Scalable Privacy-Preserving Recommendation System

by

Rushabh Tamboli
202011059

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of

MASTER OF TECHNOLOGY
in
INFORMATION AND COMMUNICATION TECHNOLOGY
to

DHIRUBHAI AMBANI INSTITUTE OF INFORMATION AND COMMUNICATION TECHNOLOGY

May, 2022

Declaration

I hereby declare that

i) the thesis comprises of my original work towards the degree of Master of Technology in Information and Communication Technology at Dhirubhai Ambani Institute of Information and Communication Technology and has not been submitted elsewhere for a degree,

ii) due acknowledgment has been made in the text to all the reference material used.

R B Tamboli

Rushabh Tamboli

Certificate

This is to certify that the thesis work entitled Scalable Privacy-Preserving Recommendation System has been carried out by Rushabh Tamboli for the degree of Master of Technology in Information and Communication Technology at *Dhirubhai Ambani Institute of Information and Communication Technology* under my/our supervision.

Manik Lal Das.

Dr. Manik Lal Das
Thesis Supervisor

Acknowledgments

I would like to thank my thesis supervisor Dr. Manik Lal Das, for his extraordinary support and expert advice throughout the journey. I am greatly honoured to work with him. It would not have been possible to do this work without his guidance.

It is impossible to extend enough thanks to my family, who gave me the support and encouragement I needed throughout this process.

To my friends, this would have been a much more difficult feat without you. Thank you all for your unwavering support.

Contents

Abstract

Recommendation systems, which are used by e-business applications, play an essential role in our daily life. For example, companies like Amazon, Flipkart, YouTube use recommendation systems to generate customized recommendations from the user's personal information. A personalized recommendation system's primary purpose is to give users helpful suggestions on different things. The service provider has to access different kinds of user data in order to make suggestions, such as prior product buying history, demographic, and identifying details. Users, on the other hand, are wary about disclosing personal information since it may be readily abused by hostile third parties. To address this challenge, we propose a privacy-preserving recommendation system using homomorphic encryption, which is able to provide recommendations without revealing user rating information. Also, in this system, a service provider can use another service provider's user rating database to improve the generated recommendation while protecting user's privacy. The implementation of the proposed system on a publicly available database shows that the system is practical and achieves higher recommendation accuracy.

Keywords: Recommendation System, Privacy, Homomorphic Encryption

List of Principal Symbols and Acronyms

E Encryption

EP_j Encrypted predicted rating on item i_j

$f_{i,j}$ Flag given by user u_i on item i_j

g Group generator

i_j jth item

m Total number of items

n Total number of users

$Perm$ Permutation function

PK Public Key of ElGamal Server

PR_j Predicted rating on item i_j

R_j Average rating of item j

$r_{i,j}$ Rating value given by user u_i on item i_j

$sim(i_j, i_k)$ Similarity between the items i_j and i_k

u_i ith user

List of Tables

List of Figures

CHAPTER 1
Introduction

Leading e-business companies like Amazon, Flipkart, and eBay sell items covering not only books, films, DVDs, and music CDs, but they also sell different products in the varieties such as groceries, baby products, food, clothes, electronic devices such as smartphones, laptops, desktops, TVs, tablets, set-top boxes, beauty products, jewelry, personal care and health care products, scientific and industrial tools, kitchen utensils, games, software, automotive items, et cetera. Having a wide range of content that is offered to the customer, a need for an automated recommendation system arises.

Nowadays, recommendation systems to recommend these products are increasingly gaining popularity and are widely used by these companies. The widespread use of this recommendation system allows customers to get various customized recommendations for the products. While recommendation systems have become highly influential in generating recommendations, they also generate considerable revenue for that businesses. A recent study shows that 75% content people watch on Netflix and 35% of Amazon sales come from the recommended products [6]. Another study done by the research firm found that in the shopping sessions, sales from the product recommendations made 11.5% of the earnings (from the higher value of items or more volume) [7].

To give more accurate and personalized recommendations, the e-business organizations have to accumulate different kinds of information such as customer's age, gender, occupation, level of education, wealth, geographical information, et cetera, and customer's historical data, for example, purchased items, browsing habits, clickstream, et cetera. This information can determine customers precisely, deducing their personal information such as age, gender, health status, habits, and financial standing. Recommendation systems also make recommendations from the data supplied by customers on other issues or by analyzing similar customers.

On the other hand, customers might not want to disclose such kind of information. They may be abused by the data used for other purposes, such as selling

such information to another party or leaked by a hacker, which may lead to a loss to the customer [8] [10]. Hence, recommendation systems can render immediate risks to the customers because of privacy violations. Providing recommendation services without preserving privacy brings risks to these e-business organizations. Customer's information can be monetized, and opponents can also examine customers' choices and behaviours to alter their business tactics.

Figure 1.1: Recommendation System [13]

Potential privacy-preserving approaches for this challenge, on the other hand, should not come at the price of e-business companies' interests. The capacity to offer reliable recommendations, in particular, may give a competitive edge that must be safeguarded.

There is an inherent trade-off between customers' privacy and getting personalized recommendations. Research into anonymization and privacy-preserving algorithms aims to enhance privacy while retaining the usability of the recommendation systems. Therefore, our most vital task is to assist customers purchase items without jeopardizing their privacy.

1.1 Related Works

To solve issues of the privacy of the user data and security of the recommendation system, some solutions are proposed. A few of them are summarized below.

1.1.1 Random perturbation

Polat and Du [12] proposed a system for collaborative filtering. They suggested a method in which the user makes minor changes to their private rating data before transmitting it to the service provider, using the Randomized Perturbation technique. This technique is helpful to hide the correct private data about the user by adding randomness to the original rating data to prevent the merchant

from learning the private user data. Although the data of the product ratings are disturbed, the authors argue that their system should still allow them to generate successful recommendation.

1.1.2 Homomorphic Encryption

Homomorphic encryption can be used to preserve the privacy of the user data while computing the recommendation for the user. This is demonstrated by Badsha et al. in [3]. It employs the ElGamal encryption system [4] to secure users' data. This technique uses individual customers' public keys and private keys to encrypt and decrypt the customer ratings to preserve the customers' privacy.

1.2 Contribution

Our thesis proposes a privacy-preserving technique for Collaborative Filtering using public-key cryptography. Our approach also provides a way to use ratings from different service providers' databases without compromising the privacy of the user ratings.
In summary, our proposed system works as follows:

- All users who participate in the system encrypt their rating and transmit it to the Merchant's server.
- Server then computes average ratings and similarity table from the ratings.
- From the similarity table and average ratings, the recommendations are generated for the user.
- If a new user is added into the system, then the Merchant's server uses previous computation to recompute average ratings and similarity table and generates the recommendation for the users.
- When a user modifies his ratings, in that case, the Merchant's server also uses previous computation to recompute average ratings and similarity table. Using the previous computation, Servers can achieve higher performance while generating recommendation for the users.

The following chapter 2 describes preliminary studies related to the recommendation system and ElGamal cryptosystem. Chapter 3 represents the proposed algorithm. Chapter 4 includes the security analysis of the system. Performance

analysis is included in the chapter 5. Finally, the conclusion is presented in chapter 6.

CHAPTER 2
Preliminaries

The recommendation system works with two main entities: users who use the recommendation service to submit ratings and get recommendations and the products rated by users. Numerical ratings, which represent the users' interests in products, are typically used as input to the recommendation system. The recommendation system's output can be recommendations or predictions.

Let the set of all n users be $U = u_1, u_2, u_3, \ldots, u_n$ in a recommendation system and the set of items be $I = i_1, i_2, i_3, \ldots, i_m$, where m is the total number of items. Let a rating matrix be denoted by R where user u_i provides his rating on item i_j is denoted by $r_{i,j}$. Because it is impossible to give a rating to all things by all users in a system, the rating matrix is usually sparse due to missing data. $r_{i,k} = 0$ represents that the rating is not given. Therefore, the ultimate objective of a recommendation system is predicting a user u_i's rating on item i_k, which the ratings are not provided by the user yet.

2.1 Similarity Calculation

To provide recommendations, the similarities / correlation between the item pairs need to be calculated. In recommendation generation, similarity metrics like Pearson Correlation Coefficient and Cosine Similarity [1] are typically employed to find the closest neighbour. For two items i_j and i_k, the similarity value is calculated as

$$sim(i_j, i_k) = \frac{\sum_{i=1}^{n} (r_{i,j} \cdot r_{i,k})}{\sqrt{r_{1,j}^2 + \cdots + r_{n,j}^2}\sqrt{r_{1,k}^2 + \cdots + r_{n,k}^2}} \tag{2.1}$$

where i_k and i_j denote two different products. The user u_i's ratings on those two items are represented by $r_{i,j}$ and $r_{i,k}$, while the total number of the users is represented by n.

2.2 Recommendation Generation

The Collaborative Filtering algorithm computes recommendations based on item ratings. The Collaborative Filtering algorithm is comprised of two steps: first, based on the ratings of the items, the similarity is calculated, and then the new rating values are predicted for the items. Assuming we have similarity values between the item pairs, the item-based Collaborative Filtering [11] calculates the prediction for user u_i for item i_k as

$$P_{i,k} = \frac{R_k \cdot \sum_{j=1}^{m} sim(i_k, i_j) + \sum_{j=1}^{m} (r_{i,j} - R_j) \cdot sim(i_k, i_j)}{\sum_{j=1}^{m} sim(i_k, i_j)} \tag{2.2}$$

Where $P_{i,k}$ represents the user u_i's prediction for item i_k. $r_{i,j}$ denotes the ratings supplied by user u_i on item i_j. $sim(i_j, i_k)$ is the similarity between items i_j and i_k. The average rating of an item is denoted by R_k and R_j, is calculated by dividing the total sum of ratings by the total number of people who have rated the given item.

2.3 Homomorphic Encryption

The ElGamal encryption system is multiplicatively and additively homomorphic [14]. This indicates that across the ciphertexts $E(T_1)$ and $E(T_2)$, there are two operations whose outcomes correspond to the new ciphertext whose decryption yields the multiplication and sum of the plaintexts T_1 and T_2.

$$\begin{aligned} E(T_1) \cdot E(T_2) &= E(T_1 + T_2) \\ E(T_1)^{T_2} &= E(T_1 \cdot T_2) \end{aligned} \tag{2.3}$$

The ElGamal cryptosystem based on Diffie-Hellman key exchange [14], is a probabilistic public-key encryption scheme and also has a homomorphic property. Any cyclic group G can be used to define the ElGamal encryption algorithm. Its safety is determined by the hardness of a specific problem in group G involving the computation of discrete logarithms [9]. The key generation, encryption, and decryption algorithms are the three algorithms that make up the ElGamal encryption system.

2.3.1 Key Generation

The private key is $x \in Z_q$, and the public key is $h = g^x$, where $x = 1, 2, \ldots, q-1$ and G is a cyclic group of order q.

2.3.2 Encryption

To encrypt the message m,

$$C = (C_1, C_2) = (g^r, m \cdot h^r) \tag{2.4}$$

where r is a random number. C is a ciphertext.

2.3.3 Decryption

The private key x is used to decrypt the ciphertext. The decryption process is as follows,

$$\frac{C_2}{(C_1)^x} = \frac{m \cdot h^r}{(g^r)^x} = \frac{m \cdot h^r}{h^r} = m \tag{2.5}$$

2.3.4 Homomorphic Property

Provided two ciphertexts,

$$\begin{aligned} C_1 &= (C_{11}, C_{12}) = (g^{r_1}, m_1 \cdot h^{r_1}) \\ C_2 &= (C_{21}, C_{22}) = (g^{r_2}, m_2 \cdot h^{r_2}) \end{aligned} \tag{2.6}$$

The following shows the process to compute multiplication of two ciphertexts:

$$\begin{aligned} (C_{11}, C_{12}) \cdot (C_{21}, C_{22}) &= (C_{11} \cdot C_{21}, C_{12} \cdot C_{22}) \\ &= (g^{r_1} g^{r_2}, (m_1 \cdot h^{r_1})(m_2 \cdot h^{r_2})) \\ E(m_1 \cdot m_2) &= (g^{r_1 + r_2}, (m_1 \cdot m_2) h^{r_1 + r_2}) \end{aligned} \tag{2.7}$$

The result ciphertext of the above calculation is the encryption of the multiplication of two plaintexts m_1 and m_2. In [14] it has been shown that distinguishing amongst the ciphertexts of any two given messages is computationally infeasible, i.e. ElGamal is semantically secure if the decisional Diffie-Hellman problem is intractable.

CHAPTER 3
Proposed Scheme

In our proposed scheme, firstly, two merchant servers calculate the required information. Then to use their information, we use the homomorphic characteristic of the ElGamal encryption system to calculate the results as if they are calculated by merging the item rating information while protecting the rating information of the users.

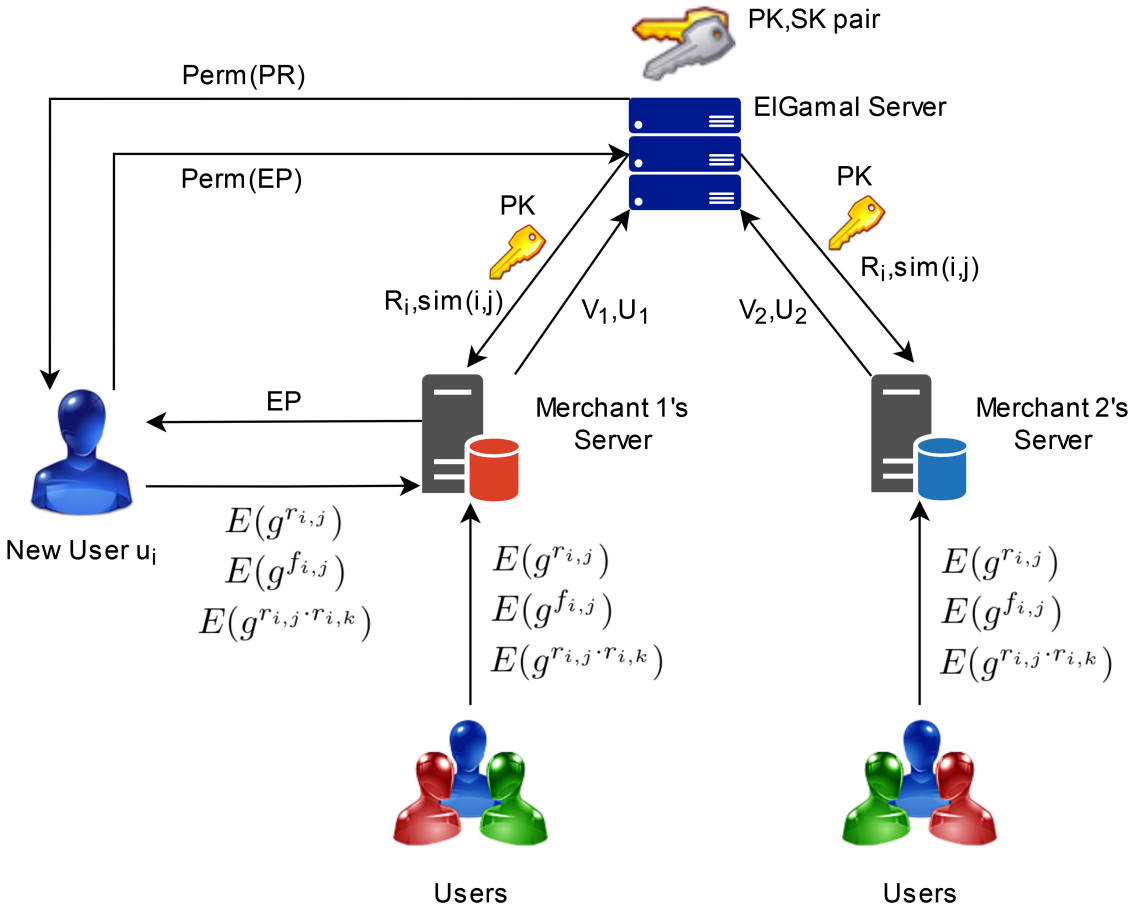

Figure 3.1: General architecture of the proposed system

Our scheme is comprised of five phases. First, we describe rating encryption.

Table 3.1: Rating Matrix

	Item i_1	Item i_2	$\cdots$	Item i_j	$\cdots$	Item i_m
User u_1	$r_{1,1}, f_{1,1}$	$r_{1,2}, f_{1,2}$	$\cdots$	$r_{1,j}, f_{1,j}$	$\cdots$	$r_{1,m}, f_{1,m}$
User u_2	$r_{2,1}, f_{2,1}$	$r_{2,2}, f_{2,2}$	$\cdots$	$r_{2,j}, f_{2,j}$	$\cdots$	$r_{2,m}, f_{2,m}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
User u_i	$r_{i,1}, f_{i,1}$	$r_{i,2}, f_{i,2}$	$\cdots$	$r_{i,j}, f_{i,j}$	$\cdots$	$r_{i,m}, f_{i,m}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
User u_n	$r_{n,1}, f_{n,1}$	$r_{n,2}, f_{n,2}$	$\cdots$	$r_{n,j}, f_{n,j}$	$\cdots$	$r_{n,m}, f_{n,m}$

Then average ratings and similarity computation. After that, we represent the recommendation generation phase. The next phase describes the computation required for the addition of the new user. The last phase represents how ratings can be modified to generate new recommendations.

In this system, there are three parties: ElGamal server, Merchant servers and users. For the whole duration of the algorithm, the ElGamal server's private key for ElGamal encryption is kept private from other parties, and only the public key is provided to all other parties. This public key is denoted as PK in the following phases. All the parties are assumed to be semi-honest parties. We assume that a safe channel has been established between the users and all of the servers for the secure exchange of any secret messages. We also assume that the different Merchant servers have similar items sets to compute recommendations.

3.1 Ratings Encryption

In this phase, the users transmit their encrypted ratings to their corresponding merchant server.

Let $u_i = u_1, u_2, \ldots, u_n$ denotes the users who have participated with the merchant server and encrypted his ratings $r_{i,j}$ and flags $f_{i,j}$ (whether a user has given a rating then the flag is 1; otherwise, it is 0) using ElGamal server's public key PK. The complete process is given below.

Step 1

All users encrypt their flags as $E(g^{f_{i,j}})$ and ratings as $E(g^{r_{i,j}})$, for all items.

Step 2

All users encrypt their ratings as $E(g^{r_{i,j} \cdot r_{i,k}})$ for all $1 \leq j \leq k \leq m$.

Step 3

Encrypted data is transmitted to the Merchant's server.

3.2 Average Computation

The Merchant's server calculates averages of items homomorphically, and the ElGamal server decrypts the results using the private key. The complete process is given below.

Step 1

Merchant's server computes the sum of ratings of all products by using this formula,

$$S(j) = \prod_{i=1}^{n} E(g^{r_{i,j}}) \tag{3.1}$$

where $j = 1, 2, \ldots, m$.

Step 2

Merchant's server computes the sum of flags of all products by using this formula,

$$F(j) = \prod_{i=1}^{n} E(g^{f_{i,j}}) \tag{3.2}$$

where $j = 1, 2, \ldots, m$.

Step 3

The vector $V_1(j) = [(S(j), F(j))]$, $j = 1, \ldots, m$, is transmitted to the ElGamal server.

Step 4

The same process is done by the other Merchant server. Other Merchant's server sends V_2 to the ElGamal server.

Step 5

The ElGamal server multiplies the corresponding pairs for all items to get the sum of the ratings and the sum of the flags over all the users participating regardless of the merchant's server they are participating from.

$$V(i) = V_1(i) \cdot V_2(i) \tag{3.3}$$

where $i = 1, 2, \ldots, m$.

Step 6

The ElGamal server decrypts the vector and solves the discrete logarithm to obtain the values as $S'(j) = \sum_{i=1}^{n} r_{i,j}$, $F'(j) = \sum_{i=1}^{n} f_{i,j}$, for $1 \leq j \leq m$.

Step 7

The ElGamal server computes the average ratings as below and it is sent to the Merchant servers.

$$R_j = \frac{S'(j)}{F'(j)} \tag{3.4}$$

where $j = 1, 2, \ldots, m$.
Note: The sum of ratings and flags are not large, so calculation of discrete logarithm is not hard.

3.3 Similarity Computation

The Merchant's server computes a similarity table of items homomorphically, and the ElGamal server decrypts the results using the private key. The complete process is given below.

Step 1

The Merchant's server computes the sum of the multiplication of ratings of all products by using this formula,

$$M(j,k) = \prod_{i=1}^{n} E(g^{r_{i,j} \cdot r_{i,k}}) \tag{3.5}$$

where $1 \leq j \leq k \leq m$.

Step 2

The vector $U_1(j,k) = [M(j,k)]$, $1 \leq j \leq k \leq m$, is transmitted to the ElGamal server.

Step 3

The same process is done by the other Merchant server. Another Merchant's server sends U_2 to the ElGamal server.

Step 4

The ElGamal server multiplies the corresponding pairs for all items to get the sum of the multiplication of the ratings over all the users participating regardless of the merchant's server they are participating from.

$$U(j,k) = U_1(j,k) \cdot U_2(j,k) \tag{3.6}$$

where $1 \leq j \leq k \leq m$.

Step 5

The ElGamal server decrypts the vector and solves the discrete logarithm to obtain the values as $M'(j,k) = \sum_{i=1}^{n} r_{i,j} \cdot r_{i,k}$, $1 \leq j \leq k \leq m$.

Step 6

The ElGamal server computes the similarity values as,

$$sim(i_j, i_k) = \frac{M'(j,k)}{\sqrt{M'(j,j)}\sqrt{M'(k,k)}} \tag{3.7}$$

where $1 \leq j \leq k \leq m$, and it is sent to the Merchant servers.
Note: The sum of the multiplication of ratings is not large, so calculation of discrete logarithm is not hard.

3.4 Precomputation for Recommendation Generation

In this phase the Merchant servers precomputes,

$$\begin{gathered} E(g^{R_j})_{j=1,\ldots,m}, \\ E(g^{\sum_{i=1}^{m} sim(i_j,i_k)})_{1\le j\le k\le m}, \\ E(g^{R_k \sum_{i=1}^{m} sim(i_j,i_k)})_{1\le j\le k\le m} \end{gathered} \tag{3.8}$$

These values are used to decrease the time required for the next recommendation generation process and also can be reused until the user ratings are changed.

3.5 Recommendation Generation

To get the recommendation for the items, the user encrypts his ratings of the items with PK and transmits them to the Merchant's server. The Merchant's server uses the precomputation and the user's rating values to generate predicted ratings for the items. The complete process is given below.

Step 1

The user u_i computes $r_i = E(g^{r_{i,j}})$, for $j = 1, \ldots, m$ and transmits it to the Merchant's server.

Step 2

Merchant's server uses equation (2.2) to compute encrypted numerator and denominator homomorphically of the predicted rating as,

$$\begin{aligned} Num_k =& E(g^{R_k \cdot \sum_{j=1}^{m} sim(i_k,i_j)}) \\ & \cdot \left(\prod_{j=1}^{m} \left(\frac{E(g^{r_{i,j}})}{E(g^{R_j})} \right)^{sim(i_k,i_j)} \right) \\ Den_k =& E(g^{\sum_{j=1}^{m} sim(i_k,i_j)}) \end{aligned} \tag{3.9}$$

where $k = 1, 2, \ldots, m$.

Step 3

The computed vectors of the encrypted numerator and denominator are sent to the user.

$$EP = [(Num_j, Den_j)]_{j=1,\ldots,m} \tag{3.10}$$

Step 4

The encrypted numerator and denominator are sent to the ElGamal server. The pairs are permuted before being sent to the ElGamal server, and the mapping is kept secret by the user.

Step 5

The ElGamal server decrypts the numerator and denominator and computes discrete logarithm for them. If its result is $[(Num'_j, Den'_j)]$ for $j = 1, \ldots, m$, then the predicted rating is calculated as,

$$PR_j = \frac{Num'_j}{Den'_j} \tag{3.11}$$

where $k = 1, 2, \ldots, m$. The PR vector is sent to the user.

Step 6

Having the original mapping between the permuted predicted ratings and the items, the user can obtain the recommendation generated by the Merchant's server while preserving privacy.
Note: The numerator and denominator of the predicted ratings are not large, so calculation of discrete logarithm is not hard.

3.6 Adding User

In this phase, a new user is added to the system. He computes his encrypted ratings and sends them to the Merchant's server. The complete process is given below.

Step 1

A new user u_i encrypts his flags as $E(g^{f_{i,j}})$ and ratings as $E(g^{r_{i,j}})$ for all items $1 \leq j \leq m$.

Step 2

A new user u_i encrypts his ratings as $E(g^{r_{i,j} \cdot r_{i,k}})$ for all $1 \leq j \leq k \leq m$.

Step 3

Encrypted data is transmitted to the Merchant's server.

Step 4

For the average ratings computation, the Merchant's server reuses the previously calculated values. If the previously calculated values are $S(j)$ and $F(j)$, for $j = 1, \ldots, m$, then,

$$\begin{aligned} S(j) &= S(j) \times E(g^{r_{i,j}}) \\ F(j) &= F(j) \times E(g^{f_{i,j}}) \end{aligned} \tag{3.12}$$

where $j = 1, 2, \ldots, m$. The new vector $V_1(j) = [(S(j), F(j))]$, $j = 1, \ldots, m$, is transmitted to the ElGamal server, and Step 4 in the above Average Ratings computation continues to get the average ratings.

Step 5

For the similarity values computation, the Merchant's server reuses the previously calculated values. If the previously calculated value is $M(j,k)$, for all $1 \leq j \leq k \leq m$, then,

$$M(j,k) = M(j,k) \times E(g^{r_{i,j} \cdot r_{i,k}}) \tag{3.13}$$

where $1 \leq j \leq k \leq m$. The new vector $U_1(j,k) = [M(j,k)]$, $1 \leq j \leq k \leq m$, is transmitted to the ElGamal server, and Step 3 in the above Similarity Table computation continues to get the similarity values.

Step 6

The Precomputation for Recommendation Generation and Recommendation Generation steps remain the same.

3.7 Update Ratings

In this phase, an existing user modifies his ratings and sends them to the Merchant's server. The complete process is given below.

Step 1

An existing user u_i encrypts his new ratings as $E(g^{r'_{i,j}})$ and new flags as $E(g^{f'_{i,j}})$ for $1 \leq j \leq m$.

Step 2

An existing user u_i encrypts his new ratings as $E(g^{r'_{i,j} \cdot r'_{i,k}})$ for all $1 \leq j \leq k \leq m$.

Step 3

Encrypted data is transmitted to the Merchant's server.

Step 4

For the average ratings computation, the Merchant's server reuses the previously calculated values $S(j)$ and $F(j)$, and computes,

$$\begin{aligned} S(j) &= \frac{S(j) \times E(g^{r'_{i,j}})}{E(g^{r_{i,j}})} \\ F(j) &= \frac{F(j) \times E(g^{f'_{i,j}})}{E(g^{f_{i,j}})} \end{aligned} \tag{3.14}$$

where $j = 1, \ldots, m$. The new vector $V_1(j) = [(S(j), F(j))]$, $j = 1, \ldots, m$, is transmitted to the ElGamal server, and Step 4 in the above Average Ratings computation continues to get the average ratings.

Step 5

For the similarity values computation, the Merchant's server reuses the previously calculated value $M(j,k)$ and computes,

$$M(j,k) = \frac{M(j,k) \times E(g^{r'_{i,j} \cdot r'_{i,k}})}{E(g^{r_{i,j} \cdot r_{i,k}})} \tag{3.15}$$

where $1 \leq j \leq k \leq m$. The new vector $U_1(j,k) = [M(j,k)]$, $1 \leq j \leq k \leq m$, is transmitted to the ElGamal server, and Step 3 in the above Similarity Table Computation continues to get the similarity values.

Step 6

The Precomputation for Recommendation Generation and Recommendation Generation steps remain the same.

CHAPTER 4
Security Analysis

Lemma 1:

ElGamal encryption algorithm is indistinguishable under a chosen-plaintext attack (IND-CPA).

Using this lemma, we have an indistinguishability experiment for ElGamal encryption as follows,

ElGamal Encryption Indistinguishability Experiment:

Let $Exp^{CPA}_{E,\mathcal{A}}(\kappa)$ an indistinguishability experiment for CPA, where κ is the security parameter for the ElGamal cryptosystem. It is defined as:

- An Adversary $\mathcal{A}$ is given N, κ and Encryption Oracle $\mathcal{O}$.
- An Adversary $\mathcal{A}$ chooses two plaintexts i and j in $1 \leq i, j \leq N$.
- Let $t \leftarrow \{i, j\}$ (that is, either i or j is chosen using the uniform distribution), and then the ElGamal encryption algorithm is executed.
- An Adversary $\mathcal{A}$ queries the encryption oracle $\mathcal{O}$, the ciphertexts resulting from an ElGamal protocol execution up to polynomial times. The input to the encryption oracle $\mathcal{O}$ is an integer. The outputs are ciphertexts resulting from the E (ElGamal encryption) algorithm.
- An Adversary $\mathcal{A}$ outputs t' by guessing the values of t.
- If $t = t'$, the output of $Exp^{CPA}_{E,\mathcal{A}}(\kappa)$ is 1 and 0 if $t \neq t'$.

According to the lemma, the following result is true.

$$Pr[Exp^{CPA}_{E,\mathcal{A}}(\kappa) = 1] \leq \frac{1}{2} + negl(\kappa) \tag{4.1}$$

PPRS Indistinguishability Experiment:

In this experiment, the Adversary $\mathcal{A}$ chooses the challenge plaintexts from R, yet the rating themselves is not known to the Adversary $\mathcal{A}$ because of the characteristics of the PPRS.

In our experiment, The plaintexts are the user ratings, $r_{i,p}$ for $1 \leq p \leq M$ and $1 \leq i \leq N$. In the recommendation system, the user ratings are by nature encrypted, and hence, Merchant servers can not see $r_{i,p}$ for any user u_i.

Thus, the two ratings $r_{i,p}$ of user u_i and $r_{j,q}$ of user u_j are chosen by the Adversary $\mathcal{A}$.

An Adversary can learn ciphertexts by querying rating values $r \in \{Z_{0,5}\}$ to an encryption oracle, denoted by $\mathcal{O}$, up to polynomial times. $Z_{0,5}$ is a set of integers between 0 and 5.

$Exp^{CPA}_{PPRS,\mathcal{A}}(\kappa)$ be an indistinguishability experiment under the chosen-plaintext attack, where the security parameter is κ for the ElGamal encryption system. It is defined as:

- An Adversary $\mathcal{A}$ is given M, N, κ and Encryption Oracle $\mathcal{O}$.
- An Adversary $\mathcal{A}$ chooses the two rating values $r_{i,p}$ and $r_{j,q}$ from R ($1 \leq i, j \leq N$ and $1 \leq p, q \leq M$).
- Let $t \leftarrow \{i, j\}$ (i.e., either $r_{i,p}$ or $r_{j,q}$ is chosen using the uniform distribution), and then the PPRS algorithm is executed.
- An Adversary $\mathcal{A}$ queries the encryption oracle $\mathcal{O}$, the ciphertexts resulting from a PPRS protocol execution up to polynomial times. Here, the input to the encryption oracle $\mathcal{O}$, i.e. $r_{i,p} \in \{Z_{0,5}\}$. The outputs are ciphertexts resulting from the $PPRS$ algorithm.
- An Adversary $\mathcal{A}$ outputs t' by guessing the values of t.
- If $t = t'$, the output of $Exp^{CPA}_{PPRS,\mathcal{A}}(\kappa)$ is 1 and 0 if $t \neq t'$.

Assume that an unknown negligible function which is smaller than the multiplicative inverse of any polynomial functions with respect to the security parameter κ, is denoted by $negl(\kappa)$.

The $PPRS$ protocol's CPA security is defined by the below definition.

Theorem 1:

In the semi-trusted model, PPRS protocol is CPA secure if and only if the below equation holds.

$$Pr[Exp^{CPA}_{PPRS,\mathcal{A}}(\kappa) = 1] \leq \frac{1}{2} + negl(\kappa) \tag{4.2}$$

Proof by contradiction:

Assume that, PPRS protocol is CPA insecure.
Therefore, we have,

$$Pr[Exp^{CPA}_{PPRS,\mathcal{A}}(\kappa) = 1] > \frac{1}{2} \tag{4.3}$$

This means that while running the PPRS protocol, an Adversary $\mathcal{A}$ can distinguish which plaintext is encrypted by the Encryption Oracle $\mathcal{O}$ with a probability greater than $\frac{1}{2}$.

In the experiment $Exp^{CPA}_{E,\mathcal{A}}(\kappa)$ and experiment $Exp^{CPA}_{PPRS,\mathcal{A}}(\kappa)$, the ciphertexts generated by the Encryption Oracle $\mathcal{O}$ are using the same ElGamal encryption.

This means that the probability of the $Exp^{CPA}_{E,\mathcal{A}}(\kappa)$ is the same as $Exp^{CPA}_{PPRS,\mathcal{A}}(\kappa)$. So,

$$Pr[Exp^{CPA}_{PPRS,\mathcal{A}}(\kappa) = 1] = Pr[Exp^{CPA}_{E,\mathcal{A}}(\kappa) = 1] \tag{4.4}$$

Putting equation (4.4) in equation (4.3), we get,

$$Pr[Exp^{CPA}_{E,\mathcal{A}}(\kappa) = 1] > \frac{1}{2} \tag{4.5}$$

The result (4.5) shows that the experiment using the ElGamal encryption system is CPA-insecure according to our assumption in the equation (4.3).

This leads to the contradiction with our Lemma 1 as shown by equation (4.1) that ElGamal encryption is secure.

So, our assumption that PPRS is CPA insecure is wrong. This means that,

$$Pr[Exp^{CPA}_{PPRS,\mathcal{A}}(\kappa) = 1] \leq \frac{1}{2} + negl(\kappa) \tag{4.6}$$

Hence, the PPRS is CPA secure. This concludes the proof. ■

CHAPTER 5
Experimental Result

5.1 Theoretical Analysis

Table 5.1 represents the costs to execute average computation, similarity calculation and recommendation generation phases by the users and servers. To represent the complexity measurements, we denote m, n, e and M for the total number of the items, the total number of the users, exponentiation and multiplication, respectively. In Table 5.1 and Table 5.2 only one recommendation is generated from the ratings given by all the n users. Table 5.3 shows the complexity analysis when another n users are added to the system. In Table 5.1 and Table 5.3 we include all servers' computation and communication complexity in the servers column.

5.2 Performance Analysis

To perform the experiments, we use Python 3.8.10 and PyCryptodome library [2] for the ElGamal algorithm key generation, encryption and decryption methods. The hardware platform has OS as Windows 10, 64-bit with Intel Core i7 2.6 GHz CPU and 16 GB memory. We have assessed our proposed algorithm using a public database by GroupLens [5]. We first assess the method's viability by examining

Table 5.1: Computation and communication cost of the Proposed model

	Computation Cost		Communication Cost	
	User	Servers	User	Servers
Initialization	$O(m^2(e+M))$	$O(nm^2M)$	$O(m^2)$	$O(nm^2)$
Average		$O(m(e+M))$		$O(m)$
Similarity		$O(m^2(e+M))$		$O(m^2)$
Recommendation	$O(m(e+M))$	$O(m^2(e+M))$	$O(m)$	$O(m)$
Total	$O(m^2(e+M))$	$O(nm^2M)$	$O(m^2)$	$O(nm^2)$

Table 5.2: Computation and communication cost of the Badsha et al.'s model

	Computation Cost		Communication Cost	
	User	Servers	User	Servers
Initialization		$O(nM)$		
Average	$O(m(e+M))$	$O(nm(e+M))$	$O(m)$	$O(nm)$
Similarity	$O(m^2(e+M))$	$O(nm^2(e+M))$	$O(m^2)$	$O(nm^2)$
Recommendation	$O(m(e+M))$	$O(m^2(e+M))$	$O(m)$	$O(m)$
Total	$O(m^2(e+M))$	$O(nm^2(e+M))$	$O(m^2)$	$O(nm^2)$

Table 5.3: Computation cost after adding n new users

	Total Computation Cost	
	User	Servers
Badsha et al.	$O(nm^2(e+M))$	$O(n^2m^2(e+M))$
Proposed Algorithm	$O(m^2(e+M))$	$O(nm^2(e+M))$

communication and computation cost. Then we analyze our protocol with the existing solution [3] in-terms of efficiency. The computation cost and communication cost are calculated in relation with the number of items and users participating. User ratings ranges from 0 to 5. When a user u_p did not give the rating to the item i_q, the corresponding rating value $r_{p,q}$ is 0. In the experiments, we implemented our proposed methods with 256-bit security, that is κ is set as 256. In the experiment, floating point values are multiplied by an integer 100 to manage the homomorphic characteristics of the ElGamal cryptosystem. The discrete log calculations are not included in the measurements. In Table 5.1 we only need to do the initialization once.

5.3 Computational and Communication Costs

The implementation of our proposed algorithm can compute one exponentiation (e) in 7.7×10^{-6} seconds and one multiplication (M) in 2.8×10^{-6} seconds. Table 5.4 shows the computational time elapsed to perform the execution of both methods. The measurements shown in Table 5.4 are executed with 100 users and 100 items. The experiment includes all phases running once.

The performance of the proposed technique is cost-effective and efficient interms of both communication and computation, as shown in the table. Figure 5.1 shows the computation time elapsed to compute average and similarity calculations. From this figure, it can be determined that having Merchant's server do all

Table 5.4: Computation and communication cost comparison for N = 100 and M = 100

	Proposed Algorithm		Badsha et al.'s Algorithm	
	Computation (s)	Communication (B)	Computation (s)	Communication (B)
Initialization	2.930	105122808	0.0004	180000
Average & Similarity	5.768	2142072	132.099	231122816
Recommendation	0.994	167440	1.259	170224
Total	9.692	107432320	133.359	231473040

Table 5.5: Computation and communication cost comparison for N = 100, M = 25 and adding 10 new users

	Proposed Algorithm		Badsha et al.'s Algorithm	
	Computation (s)	Communication (B)	Computation (s)	Communication (B)
Initialization	1.993	9556016	0.003	2079000
Average & Similarity	3.108	2062272	84.370	196141536
Recommendation	0.944	463760	1.499	470096
Total	6.045	12082048	85.872	198690632

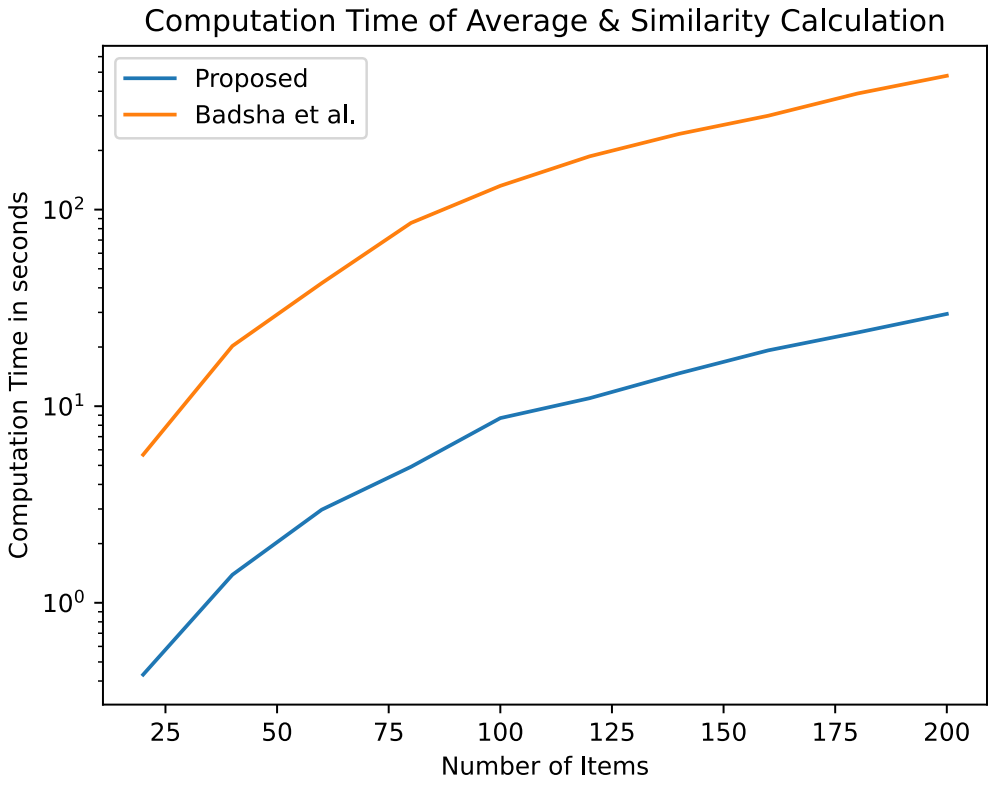

Figure 5.1: Computation Time of Average & Similarity Calculation

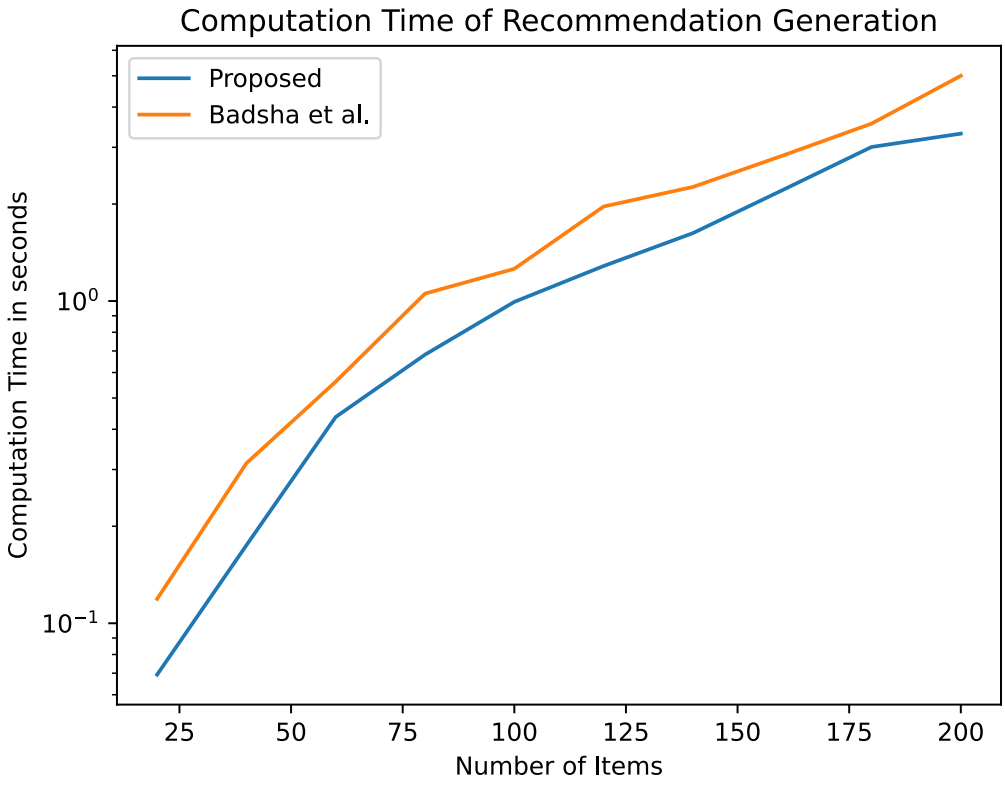

Figure 5.2: Computation Time of Recommendation Generation

the computation alone makes the system efficient. The result demonstrates that our proposed system takes 8.7 seconds to compute steps up to the similarity table calculation for 100 users and 200 items, while the system in [3] needs 132 seconds. This is a significant improvement over the previous system.

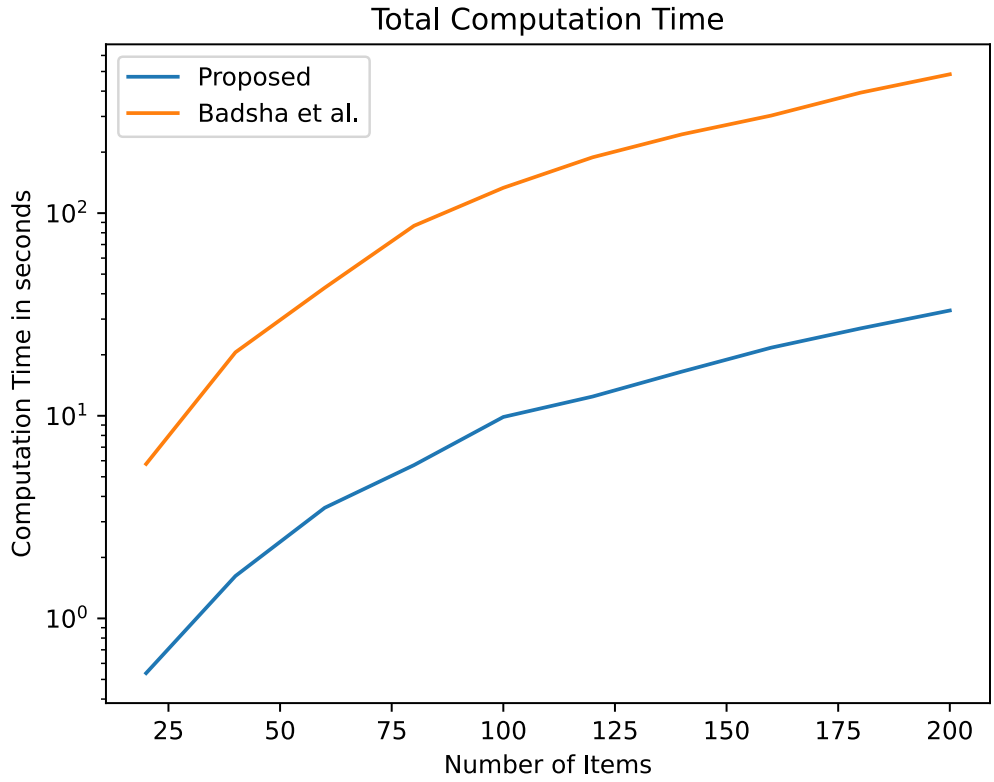

Figure 5.3: Total Computation Time

Table 5.5 shows the performance analysis of the systems when there are 100 users and 25 items, and then one new user is added to the system and requests recommendations. This process is done up to 10 times. So, there are 110 users after the execution of the experiment. We can see from these analyses that the average and similarity calculation takes longer than the other calculations since it involves a significant number of computations. In our proposed method, the previous computations help to achieve better performance when calculating average and similarity tables, which makes our system scalable in terms of computational costs.

Figure 5.4 shows the total computational time required to add 10 users and generate recommendations for them. From the figure, it is clear that the proposed method requires 6.0 seconds, while [3] requires 85.8 seconds to complete the execution. Therefore, from Figure 5.4, we can observe that our proposed system

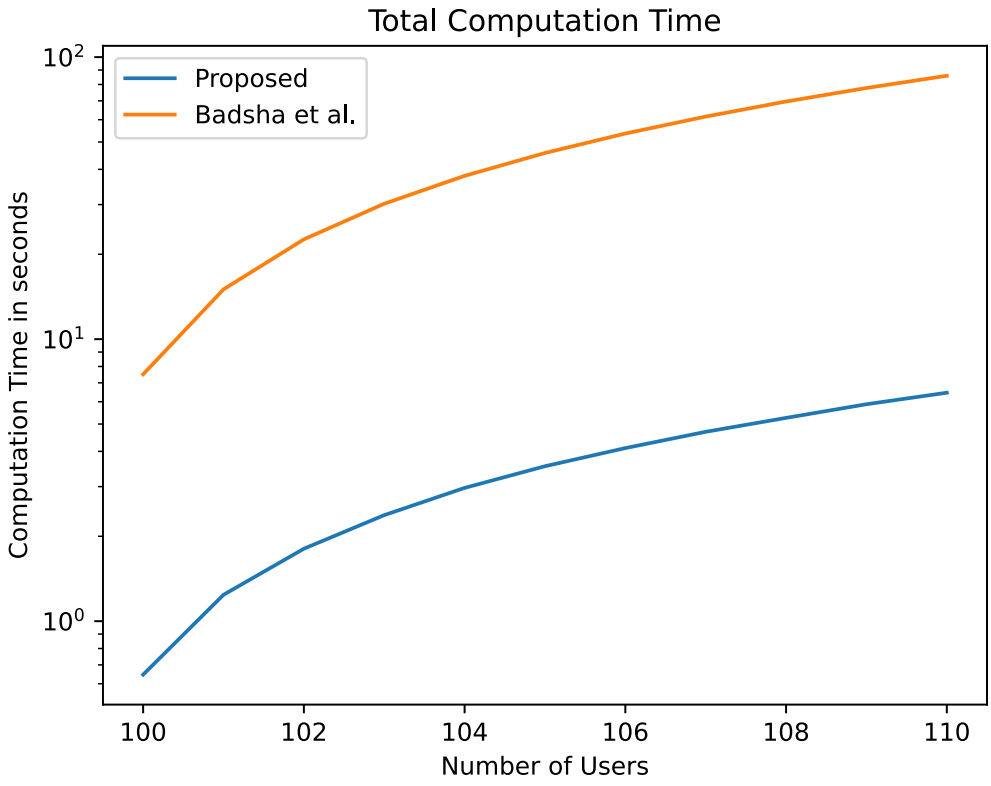

Figure 5.4: Total Computation Time After Adding 10 Users

outperforms [3].

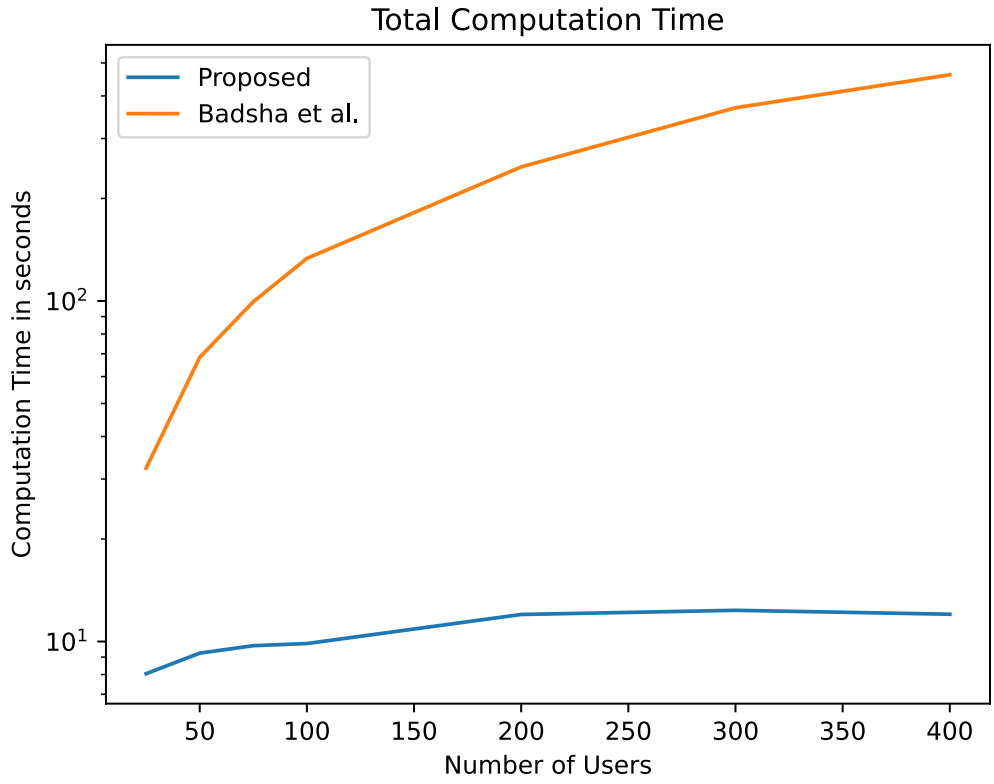

Figure 5.5: Total Computation Time

Figure 5.5 shows the total computational time required to perform the whole process of our proposed algorithm with 100 items and different number of users.

5.4 Recommendation Accuracy

There is no accuracy loss in the proposed method during the recommendation generation upto two decimal places. The experiments were carried out using both our proposed method and without using any cryptography algorithms and compared for the result.

CHAPTER 6
Conclusion and Future Work

This thesis work presents a privacy-preserving recommendation system, which provides privacy to the user ratings from any service provider or other users. Also, in this approach, service providers can use other service provider's database to increase the recommendation accuracy without disclosing the real rating data.

The experimental results demonstrate that our proposed method is feasible and capable of providing recommendations while maintaining privacy. It is also more cost-efficient than the system proposed by Badsha et al [3]. Our approach provides scalability to the privacy-preserving recommendation system as service providers and participating users do not need to recompute the sum of the ratings, sum of the flags and sum of the multiplication of the ratings of all users when a user is added to the system. This leads to a reduction in the time required to generate a recommendation for the users.

Furthermore, there is no loss of accuracy in terms of recommendations using the proposed method. In the future, we intend to extend our system for scalability while adding new items in the system.

References

[1] Collaborative filtering. https://en.wikipedia.org/wiki/Collaborative_filtering#Memory-based.

[2] Python package of cryptographic primitives. https://github.com/Legrandin/pycryptodome.

[3] S. Badsha, X. Yi, and I. Khalil. A practical privacy-preserving recommender system. *Data Science and Engineering*, 1(3):161–177, 2016.

[4] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469–472, 1985.

[5] F. M. Harper and J. A. Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

[6] S. N. Ian MacKenzie, Chris Meyer. How retailers can keep up with consumers. https://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers. 2021.

[7] H. J. The power of personalized product recommendations. https://www.intelliverse.com/blog/the-power-of-personalized-product-recommendations/. 2021.

[8] A. J. Jeckmans, M. Beye, Z. Erkin, P. Hartel, R. L. Lagendijk, and Q. Tang. Privacy in recommender systems. In *Social media retrieval*, pages 263–281. Springer, 2013.

[9] Ç. K. Koç, F. Özdemir, and Z. Ö. Özger. *Partially Homomorphic Encryption*. Springer, 2021.

[10] S. K. Lam, D. Frankowski, J. Riedl, et al. Do you trust your recommendations? an exploration of security and privacy issues in recommender systems. In *International conference on emerging trends in information and communication security*, pages 14–29. Springer, 2006.

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

[12] H. Polat and W. Du. Privacy-preserving collaborative filtering. *International journal of electronic commerce*, 9(4):9–35, 2005.

[13] C. Wang, Y. Zheng, J. Jiang, and K. Ren. Toward privacy-preserving personalized recommendation services. *Engineering*, 4(1):21–28, 2018.

[14] X. Yi, R. Paulet, and E. Bertino. Homomorphic encryption. In *Homomorphic encryption and applications*, pages 27–46. Springer, 2014.