The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Privacy Preserving Point-of-Interest Recommendation Using Decentralized Matrix Factorization

Chaochao Chen, Ziqi Liu, Peilin Zhao,* Jun Zhou, Xiaolong Li
AI Department, Ant Financial Services Group, China
{chaochao.ccc, ziqiliu, peilin.zpl, jun.zhoujun, xl.li}@antfin.com

Abstract

Points of interest (POI) recommendation has been drawn much attention recently due to the increasing popularity of location-based networks, e.g., Foursquare and Yelp. Among the existing approaches to POI recommendation, Matrix Factorization (MF) based techniques have proven to be effective. However, existing MF approaches suffer from two major problems: (1) Expensive computations and storages due to the centralized model training mechanism: the centralized learners have to maintain the whole user-item rating matrix, and potentially huge low rank matrices. (2) Privacy issues: the users' preferences are at risk of leaking to malicious attackers via the centralized learner. To solve these, we present a Decentralized MF (DMF) framework for POI recommendation. Specifically, instead of maintaining all the low rank matrices and sensitive rating data for training, we propose a random walk based decentralized training technique to train MF models on each user's end, e.g., cell phone and Pad. By doing so, the ratings of each user are still kept on one's own hand, and moreover, decentralized learning can be taken as distributed learning with multi-learners (users), and thus alleviates the computation and storage issue. Experimental results on two real-world datasets demonstrate that, comparing with the classic and state-of-the-art latent factor models, DMF significantly improvements the recommendation performance in terms of precision and recall.

Introduction

Nowadays, location-based networks, e.g., Foursquare and Yelp, are becoming more and more popular. These platforms provide kinds of point of interests (POIs) such as hotels, restaurants, and markets, which makes our lives much easier than before. Meanwhile, the problem of "where to go" starts to bother people, since it is time-consuming for people to find their desired places from so many POIs. POI recommendation appears to address such a problem by helping users filter out uninteresting POIs and saving their decision-making time (Ye et al. 2011; Gao et al. 2015).

Among the existing research from POI recommendation communities, Matrix Factorization (MF) techniques draws a lot of attention (Cheng et al. 2011; 2012; Yang et al. 2013), and it has proven to be attractive in many recommendation applications (Koren et al. 2009; Chen et al. 2014; Koenigstein, Dror, and Koren 2011; Chen et al. 2016). However, existing MF approaches train the recommendation model in a centralized way. That is, the recommender system, actually the organization who built it, first needs to collect the action data of all the users on all the POIs, and then trains a MF model. The shortcomings of this kind of centralized MF methods are mainly three-folds. (1) *High cost of resources*. Centralized MF not only needs large storage to store the collected user action data on POIs, but also requires huge computing resources to train the model, especially when datasets are large. (2) *Low model training efficiency*. Centralized MF model is trained on a single machine or a cluster of machines. Thus, the model training efficiency is restricted to the number of machines available. (3) *Shallow user privacy protection*. In recommender systems, user privacy is an importance concern (Canny 2002), especially in POI scenarios. Most people do not want to release their activities on location based network, not only to other people, but also to the platforms, for privacy concerns. However, centralized MF trains model on the basis of having all the users' activities data. In fact, the safest way is that users' data are kept in their own hand, without sending them to anyone or any organization (platform).

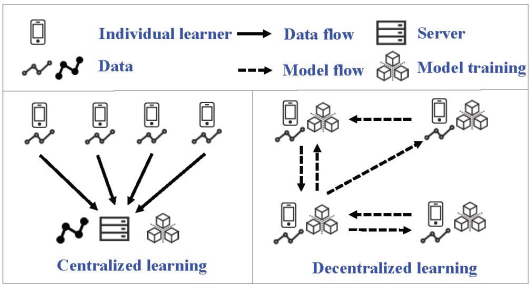


Figure 1: Comparison between centralized learning and decentralized learning.

To solve the above problems caused by centralized training of MF, we propose Decentralized MF (DMF) framework. That is, instead of training user and item latent factors in a centralized way by obtaining all the ratings, we

*Corresponding author.

train MF model in each user's end, e.g., user's cell phone and Pad. Figure 1 shows the difference between centralized and decentralized learning. DMF treats each user's device as an autonomous learner (individual computation unit). As we know, the essence of MF is that user and item latent factors are learnt collaboratively, so should be DMF. Three key challenges exist when making users collaborate in DMF meanwhile preserving user privacy. The first challenge is *which user should be communicated*. We answer this question by analyzing the data in POI recommendation scenarios, and propose nearby user communication on user adjacent graph, which is built based on users' geographical information. Along with this, the second challenge naturally appears: *how far should users communicate with their neighbors*. To address this, we present a random walk method to enhance local user communication. That is, we use random walk technique to intelligently select users' high-order neighbors instead of only their direct neighbors. The third and also the most serious challenge is *what information should users communicate with each other* without leaking their data privacy. To solve this challenge, we decompose item preference into global (common) and local (personal) latent factors, and allow users communicate with each other by sending the gradients of the global item latent factors.

Our proposed DMF framework successfully deals with the shortcomings of centralized MF. (1) Each user only needs to store his own latent factor, and items' latent factors. The computation is also cheap: each user only need to update the corresponding user and item latent factors when he (his neighbors) rates (rate) an item. (2) Since each user updates the DMF model on his own side, it can be taken as a distributed learning system with #users as #machines, which makes the model efficient to train. (3) The ratings of each user on items are still kept on one's own hand, which avoids user's privacy being disclosed.

We summarize our main contributions as follows:

- We propose a novel DMF framework for POI recommendation, which is scalable and is able to preserve user privacy. To our best knowledge, it is the first attempt in literature.
- We propose an efficient way to train DMF. Specifically, when a user rates an item on his side, the user and item gradients are first calculated. We then present a random walk based technique for users to send the item gradient to their neighbors. Finally, the corresponding user/item latent factors are updated using stochastic gradient descent.
- Experimental results conducted on two real-world datasets demonstrate that DMF can achieve even better performance compared with the classic and state-of-the-art latent factor models in terms of precision and recall. Parameter analysis also shows the effectiveness of our proposed random walk based optimization method.

Background

In this section, we review some necessary backgrounds which form the basis of our work, i.e., (1) Matrix Factorization (MF) in POI Recommendation, (2) decentralized learning.

Matrix Factorization in POI Recommendation

MF aims to learn user and item (POI) latent factors through regressing over the existing user-item ratings (Koren et al. 2009; Mnih and Salakhutdinov 2007), which can be formalized as follows,

$$\min_{u_i,v_j} \sum_{i=1}^{I}\sum_{j=1}^{J}\left(r_{ij}-u_i^T v_j\right)^2+\lambda\left(\sum_{i=1}^{I}||u_i||^2+\sum_{j=1}^{J}||v_j||^2\right), \tag{1}$$

where u_i and v_j denote the latent factors of user i and item j, respectively, and r_{ij} denotes the known rating of user i on item j. We will describe other parameters in details later.

MF and its variants have been extensively applied to POI recommendation due to their promising performance and scalability (Cheng et al. 2011; 2012; Yang et al. 2013). However, these methods are all trained by using the centralized mechanism. This centralized MF training results in expensive resources required, low model training efficiency, and shallow protection of user privacy.

Decentralized Learning

Decentralized learning appears to solve the above problems of centralized learning (Nedic and Ozdaglar 2009; Yan et al. 2013). Recently, it has been applied in many scenarios such as multiarmed bandit (Kalathil, Nayyar, and Jain 2014), network distance prediction (Liao, Geurts, and Leduc 2010), hash function learning (Leng et al. 2015), and deep networks (Shokri and Shmatikov 2015; McMahan et al. 2016).

The most similar existing works to ours are decentralized matrix completion (Ling et al. 2012; Yun et al. 2014). However, we summarize the following two major differences. (1) They either allow each learner (user) to communicate with those who have rated the same items or communicate with all the learners, and thus have low accuracy or high communication cost. In practice, users always collaborate with their affinitive users. To capture this, in this paper, we propose a random walk approach for users from the adjacent graph to collaboratively communicate with each other. (2) They allow directly exchange of item preference among learners, which may cause information leakage. For example, it is easy to be hacked by using the idea of collaborative filtering (Sarwar et al. 2001), i.e., similar items tend to be preferred by similar users. Assume user i is a malicious user, he has his own latent factor of item j (v_j^i). He also gets the latent factor of item j from user i' ($v_j^{i'}$). If i likes item j, and v_j^i and $v_j^{i'}$ are similar, then i will know i' likes j as well. In contrast, in this paper, we propose a gradient exchange scheme to limit the possibility of privacy leakage.

The Proposed Model

In this section, we first formally describe the Decentralized Matrix Factorization (DMF) problem. We then discuss a nearby user communication scheme for users to collaborate with each other. Next, we propose an enhanced version by applying random walk theory. Then, we present a privacy preserving nearby user collaboration algorithm to optimize DMF. We analyze the model complexity in the end.

Preliminary

Formally, let $\mathcal{U}$ and $\mathcal{V}$ be the user and item (POI) set with I and J denoting user size and item size, respectively. Let (i, j) be an interaction between user $i \in \mathcal{U}$ and item $j \in \mathcal{V}$, and r_{ij} be the rating of user i on item j. Without loss of generality, we assume $r_{ij} \in [0, 1]$ in this paper. Let $\mathcal{O}$ be the training dataset, where all the user-item ratings in it are known.

For the traditional centralized MF, it first collects all the $r_{ij} \in \mathcal{O}$, and then learns $\mathbf{U} \in \mathbb{R}^{K \times I}$ and $\mathbf{V} \in \mathbb{R}^{K \times J}$ using MF technique (Equation 1). Here, $\mathbf{U} \in \mathbb{R}^{K \times I}$ and $\mathbf{V} \in \mathbb{R}^{K \times J}$ denote the user and item latent factor matrices, with their column vectors u_i and v_j be the K-dimensional latent factors for user i and item j, respectively.

For DMF, to guarantee the privacy of each user, we need to keep all the known ratings and latent factors on each user's end during the whole training procedure. To do this, we use $\mathbf{U} \in \mathbb{R}^{K \times I}$ to denote user latent factor matrix, with each column vector u_i denotes the K-dimensional latent factors for user i. We also use $\mathbf{V} \in \mathbb{R}^{I \times K \times J}$ to denote item latent factor tensor, with $\mathbf{V}^i \in \mathbb{R}^{K \times J}$ denotes the item latent factor matrix for user i, and further with v_j^i denotes the K-dimensional latent factors for item j of user i. Thus, each user i only needs to store i's own K-dimensional latent factor u_i, and i's item latent factor matrix $\mathbf{V}^i$.

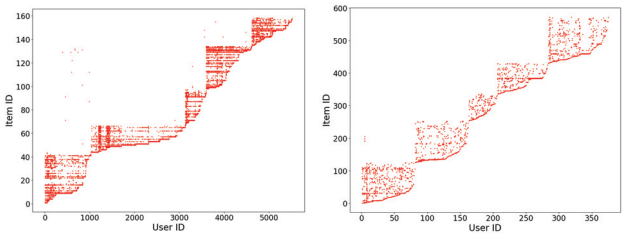
Besides, users need to collaboratively learn their stored factors, i.e., u_i and $\mathbf{V}^i$, in DMF scenario. For centralized MF, all the users share the same item latent factor matrix, i.e., $\mathbf{V}^i = \mathbf{V}^{i'}, \forall\, i, i' \in \mathcal{U}$. For DMF, each user stores his u_i and $\mathbf{V}^i$, and they should be trained collaboratively with other users—which we call 'neighbors'. Suppose we have a user adjacent graph $\mathcal{G}$, we use $\mathbf{W} \in \mathbb{R}^{I \times I}$ to denote user adjacency matrix, where each element $w_{i,i'} \in [0, 1]$ denotes the degree of relationship between user i and i'. Of course, user i and i' have no relationship if $w_{i,i'} = 0$. We use $\mathcal{N}^d(i)$ to denote the d^{st} order neighbors of i on $\mathcal{G}$, $|\mathcal{N}^d(i)|$ as the neighbor size, and $|\mathcal{N}^D(i)| = \sum_d^D |\mathcal{N}^d(i)|$. Obviously, $\mathcal{N}^1(i)$ denotes the direct neighbors of i. Besides, to save communication cost, we use N to denote the maximum number of direct neighbors of each user.

DMF aims to learn u_i and $\mathbf{V}^i$ for each user, and the model learning procedure is performed on one's own side, e.g., cell phone and Pad.

Nearby User Communication

The essence of MF is that the user and item latent factors are learnt collaboratively, so should be DMF. Thus, *which user should be communicated under DMF framework* becomes the first challenging question. We answer this question by first analyze the data in POI recommendation scenarios.

Observation. Figure 2 shows the user-POI check-in distributions on two real datasets, i.e., *Foursquare* and *Alipay*. Both datasets contain user-item-check-in-location records, and we divide locations into different cities. We randomly select the user-item check-in records in five cities from both datasets, and plot their check-in distributions in Figure 2,



(a) Foursquare dataset (b) Alipay dataset

Figure 2: Data analysis of Foursquare and Alipay datasets.

where each dot denotes a user-item check-in record. From it, we have the following observation: in POI scenarios, most users are only active in a certain city, and we call it "location aggregation". Only a few users be active in multi-cities, which is neglectable.

User Adjacent Graph. We represent the affinities among users based on the definition of a user adjacency graph. It can be built using whatever information that is available, e.g., rating similarity (Su and Khoshgoftaar 2009) and user social relationship (Yang et al. 2011). However, in DMF for POI recommendation scenario, users' ratings are on their own hand, and social relationship is not always available. Thus, we focus on using user geographical information to build user adjacent graph, similar as the existing researches (Ye et al. 2011; Cho, Myers, and Leskovec 2011; Cheng et al. 2012). Specifically, suppose $d_{i,i'}$ is the distance between user i and i', the relationship degree between i and i' is defined as

$$w_{i,i'} = I^{i,i'} \cdot f(d_{i,i'}), \tag{2}$$

where $w_{i,i'} \in [0, 1]$, $I^{i,i'}$ is the indicator function that equals to 1 if i and i' are in the same city and 0 otherwise, and $f(d_{i,i'})$ is a mapping function of distance and relationship degree, and the smaller the distance of i and i' is, the bigger their relationship degree is. Existing research has proposed different such mapping functions (Zhao, King, and Lyu 2016). In practice, it's extremely expensive if we maintain the communications for those super-users who have a huge number of neighbors. Thus, we set N to be the maximum number of neighbors each user can have. With nearby user communication schema, the first question is answered.

Random Walk Enhanced Nearby User Communication

With the adjacency matrix W representing the communication graph among users, *how far should users communication with their neighbors* becomes the second challenging question. Practically, a user's decision on an item is not only affected by his direct neighbors, but also the further neighbors, e.g., the neighbors of neighbors. Thus, when a user i rates an item j, this information should not only be sent to the directly neighbors of user i, but also his further neighbors, as shown in Figure 3. The challenge remains to be how far to explore the network, since there is a tradeoff between decentralization and communication/computation cost: the

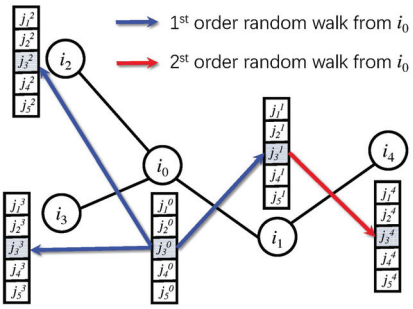


Figure 3: Random walk enhanced nearby user communication. User i_0 communicates with his neighbors once he has an interaction with POI j_3.

further the communication is, the more users can collaborate, meanwhile, the more communication and computation need to be done. We propose to solve this challenge by using random walk theory, which has been used to model trust relationship between users (Jamali and Ester 2009).

Random walk. We aim to find an intelligent way to determine how far a user communicate with his neighbors. Assuming user i wants to communicate with his direct neighbors ($k \in \mathcal{N}^1(i)$), we define n_i as the activity of user i selecting a user from his neighbor set, and thus

$$P(n_i = k) = \frac{w_{ik}}{\sum_{i' \in \mathcal{N}^1(i)} w_{ii'}}. \tag{3}$$

According to the Markov property (Aldous and Fill 2002), the probability of user i choosing his 2^{st} order of neighbors ($k' \in \mathcal{N}^2(i)$) is

$$P(n_i = k') = \sum_k P(n_i = k)P(n_k = k') \propto \sum_k w_{ik}w_{kk'}. \tag{4}$$

We use D to denote the max distance of random walk, and generally, the adjacent matrix of d^{st} order of neighbors is W^d. With random walk theory on user adjacent graph, the second question is answered.

DMF: Privacy Preserving Nearby User Communication.

The random walk based nearby user communication is an intelligent way for selecting neighbors to be communicated within POI recommendation scenarios, but the third challenging question remains: *what information should users communicate with each other*, that is, how should users collaboratively learn the DMF model without leaking their data privacy. The original rating, of course, can clearly reflect his preference on this item. However, the rating itself discloses the user's privacy too much. Inspired by the work (Yan et al. 2013), we propose a privacy preserving collaborative approach for decentralized POI recommendation scenarios. Specifically, we suppose that for each user, the corresponding j-th item latent factor v_j^i can be decomposed as follows:

$$v_j^i = p_j + q_j^i, \tag{5}$$

which implies that the latent factor of item j for user i is the sum between *one common (global) latent factor* p_j and *one personal (local) latent factor* q_j^i, where the common factor represents the common preference of all the users while the personal factor shows the personal favor of user i. Under this assumption, the DMF model can be formulated as

$$\begin{aligned} \min_{u_i, v_j^i \in \mathbb{R}^K} \mathcal{L} = &\sum_{i=1}^{I} l(r, u_i, v^i) + \frac{\alpha}{2}\sum_{i=1}^{I}||u_i||_F^2 \\ &+ \frac{\beta}{2}\sum_{j=1}^{J}||p_j||_F^2 + \frac{\gamma}{2}\sum_{i=1}^{I}\sum_{j=1}^{J}||q_j^i||_F^2 \\ &\text{s.t. } v_j^i = p_j + q_j^i, \end{aligned} \tag{6}$$

where $l(r, u_i, V^i)$ can be least square loss

$$l(r, u_i, V^i) = \frac{1}{2}\sum_{j=1}^{J}\left(r_{ij} - u_i^T v_j^i\right)^2, \tag{7}$$

which minimizes the error between real ratings and predicted ratings (Mnih and Salakhutdinov 2007), or listwise loss (Shi, Larson, and Hanjalic 2010), as well as pairwise loss (Rendle et al. 2009). In this paper, we will take least square loss as an example. The last three terms in Equation (6) are regularizers to prevent overfitting.

The factors u_i and q_j^i only depend on the information stored in user i, while the item common latent factor p_j depends on the information of all the users. Practically, in decentralized learning scenario, p_j is also saved on each user's (learner's) hand, and thus, for each user i, p_j is actually saved as p_j^i. Consequently, Equation (5) becomes

$$v_j^i = p_j^i + q_j^i. \tag{8}$$

Thus, it needs one protocol for users to exchange p_j^i to learn a global p_j. To solve this issue, inspired by (Yan et al. 2013), we propose to send the gradient of the loss $\mathcal{L}$ with respect to p_j, i.e., p_j^i for each user i, to his neighbors, to help learn a global p_j. This gradient exchange method has been successfully applied in decentralized learning scenarios (Nedic and Ozdaglar 2009; Yan et al. 2013), which not only guarantees model convergency, but also protects the privacy of user raw data. For each user i, the gradients of $\mathcal{L}$ with respect to u_i, p_j^i, and q_j^i are:

$$\frac{\partial \mathcal{L}}{\partial u_i} = -\left(r_{ij} - u_i^T v_j^i\right)v_j^i + \alpha u_i, \tag{9}$$

$$\frac{\partial \mathcal{L}}{\partial p_j^i} = -\left(r_{ij} - u_i^T v_j^i\right)u_i + \beta p_j^i, \tag{10}$$

$$\frac{\partial \mathcal{L}}{\partial q_j^i} = -\left(r_{ij} - u_i^T v_j^i\right)u_i + \gamma q_j^i. \tag{11}$$

Based on the above gradient exchange protocol, users collaboratively learn a global p_j. Figure 3 shows a demo of this protocal: i_0 will send $\partial\mathcal{L}/\partial p_3^0$ to his neighbors, to collaboratively learn p_3. Combining our proposed random walk enhanced nearby user communication method and gradient exchange protocol, we summarize our proposed privacy preserving DMF optimization framework for POI recommendation (Equation 6) in Algorithm 1. As we can see that users

Algorithm 1: Random Walk Enhanced Nearby Collaborative DMF Optimization

Input: training ratings ($\mathcal{O}$), learning rate (θ), user adjacency matrix (W), regularization strength(α, β, γ), maximum random walk distance (D), and maximum iterations (T)
Output: user latent factor (u_i), common item latent factor (p_j^i), and personal item latent factor (q_j^i)

1 **for** $i = 1$ *to* I **do**
2 Initialize u_i, p_j^i, and q_j^i
3 **end**
4 **for** $t = 1$ *to* T **do**
5 Shuffle training data $\mathcal{O}$
6 **for** r_{ij} *in* $\mathcal{O}$ **do**
7 Calculate $\frac{\partial \mathcal{L}}{\partial u_i}$ based on Equation (9)
8 Calculate $\frac{\partial \mathcal{L}}{\partial p_j^i}$ based on Equation (10)
9 Calculate $\frac{\partial \mathcal{L}}{\partial q_j^i}$ based on Equation (11)
10 Update u_i by $u_i \leftarrow u_i - \theta \frac{\partial \mathcal{L}}{\partial u_i}$
11 Update p_j^i by $p_j^i \leftarrow p_j^i - \theta \frac{\partial \mathcal{L}}{\partial p_j^i}$
12 Update q_j^i by $q_j^i \leftarrow q_j^i - \theta \frac{\partial \mathcal{L}}{\partial q_j^i}$
13 **for** *user* i' *in* $\mathcal{N}^d(i), d \in \{1, 2, ..., D\}$ **do**
14 Receive $\frac{\partial \mathcal{L}}{\partial p_j^i}$ from user i
15 Update $p_j^{i'}$ by $p_j^{i'} \leftarrow p_j^{i'} - \theta |\mathcal{N}^d(i)| W_{ii'} \frac{\partial \mathcal{L}}{\partial p_j^i}$
16 **end**
17 **end**
18 **end**
19 **return** u_i, p_j^i, and q_j^i

communicate with each other by sending the common item latent factor gradients instead of raw data, i.e., ratings, which significantly reduces the possibility of information leakage. With the gradient exchange protocol, the third question is answered.

From the objective function of DMF in Equation (6), we can easily make the following observations:

- If β is very large, then $p_j \to 0$, and thus users will not exchange item common preferences, which means that item preference is learnt only based on their own data.
- If γ is very large, then $q_j^i \to 0$, which indicates that users will not save their personal favor on items anymore. It will work more like centralized MF.

The values of β and γ determine how well item (common and personal) preferences are learnt. We will empirically study their effects on our model performance in experiments.

Unobserved rating sample. A universal problem appears in POI recommendation is that the observations are extremely sparse. Unless we have access to negative observations, we will probably obtain an estimator that tends to predict all the unknown (i, j) as 1. Following the existing researches (Yang et al. 2011), we solve this problem by sampling unobserved (i, j) during SGD optimization. Specifically, for each $r_{ij} \in \mathcal{O}$, we randomly sample m missing entries $r_{ij',j'=1:m}$ and treat them as negative examples, i.e., $r_{ij'} = 0$. However, a missing entry $r_{ij'}$ can denote either i does not like j' or i does not know the existing of j'. Therefore, we decrease the confidence of $r_{ij'}$ to $1/m$.

Complexity Analysis

Here we analyze the communication and computation complexity of Algorithm 1. Recall that K denotes the dimension of latent factor, N denotes the maximum number of direct neighbors of each user, D denotes the max distance of random walk, $\mathcal{O}$ as the training data, and $|\mathcal{O}|$ as its size.

Communication Complexity. The communication cost depends on both the length of item gradient and number of neighbor to be communicated. Each item gradient contains $4K$ bytes information, since it is a K dimensional real-valued vector. For user i, the max number of neighbors to be communicated of D^{st} order random walk is $\min(|C^i|, \mathcal{N}^D(i))$, where $|C^i|$ is the number of users in the current city of i. Thus, for each $r_{ij} \in \mathcal{O}$, the communication cost is $\min(|C^i|, \mathcal{N}^D(i)) \cdot 4K$ bytes. It will be $|\mathcal{O}| \cdot \min(|C^i|, \mathcal{N}^D(i)) \cdot 4K$ bytes for passing the training dataset once. The values of N, D, and K are usually small, and thus the communication cost is linear with the training data size.

Computation Complexity. The computation cost mainly relies on two parts, (1) calculating gradients, i.e., Line $7-9$ in Algorithm 1, and (2) updating user and item latent factors, i.e., Line $10-12, 15$ in Algorithm 1. For a single pass of the training data, the time complexity of (1) is $|\mathcal{O}| \cdot K$, and the time complexity of (2) is $|\mathcal{O}| \cdot \min(|C^i|, \mathcal{N}^D(i)) \cdot K$. Therefore, the total computational complexity in one iteration is $|\mathcal{O}| \cdot \min(|C^i|, \mathcal{N}^D(i)) \cdot K$. The values of N, D, and K are usually small, and thus the time complexity is linear with the training data size.

The above communication and computation complexity analysis shows that our proposed approach is very efficient and can scale to very large datasets.

Experiments

In this section, we empirically compare the performance of DMF with the classic centralized MF models, we also study the effects of parameters on model performance.

Setting

We first describe the datasets, metrics, and comparison methods we use during our experiments.

Datasets. We use two real-world datasets, i.e., *Foursquare* and *Alipay*. *Foursquare* is a famous benchmark dataset for evaluating a POI recommendation model (Yang, Zhang, and Qu 2016). We randomly choose two cities for each country from the original dataset, and further remove the users and POIs that have too few or too many interactions[1]. Our *Alipay* dataset is sampled from user-merchant offline check-in

[1]The reason we sample a small dataset is: we mock decen-

Table 1: Dataset statistics.

Dataset	#user	#item	#rating	#cities
Foursquare	6,524	3,197	26,186	117
Alipay	5,996	7,404	18,978	298

records during 2017/07/01 to 2017/07/31, and we also perform similar preprocess on it. Table 1 shows the statistics after process for both datasets, with which we randomly sample 90% as training set and the rest 10% as test set.

Metrics. We adopt two metrics to evaluate our model performance, i.e., $P@k$ and $R@k$, which are commonly used to evaluate POI recommendation performance (Cheng et al. 2012; Gao et al. 2015). For user i, they are defined as

$$P@k = \frac{|S_i^T \cap S_i^R|}{k}, \quad R@k = \frac{|S_i^T \cap S_i^R|}{|S_i^T|},$$

where S_i^T denotes the visited POI set of user i in the test data, and S_i^R denotes the recommended POI set of user i which contains k POIs.

Comparison methods. Our proposed DMF framework is a novel decentralized algorithm for POI recommendation, which is a decentralized version for the classic MF model (Mnih and Salakhutdinov 2007). We compare our proposed DMF with the following classic and state-of-the-art latent factor models, including several variants of DMF:

- **MF** (Mnih and Salakhutdinov 2007) is a classic centralized latent factor model which uses least square loss.
- **Bayesian Personalized Ranking (BPR)** (Rendle et al. 2009) is the state-of-the-art centralized latent factor model which uses pairwise loss.
- **Global DMF (GDMF)**. Users will not save their personal favor (q_j^i) anymore, and they tend to share similar latent factor for the same item. This is a special case of our proposed DMF model, i.e., when γ is very large.
- **Local DMF (LDMF)**. Users will not exchange preferences and they learn the model only based on their own data. This is also a special case of our proposed DMF model, i.e., when β is very large.

Note that we do not compare with the state-of-the-art POI recommendation methods. This is because, (1) most of them are the improvement of the classic MF model by using additional information, e.g. user social information and contextual information (Cheng et al. 2012; Yang et al. 2013; Gao et al. 2015), which are not fair to compare with, and (2) our focus is to compare the effectiveness of the traditional centralized latent factor models and our proposed decentralized MF model.

Hyper-parameters. During comparison, we set user regularizer $\alpha = 0.1$, learning rate $\theta = 0.1$, and the returned number of POI $k \in \{5, 10\}$. We also set the maximum number of neighbor $N = 2$, and the number of sampled unobserved ratings $m = 3$. After we build the user adjacent graph, we simply set $w_{i,i'} = 1$ to eliminate the effect of mapping function on model performance, since this is not the focus of this paper. For the latent factor dimension K, we vary its values in $\{5, 10, 15\}$. For the random walk distance D, we vary its values in $\{1, 2, 3, 4\}$. We also vary β and γ in $\{10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}\}$ to study their effects on DMF. We tune parameters of each model to achieve their best performance for comparison.

tralized learning during our experiments, that is, there will be $2I \times \mathbb{R}^{K \times J}$ POI (global and local) latent matrices in total, which actually is not a small scale.

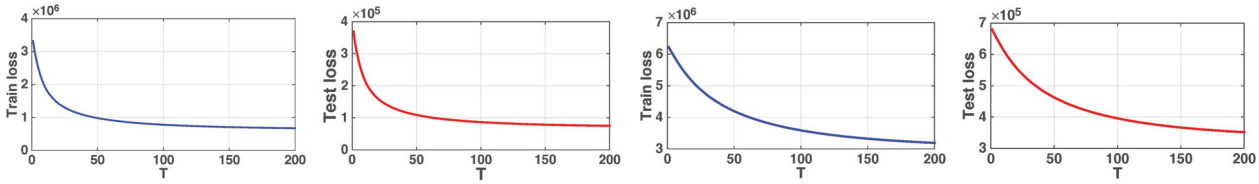
Table 2: Performance comparison on *Foursquare* dataset.

Metrics	P@5	R@5	P@10	R@10
Dimension	*K=5*			
MF	0.0291	0.1030	0.0242	0.1697
BPR	0.0293	0.1168	0.0247	0.2058
GDMF	0.0307	0.1245	0.0263	0.2115
LDMF	0.0025	0.0125	0.0025	0.0125
DMF	**0.0337**	**0.1562**	**0.0291**	**0.2418**
Dimension	*K=10*			
MF	0.0289	0.1250	0.0263	0.2226
BPR	0.0370	0.1533	0.0302	0.2514
GDMF	0.0281	0.1286	0.0269	0.2398
LDMF	0.0133	0.0565	0.0133	0.1121
DMF	**0.0386**	**0.1553**	**0.0340**	**0.2774**
Dimension	*K=15*			
MF	0.0406	0.1711	0.0316	0.2638
BPR	**0.0448**	**0.1898**	0.0342	0.2872
GDMF	0.0316	0.1388	0.0287	0.2500
LDMF	0.0187	0.0714	0.0170	0.1511
DMF	0.0421	0.1801	**0.0370**	**0.3035**

Comparison Results

We report the comparison results on both *Foursquare* and *Alipay* datasets in Table 2 and Table 3, respectively. From them, we find that:

- All the model performances increase with the dimension of latent factor (K). This is because the latent factor with a bigger K contains more information, and thus user and POI preferences can be learnt more precisely.
- GDMF achieves comparable performance with MF, since users collaboratively learn the global item latent factor, which works similarly as the traditional MF.
- LDMF behave the worst, since each user learn item preference only based on his own check-in data which is very sparse. This indicates the importance of user collaboration during recommendation.
- DMF consistently outperforms MF, GDMF, and LDMF, and even beats the pairwise ranking model (BPR) in most cases. Take the result of DMF on *Alipay* dataset for example, $P@5$ and $R@5$ of DMF improve those of MF by 27.75% and 25.89% when $K = 5$. This is because, the item preference of DMF contains not only common preference that obtained from all the users' data, but also each user's personal preference that learnt from his own data. Moreover, the random walk technique intelligently help

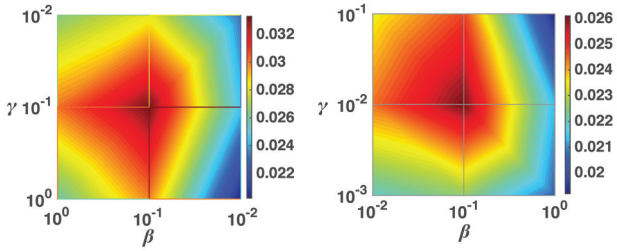


(a) *Foursquare* training loss (b) *Foursquare* test loss (c) *Alipay* training loss (d) *Alipay* test loss

Figure 4: The training and test loss of DMF with respect to the maximum iteration (T) on *Foursquare* and *Alipay* datasets.

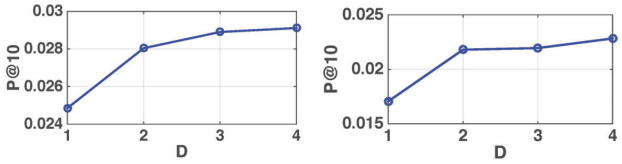
Table 3: Performance comparison on *Alipay* dataset.

Metrics	P@5	R@5	P@10	R@10
Dimension	*K=5*			
MF	0.0209	0.0896	0.0134	0.1194
BPR	0.0243	0.1027	0.0166	0.1408
GDMF	0.0209	0.0937	0.0144	0.1231
LDMF	0.0017	0.0042	0.0008	0.0042
DMF	**0.0267**	**0.1128**	**0.0228**	**0.1824**
Dimension	*K=10*			
MF	0.0343	0.1493	0.0261	0.2239
BPR	0.0353	0.1512	0.0286	0.2439
GDMF	0.0304	0.1146	0.0277	0.2202
LDMF	0.0013	0.0065	0.0013	0.0131
DMF	**0.0357**	**0.1612**	**0.0291**	**0.2537**
Dimension	*K=15*			
MF	0.0378	0.1538	0.0266	0.2308
BPR	**0.0448**	**0.1898**	0.0342	0.2872
GDMF	0.0355	0.1383	0.0267	0.2123
LDMF	0.0137	0.0684	0.0103	0.0983
DMF	0.0413	0.1942	**0.0370**	**0.3035**



(a) *Foursquare* (b) *Alipay*

Figure 5: Effect of β and γ on DMF.



(a) *Foursquare* (b) *Alipay*

Figure 6: Effect of D on DMF.

users to choose neighbors to communicate with. Therefore, item preferences are learnt more precisely, to better match each user's favor.

Parameter Analysis

Finally, we study the effects of parameters on DMF, including item global and local regularizer (β and γ), maximum random walk distance (D), and maximum iteration (T).

Effect of β and γ.

The item global regularizer (β) and local regularizer (γ) controls the proportion of one's item preference comes from his own data or other users' data. The bigger β is, the more one's item preference comes from his own data, and similarly, the bigger γ is, the more one's item preference comes from other users' data through global item gradient ($\partial\mathcal{L}/\partial p_j^i$) exchange. Figure 5 shows their effects on DMF on both datasets. From it, we can find that, with the good choices of β and γ, DMF can make full use of one's own data and his neighbors' data, and thus, achieves the best performance.

Effect of maximum random walk distance (D). The maximum random walk distance determinates how many neighbors will be communicated after a user has interaction with a POI, as we described in complexity analysis section. Figure 6 shows its effect on DMF model on both *Foursquare* and *Alipay* datasets, where we set $K = 5$ and fix other parameters to their best values. From it, we see that with the increase of D, our model performance increases, and tends to be relative stable when D is bigger than 3. This shows that DMF achieves a good performance with only a small value of D, which indicates a low cost of communication complexity.

Effect of maximum iteration (T). As we analyzed above, the computing time complexity is linear with the training data size, and thus, the converge speed determines how long DMF should be trained. Figure 4 shows the effect of T on training loss and test loss on both *Foursquare* and *Alipay* datasets. As we can observe, DMF converges steadily with the increase of T, and it takes about 100 epochs to converge on *Foursquare* and about 200 epochs on *Alipay*.

Conclusion and Future Work

In this paper, we proposed a Decentralized MF (DMF) framework for POI recommendation. Specifically, we proposed a random walk based nearby collaborative decentralized training technique to train DMF model in each user's

end. By doing this, the data of each user on items are still kept on one's own hand, and moreover, decentralized learning can be taken as distributed learning with multi-learner (user), and thus solves the efficiency problem. Experimental results on two real-world datasets demonstrate that, comparing with the classic and state-of-the-art latent factor models, DMF significantly improvements the recommendation performance in terms of precision and recall.

We would like to take model compression as our future work. Currently, each user needs to store the real-valued item latent matrix. A binary type of latent matrix will significantly reduce the storage cost. How to balance the model storage and model accuracy will be our next stage of work.

References

Aldous, D., and Fill, J. 2002. Reversible markov chains and random walks on graphs.

Canny, J. 2002. Collaborative filtering with privacy. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, 45–57. IEEE.

Chen, C.; Zheng, X.; Wang, Y.; Hong, F.; and Lin, Z. 2014. Context-aware collaborative topic regression with social matrix factorization for recommender systems. In *AAAI*, volume 14, 9–15.

Chen, C.; Zheng, X.; Wang, Y.; Hong, F.; Chen, D.; et al. 2016. Capturing semantic correlation for item recommendation in tagging systems. In *AAAI*, 108–114.

Cheng, Z.; Caverlee, J.; Lee, K.; and Sui, D. Z. 2011. Exploring millions of footprints in location sharing services. *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media* 2011:81–88.

Cheng, C.; Yang, H.; King, I.; and Lyu, M. R. 2012. Fused matrix factorization with geographical and social influence in location-based social networks. In *AAAI*, volume 12, 17–23.

Cho, E.; Myers, S. A.; and Leskovec, J. 2011. Friendship and mobility: user movement in location-based social networks. In *SIGKDD*, 1082–1090. ACM.

Gao, H.; Tang, J.; Hu, X.; and Liu, H. 2015. Content-aware point of interest recommendation on location-based social networks. In *AAAI*, 1721–1727.

Jamali, M., and Ester, M. 2009. Trustwalker: a random walk model for combining trust-based and item-based recommendation. In *SIGKDD*, 397–406. ACM.

Kalathil, D.; Nayyar, N.; and Jain, R. 2014. Decentralized learning for multiplayer multiarmed bandits. *IEEE Transactions on Information Theory* 60(4):2331–2345.

Koenigstein, N.; Dror, G.; and Koren, Y. 2011. Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In *Proceedings of the 5th ACM conference on Recommender Systems*, 165–172. ACM.

Koren, Y.; Bell, R.; Volinsky, C.; et al. 2009. Matrix factorization techniques for recommender systems. *Computer* 42(8):30–37.

Leng, C.; Wu, J.; Cheng, J.; Zhang, X.; and Lu, H. 2015. Hashing for distributed data. In *ICML*, 1642–1650.

Liao, Y.; Geurts, P.; and Leduc, G. 2010. Network distance prediction based on decentralized matrix factorization. *Proceedings of the 2010 International Conference on Research in Networking* 15–26.

Ling, Q.; Xu, Y.; Yin, W.; and Wen, Z. 2012. Decentralized low-rank matrix completion. In *ICASSP*, 2925–2928. IEEE.

McMahan, H. B.; Moore, E.; Ramage, D.; Hampson, S.; et al. 2016. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*.

Mnih, A., and Salakhutdinov, R. 2007. Probabilistic matrix factorization. In *NIPS*, 1257–1264.

Nedic, A., and Ozdaglar, A. 2009. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control* 54(1):48–61.

Rendle, S.; Freudenthaler, C.; Gantner, Z.; and Schmidt-Thieme, L. 2009. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, 452–461. AUAI Press.

Sarwar, B.; Karypis, G.; Konstan, J.; and Riedl, J. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW*, 285–295. ACM.

Shi, Y.; Larson, M.; and Hanjalic, A. 2010. List-wise learning to rank with matrix factorization for collaborative filtering. In *Proceedings of the 4th ACM conference on Recommender systems*, 269–272. ACM.

Shokri, R., and Shmatikov, V. 2015. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 1310–1321. ACM.

Su, X., and Khoshgoftaar, T. M. 2009. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence* 2009:4.

Yan, F.; Sundaram, S.; Vishwanathan, S.; and Qi, Y. 2013. Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties. *TKDE* 25(11):2483–2493.

Yang, S.-H.; Long, B.; Smola, A.; Sadagopan, N.; Zheng, Z.; and Zha, H. 2011. Like like alike: joint friendship and interest propagation in social networks. In *WWW*, 537–546. ACM.

Yang, D.; Zhang, D.; Yu, Z.; and Wang, Z. 2013. A sentiment-enhanced personalized location recommendation system. In *Proceedings of the 24th ACM Conference on Hypertext and Social Media*, 119–128. ACM.

Yang, D.; Zhang, D.; and Qu, B. 2016. Participatory cultural mapping based on collective behavior data in location-based social networks. *TIST* 7(3):30.

Ye, M.; Yin, P.; Lee, W.-C.; and Lee, D.-L. 2011. Exploiting geographical influence for collaborative point-of-interest recommendation. In *SIGIR*, 325–334. ACM.

Yun, H.; Yu, H.-F.; Hsieh, C.-J.; Vishwanathan, S.; and Dhillon, I. 2014. Nomad: Non-locking, stochastic multi-machine algorithm for asynchronous and decentralized matrix completion. *VLDB* 7(11):975–986.

Zhao, S.; King, I.; and Lyu, M. R. 2016. A survey of point-of-interest recommendation in location-based social networks. *arXiv preprint arXiv:1607.00647*.