the 7th ACM International Conference on Web Search and Data Mining, New York, NY, USA, 2014, pp. 123–132.

Vita

Personal Information

- Name: Longyin Cui
- Place of Birth: Shijiazhuang, Hebei, China

Education

- Ph.D. in Computer Science, University of Kentucky, Lexington, KY, USA, 2017 - Present
- M.S. in Computer Science, Kentucky State University, Frankfort, KY, USA, 2016
- M.A. in English Communication, Muskingum University, New Concord, OH, USA, 2013
- B.S. in Biotechnology, Sichuan Normal University, Chengdu, Sichuan, China, 2011

Research Experience

- GAN-based User Data Synthesizing for POI Recommender Systems, University of Kentucky, Lexington, KY, May 2022 - August 2022
- Developing Recommender System Assistive University Course Selection System, Northeastern Illinois University, Chicago, IL, May 2021 - August 2021
- Preliminary GPU-Accelerated Computing Implementation, Kentucky State University, Frankfort, KY, March 2015 - May 2016
- RNA-Seq Data processing, Kentucky State University, Frankfort, KY, June 2014 - August 2014
- Research in Different Chemical Effects on EAC, Hebei Medical University, Shijiazhuang, China, March 2011 - April 2011

Teaching Experience

- Teaching Assistant, University of Kentucky, Lexington, KY, August 2017 - Present
- Instructor, University of Kentucky, Lexington, KY, December 2020 - January 2021, May 2020 - August 2020

Publications

1. Cui, L., & Wang, X. (2022). A Cascade Framework for Privacy-Preserving Point-of-Interest Recommender System. *Electronics*, 11(7), 1153.
2. Wang, X., Nguyen, M., Carr, J., Cui, L., & Lim, K. (2020). A group preference-based privacy-preserving POI recommender system. *ICT Express*, 6(3), 204-208.
3. Cui, L., Wang, X. & Zhang, J. (2021). Conference Vendor-Based Privacy-Preserving POI Recommendation Network. *International Conference on Mobile Multimedia Communications* (pp. 477-490). Springer, Cham.
4. Wang, X., Cui, L., Bangash, M., Bilal, M., Rosales, L., & Chaudhry, W. (2022). A Machine Learning-based Course Enrollment Recommender System. *CSEDU*, (1), 436-443.
5. Cui, L., Wang, X & Ting, G (In Review). A Generic Data Synthesis Framework for Privacy-Preserving Point-of-Interest Recommender Systems.
6. Cui, L., Zhang, C & Zongming, F (In Review).
7. Cui, L., Wang, X & Ting, G (In Review). A Generic Data Synthesis Framework for Privacy-Preserving Point-of-Interest Recommender Systems.

Awards & Honors

- Outstanding Teaching Assistant, University of Kentucky, Lexington, KY, April 2022
- Best Presenter, Eastern Kentucky University, Lexington, KY, March 2020
- Outstanding Tutor, Kentucky State University, Student Center, Lexington, KY, April 2015