This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2015.2426677, IEEE Journal of Selected Topics in Signal Processing

A Belief Propagation Approach to Privacy-Preserving Item-Based Collaborative Filtering

Jun Zou and Faramarz Fekri, *Senior Member, IEEE*

***Abstract*—Collaborative Filtering (CF) is the most popular recommendation algorithm, which exploits the collected historic user ratings to predict unknown ratings. However, traditional recommender systems run at the central servers, and thus users have to disclose their personal rating data to other parties. This raises the privacy issue, as user ratings can be used to reveal sensitive personal information. In this paper, we propose a semi-distributed Belief Propagation (BP) approach to privacy-preserving item-based CF recommender systems. Firstly, we formulate the item similarity computation as a probabilistic inference problem on the factor graph, which can be efficiently solved by applying the BP algorithm. To avoid disclosing user ratings to the server or other user peers, we then introduce a semi-distributed architecture for the BP algorithm. We further propose a cascaded BP scheme to address the practical issue that only a subset of users participate in BP during one time slot. We analyze the privacy of the semi-distributed BP from a information-theoretic perspective. We also propose a method that reduces the computational complexity at the user side. Through experiments on the MovieLens dataset, we show that the proposed algorithm achieves superior accuracy.**

***Index Terms*—Collaborative filtering, recommender systems, privacy, belief propagation, factor graph.**

I. Introduction

THE thriving of the Internet and online services has overwhelmed users with an explosive amount of product information. It is too exhausting for users to go through the complete list of thousands of items, e.g., books and movies, to find items interest them. Recommender systems have been widely used in e-commerce websites, such as Amazon.com and Netflix.com, to suggest to users items that they might like [1]. Good recommendation services increase user satisfaction and boost business. Since users have different tastes, it necessitates personalized recommendation services that meet an individual's preferences.

Collaborative Filtering (CF) is the most popular recommendation algorithm [2], where users express opinions on items by rating them, and the CF algorithm exploits the collected historic user ratings to predict ratings on unseen items for an individual user, and recommends to the user the items with the highest predicted ratings. The CF recommendation can be divided into user-based and item-based methods. The user-based CF method recommends to an active user new items favorably rated by other users with similar tastes to the active user [3]. The item-based CF method on the other hand analyzes the similarity between items using the aggregated user ratings, and recommends to an active user new items that are similar to the items he liked in the past [4]. Closely related to the item-based CF is the content-based recommendation method, which also recommends similar items. However, the content-based method evaluates item similarity based on the explicit features that describe the items, and hence it is limited by the content analysis techniques, as it is difficult to automatically extract features from multimedia data [2]. The item-based CF recommendation is immune to such problems, an important reason for the popularity of the CF recommendation.

In traditional recommender systems, the recommendation algorithm is run at the central server of the commercial websites or other third party providers, and thus the users have to disclose their personal information, such as preferences, age and gender, to the server in order to receive satisfactory recommendation services. This raises the privacy issue, as users simply have no control over how their personal data will be disseminated and used [5]. It is not uncommon that websites sell to other parties such data, which are valuable for targeted advertising. As users become more concerned about their online privacy, they are less willing to directly release their personal information. Since detailed user profiles are difficult to obtain, most recommender systems employ the CF recommendation approach which only relies on the user ratings. Although rating data do not directly tell personal details, it is still possible to infer user demographics, such as age and gender, from their ratings [6], and even uncover user identities and reveal sensitive personal information with access to other databases [7]. While personalized recommendations at e-commerce websites have been very successful [8], users are increasingly worried about online privacy [9]. Privacy-preserving CF recommender systems are thus in urgent need. The challenge stems from the conflict between accuracy and privacy. That is, to provide recommendations that better match the user's tastes, the system needs to know more about the user.

Most privacy-preserving recommender systems are developed either by obfuscating user data with random noise [10] or fake data [11] at the cost of recommendation accuracy, or by encrypting user data using cryptography techniques [12] which require careful key management. In this paper, we hope to build a recommender system with intrinsic privacy-preserving properties. To achieve this goal, it is evident that any communications between the server and users or between user peers should avoid exposing user ratings. An interesting work in [13] utilized concordance measure for computing user

Copyright (c) 2014 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

J. Zou and F. Fekri are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, 30332 USA (e-mail: {junzou, fekri}@ece.gatech.edu).

This material is based upon work supported by the National Science Foundation under Grant No. IIS-1115199.

Manuscript received September 29, 2014; revised February 19, 2015; accepted April 20, 2015.

1932-4553 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2015.2426677, IEEE Journal of Selected Topics in Signal Processing

similarity required by the user-based CF algorithm, where computation is conducted between users without exposure of their ratings to each other, but unfortunately, to collaboratively generate recommendation, users need to reveal ratings to other users. We notice that user-based CF relies on direct collaboration among users, i.e., a user needs to know what other people like to find out what he might like, whereas item-based CF exploits the consistency in an individual user's taste, i.e., if a user likes an item, he might also like other items similar to it. We consider item-based CF a better choice for privacy-preserving recommender systems. The accuracy of the item-based CF system depends on the measure of item similarity, which has to be estimated based on ratings on items, and further, subject to the privacy constraint.

In this paper, we introduce a semi-distributed Belief Propagation (BP) approach to privacy-preserving item-based CF recommender systems. Firstly, we formulate the item similarity computation, a key step of item-based CF, as a probabilistic inference problem on a proper factor graph, which can then be efficiently solved using BP. Then we develop a semi-distributed architecture for BP, where probabilistic messages on item similarity are exchanged between the server and users, without disclosing user ratings to the server or other peer users. We further propose a cascaded BP scheme to address the practical issue that only a subset of users are available for BP during one time slot. Each user locally generates rating predictions by averaging his own ratings on items weighted according to their similarities to other unseen items. Hence, the proposed item-based CF algorithm preserves user privacy in both item similarity computation and rating prediction. We analyze the achieved privacy of the semi-distributed BP from a information-theoretic perspective. We also propose a method that reduces the computational complexity of BP at the user side. Through experiments on the MovieLens dataset, we show that our algorithm achieves superior accuracy.

The rest of the paper is organized as follows. In Sec. II, we introduce the preliminaries. In Sec. III, we present the modelling and computation of item similarity on factor graphs as a key step of item-based CF. In Sec. IV, we describe the semi-distributed privacy-preserving architecture as well as the cascaded BP scheme. In Sec. V, we present the information-theoretic privacy analysis. In Sec. VI, we analyze and reduce the computational complexity. In Sec. VII, we evaluate the recommendation performance of the proposed algorithm on the MovieLens dataset, and compare it against other well-known item-based CF algorithms. In Sec. VIII, we review the related works on privacy-preserving information processing techniques and applications of BP to recommender systems. In Sec. IX, we conclude this paper [1].

II. Preliminaries

A. Collaborative Filtering

Assuming there are M users and N items in a recommender system, let $\mathbb{U} = \{1, \ldots, M\}$ represent the set of all users, and $\mathbb{I} = \{1, \ldots, N\}$ represent the set of all items. A user u expresses his opinion on item i in the form of rating r_{ui} which takes values from a finite discrete set $\Gamma = \{r | 1 \leq r \leq T, r = 1, 2, \ldots\}$, e.g, $\Gamma = \{1, 2, 3, 4, 5\}$. We arrange the collection of all ratings in an incomplete $M \times N$ matrix $\mathbb{R}$, with r_{ui} at the intersection of u-th row and i-th column. The entries of unknown ratings are unfilled. Let I_u denote the subset of items rated by user u, and U_i denote the subset of users who have rated item i. The task of a recommender system is to predict the ratings for an active user u on the subset of unseen items $\mathbb{I} \backslash I_u$. Throughout this paper, we focus on the item-based CF recommendation algorithm [4].

To predict the rating r_{ui} for user u on an unseen item i, the algorithm sorts the items in I_u according to their similarity to item i in descending order, and finds a subset of top K most similar items, denoted by $\mathcal{N}_{ui}$, where $|\mathcal{N}_{ui}| = K$. We refer to $\mathcal{N}_{ui}$ as the neighbourhood of item i from user u's perspective, and K as the neighborhood size. Then r_{ui} is predicted by

$$\hat{r}_{ui} = \frac{\sum_{j \in \mathcal{N}_{ui}} s_{ij} \times r_{uj}}{\sum_{j \in \mathcal{N}_{ui}} |s_{ij}|}, \tag{1}$$

where s_{ij} is the similarity between items i and j. The accuracy of the algorithm depends on the measure of item similarity s_{ij}, which is computed based on the observed ratings on items. The user-based CF algorithm uses a similar formula to (1), but based on ratings from other users.

Several well-known methods for item similarity computation include Cosine Similarity (CS), Pearson Correlation Similarity (PCS), and Adjusted Cosine Similarity (ACS) [4]. Yet, all of those mentioned methods directly operate on ratings from different users, which requires users to disclose ratings to the server or other users. In this paper, we will introduce a semi-distributed BP algorithm for item similarity computation with intrinsic privacy-preserving property.

B. Factor Graphs and Belief Propagation

A factor graph is a bipartite graph that expresses the factorization structure of a global function of many variables, where variable nodes and factor nodes represent variables and local functions, respectively, and an edge connects a variable node to a factor node if and only if the variable is an argument of the local function represented by the factor node [15].

The sum-product BP algorithm is a message-passing algorithm that operates on the factor graph, in which messages are exchanged between the factor nodes and variable nodes. It essentially exploits the factorization structure to efficiently compute marginal functions from the global function [15], [16].

III. Modelling Item Similarity on Factor Graphs

A. Probabilistic Problem Formulation

We model the similarity s_{ij} between items i and j as a discrete random variable that takes values from a predefined set $\mathcal{S}$. The total number of possible values is $L = |\mathcal{S}|$. Let $\mathbb{S}_i = \{s_{ij} : 1 \leq j \leq N, j \neq i\}$ be the set of item similarities between item i and other items. We denote by $P(\mathbb{S}_i | \mathbb{R})$ the joint posterior probability distribution of $\mathbb{S}_i$. To obtain s_{ij},

[1] Part of this work was previously published in [14].

1932-4553 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2015.2426677, IEEE Journal of Selected Topics in Signal Processing

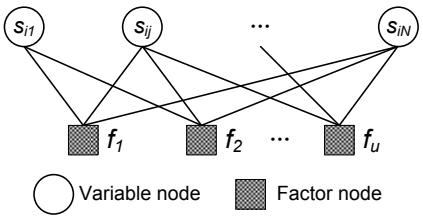


Fig. 1: The factor graph $\mathcal{G}_i$.

we need the marginal posterior probability distribution of s_{ij}, which can be derived by

$$P\left(s_{ij}|\mathbb{R}\right) = \sum_{s_{i1}\in\mathcal{S}} \cdots \sum_{s_{i(j-1)}\in\mathcal{S}} \sum_{s_{i(j+1)}\in\mathcal{S}} \cdots \sum_{s_{iN}\in\mathcal{S}} P\left(\mathbb{S}_i|\mathbb{R}\right). \tag{2}$$

For notational convenience, we rewrite (2) for the sum over all variables in $\mathbb{S}_i$ except s_{ij} as

$$P\left(s_{ij}|\mathbb{R}\right) = \sum_{\mathbb{S}_i \backslash s_{ij}} P\left(\mathbb{S}_i|\mathbb{R}\right). \tag{3}$$

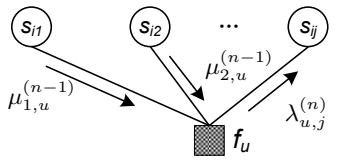
However, direct computation using (3) is not appealing, because it must be performed by a central server, and thus users are required to disclose their ratings to the server. Moreover, (3) incurs an exponential complexity of $\mathcal{O}\big(L^N\big)$. We instead resort to the factor graph to express the factorization of $P\left(\mathbb{S}_i|\mathbb{R}\right)$, and apply the efficient sum-product BP algorithm to infer the marginal posterior probability distributions. More importantly, as will be described in Sec. IV, the BP algorithm can be carried out in a semi-distributed manner, so that the part of computation that requires knowledge of user ratings can be locally performed by the user, eliminating the need to disclosing user ratings.

B. Modelling with Factor Graphs

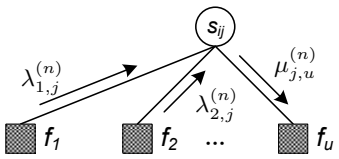
We first find a proper factorization for $P\left(\mathbb{S}_i|\mathbb{R}\right)$. For each user $u \in U_i$, we denote $\mathbb{S}_{ui} = \{s_{ij} : j \in I_u \backslash i\}$ as the set of item similarities between item i and other items user u has rated, and use a local function $f_u(\mathbb{S}_{ui})$ to model the dependencies among variables in $\mathbb{S}_{ui}$ from user u's perspective. Hence, $P\left(\mathbb{S}_i|\mathbb{R}\right)$ factorizes into local functions as follows

$$P\left(\mathbb{S}_i|\mathbb{R}\right) = \frac{1}{Z} \prod_{u\in U_i} f_u(\mathbb{S}_{ui}), \tag{4}$$

where Z is a normalization constant. We construct a factor graph $\mathcal{G}_i$ for the factorization in (4) as illustrated in Fig. 1, where there are $|\mathbb{S}_i|$ variable nodes and $|U_i|$ factor nodes. Each $s_{ij} \in \mathbb{S}_i$ is represented by variable node j in $\mathcal{G}_i$, and each local function $f_u(\mathbb{S}_{ui})$ is represented by factor node u. The subset of variable nodes for $\mathbb{S}_{ui}$ are connected to factor node u via edges. Let U_{ij} denote the set of common users of items i and j, $U_{ij} = U_i \cap U_j$. Hence, variable node j is connected to $|U_{ij}|$ factor nodes in $\mathcal{G}_i$. Essentially, if a user u has rated item i and other items in $I_u \backslash i$, then this user has a belief on the similarity s_{ij}, $\forall j \in I_u \backslash i$, from his perspective. The factor graph allows users' beliefs be exchanged and aggregated following the principle of sum-product message passing. To solve for all item similarities $\{\mathbb{S}_i : 1 \le i \le N\}$, a total of N such factor graphs need to be constructed.



(a) The λ-message $\lambda_{u,j}^{(n)}(s_{ij})$.



(b) The μ-message $\mu_{j,u}^{(n)}$.

Fig. 3: Illustration of message passing at iteration n.

The local function $f_u(\mathbb{S}_{ui})$ determines how user u estimates item similarity based on his own ratings. It should be properly designed with regard to the eventual goal to predict ratings using (1). For a user u who has rated item i, we assume for now rating r_{ui} is unknown, and let $\hat{I}_u = I_u \backslash i$. Given a configuration of item similarities in $\mathbb{S}_{ui}$, user u predicts r_{ui} as

$$\hat{r}_{ui}(\mathbb{S}_{ui}) = \frac{\sum_{j\in\hat{I}_u} s_{ij} \times r_{uj}}{\sum_{j\in\hat{I}_u} |s_{ij}|}. \tag{5}$$

Note that (5) has a similar form to (1). Then user u checks $\hat{r}_{ui}(\mathbb{S}_{ui})$ against the actual rating r_{ui} using the following factor node function

$$f_u\left(\mathbb{S}_{ui}\right) = \frac{1}{Z_u} \exp\left\{-\frac{1}{\sigma^2}\left(\hat{r}_{ui}(\mathbb{S}_{ui}) - r_{ui}\right)^2\right\}, \tag{6}$$

where Z_u is a normalization constant, and σ is a designing parameter that controls the sensitivity of $f_u\left(\mathbb{S}_{ui}\right)$ to the discrepancy between $\hat{r}_{ui}(\mathbb{S}_{ui})$ and r_{ui}. We note that $f_u\left(\mathbb{S}_{ui}\right)$ decreases with increasing discrepancy.

C. BP for Similarity Computation

We describe the sum-product BP algorithm to infer the marginal posterior distribution $P\left(s_{ij}|\mathbb{R}\right)$, $\forall s_{ij} \in \mathbb{S}_i$, on the factor graph $\mathcal{G}_i$, without worrying about privacy. Later in Sec. IV-A, we will introduce the semi-distributed implementation of BP for privacy, yet with no impacts on the computed similarities. Since the constructed factor graph has loops, we apply the "loopy" BP algorithm that iteratively exchanges messages between factor nodes and variable nodes along the edges until convergence [15]. As in the sum-product principle, there are two types of messages passed on $\mathcal{G}_i$: (i) The λ-message $\lambda_{u,j}(s_{ij})$ sent from a factor node u to a variable node j, and (ii) the μ-message $\mu_{j,u}(s_{ij})$ sent from a variable node j to a factor node u.

We illustrate the message passing in Fig. 3. In each iteration, each node (factor node or variable node) in the factor graph generates and sends messages to the neighbor nodes connected to it, based on incoming messages. In iteration n, factor node u generates the λ-message $\lambda_{u,j}^{(n)}(s_{ij})$ sent to variable node j by computing the product of local factor function $f_u(\mathbf{s}_{ui})$ with all μ-messages received in the previous iteration from neighbor

1932-4553 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2015.2426677, IEEE Journal of Selected Topics in Signal Processing

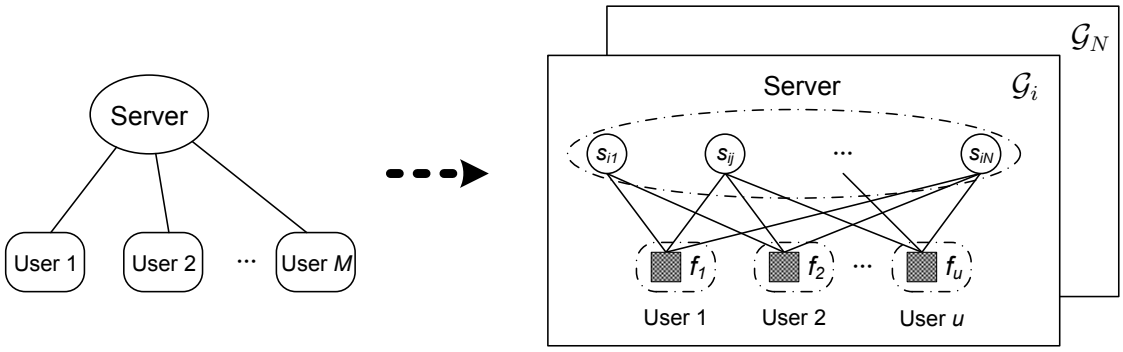


Fig. 2: Architecture of semi-distributed BP.

variable nodes of factor node u, excluding the message from the recipient variable node j, and sums out all variables except s_{ij} as follows

$$\lambda_{u,j}^{(n)}(s_{ij}) \propto \sum_{\mathbb{S}_{ui}\setminus s_{ij}} f_u(\mathbb{S}_{ui}) \prod_{h\in\hat{I}_u\setminus j} \mu_{h,u}^{(n-1)}(s_{ih}). \quad (7)$$

The λ-message $\lambda_{u,j}^{(n)}(s_{ij})$ is a list of the beliefs on the similarity $s_{ij}=s, \forall s\in\mathcal{S}$, perceived from user u's perspective, given the current collective knowledge of the similarity between other items in $\hat{I}_u\setminus j$ and item i.

Variable node j generates the μ-message $\mu_{j,u}^{(n)}(s_{ij})$ sent to factor node u as the product of all incoming λ-messages received in the current iteration from all factor nodes connected to variable node j, excluding the one from the recipient factor node u as follows

$$\mu_{j,u}^{(n)}(s_{ij}) \propto \prod_{f\in F_j\setminus u} \lambda_{f,j}^{(n)}(s_{ij}), \quad (8)$$

where F_j denotes the set of factor nodes connected to variable node j. Here in graph $\mathcal{G}_i$, $F_j=\{u: u\in U_{ij}\}$. The μ-message $\mu_{j,u}^{(n)}(s_{ij})$ is a list of beliefs on the similarity $s_{ij}=s, \forall s\in\mathcal{S}$, which is generated by aggregating beliefs from other users in $U_j\setminus u$ on the similarity s_{ij}.

After convergence, the marginal posterior distribution $P(s_{ij}|\mathbb{R})$ is computed at variable node j as product of all λ-messages received from neighbor factor nodes connected to variable node j as

$$P(s_{ij}|\mathbb{R}) = \frac{1}{Z_{ij}} \prod_{f\in F_j} \lambda_{f,j}^{(n)}(s_{ij}), \quad (9)$$

where Z_{ij} is a normalization constant. Finally, based on the marginal posterior distribution $P(s_{ij}|\mathbb{R})$, the item similarity s_{ij} can be estimated in various ways. We consider using the minimum mean squared error criterion, for which the optimal estimated s_{ij} is given by the expectation

$$\hat{s}_{ij} = \sum_{s\in\mathcal{S}} s \times P(s_{ij}=s|\mathbb{R}). \quad (10)$$

We summarize the BP algorithm for computing $\mathbb{S}_i$ on factor graph $\mathcal{G}_i$ in Alg. 1.

Algorithm 1 BP on $\mathcal{G}_i$ for computing item similarity $\mathbb{S}_i$.

- Initialize all messages as $\lambda_{u,j}^{(0)}(s_{ij}=s)=\frac{1}{L}$ and $\mu_{j,u}^{(0)}(s_{ij}=s)=\frac{1}{L}, \forall s\in\mathcal{S}$, and set iteration $n=1$.
- Iterative message passing until convergence.
  (a) Update all λ-messages using (7);
  (b) Update all μ-messages using (8);
  (c) $n=n+1$. If not convergent, repeat (a) and (b).
- Compute marginal posterior probability distributions of $s_{ij}\in\mathbb{S}_i$ using (9).
- Compute item similarity $s_{ij}\in\mathbb{S}_i$ using (10).

IV. Semi-Distributed Privacy-Preserving CF

Our goal is to build a privacy-preserving recommender system. Specifically, we would like to preserve user privacy for both item similarity computation and recommendation generation. In the following, we introduce the semi-distributed architecture of the proposed BP algorithm for item similarity computation without exposing user ratings, and the user-side recommendation generation as well.

A. Semi-Distributed BP

From Sec. III-C, we know that the messages exchanged between factor nodes and variable nodes are probabilistic statements on item similarity, and such messages do not directly reveal user ratings. However, the user rating data are used to compute $f_u(\mathbb{S}_{ui})$ for generating λ-messages using (7). Hence, the key to preserve user privacy is to compute λ-messages at the user side. The μ-messages can be generated at a central server by performing multiplication operations on received λ-messages. This leads to the semi-distributed implementation of the BP algorithm. As we will see next, the computation follows the BP algorithm described in Sec. III-C, and thus the semi-distributed BP does not impact the computed results.

The architecture of the semi-distributed BP is shown in Fig. 2. The message passing on graph $\mathcal{G}_i$ is carried out by exchanging messages between the server and the users. Users locally store their personal ratings, and generate λ-messages according to (7) without disclosing their ratings to the server or other users. User u sends the λ-message $\lambda_{u,j}^{(n)}(s_{ij})$ to the server in the format shown in Table I, where the L-entry

1932-4553 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2015.2426677, IEEE Journal of Selected Topics in Signal Processing

TABLE I: The format of λ-messages.

Graph i	User u	Item j	Iteration n	L-entry vector $\overrightarrow{\lambda}$

TABLE II: The format of μ-messages.

Graph i	User u	Item j	Iteration n	L-entry vector $\overrightarrow{\mu}$

vector $\overrightarrow{\lambda}$ stores the values of $\lambda_{u,j}^{(n)}(s_{ij}=s)$, $\forall s \in \mathcal{S}$. The server is responsible for generating μ-messages. To compute the μ-message $\mu_{j,u}^{(n)}(s_{ij})$, the server checks if all λ-messages $\lambda_{u,j}^{(n-1)}(s_{ij})$ from users in U_j have been received. The server then performs multiplication operations on the λ-messages to generate μ-messages according to (8), and sends the μ-message $\mu_{j,u}^{(n)}(s_{ij})$ to user u in the format shown in Table II, where the L-entry vector $\overrightarrow{\mu}$ stores the values of $\mu_{j,u}^{(n)}(s_{ij}=s)$, $\forall s \in \mathcal{S}$. The server checks the convergence of messages, and obtains the item similarity after convergence using (10). Although we have applied synchronous BP here, asynchronous BP can be readily used, where messages are updated whenever new messages arrive. Indeed, round-robin asynchronous BP converges at least as fast as synchronous BP [17].

The computation of all item similarities in $\{\mathbb{S}_i : 1 \le i \le N\}$ requires N factor graphs, with $\mathbb{S}_i$ computed using factor graph $\mathcal{G}_i$. We introduce three protocols for coordinating the message passing on different graphs: the serial protocol, the parallel protocol, and the pipeline protocol. In the serial protocol, the message passing on factor graphs is performed in a serial manner, that is at any time, all generated messages belong to one factor graph, and unless inference on that graph is finished, no messages on other factor graphs are generated. Alternatively, we can adopt the parallel protocol. If the bandwidth of the communication channel between users and the server is sufficient, and if the user's computational capability allows, message passing on multiple factor graphs can be performed in parallel to accelerate the inference process. The pipeline protocol, which is a combination of the serial and parallel protocols, is favorable when the delay in the network is large. While waiting for the messages on one factor graph to arrive, users can compute messages on other factor graphs and continue to send them to the server, so as to increase throughput and make more efficient use of computational resources as well.

Aside from the semi-distributed implementation, a fully distributed implementation is also possible. For example, [18] proposed a distributed BP-based trust management algorithm for P2P networks. Users directly send λ-messages to other users instead of to the server, and also generate μ-messages locally using received λ-messages, without relying on the server. However, there is a significant increase in communication overhead, considering that a λ-message $\lambda_{u,j}(s_{ij})$ needs to be sent to each of the users in U_j, as they require this λ-message for updating μ-messages. Moreover, for each item a user has rated, he needs to find out other users who also have rated that item, i.e., the graph must be known by the users. Further, since the item similarity is locally computed by users, and user $u \in U_i$ only obtains $\{s_{ij} : j \in \hat{I}_u\}$, the users need to share with each other the item similarities. Thus, a more sophisticated protocol needs to be developed for a fully distributed system.

B. Cascaded BP

The semi-distributed BP architecture in Sec. IV-A requires that all users be active and participate in BP message propagation at the same time. Yet, in practical scenarios, it is difficult to meet this stringent requirement due to various reasons, e.g., some users may not be active temporally. To address this challenge, we adapt the original BP architecture to perform BP in cascade. Rather than wait until all users become active, we can perform BP on the subgraph constructed based only on the subset of users active during the current time slot, and store the inferred knowledge for use in the next time slot. Let $\mathbb{U}^{(1)}, \mathbb{U}^{(2)}, \ldots, \mathbb{U}^{(t)}$ denote the available subsets of users at time slots $1, 2, \ldots, t$, respectively. We construct $\mathcal{G}_i^{(t)}$ based on $\mathbb{U}^{(t)}$ at each time slot t, and perform inference in $\mathcal{G}_i^{(t)}$. To incorporate the knowledge from the previous graph $\mathcal{G}_i^{(t-1)}$ into $\mathcal{G}_i^{(t)}$ at time slot t, we introduce the priors $P(s_{ij}|\mathcal{G}_i^{(t-1)})$, $\forall s_{ij} \in \mathbb{S}_i$, where $P(s_{ij}|\mathcal{G}_i^{(t-1)})$ is the marginal distribution of s_{ij} computed from $\mathcal{G}_i^{(t-1)}$. With these priors, the server computes the μ-messages according to

$$\mu_{j,u}^{(n)}(s_{ij}) \propto \prod_{f \in F_j \setminus u} \lambda_{f,j}^{(n)}(s_{ij}) \times P(s_{ij}|\mathcal{G}_i^{(t-1)}). \tag{11}$$

Upon convergence, the server computes the marginal distribution $P(s_{ij}|\mathcal{G}_i^{(t)})$ as follows

$$P\left(s_{ij}|\mathcal{G}_i^{(t)}\right) = \frac{1}{Z_{ij}} \prod_{f \in F_j} \lambda_{f,j}^{(n)}(s_{ij}) \times P(s_{ij}|\mathcal{G}_i^{(t-1)}). \tag{12}$$

Note that $P(s_{ij}|\mathcal{G}_i^{(t-1)})$ is actually computed by the server in the previous graph $\mathcal{G}_i^{(t-1)}$, and hence it is directly available for the server. The computation of λ-messages at the user side is still the same as (7). Hence, during cascaded BP, the priors are always directly computed and used at the server side.

The intermediate item similarity s_{ij} at time slot t can be computed as

$$\hat{s}_{ij} = \sum_{s \in \mathcal{S}} s \times P\left(s_{ij} = s|\mathcal{G}_i^{(t)}\right). \tag{13}$$

We will also update s_{ij} at each time slot of the cascaded BP. This allows us to generate recommendations using the most up-to-date item similarity.

C. User-Side Recommendation

Thus far, we have focused on the item similarity computation using the semi-distributed BP algorithm. To complete the privacy-preserving item-based CF recommender system, it remains to specify the recommendation generation. As introduced in Sec. II, the item-based CF computes rating prediction for user u on item i using (1), which takes as its input the past ratings of user u and item similarities. To avoid revealing user ratings, users would then locally generate rating predictions. Since the item similarities are obtained at

1932-4553 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2015.2426677, IEEE Journal of Selected Topics in Signal Processing

TABLE III: Summarization of entity functions.

Entity	Function
Server	Coordinate message passing; Generate μ-messages; Compute and store item similarity.
User	Store personal rating data; Generate λ-messages; Generate recommendations.

the server side in the semi-distributed BP algorithm, the server should send to users the required item similarities. To predict ratings on all unseen items in $\mathbb{I}\backslash I_u$, user u only needs item similarities in $\{s_{ij} : i \in \mathbb{I}\backslash I_u, j \in I_u\}$. After computing rating predictions, users can locally store the item similarities received from the server for future uses, and only update them periodically. We summarize the functions of the server and users in Table III.

It is worth noting that in addition to preserving privacy, user-side recommendation also enhances user trust in e-commerce. Traditionally, centralized recommender systems owned by the commercial websites can manipulate the recommendations in various ways for revenue. A website might place the items with the highest profits on top of the recommendation list, or even employ recommendations as tools for advertisement of new products. This trust issue is well addressed by user-side recommendation, where users locally generate recommendations on their personal computers.

D. Accuracy and Communication Overhead

Assuming all users participate in BP, the semi-distributed architecture does not compromise the accuracy. Basically, we let users locally conduct the computations that require private rating data, but all computations are carried out exactly as in a centralized approach in Sec. III-C, and the user-side recommendation generates rating predictions using exactly (1). When not all users are available for BP at the same time slot, we can apply the cascaded BP scheme to improve performance over time. We will present performance evaluation later in Sec. VII.

The proposed algorithm incurs additional communication overhead. In the semi-distributed BP on graph $\mathcal{G}_i$, for each user $u \in U_i$, there are $|I_u|$ λ-messages and $|I_u|$ μ-messages exchanged between the server and use u in each iteration. And the server needs to send a total of $|I_u|(N - |I_u|)$ item similarities to user u for user-side recommendation. Whereas in the centralized approach, only the generated recommendations need to be sent to the user.

V. Information-Theoretic Privacy Analysis

We analyze the achieved privacy of the proposed algorithm for item-based CF by charactering the information leakage from the information-theoretic perspective. According to Sec. IV, the λ-messages and μ-messages are exchanged between the server and users, which can cause privacy degradation.

A. User Privacy Loss Due to λ-Messages

The λ-message $\lambda_{u,j}(s_{ij})$ in (7) is sent from user u to the server. Suppose the server tries to infer information from λ-messages about user u's ratings on items. We characterize the upper bound of information-theoretic privacy loss. To simplify the analysis, we assume s_{ij} can only take values from $\mathcal{S} = \{1, 2\}$. From (7) and (6), we can see that

$$\begin{cases} \lambda_{u,j}(s_{ij}=1) < \lambda_{u,j}(s_{ij}=2) & \text{if } |r_{uj} - r_{ui}| < \delta_{ui} \\ \lambda_{u,j}(s_{ij}=1) \geq \lambda_{u,j}(s_{ij}=2) & \text{if } |r_{uj} - r_{ui}| \geq \delta_{ui} \end{cases} \tag{14}$$

where $\delta_{ui} > 0$ is a parameter whose value is determined by ratings r_{uh} on other items $h \in \hat{I}_u\backslash j$ and the messages $\mu_{h,u}(s_{ih})$, $\forall h \in \hat{I}_u\backslash j$. Since the server does not have direct access to user ratings, δ_{ui} is unknown to it. To derive an upper bound, we assume that δ_{ui} is close to 1 with large probability, as the empirical results in [4] show that the item-based rating prediction method generates a rating with absolute error less than 1 on average using datasets with similar rating scale to Γ. Using this prior knowledge and (14), the server can infer about user ratings as follows

$$\begin{cases} r_{uj} = r_{ui} & \text{if } \lambda_{u,j}(s_{ij}=1) < \lambda_{u,j}(s_{ij}=2); \\ r_{uj} \neq r_{ui} & \text{if } \lambda_{u,j}(s_{ij}=1) \geq \lambda_{u,j}(s_{ij}=2). \end{cases} \tag{15}$$

We define the total privacy of user u, i.e., his rating information unknown to the server, as

$$\begin{aligned} H_u = H(\{r_{uk}|k \in I_u\}) &= \sum_{k \in I_u} H(r_{uk}) \\ &= |I_u| \log |\Gamma|, \end{aligned} \tag{16}$$

where $H(\cdot)$ denotes entropy. Note that we have assumed that user u rates different items independently, and the user rating on each item takes values from Γ with equal probabilities.

Next we compute the privacy loss $\Delta I(\lambda_{u,j}(s_{ij}))$ as the reduction of unknown information of user u's ratings after the server observes $\lambda_{u,j}(s_{ij})$

$$\begin{aligned} &\Delta I(\lambda_{u,j}(s_{ij})) \\ &= H(\{r_{uk}|k \in I_u\}) - H\Big(\{r_{uk}|k \in I_u\}\big|\lambda_{u,j}(s_{ij})\Big) \\ &= H(\{r_{uk}|k \in I_u\}) - \Big[H\Big(r_{uj}\big|\lambda_{u,j}(s_{ij}), \{r_{uk}|k \in I_u\backslash j\}\Big) \\ &\qquad + H\Big(\{r_{uk}|k \in I_u\backslash j\}\big|\lambda_{u,j}(s_{ij})\Big)\Big] \end{aligned} \tag{17}$$

Since $\lambda_{u,j}(s_{ij})$ only introduces dependency between r_{ui} and r_{uj}, we have

$$H\Big(r_{uj}\big|\lambda_{u,j}(s_{ij}), \{r_{uk}|k \in I_u\backslash j\}\Big) = H\left(r_{uj}|\lambda_{u,j}(s_{ij}), r_{ui}\right),$$

$$H\Big(\{r_{uk}|k \in I_u\backslash j\}\big|\lambda_{u,j}(s_{ij})\Big) = H(\{r_{uk}|k \in I_u\backslash j\}).$$

Hence,

$$\begin{aligned} &\Delta I(\lambda_{u,j}(s_{ij})) \\ &\quad = H(\{r_{uk}|k \in I_u\}) - H\left(r_{uj}|\lambda_{u,j}(s_{ij}), r_{ui}\right) \\ &\qquad\quad - H(\{r_{uk}|k \in I_u\backslash j\}) \\ &\quad = H(r_{uj}) - H(r_{uj}|\lambda_{u,j}(s_{ij}), r_{ui}). \end{aligned} \tag{18}$$

1932-4553 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2015.2426677, IEEE Journal of Selected Topics in Signal Processing

Specifically, for the two cases in (15), we have:
(1) If $\lambda_{u,j}(s_{ij}=1) < \lambda_{u,j}(s_{ij}=2)$

$$\begin{aligned}\Delta I_1(\lambda_{u,j}(s_{ij})) &= H(r_{uj}) - H(r_{uj}|r_{uj}=r_{ui}, r_{ui}) \\ &= \log|\Gamma|; \end{aligned} \tag{19}$$

(2) If $\lambda_{u,j}(s_{ij}=1) \geq \lambda_{u,j}(s_{ij}=2)$

$$\begin{aligned}\Delta I_2(\lambda_{u,j}(s_{ij})) &= H(r_{uj}) - H(r_{uj}|r_{uj}\neq r_{ui}, r_{ui}) \\ &= \log|\Gamma| - \log(|\Gamma|-1). \end{aligned} \tag{20}$$

The expected privacy loss per λ-message can be given by

$$\begin{aligned}&\mathbb{E}\left\{\Delta I(\lambda_{u,j}(s_{ij}))\right\} \\ &= \Delta I_1(\lambda_{u,j}(s_{ij}))P(r_{uj}=r_{ui}) \\ &\quad + \Delta I_2(\lambda_{u,j}(s_{ij}))P(r_{uj}\neq r_{ui}) \\ &= \log|\Gamma| - \frac{|\Gamma|-1}{|\Gamma|}\log(|\Gamma|-1), \end{aligned} \tag{21}$$

where $P(r_{uj}=r_{ui})$ is the probability that user u rates items i and j with the same rating, $P(r_{uj}=r_{ui})=\frac{1}{|\Gamma|}$ and $P(r_{uj}\neq r_{ui})=1-P(r_{uj}=r_{ui})$.

Taking into account all λ-messages from user u on graph $\mathcal{G}_i$, we can compute the λ-message privacy loss for user u on graph $\mathcal{G}_i$ as

$$\begin{aligned}&\Delta I\left(\{\lambda_{uj}(s_{ij})|j\in\hat{I}_u\}\right) \\ &= H\left(\{r_{uk}|k\in I_u\}\right) \\ &\quad - H\left(\{r_{uk}|k\in I_u\}\big|\{\lambda_{u,j}(s_{ij})|j\in\hat{I}_u\}\right) \\ &= H\left(\{r_{uk}|k\in I_u\}\right) \\ &\quad - H\left(\{r_{uk}|k\in I_u\backslash i\}\big|\{\lambda_{u,j}(s_{ij})|j\in\hat{I}_u\}, r_{ui}\right) \\ &\quad - H\left(r_{ui}\big|\{\lambda_{u,j}(s_{ij})|j\in\hat{I}_u\}\right). \end{aligned} \tag{22}$$

Observing that

$$\begin{aligned}&H\left(\{r_{uk}|k\in I_u\backslash i\}\big|\{\lambda_{u,j}(s_{ij})|j\in\hat{I}_u\}, r_{ui}\right) \\ &\qquad = \sum_{j\in I_u\backslash i} H\left(r_{uj}\big|\lambda_{u,j}(s_{ij}), r_{ui}\right), \end{aligned} \tag{23}$$

$$H\left(r_{ui}\big|\{\lambda_{u,j}(s_{ij})|j\in\hat{I}_u\}\right) = H(r_{ui}), \tag{24}$$

we have

$$\begin{aligned}&\Delta I\left(\{\lambda_{uj}(s_{ij})|j\in\hat{I}_u\}\right) \\ &= \sum_{j\in\hat{I}_u} H(r_{uj}) - \sum_{j\in\hat{I}_u} H(r_{uj}|\lambda_{u,j}(s_{ij}), r_{ui}) \\ &= \sum_{j\in\hat{I}_u} \Delta I(\lambda_{u,j}(s_{ij})). \end{aligned} \tag{25}$$

Hence, the expected privacy loss resulting from all λ-messages from user u on graph $\mathcal{G}_i$ is

$$\begin{aligned}&\mathbb{E}\left\{\Delta I\left(\{\lambda_{u,j}(s_{ij})|j\in\hat{I}_u\}\right)\right\} \\ &= \sum_{j\in\hat{I}_u} \mathbb{E}\left\{\Delta I(\lambda_{u,j}(s_{ij}))\right\} \\ &= |\hat{I}_u|\left(\log|\Gamma| - \frac{|\Gamma|-1}{|\Gamma|}\log(|\Gamma|-1)\right). \end{aligned} \tag{26}$$

As one example, suppose user u has rated 20 items with the rating scale $\Gamma=\{1,2,3,4,5\}$. His total privacy is $H_u=46.44$ bits, and the expected privacy loss due to λ-messages on one graph is 8.02 bits.

B. Total User Privacy Lost to the Server

In the previous analysis, we focused on the privacy loss due to messages on a single graph, but each user may participate in as many graphs as the number of items he rated, e.g., user u may send λ-messages on the set of graphs $\{\mathcal{G}_i|i\in I_u\}$. We are interested to know the total privacy loss of user u as a result of participating in multiple graphs. However, it is not simply the sum of privacy loss on all graphs, since the λ-messages on different graphs are not independent and thus there is significant redundancy in the revealed information. We next derive an upper bound of the total privacy loss.

Suppose the server is intelligent, and by combining the pairwise relationships between item ratings of user u inferred from λ-messages on multiple graphs using (15), it can divide the unknown ratings of user u in to $|\Gamma|$ groups, where the items in each group g have the same rating $r_g\in\Gamma$ while items in different groups have different ratings. Let G_g denote the set of items in group g, then $r_{uk}=r_g$, $\forall k\in G_g$. The unknown information of item ratings in each group g is

$$\begin{aligned}H(\{r_{uk}|k\in G_g\}) &= H\left(\{r_{uk}|k\in G_g\backslash i\}\,\big|r_{ui}\right) + H(r_{ui}) \\ &= H(r_{ui}) \\ &= H(r_g). \end{aligned} \tag{27}$$

The unknown information of item ratings in all groups can be derived as

$$\begin{aligned}&H\left(\{r_{uk}|k\in G_1\}, \{r_{uk}|k\in G_2\}, \ldots, \{r_{uk}|k\in G_{|\Gamma|}\}\right) \\ &= H(r_1, r_2, \ldots, r_{|\Gamma|}) \\ &= H(r_1) + H(r_2|r_1) + \ldots + H(r_{|\Gamma|}|r_1, r_2, \ldots, r_{|\Gamma|-1}) \\ &= \log|\Gamma| + \log(|\Gamma|-1) + \ldots + \log 1 \\ &= \log(|\Gamma|!). \end{aligned} \tag{28}$$

Hence, the total privacy loss of user u is

$$\Delta I_u^{\text{total}} = |I_u|\log|\Gamma| - \log(|\Gamma|!). \tag{29}$$

In the example of 20 rated items with $|\Gamma|=5$, the privacy loss is 39.53 bits versus the total privacy of 46.44 bits. However, without knowing exactly the rating of each group, the server will not be able to launch effective attacks against users to reveal sensitive personal information [6], [7]. Yet, the users are suggested to randomly participate in only a subset of graphs to better preserve their privacy.

C. User Privacy Lost to Other Users Due to μ-Messages

The μ-messages are sent from the server to users, and no direct messages are exchanged between users. Suppose a curious user u tries to extract from μ-messages information about ratings of other users. We notice that $\mu_{j,u}(s_{ij})$ in (8) mixes the λ-messages from multiple users, but does not contain any information about who are the source users that send those λ-messages. Thus, even if user u can extract any

1932-4553 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2015.2426677, IEEE Journal of Selected Topics in Signal Processing

information from μ-messages, he does not know whom the information should be related to.

Nevertheless, we are interested in characterizing the amount of information a curious user can extract from μ-messages about ratings of other users, assuming the server colludes with some curious users and provides the graph structure information of $\mathcal{G}_i$ to them. We compute the upper bound of the rating information of other users a curious user u can infer about from the received μ-message $\mu_{j,u}(s_{ij})$, as well as the privacy of a particular user lost to user u.

From (8), we know that $\mu_{j,u}(s_{ij})$ is the product of λ-messages from multiple users in $U_{ij}\backslash u$. To simplify analysis, we consider the worst case of privacy loss where $\mu_{j,u}(s_{ij})$ is consistent with each λ-message $\lambda_{v,j}(s_{ij})$, i.e., user u can infer about λ-messages as follows

(1) If $\mu_{j,u}(s_{ij}=1) < \mu_{j,u}(s_{ij}=2)$

$$\lambda_{v,j}(s_{ij}=1) < \lambda_{v,j}(s_{ij}=2);$$

(2) If $\mu_{j,u}(s_{ij}=1) \geq \mu_{j,u}(s_{ij}=2)$

$$\lambda_{v,j}(s_{ij}=1) \geq \lambda_{v,j}(s_{ij}=2).$$

With the inferred λ-messages, user u can further infer about the ratings from other users in $U_{ij}\backslash u$. The inference process is the same as described in Sec. V-A. Hence, the upper bound of the rating information of other users extracted from μ-message $\mu_{uj}(s_{ij})$ can be given by

$$\Delta I(\mu_{j,u}(s_{ij})) = \sum_{v\in U_{ij}\backslash u} \Delta I(\lambda_{v,j}(s_{ij})). \tag{30}$$

Using (21), we can obtain the expected amount of information extracted by user u from $\mu_{j,u}(s_{ij})$ as

$$\begin{aligned}&\mathbb{E}\left\{\Delta I(\mu_{j,u}(s_{ij}))\right\}\\ &= \sum_{v\in U_{ij}\backslash u} \mathbb{E}\left\{\Delta I(\lambda_{v,j}(s_{ij}))\right\}\\ &= (|U_{ij}|-1)\left(\log|\Gamma| - \frac{|\Gamma|-1}{|\Gamma|}\log(|\Gamma|-1)\right).\end{aligned} \tag{31}$$

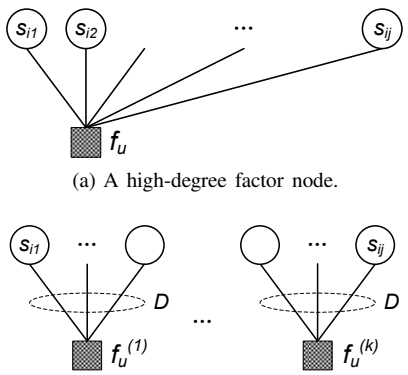
The privacy of user v lost to a curious user u are essentially caused by the μ-messages on graph $\mathcal{G}_i$ that mix λ-messages from user v and are forwarded to user u by the server. Hence, we can derive the expected privacy of user v lost to user u on graph $\mathcal{G}_i$ as

$$\begin{aligned}&\mathbb{E}\left\{\Delta I\left(\{\lambda_{v,j}(s_{ij})|j\in\hat{I}_{uv}\}\right)\right\}\\ &= |\hat{I}_{uv}|\left(\log|\Gamma| - \frac{|\Gamma|-1}{|\Gamma|}\log(|\Gamma|-1)\right),\end{aligned} \tag{32}$$

where $\hat{I}_{uv} = \hat{I}_u \cap \hat{I}_v$ denotes the set of common items rated by users u and v.

VI. Complexity Analysis and Reduction

The computational complexity of the BP algorithm is determined by the computation of λ-messages and μ-messages. While the complexity of generating a μ-message using (8) is $\mathcal{O}(|I_u|)$, the complexity of generating a λ-message using (7) is $\mathcal{O}\left(|I_u|L^{|I_u|}\right)$, where $L=|\mathcal{S}|$, which is exponential in the degree of the factor node, i.e., the number of items user u has rated. Unfortunately, in recommender systems, a user can rate over hundred of items. We thus propose a complexity reduction technique by controlling the degree of the factor node.



(a) A high-degree factor node.

(b) Multiple D-degree factor nodes.

Fig. 4: Illustration of complexity reduction via grouping.

A. Complexity Reduction

We randomly divide the variable nodes in $\hat{I}_u$ at factor node u into small groups of size D where D is a small integer. There are $G_u = \lceil|\hat{I}_u|/D\rceil$ such groups. Let $D_u^{(k)}$ denote the size of group k, $D_u^{(k)} = D$ for $1\leq k\leq G_u-1$ and $D_u^{(k)} = |\hat{I}_u| - D(G_u-1)$ for $k=G_u$. Let $I_u^{(k)}$ denote the variable nodes in group k. We set an indicator $j_u^{(k)}=1$ if $j\in I_u^{(k)}$ and $j_u^{(k)}=0$ otherwise. Since each variable node $j\in\hat{I}_u$ only belongs to one group, we have $\sum_{k=1}^{G_u} j_u^{(k)} = 1$. Let $\mathbb{S}_{ui}^{(k)} = \{s_{ij} : j\in I_u^{(k)}\}$ denote the variable nodes in group k of $\hat{I}_u$. Assuming independence of variable nodes in different groups, we replace the factor node function $f_u(\mathbb{S}_{ui})$ in (6) with

$$\prod_{k=1}^{G_u} f_u^{(k)}\left(\mathbb{S}_{ui}^{(k)}\right), \tag{33}$$

where $f_u^{(k)}$ is the local function of variables in group k. Hence, we modify the factor graph $\mathcal{G}_i$ accordingly to obtain a new factor graph $\hat{\mathcal{G}}_i$ for complexity reduction. Instead of connecting all variable nodes in $\hat{I}_u$ to one factor node u, we connect a separate factor node $u^{(k)}$ to the group of variable nodes in $I_u^{(k)}$. We illustrate the complexity reduction at one factor node in Fig. 4.

A more intuitive understanding is that, user u first randomly divides the items in $\hat{I}_u$ into multiple groups of size D, and decides item similarity on the basis of groups, as if items in different groups were rated independently by user u. Since variable node j is associated with item j, the variable nodes in $\hat{I}_u$ are grouped exactly as the grouping among items.

The local function at factor node $u^{(k)}$ can be similarly derived as (6). Assuming r_{ui} on item i is unknown, then using ratings from items within group k, user u predicts r_{ui} as

$$\hat{r}_{ui}(\mathbb{S}_{ui}^{(k)}) = \frac{\sum_{j\in I_u^{(k)}} s_{ij}\times r_{uj}}{\sum_{j\in I_u^{(k)}}|s_{ij}|}. \tag{34}$$

1932-4553 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2015.2426677, IEEE Journal of Selected Topics in Signal Processing

We substitute $\hat{r}_{ui}(\mathbb{S}_{ui})$ with $\hat{r}_{ui}(\mathbb{S}_{ui}^{(k)})$ in (6) to obtain the new local function of factor node $u^{(k)}$

$$f_u^{(k)}\left(\mathbb{S}_{ui}^{(k)}\right) = \frac{1}{Z_u^{(k)}} \exp\left\{-\frac{1}{\sigma^2}\left(\hat{r}_{ui}(\mathbb{S}_{ui}^{(k)}) - r_{ui}\right)^2\right\}, \quad (35)$$

where $Z_u^{(k)}$ is a normalization constant.

On the new factor graph $\hat{\mathcal{G}}_i$, we apply the BP algorithm described in Sec. III-C. The λ-messages and μ-messages are exchanged between the new factor nodes and variable nodes. The λ-message $\lambda^{(n)}_{u^{(k)},j}(s_{ij})$ sent from factor node $u^{(k)}$ to variable node j is given by

$$\lambda^{(n)}_{u^{(k)},j}(s_{ij}) \propto \sum_{\mathbb{S}_{ui}^{(k)} \setminus s_{ij}} f_u(\mathbb{S}_{ui}^{(k)}) \prod_{h \in I_u^{(k)} \setminus j} \mu^{(n-1)}_{h,u^{(k)}}(s_{ih}). \quad (36)$$

And the μ-message $\mu^{(n)}_{j,u^{(k)}}(s_{ij})$ sent from variable node j to factor node $u^{(k)}$ is given by

$$\mu^{(n)}_{j,u^{(k)}}(s_{ij}) \propto \prod_{f \in \hat{F}_j \setminus u^{(k)}} \lambda^{(n)}_{f,j}(s_{ij}), \quad (37)$$

where $\hat{F}_j = \{v^{(k)} : v \in U_{ij},\, j_v^{(k)} = 1,\, 1 \le k \le G_v\}$.

The complexity of updating a λ-message is effectively reduced to $\mathcal{O}\left(DL^D\right)$ from $\mathcal{O}\left(|I_u|L^{|I_u|}\right)$ by using (36). Meanwhile, the total number of λ-messages need to be generated from user u's perspective in each iteration remains $|\hat{I}_u|$, which is the same as in $\mathcal{G}_i$. There is no change in complexity regarding the μ-messages. The overall complexity of the BP algorithm on $\hat{\mathcal{G}}_i$ with complexity reduction in each iteration is $\mathcal{O}\left(\bar{M}\bar{N}DL^D + N\bar{M}^2\right)$, where $\bar{N}$ is the average number of items rated by one user, and $\bar{M}$ is the average number of users of one item. Since the number of items a user can consume is limited by his time and money, we can assume $\bar{N}$ is much smaller than N. As for $\bar{M}$, we assume $\bar{M}$ grows in the order of $M^{1-\epsilon}$, where $0 < \epsilon < 1$. Then we can rewrite the computation complexity on $\hat{\mathcal{G}}_i$ as $\mathcal{O}\left(M^{1-\epsilon}\bar{N}DL^D + NM^{2(1-\epsilon)}\right)$, and when N and M is large, it is dominated by the second term, so we have $\mathcal{O}\left(NM^{2(1-\epsilon)}\right)$.

B. Impact on Semi-Distributed BP

The complexity reduction part can be easily integrated into the semi-distributed BP architecture in Sec. IV. Indeed, the complexity reduction is for reducing the computational burden of the users, and users can locally perform the required grouping. Also, the grouping information is not needed at the server side for BP. The server does not need to know from which group of user u the λ-message is generated, since only one $\lambda^{(n)}_{u,j}(s_{ij})$ message is generated by user u that is destined to variable node j on graph $\mathcal{G}_i$. Meanwhile, at the user side, user u can easily recover the group information "k" from the received message $\mu^{(n)}_{j,u}(s_{ij})$ by looking up for k with $j_u^{(k)} = 1$, and obtain $\mu^{(n)}_{j,u^{(k)}}(s_{ij})$, as if it was computed using (37), which can then be used to update $\lambda^{(n)}_{u^{(k)},j}(s_{ij})$ in (36).

Therefore, the grouping step is transparent to the server. The computational complexity on graph $\mathcal{G}_i$ at each user is only $\mathcal{O}\left(\bar{N}DL^D\right)$, regardless of N and M, and users can autonomously adjust D according to the availability of local computation resources and power.

VII. Experimental Evaluation

We evaluate the accuracy of the proposed privacy-preserving item-based CF algorithm on the 100K MovieLens dataset[2], which consists of $100,000$ ratings on 1682 items (movies) by 943 users. Each rating is an integer between 1 and 5. We randomly divide the dataset into two disjoint sets: a training set containing 80% of the ratings, and a test set containing the rest 20% of the ratings. The ratings in the training set are used as memory for the item-based CF algorithm to compute item similarities and predict unknown ratings. We compare the predicted ratings with the actual ratings in the test set to evaluate the accuracy of the recommendation algorithms in terms of Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), which are computed as follows

$$\text{MAE} = \frac{1}{|\mathbb{T}|}\sum_{r_{ui}\in\mathbb{T}} |r_{ui} - \hat{r}_{ui}|,$$

$$\text{RMSE} = \sqrt{\frac{1}{|\mathbb{T}|}\sum_{r_{ui}\in\mathbb{T}} (r_{ui} - \hat{r}_{ui})^2},$$

where $\hat{r}_{ui}$ is the predicted rating, and r_{ui} is the actual rating in the test set denoted by $\mathbb{T}$. The smaller the MAE and RMSE, the better accuracy. Note that RMSE is more sensitive to large errors than MAE. In the experiments in Secs. VII-A and VII-B, we assume all users are active and participate in BP during the same time slot. In Sec. VII-C we examine the performance of the cascaded BP where only a subset of users are active during each time slot.

A. Performance Comparison

We compare our proposed privacy-preserving algorithm with other item-based CF algorithms using the well-known similarity measures, including the CS, PCS and ACS methods as introduced in Sec. II. In all cases, rating predictions are generated by (1). We assume no privacy requirement is imposed on other algorithms, so the CS, PCS and ACS measures are directly applied to original rating data, and thus their results are not compromised by privacy techniques such as obfuscation. In particular, the presented results of the CS measure represent the best achievable performance of the distributed personal recommender system proposed in [19], as the item similarity between two items is computed based on the complete rating vectors associated with them in $\mathbb{R}$, rather than computed in an incremental manner as in [19].

The PCS and ACS methods compute the item similarity s_{ij} between two items i and j using the ratings from the set of their common users $U_{ij} = U_i \cap U_j$. We denote by N_c the minimum number of common users required for computing the item similarity s_{ij} using PCS and ACS. If $|U_{ij}| < N_c$, then neither item i nor item j will be used to predict each other's ratings. Let $\mathbb{N}_i$ be the set of all valid items for item i under N_c. To predict ratings on item i for user u, the neighborhood $\mathcal{N}_{ui}$ used in (1) is formed from the set of items in $\mathbb{N}_{ui} = \mathbb{N}_i \cap I_u$. In addition, given a required neighborhood size K, if $|\mathbb{N}_{ui}| < K$, we simply say the unknown rating r_{ui} in the test set $\mathbb{T}$ is

[2]Available at: http://www.grouplens.org/node/73.

1932-4553 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2015.2426677, IEEE Journal of Selected Topics in Signal Processing

TABLE IV: MAE performance comparison of various item-based CF algorithms under different neighborhood size K.

Algorithms	$N_c = 3$ ($P_{\text{pred}} = 74\%$)					$N_c = 8$ ($P_{\text{pred}} = 67\%$)				
	$K = 10$	$K = 20$	$K = 30$	$K = 40$	$K = 50$	$K = 10$	$K = 20$	$K = 30$	$K = 40$	$K = 50$
PCS	0.8839	0.8486	0.8370	0.8326	0.8418	0.8330	0.8073	0.8009	0.7989	0.8127
ACS	0.7812	0.7585	0.7609	0.8029	0.9278	0.7390	0.7264	0.7389	0.8044	0.9765
CS	0.7567	0.7601	0.7676	0.7751	0.7812	0.7418	0.7428	0.7494	0.7566	0.7632
Proposed	0.7512	0.7437	0.7486	0.7557	0.7632	0.7283	0.7255	0.7293	0.7359	0.7450

TABLE V: RMSE performance comparison of various item-based CF algorithms under different neighborhood size K.

Algorithms	$N_c = 3$ ($P_{\text{pred}} = 74\%$)					$N_c = 8$ ($P_{\text{pred}} = 67\%$)				
	$K = 10$	$K = 20$	$K = 30$	$K = 40$	$K = 50$	$K = 10$	$K = 20$	$K = 30$	$K = 40$	$K = 50$
PCS	1.0993	1.0562	1.0435	1.0392	1.0560	1.0403	1.0096	1.0031	1.0013	1.0208
ACS	0.9896	0.9622	0.9671	1.0473	1.2712	0.9433	0.9264	0.9475	1.0755	1.3740
CS	0.9754	0.9791	0.9876	0.9942	0.9992	0.9577	0.9583	0.9651	0.9716	0.9768
Proposed	0.9680	0.9543	0.9580	0.9637	0.9703	0.9397	0.9322	0.9349	0.9400	0.9485

unpredictable by the PCS and ACS. We denote $\mathbb{T}_K$ as the subset of all predictable ratings in $\mathbb{T}$ at neighborhood size K. Note that $\mathbb{T}_K$ changes with N_c, since N_c impacts $\mathbb{N}_i$. For fair comparison of different algorithms, we apply the same N_c to all evaluated algorithms.

In Tables IV and V, we examine the MAE and RMSE performance of various algorithms. The parameters of the proposed algorithm are set as $D = 4$, $\mathcal{S} = \{1, 2\}$, and $\sigma = 0.5$. We show the results for $N_c = 3$ and 8 with K varying from 10 to 50 in steps of 10. Under each N_c, to fairly compare the performance of different K's, all results are obtained on the same subset of test ratings $\mathbb{T}_{50}$, and the percentage of ratings in $\mathbb{T}$ used for evaluation is

$$P_{\text{pred}} = |\mathbb{T}_{50}|/|\mathbb{T}| \times 100\%.$$

Our proposed algorithm achieves superior performance compared to other algorithms in terms of both MAE and RMSE. As K increases from 10, the performance of the proposed algorithm, as well as PCS and ACS algorithms, first improves but then degrades when K becomes too large, because ratings from neighbor items with smaller similarity to the active item, on which the rating is predicted, corrupt the prediction accuracy. Thus, to achieve the best performance, a proper neighborhood size K should be chosen. The computational complexity cost of our algorithm to solve for the similarities between item i and other items on graph $\mathcal{G}_i$ is $\mathcal{O}\left(NM^{2(1-\epsilon)}\right)$ for large N and M as discussed in Sec. VI, whereas for the CS method the complexity is $\mathcal{O}\left(NM\right)$, and for PCS and ACS, the complexity is $\mathcal{O}\left(NM^{(1-\epsilon)}\right)$. Finally, as we will see next, the performance of our algorithm can be further improved if a higher degree D is used, but at the cost of increased computational complexity at users.

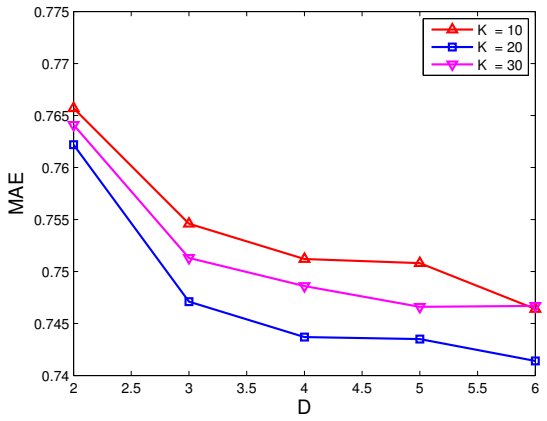


Fig. 5: Impact of D on MAE of the proposed algorithm.

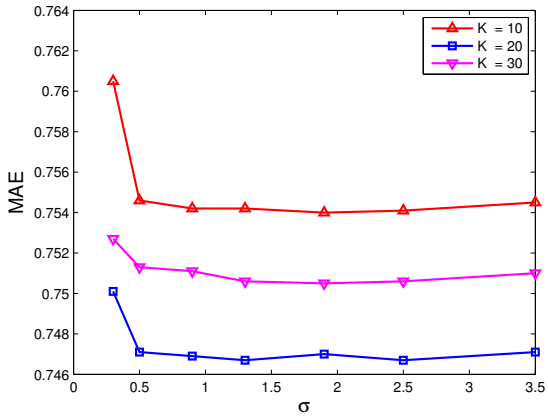


Fig. 6: Impact of σ on MAE of the proposed algorithm.

B. Impact of Parameters

In the following experiments, we investigate the impact of the parameters D, σ, and $\mathcal{S}$ on the performance of our proposed algorithm. We always set $N_c = 3$ and evaluate the proposed algorithm on $\mathbb{T}_{50}$. In Fig. 5, we investigate the influence of group size D on the accuracy of the proposed algorithm, where we fix other parameters as $\mathcal{S} = \{1, 2\}$, and $\sigma = 0.5$. It can be seen that increasing D slightly improves accuracy. However, since the computational complexity at user side is $\mathcal{O}\left(\bar{N}DL^D\right)$, which is exponential in D, users need to wisely choose D according to their computational capability. In Fig. 6, we show the results for different σ's, where $D = 3$ and $\mathcal{S} = \{1, 2\}$. The performance slightly degrades if σ is too

1932-4553 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2015.2426677, IEEE Journal of Selected Topics in Signal Processing

TABLE VI: Impact of $\mathcal{S}$ on MAE of the proposed algorithm.

$\mathcal{S}$	MAE		
	$K=10$	$K=20$	$K=30$
$\mathcal{S}_2$	0.7546	0.7471	0.7513
$\mathcal{S}_5$	0.7504	0.7500	0.7545

small or too large, since for a small σ, the curve of the factor function (6) quickly flattens out with respect to $|\hat{r}_{ui} - r_{ui}|$, while for a large σ, the curve becomes too flat. But overall, the algorithm is not sensitive for large dynamic ranges of σ.

In Table VI, we show the results of the proposed algorithm for different $\mathcal{S}$'s, where $\mathcal{S}_l = \{1, 2, \ldots, l\}, \forall l \in \{2, 5\}$, $D = 3$, and $\sigma = 0.5$. We observe that $\mathcal{S}_2$ and $\mathcal{S}_5$ achieve their best performance when $K = 20$, but $\mathcal{S}_2$ actually provides better accuracy than $\mathcal{S}_5$. This is because large l could cause overfitting, that is the obtained item similarity is strongly biased towards the memory data, and does not generalize well when used for prediction. Meanwhile, the computational complexity $\mathcal{O}\left(\bar{N}DL^D\right)$ at user side significantly increases with $L = |\mathcal{S}_l|$, depending on D.

C. Cascaded BP

We now investigate the more challenging scenarios where only a subset of users are available for BP at a given time slot. Firstly, we consider applying the basic BP algorithm without introducing priors, i.e., the inferred knowledge from previous time slots are not incorporated. The parameters are set as $\mathcal{S} = \{1, 2\}$, $D = 3$, $\sigma = 0.5$, and $K = \{10, 20, 30\}$. In Fig. 7, we show the MAE results with the percentage of participating users ranging from 10% to 100%. The performance degrades significantly as the percentage of users decreases. Thus, relying only on the users available at the current time slot is not satisfactory.

Next, we apply the cascaded BP proposed in Sec. IV-B. In the experiment, the users are randomly divided into 5 subsets with equal sizes, i.e., only 20% of users can participate in BP at each time slot. In Fig. 8, we show the MAE performance versus the number of elapsed time slots, where at each time slot the updated item similarity results are used to predict ratings. We also show the results of the original BP algorithm with all users participating at the same time slot. We can see that as the number of elapsed time slots increases, the MAE performance improves. After 5 time slots, the performance of the cascaded BP approaches that of the original BP algorithm with all users. Therefore, incorporating knowledge from previous time slots can effectively improve the performance.

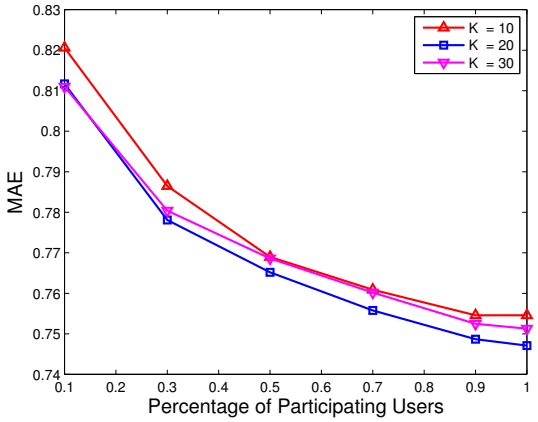


Fig. 7: MAE versus percentage of participating users.

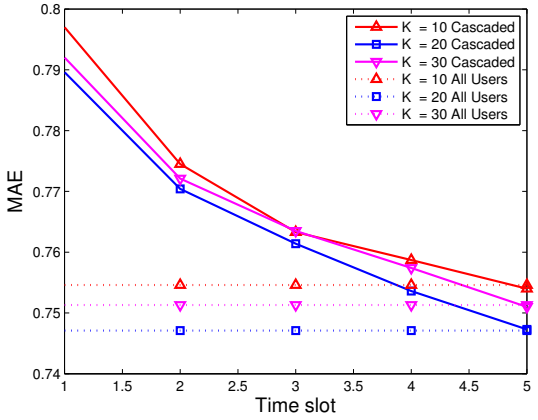


Fig. 8: MAE performance of cascaded BP.

VIII. Related Works

Recently, privacy-preserving information processing and data mining techniques have been widely studied. In [20], the authors proposed distributed online learning with intrinsic privacy-preserving properties, where local parameters at each participating user are updated based on local data sources, and parameters are periodically exchanged among a small subset of neighbors, and they showed that malicious users cannot reconstruct the subgradients of other users. In [21], the authors applied homomorphic encryption to privacy-preserving distributed aggregation of energy consumption metering data in smart grids. A number of works also investigated privacy-preserving information processing in semi-distributed or centralized cloud computing. [22] proposed privacy-preserving back-propagation neural network learning via cloud computing, where private data of each party are first encrypted before being uploaded to the cloud, and operations over ciphertexts are supported via doubly homomorphic encryption. [23] addressed multi-keyword ranked search over encrypted data in cloud computing using secure inner product computation.

There are existing works on both centralized and distributed privacy-preserving recommendation systems. The privacy-preserving techniques that have been applied to centralized systems mainly include data obfuscation and cryptography approaches. The data obfuscation approach is to obfuscate user ratings with random noise before releasing them to the server, e.g., differentially private recommender system [24] and the perturbation technique [10]. In [6], [11], the authors proposed to disguise genuine user profiles by adding extra fake data. However, obfuscation and random noise undermine accuracy, and users have to sacrifice more privacy for better recommendation. In [25], the authors proposed a data pertur-

1932-4553 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2015.2426677, IEEE Journal of Selected Topics in Signal Processing

bation approach based on differential privacy with accuracy guarantees.

The cryptography approach encrypts users data using advanced cryptography techniques to hide user information, e.g., item ratings, from the recommender server, which only operates on the encrypted user data [26], [27]. In particular, [27] proposed an item-based privacy-preserving recommender system with homomorphic encryption, where the server maintains an item-item similarity model, and generates rating predictions blindly by performing homomorphic operations on the encrypted ratings. However, the authors assumed the similarity model is known *a priori*, and did not provide any privacy-preserving solution for that. Besides, key management required by cryptography tools could be demanding in practice. [28] applied the garbled circuits cryptographic technique to matrix factorization collaborative filtering, where a third-party crypto-service provider is required for private computation. [29] discussed practical implementations of privacy-preserving collaborative filtering deployed in cloud computing infrastructures.

Alternatively, distributed privacy-preserving recommender systems let users store data in their devices. [19] presented a personal collaborative filtering recommender running on the user side. Each user stores his data locally, and constructs an item-item similarity model using the cosine similarity measure by incrementally incorporating ratings from other neighbour users in a peer-to-peer (P2P) environment. The quality of the locally constructed similarity model depends on the set of neighbours a user can find and contact. The best performance is achieved by using ratings from all common users of two items to evaluate the item similarity, which is much harder to implement in P2P architectures than in a centralized CF system. Moreover, the user privacy is not guaranteed as users need to share their ratings with each other. An attacker can easily mimic the behaviour of genuine users to acquire their personal data. As in centralized systems, cryptography [12] and obfuscation techniques [30] can be applied to distributed systems to avoid disclosing data to other users. A client-side content-based mobile shopping recommender was proposed in [31], where user data, e.g., historical purchases, are locally stored on users' devices, e.g., smartphones, and each device runs its own content-based recommendation algorithm to find products that match the user's profile based on product descriptions. However, the complete product catalog can be challenging for mobile devices to process, and incurs significant data traffic as well. In [32], the authors proposed social connection recommendation in mobile social networking, where matching user attributes are computed via secure multi-party computation (SMC) techniques based on secret sharing.

Previously, BP has been applied to recommender systems without considering privacy in [33]–[35]. In [33], the proposed algorithm therein follows the philosophy of the user-based CF algorithm. To receive recommendations, an active user discloses his ratings to other users, who then compare their own ratings with the active user's ratings on common items in order to update their "confidence", which can be understood as similarity to the active user. The central server collects "confidences" as well as ratings of all users, and sends back to users probabilistic messages regarding predicted ratings on items. In [34], each user combines his "confidence" and ratings to form probabilistic messages on ratings, and sends them to the server. The work in [35] proposed to predict ratings for an active user on a Pairwise Markov Random Field (PMRF), where the local evidence for each unknown rating is the aggregated ratings from other users similar to the active user, and probabilistic messages on predicted ratings are exchanged between similar items. Whereas in this work, we are concerned with preserving privacy in item-based CF recommendation. Instead of directly using BP for rating prediction as in [33]–[35], we employ BP for item similarity computation in a semi-distributed fashion, where messages are exchanged between the server and users, without disclosing user ratings. The rating prediction for an active user is then locally computed at the user side by combining his own ratings and item similarity. We note that we can further apply the distributed privacy-preserving belief propagation [36] that sends only masked messages to each party for enhanced privacy.

IX. Conclusion

In this work, we proposed a semi-distributed BP approach to item-based CF recommender system that preservers user privacy. The proposed algorithm computes item similarity by exchanging probabilistic messages between the server and users without directly exposing user ratings. To address the issue that only a subset of users participate in BP at one time slot, we proposed the cascaded BP scheme which accumulates the inferred knowledge from the previous time slots. With the computed similarities, a user then locally generates rating predictions as the weighted average of his own ratings on items. Through information-theoretic analysis, we showed that the proposed semi-distributed BP algorithm effectively preserves user privacy. In addition, we proposed a complexity reduction technique for efficient computation at the user side. The experimental results on the MovieLens dataset demonstrated that the BP algorithm with all users participating at the same time slot achieves superior performance, and the cascaded BP algorithm can improve performance as the number of elapsed time slots increases.

References

[1] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, Jan. 2003.

[2] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, pp. 734–749, Jun. 2005.

[3] P. Resnick, N. Iakovou, M. Sushak, P. Bergstrom, and J. Riedl, "Grouplens: An open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 Computer Supported Cooperative Work Conference*, 1994, pp. 175–186.

[4] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th International Conference on World Wide Web*, 2001, pp. 285–295.

[5] M. S. Ackerman, L. F. Cranor, and J. Reagle, "Privacy in e-commerce: Examining user scenarios and privacy preferences," in *Proceedings of the 1st ACM Conference on Electronic Commerce*, Denver, CO, USA, 1999, pp. 1–8.

1932-4553 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTSP.2015.2426677, IEEE Journal of Selected Topics in Signal Processing

[6] U. Weinsberg, S. Bhagat, S. Ioannidis, and N. Taft, "BlurMe: Inferring and obfuscating user gender based on ratings," in *Proceedings of the Sixth ACM Conference on Recommender Systems*, Dublin, Ireland, 2012, pp. 195–202.

[7] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, 2008, pp. 111–125.

[8] X. Amatriain, "Beyond data: from user information to business value through personalized recommendations and consumer science," in *Proc. of the 22nd ACM International Conference on Information and Knowledge Management (CIKM'13)*, 2013, pp. 2201–2208.

[9] L. Rainie, S. Kiesler, R. Kang, and M. Madden, "Anonymity, privacy, and security online," 2013, Pew Research Center [Online] http://www.pewinternet.org/2013/09/05/anonymity-privacy-and-security-online/.

[10] H. Polat and W. Du, "Privacy-preserving collaborative filtering using randomized perturbation techniques," in *Proceedings of the Third IEEE International Conference on Data Mining*, 2003, pp. 625–628.

[11] R. Shokri, P. Pedarsani, G. Theodorakopoulos, and 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.

[12] J. F. Canny, "Collaborative filtering with privacy," in *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, 2002, pp. 45–57.

[13] N. Lathia, S. Hailes, and L. Capra, "Private distributed collaborative filtering using estimated concordance measures," in *Proceedings of the 2007 ACM Conference on Recommender Systems*, 2007, pp. 1–8.

[14] J. Zou, A. Einolghozati, and F. Fekri, "Privacy-preserving item-based collaborative filtering using semi-distributed belief propagation," in *Proceedings of the First IEEE Conference on Communications and Network Security (CNS'13)*, Washington, D.C., USA, 2013.

[15] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[16] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc., 1988.

[17] G. Elidan, I. McGraw, and D. Koller, "Residual belief propagation: Informed scheduling for asynchronous message passing," in *Proc. of the 22nd Conference on Uncertainty in AI (UAI'06)*, 2006.

[18] E. Ayday and F. Fekri, "BP-P2P: Belief propagation-based trust and reputation management for P2P networks," in *Proceedings of the 9th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, 2012, pp. 578–586.

[19] B. N. Miller, J. A. Konstan, and J. Riedl, "Pocketlens: Toward a personal recommender system," *ACM Transactions on Information Systems*, vol. 22, no. 3, pp. 437–476, Jul. 2004.

[20] F. Yan, S. Sundaram, S. Vishwanathan, and Y. Qi, "Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 11, pp. 2483–2493, Nov. 2013.

[21] C. Rottondi, G. Verticale, and C. Krauss, "Distributed privacy-preserving aggregation of metering data in smart grids," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 7, pp. 1342–1354, July 2013.

[22] J. Yuan and S. Yu, "Privacy preserving back-propagation neural network learning made practical with cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 212–221, Jan. 2014.

[23] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 222–233, Jan. 2014.

[24] F. McSherry and I. Mironov, "Differentially private recommender systems: Building privacy into the net," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009, pp. 627–636.

[25] Y. Shen and H. Jin, "Privacy-preserving personalized recommendation: An instance-based approach via differential privacy," in *Proc. of IEEE Int. Conf. on Data Mining (ICDM'14)*, 2014, pp. 540–549.

[26] Z. Erkin, T. Veugen, T. Toft, and R. L. Lagendijk, "Generating private recommendations efficiently using homomorphic encryption and data packing," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 1053–1066, Jun. 2012.

[27] Z. Erkin, M. Beye, T. Veugen, and R. L. Lagendijk, "Privacy-preserving content-based recommender system," in *Proceedings of the 14th ACM Workshop on Multimedia and Security*, 2012, pp. 77–84.

[28] V. Nikolaenko, S. Ioannidis, U. Weinsberg, M. Joye, N. Taft, and D. Boneh, "Privacy-preserving matrix factorization," in *Proc. of ACM SIGSAC Conf. on Computer and Communications Security (CCS'13)*, 2013, pp. 801–812.

[29] A. Basu, J. Vaidya, H. Kikuchi, and T. Dimitrakos, "Privacy-preserving collaborative filtering on the cloud and practical implementation experiences," in *Proc. of IEEE Sixth International Conference on Cloud Computing (CLOUD'13)*, 2013, pp. 406–413.

[30] S. Berkovsky, Y. Eytani, T. Kuflik, and F. Ricci, "Enhancing privacy and preserving accuracy of a distributed collaborative filtering," in *Proceedings of the 2007 ACM Conference on Recommender Systems*, 2007, pp. 9–16.

[31] T. D. Pessemier, K. Vanhecke, and L. Martens, "A hybrid strategy for privacy-preserving recommendations for mobile shopping," in *Proc. of Workshop on New Trends in Content-based Recommender Systems (CBRecSys'14)*, 2014, pp. 22–25.

[32] M. Li, S. Yu, N. Cao, and W. Lou, "Privacy-preserving distributed profile matching in proximity-based mobile social networks," *IEEE Transactions on Wireless Communications*, vol. 12, no. 5, pp. 2024–2033, May 2013.

[33] E. Ayday and F. Fekri, "A belief propagation based recommender system for online services," in *Proceedings of the Fourth ACM Conference on Recommender Systems*, Barcelona, Spain, 2010, pp. 217–220.

[34] E. Ayday, A. Einolghozati, and F. Fekri, "BPRS: Belief propagation based iterative recommender system," in *Proceedings of the 2012 IEEE International Symposium on Information Theory*, Cambridge, MA, 2012, pp. 1992–1996.

[35] E. Ayday, J. Zou, A. Einolghozati, and F. Fekri, "A recommender system based on belief propagation over pairwise Markov random fields," in *Proceedings of the 50th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, USA, 2012.

[36] M. Kearns, J. Tan, and J. Wortman, "Privacy-preserving belief propagation and sampling," in *Proceedings of the 21st Annual Conference on Neural Information Processing Systems (NIPS'07)*, 2007.

Jun Zou received his B.S. degree in electronic engineering from Shanghai Jiao Tong University, Shanghai, China, in 2009, and his M.S. degree in electrical and computer engineering from Georgia Institute of Technology, Atlanta, GA, USA, in 2012. He is now pursuing his Ph.D. degree in electrical and computer engineering at the Georgia Institute of Technology, Atlanta, GA. His current research interests include belief propagation, probabilistic graphical models, privacy and security techniques, and their applications to collaborative filtering.

Faramarz Fekri received Ph.D. degree from the Georgia Institute of Technology in 2000. Since 2000, Dr. Fekri has been with the faculty of the School of Electrical and Computer Engineering at the Georgia Institute of Technology where he currently holds a Professor position. He serves on the Technical Program Committees of several IEEE conferences. In the past, he was on the editorial board of the IEEE Transactions on Communications, and the Elsevier Journal on PHYCOM.

Prof. Fekri's current research interests are in the area of communications and signal processing, in particular source and channel coding, information theory in biology, statistical inference in large data, information processing for wireless and sensor networks, and communication security. Dr. Fekri received the National Science Foundation CAREER Award (2001), Southern Center for Electrical Engineering Education (SCEEE) Young Faculty Development Award (2003), and Outstanding Young faculty Award of the School of ECE (2006). He is a Senior Member of the IEEE.

1932-4553 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.