Incorporating Contextual Information into Personalized Mobile Applications Recommendation

Ke Zhu (zabc_k@163.com)
Tianjin University of Technology https://orcid.org/0000-0003-3178-3493
Yingyuan Xiao
Tianjin University of Technology
Wenguang Zheng
Tianjin University of Technology
Xu Jiao
Tianjin Foreign Studies University
Chenchen Sun
Tianjin University of Technology
Ching-Hsien Hsu
Asia University

Research Article

Keywords:

Posted Date: April 28th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-393448/v1

Version of Record: A version of this preprint was published at Soft Computing on July 10th, 2021. See the published version at https://doi.org/10.1007/s00500-021-05988-8.

Noname manuscript No.
(will be inserted by the editor)

Incorporating Contextual Information into Personalized Mobile Applications Recommendation

Ke Zhu [1,2] · Yingyuan Xiao [1,2] · Wenguang Zheng[1,2] · Xu Jiao[3] · Chenchen Sun[1,2] · Ching-Hsien Hsu[4]

the date of receipt and acceptance should be inserted later

Abstract With the rise of the mobile internet, the number of mobile applications (apps) has shown explosive growth, which directly leads to the apps data overload. Currently, the recommender system has become the most effective method to solve the app data overload. App has the functional exclusiveness feature, which means the target users will not reuse apps with the same function in a certain spatiotemporal information. Most of the existing recommended methods for apps ignore the functional exclusiveness feature which makes it difficult to further improve the recommendation performance of the app recommendation. To solve this problem, we aim to improve the app recommendation performance, and propose a Personalized Context-aware Mobile App Recommendation Approach, called PCMARA. PCMARA comprehensively considers the user and app contextual information, which can mine the users app usage preference effectively. Specifically, (1) PCMARA explores the contextual characteristic of app, and constructs the app contextual factors for app which represent the function of app. (2) For the app functional exclusiveness problem, PCMARA leverages the app contextual factor to design a novel app similarity model, which enable to effectively eliminate this problem. (3) PCMARA considers the contextual information of users and apps to generates a recommendation list for target users based on the target users' current time and location. We applied the PCMARA to a real-world dataset and conducted a large-scale recommendation effect experiment. The experimental results show that the recommendation effect of PCMARA is satisfactory.

Keywords App recommendation, Gaussian mixture model, Item-based collaborative filter, Density clustering

1 Introduction

With the rise of the mobile Internet and the increase in the number of mobile devices, the number of mobile applications (apps) has expanded dramatically. The huge number of apps makes it difficult for people to find the apps that fit their preferences quickly and easily, causing app data overload problems. In recent years, as the most successful method for solving the app data overload problem, the recommender system (RS) has been a research hotspot. The RS recommends user apps that meet their preferences, which not only helps the user to find and use apps quickly and easily but also helps app developers quickly promote their products and provide quality services to their users.

Unlike the items in the traditional recommendation domain, such as film [28], news [27], POI [14] and so on, apps have functional exclusiveness features, i.e., if a user has downloaded an app with a certain function to meet his/her specific needs in a certain contextual

✉Yingyuan Xiao, Wenguang Zheng
E-mail: yyxiao@tjut.edu.cn, wenguangz@tjut.edu.cn

1
Engineering Research Center of Learning-Based Intelligent System, Ministry of Education, Tianjin, China.

2
Tianjin Key Laboratory of Intelligence Computing and Novel Software Technology, Tianjin University of Technology, Tianjin, China.

3
College of General Education, Tianjin Foreign Studies University, Tianjin, China.

4
Department of Computer Science and Information Engineering, Asia University, Taichung, Taiwan.

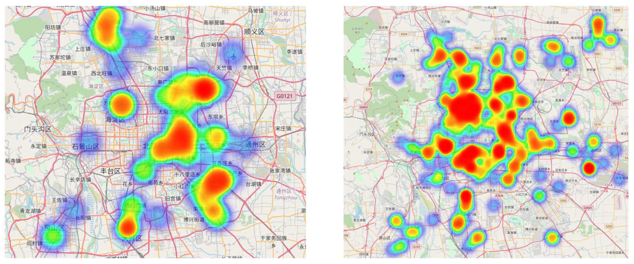


Fig. 1 Two App usage location heat maps of a travel app and an electronic payment app, Monday from 8:00 to 11:00, at Beijing city.

condition, then the user is likely not interested in other apps with similar functions. The functional exclusiveness features of the app makes the item similarity calculation methods in the traditional recommender system invalid for apps. We conjecture that the ignorance of app functional exclusiveness in the process of app recommendation model construction is the reason that the recommendation effect is difficult to further improves. Therefore, we explore the app functional exclusiveness features, and in anticipation of improveing the performance of app recommendation in view of incorporating the contextual information.

We observed the user-app usage log file in the real world and visually analyzed the app usage location data in the form of a heat map. The heat map shows an app usage geographic distribution in a specific time slot and a specific geographical area. The geographical distribution is made up of density contours. Different density contours are represented by different colors.

Based on the observation of users' use of app data in the real world, we find that within a certain geographic area and time slots, the usage location of the apps exhibits significant spatiotemporal aggregation phenomena. For example, Figure 1 shows the usage location of two apps under a specific time conditions, and the usage location of both apps show specific spatiotemporal aggregation shapes. Moreover, some apps with different functions have overlapping phenomena of spatiotemporal aggregation. We obtain the following conclusions by observation:

– The phenomenon of app spatiotemporal aggregation reflects the strong correlation between the function of the app and the contextual information, i.e., the app with a certain function is suitable for the context that meets this function. For example, users use video apps to watch videos at home during non-working hours but rarely use the video apps during work. This phenomenon occurs because the function of video apps is mainly entertainment, which is suitable for a leisure environment. This phenomenon provides a new idea for us to explore the context characteristics of apps.
– The phenomenon of the spatiotemporal aggregation overlapping between different functional apps also reflects the internal functional complementarity between apps. For example, users prefer to take photos with a camera app at a tourist attraction and use the retouching tool app to beautify the photos, and then, they share the photos on the social app. The phenomenon of jointly using multiple app functions to meet users' needs reflects the strong functional complementarity of apps in the contextual conditions.

In this paper, the spatiotemporal information is the contextual information. We named the phenomenon of apps' spatiotemporal aggregation the app contextual factor which is related to the app function. We model the function of app by using the app contextual factor to integrate the app with the contextual information. So that, the app features can be expressed more abundantly and accurately.

Based on the aforementioned conclusions, we can regard the functional complementarity of apps under the contextual conditions as a special kind of app similarity. Because we use the app contextual factor to model the function of apps, and the functional complementarity of apps is the similarity of apps in this paper, we can leverage the app contextual factor to build a novel app similarity model, which can makes up for the deficiency of traditional similarity calculation methods that do not consider app functional exclusiveness. Based on this consideration, we can reveal users' preferences for similar app functions in specific contextual information to explore users' app preferences more accurately.

In summary, to improve the performance of app recommendation, we proposed a personalized context-aware mobile application recommendation approach, called PCMARA. PCMARA uses the app contextual factor to represent the spatiotemporal characteristics of apps and uses the app contextual factor to calculate the similarity of apps. Then, PCMARA leverages a user-app-context tensor model to mine target users' app usage preferences. Finally, PCMARA combines the target user's current context with the similarity of the apps and uses the item-based collaborative filtering technique to generate the app recommendation list.

The main contributions of this work are summarized as follows:

1. PCMARA effectively mines the geographical distribution characteristics of an app under spatiotemporal conditions and innovatively proposes a method for constructing an app contextual factor.

2. PCMARA builds a novel app similarity model based on the app contextual factor. The app similarity model combines the spatiotemporal characteristics of the app and leverages the app functional complementarity to efficiently overcome the app functional exclusiveness.
3. Based on the proposed app similarity model and the target users' current contextual information, PCMARA uses the item-based collaborative filtering idea to generate an app recommendation list for target users, which meets their app usage preferences.
4. Based on a real-world dataset of users use apps, we verified that the recommendation performance of PCMARA is superior to the benchmark methods through large-scale experiments.

In the rest of the paper, we first present the related work about app recommendations in Section 2. Then, we define the problem and symbols in Section 3. In addition, in Section 4, we introduce the PCMARA and describe the inference method and the recommendation approach. To test the proposed model, we conduct experiments for evaluation in Section 5. Finally, we present a discussion and conclusion in Section 6.

2 Related Work

The main idea of an app recommendation method is based on collaborative filtering recommendation, i.e., to find similar users of the target user and recommend the apps of the similar users to the target user. For example, [26] constructed a user-app feature matrix, used the matrix factorization method to obtain the app factor, and then constructed an app similarity model to recommend apps for target users. [22] analyzed the sentence structure, semantics and sentiment of the user's comment information on the app so that users can be classified. [40] defines the concept of a location block and uses the similarity of the location block when users use the app to generate a recommendation list for users. [34] The location information of the user is employed to use the app to find similar users to the target user, thereby generating a recommendation list. [12] used user behavior logs to calculate user similarity. In addition, [1,5,8,10,16,20,25,30,31,37,39] used the app's own characteristics, such as app review text, app version evolution and other information, to construct an app similarity model and generate a recommendation list for users.

In the process of constructing the similarity model, the user or app vector is always extremely sparse, which causes difficulty in constructing the similarity model. Therefore, to solve the problem of data sparseness, the model-based recommendation method has been proposed. Model-based recommendation usually uses additional information generated during the user's interaction with the app, for example, the user's time information, location information, comment information, etc., to build a recommendation model. For example, [4] proposed a matrix decomposition app recommendation method based on app popularity to counter the sparse user-app interaction information. [18,3,32] propose app recommendation methods that use app version information. [33] proposes to mine the associations between user interactions and contexts captured by mobile devices, or behavior patterns for short, from context logs to characterize the habits of mobile users. [2] presents a framework that allows for usage-centric evaluation considering different stages of application engagement. [35] proposes a new topic model-based similarity and recommendation algorithm. [19] developed a structural user choice model to learn fine-grained user preferences by exploiting the hierarchical taxonomy of apps as well as the competitive relationships among apps. [11] proposes a novel probabilistic model named the goal-oriented exploratory model (GEM), integrating exploratory behavior identification with personalized item recommendation. [41] proposes the GTRM, a recommendation model that builds the top-N app list by optimizing the metric group-oriented mean average precision.

In addition, there are many works that have explored the contextual information when users interact with apps, such as the influence of time and location on the user's preference for app selection[17]. For example, [15] proposes the Djinn model, a context-aware collaborative filtering algorithm for implicit feedback data that is based on tensor factorization. [9] proposes a probabilistic method to recommend apps appropriate to the current time and location of a user. The study in [21] gives rise to the context-aware recommendation of apps. [6] presents a context-aware recommender system for mobile applications that produces recommendations from the first use. [38] leverages the app usage log file to build a personalized context-aware recommender system. [24] describes a framework that facilitates the development of context-aware recommendation systems for mobile environments. [35] proposes a broad learning approach for context-aware app recommendation with tensor analysis. [29] proposed a neural approach that learns the embeddings of both users and apps and then predicts a users preference for a given app.

3 Preliminaries

In this section, we give the formal description of the dataset and the symbols used in the PCMARA.

3.1 Problem definition

The goal of PCMARA is to build an app similarity model based on the historical location and time information of target users uses apps, and leverage the current time and location information of target users to mine app usage preference of target users and generate an app recommendation list.

Through the above analysis and conclusion, to recommend the apps for users under the context condition, we need to solve the following problems: (1) How to define the app contextual factor. (2) How to measure the similarity between apps and construct the app similarity model. (3) How to combine the app similarity model with the user's current context to generate a recommendation list for target users.

– For the first problem, (i) since the distribution of the app usage location has the spatial and temporal aggregation feature, the density of the location points in the distribution conforms to the Gaussian distribution, so we use the Gaussian distribution to describe the app. (ii) The app usage location distribution has multiple Gaussian density peaks. These Gaussian density peaks constitute the location distribution of the app. Therefore, we use a Gaussian mixture model to describe the app contextual factor. In Section 4, we explain the app contextual factor construction method in detail.
– For the second problem, after we obtain all app Gaussian mixture models, we simplify each app's Gaussian mixture model into joint confidence ellipses that are located in the spatial and temporal environment. For the calculation of the similarity of any two apps, we calculate the intersection area of the joint confidence ellipses of the two apps in the same context and use the Jaccard similarity to obtain the app similarity model.
– For the third problem, (i) we first find apps similar to the apps installed by the target user according to the app similarity model to form an initial app recommendation list by using the item-based collaborating filter idea. (ii) Then, we look for the apps that match the target user's current context information from the initial recommendation list to form the final app recommendation list.

To address the aforementioned challenges, we will give the specific implementation method of PCMARA in the next section.

Table 1 Users' app usage log files

App_id	User_id	Time stamp	Longitude	Latitude
1778	3098	3487	115.05	30.25
2358	3098	3601	115.05	30.26
2147	3098	3654	114.96	30.39
1922	2107	3682	114.94	30.42
2147	3098	3689	114.94	30.42
1922	2107	3702	114.94	30.42

3.2 Data description

We use a real-world user app usage log file to model the PCMARA. The log file is composed of App_id, User_id, Time stamp, Longitude and Latitude. Table 1 shows the data structure of the log file.

3.3 Symbols

to facilitate the statement of the content of this paper, we give the notations and descriptions of the symbols used in PCMARA, as shown in Table 2.

Table 2 The descriptions of symbols used in this article

Symbols	Descriptions
u_m	The m-th user
a_n	The n-th mobile app
c_k	The k-th context point
U	The set of target users
A	The set of mobile apps
T	The set of app usage times
E_n	The app contextual factor of a_n
e_i^n	The i-th app confidence ellipse of a_n
L_m	The set of usage locations of app a_m
α_k^m	The proportion of the k-th cluster
μ_k^m	The center point of the k-th cluster
σ_k^m	The shape parameter of the k-th cluster

4 Proposed work

In this section, we will describe the PCMARA in detail. First, we introduce the system framework of PCMARA and then explain the key methods and algorithms used for each model in PCMARA.

4.1 Framework

In this paper, whether an app meets the target user's preferences depends on the following two aspects: (1) whether the function of the app matches the target user's current contextual information, and (2) whether the function of the app with the target user installed

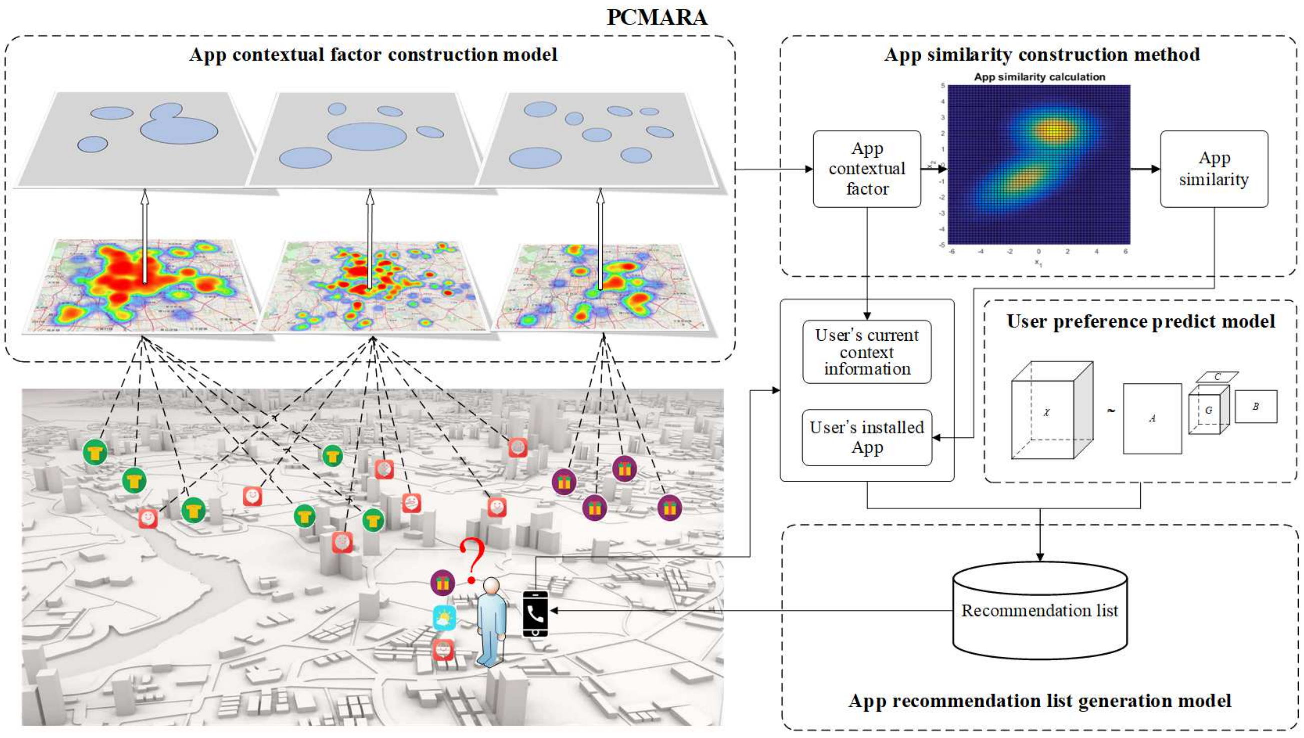

PCMARA
App contextual factor construction model
App similarity construction method
App similarity calculation
App contextual factor
App similarity
User's current context information
User's installed App
User preference predict model
Recommendation list
App recommendation list generation model

Fig. 2 The architecture of PCMARA.

app's function is complementary. Therefore, PCMARA uses four major modules to bridge the above issues: the app contextual factor construction model, the app similarity model, user preferences to predict the model and the app recommendation list generation model. We illustrate the framework of PCMARA, as shown in Figure 2.

1. The app contextual factor construction model uses the usage time and geographic location information to build an app context Gaussian mixture model (ACGMM) that matches the context characteristics of the app and uses the ACGMM's parameters to represent the app contextual factor.
2. The app similarity construction method maps the app contextual factor to the influence range composed of joint confidence ellipses in a limited geographic area and uses the intersection area of the joint confidence ellipses to calculate the similarity between two apps.
3. The user preference prediction model leverages the tensor model constructed by the target users' historical contextual information and app usage information to model the target users' app usage preferences.
4. The app recommendation list generation model generates an app recommendation list for the target user based on the current contextual information and the installed apps of the target user.

The specific details of the four modules are as follows.

4.2 App contextual factor construction model

To represent the functional characteristics of the app, PCMARA uses the contextual information of the app to design the app contextual factor construction method. The detailed model construction steps are as follows:

1. We first divide one day into 6 time slots according to the habits of users using apps (0:00-7:00, 7:00-11:30, 11:30-13:30, 13:30-17:00, 17:00-19:00, and 19:00-23:59).
2. For an app in a certain time slot, we extract this app usage location point to form spatiotemporal scattered points of the app usage location.
3. We construct a Gaussian mixture model for the spatiotemporal scattered points of the app and then use a joint confidence ellipse to map the Gaussian mixture model to a plane. The joint confidence ellipse under this contextual condition is the app contextual factor of target app.

For step 3, we use the following three steps to build the app contextual factor.

4.2.1 Find the number of density peaks

First, PCMARA needs to determine the number of density peaks of a target app to provide parameter support for subsequent steps.

For a target app usage location point set, we first determine the number of density peaks. In this paper, the method to quickly find the number of cluster center points is employed [36], [23]. The detailed steps of the method are shown as follows.

1. Given a usage location set of a target app a_m in a certain time slot: $L_m=\{l_1^m, l_2^m, ..., l_n^m\}$, we first calculate the coordinates distance between l_i^m and l_j^m for all locations, $d_{ij}^m=\sqrt{(l_i^m - l_j^m)^2}$;
2. We sort all distances d_{ij}^m in increasing order to determine the 2% d_{ij}^m and define it as d_c^m;
3. Then, we calculate the local density ρ_i^m of each point l_i^m, where the specific calculation method of ρ_i^m is given as Equation 1:

$$\rho_i^m = \sum_{j \in I_S} exp(-\frac{d_{ij}^m}{d_c^m})^2 \tag{1}$$

   where $I_S=\{1, 2, ...n\}$ is the indicator set;
4. Calculate the distance δ_i^m of each point l_i^m, where the detailed calculation is given as Equation 2;

$$\delta_i^m = \begin{cases} \min_{j \in I_S^i} d_{ij}^m, I_S^i \neq \emptyset \\ \max_{j \in I_S} d_{ij}^m, I_S^i = \emptyset \end{cases} I_S^i = \{k \in I_S : \rho_k^m > \rho_i^m\} \tag{2}$$

5. For all of the app usage location points $l_i^m(\rho_i^m, \delta_i^m)$, $i \in I_S$, calculate the $\gamma_i^m = \rho_i^m \cdot \delta_i^m$ and sort all γ_i^m in descending order. Select K^m as the number of cluster centers of a_m' usage location point set, where the K^m-th location point is the point before the γ_i^m value sharply decreases. K^m is the number of the Gaussian components of the target app a_m.

4.2.2 Construct the ACGMM

Then, we use the EM algorithm [7] to construct the Gaussian mixture model of target app a_m's usage location point set L_m and record the model parameters. We use the following equations to present the Gaussian mixture model:

$$p(L_m|\Theta^m) = \sum_{k=1}^{K^m} \alpha_k^m \cdot \frac{1}{2\pi\sqrt{|\sigma_k^m|}} exp(M) \tag{6}$$

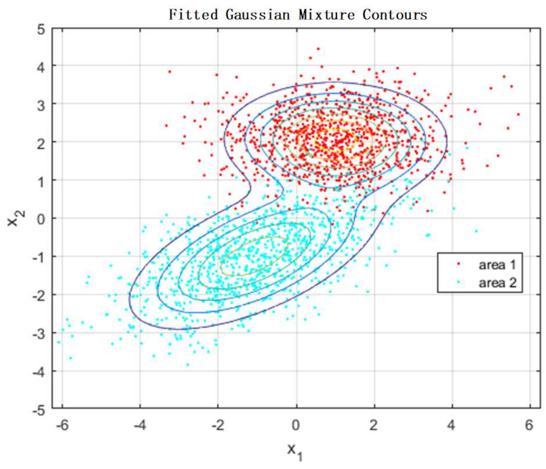


Fig. 3 Gaussian mixture contours

```
Algorithm 1: ACGMM construction
   Input: Target app $a_m$'s usage location in a certain
          time slot: $L^m=\{l_1^m, l_2^m, ..., l_n^m\}$; the number of
          cluster centers: $K^m$; maximum number of
          iterations $J$.
   Output: The parameters of the model $\Theta^m$
 1 Choose a random initial setting for the parameters
   $\Theta^m$;
 2 Initialization of joint distribution $P(l_i, z_i;\ \Theta^m)$, and
   likelihood function $P(z_i|l_i;\ \Theta^m)$;
 3 for $j \leftarrow 1$ to $J$ do
 4     Calculate the conditional probability expectations
       for the joint distribution:
 5
       $Q_i(z_i) = P(z_i|l_i;\ \Theta_j^m)$                                (3)
 6
       $M(\Theta_j^m, \Theta_{j+1}^m) = \sum_{i=1}^{n}\sum_{z_i} Q_i(z_i) log P(l_i,\ z_i; \Theta_j^m)$   (4)
 7     Maximize $M(\Theta_j^m, \Theta_{j+1}^m)$ to obtain the $\Theta_{j+1}^m$:
 8
       $\Theta_{j+1}^m = \arg\max_{\Theta} m(\Theta_j^m, \Theta_{j+1}^m)$                 (5)
 9     if $\Theta_{j+1}^m$ has converged then
10         the value of $\Theta^m$ is $\Theta_{j+1}^m$;
11     else
12         return to E-step to iterate.
13     end
14 end
```

$$M = \frac{(L_m - \mu_k^m)^T \sigma_k^{m-1} (L_m - \mu_k^m)}{-2} \tag{7}$$

In Equations 6 and 7, L_m is the target app a_m's usage location point set, K^m is the number of Gaussian components, which is obtained in Section 4.2.1; α_k^m is the proportion of the k-th cluster in the point set, and $\sum_{k=1}^{K^m} \alpha_k^m = 1$. μ_k^m is the cluster center position of the k-th cluster, and σ_k^m determines the shape of the k-th cluster. In this section, Algorithm 1 to train the parameters Θ^m ($\theta_k^m \in \Theta^m, \theta_k^m = \{\alpha_k^m, \mu_k^m, \sigma_k^m\}$) of the Gaussian model is employed.

In Algorithm 1, z_i represents the latent variables of l_i. $Q_i(z_i)$ is the estimate of the z_i, and $\sum_{z_i} Q_i(z_i)$=1. We use Figure 3 to illustrate the ACGMM. In Figure 3, the scatter points are the geographical location of an app in a certain time slot, and the color of the point refers to the location of the point in different clusters. We leverage the above method to get the context Gaussian mixture model of the app.

4.2.3 Construct app contextual factor

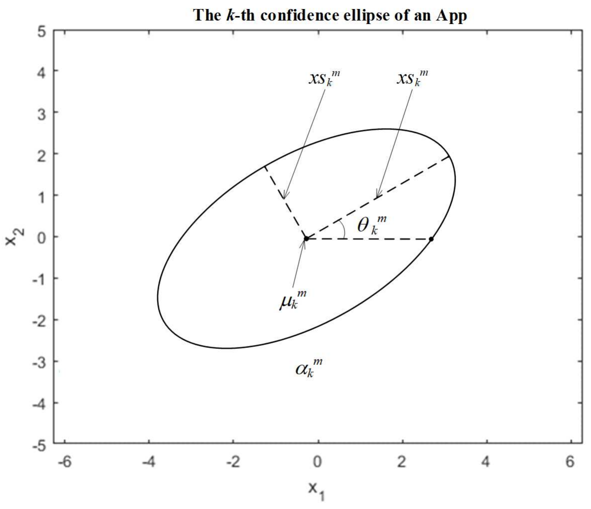


Fig. 4 The K-th confidence ellipse of an App

Finally, we use the joint confidence ellipse E_m to simplify the ACGMM of the target app a_m and map the joint confidence ellipse to the plane. The joint confidence ellipse of a target app is the app contextual factor of this target app.

For an ACGMM of a target app a_m, we use a set $E_m : \{e_1^m, e_2^m, \cdots, e_{K^m}^m\}$ to simplify the ACGMM of a_m, where e_k^m ($e_k^m \in E_m$) is the confidence ellipse of the k-th density peak of the target ACGMM that was obtained in Section 4.2.2. The specific expression of e_k^m is shown in Equation 8. Algorithm 2 describes the construction method of e_k^m in detail.

$$e_k^m = \begin{cases} x_k^m = xl_k^m \cos\theta^k + x^{\mu_k^m} \\ y_k^m = xs_k^m \sin\theta^k + y^{\mu_k^m} \end{cases}, \theta_k^m \in (0, 2\pi) \tag{8}$$

For the convenience of display, we use Figure 4 to illustrate an example regarding the k-th confidence ellipse e_k^m of a_m. In Figure 4, μ_k^m represents the center point of the k-th ellipse; xl_k^m and xs_k^m are the long and short axis of the k-th ellipse, respectively; $x^{\mu_k^m}$ and $y^{\mu_k^m}$ are the coordinates of the point μ_k^m on the X-axis and Y-axis, respectively; and θ_k^m is the angle of the long axis and the line that is composed of a point in the ellipse and the μ_k^m.

Algorithm 2: App contextual factor construction

```
Input: The target app $a_m$'s ACGMM; the app usage
       location in a time slot $L^m$; all of the center
       points of each $a_m$'s Gaussian component $\mu_k^m$,
       the number of Gaussian components $K$
Output: The joint confidence ellipse of the app:
       $E_m : \{e_1^m, e_2^m, ..., e_k^m\}$
1  for the target ACGMM do
2      extract the point set $EL_k^m$ ($EL_k^m \subseteq L^m$) of each
       Gaussian component;
3      scatter $EL_k^m$ on a plane with longitude as the
       $X$-axis and latitude as the $Y$-axis,
       $\sum_{k=1}^K EL_k = L^m$;
4      foreach $EL_k^m$ do
5          calculate eigenvectors and eigenvalues;
6          $\lambda_{m,k}^{max} \leftarrow$ the largest eigenvalue;
7          $\lambda_{m,k}^{min} \leftarrow$ the smallest eigenvalue;
8          select the 95% confidence ellipse as the target
           ellipse to obtain $a_k^m$ and $b_k^m$;
9          $xl_k^m \leftarrow \sqrt{0.5991 \cdot \lambda_{m,k}^{max}}$;
10         $xs_k^m \leftarrow \sqrt{0.5991 \cdot \lambda_{m,k}^{min}}$;
11         the $k$-th Gaussian component $e_k^m$'s confidence
           ellipse of $a_m$ is obtained by the $\mu_k^m$, $xl_k^m$ and
           $xs_k^m$, and $e_k^m$ is represented as Equation 8.
12     end
13 end
```

4.3 App similarity construction method

The app similarity construction method is an important part of the PCMARA. We leverage the app contextual factor in a certain time slot to construct the app similarity model. The detailed app similarity model construction steps are given as Algorithm 3.

Algorithm 3 calculate the ratio of the intersection area and its union area between two app contextual factors and records the ratio as the similarity between the two apps. Figure 5 illustrates the similarity between two apps. The ratio of the area where the black horizontal line is located to the sum of the confidence ellipse areas of the two apps is the similarity of the two apps.

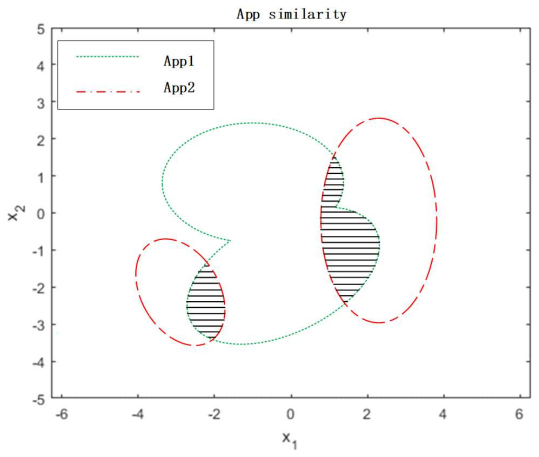


Fig. 5 The similarity between two apps

Algorithm 3: App similarity construction

```
   Input: The set of all target app contextual factors:
          $SE:\{E_1, E_2, \cdots, E_m\}$; a set of random points
          $P:\{p_1,\ p_2, \cdots, p_n\}$
   Output: The similarity between any two apps.
 1 Initialize $n$;
 2 the number of points in the intersection area of two
   apps: $IA = 0$;
 3 the number of points in the total area of two apps:
   $TA = 0$;
 4 for $E_i, E_j \in SE$ do
 5     Randomly scatter the points in the set of $P$ on the
       finite plane containing $E_i$ and $E_j$;
 6     for $p_k \in P$ do
 7         if $p_k \in E_i \cap E_j$ then
 8             $IA \leftarrow IA + 1$;
 9         else
10             if $p_k \in E_i \cup E_j$ then
11                 $TA \leftarrow TA + 1$;
12             end
13         end
14     end
15     calculate the similarity $s_{ij}$ between $E_i$ and $E_j$
       with the following equation: $s_{ij} = \frac{IA}{IA+TA}$
16 end
```

4.4 The user preference prediction model

In PCMARA, we need to predict the probability that the user will use the app in the current time. To solve the above problem, we decompose the 3-dimensional tensor composed of the user-app-time to obtain the user's app usage preference probability in the time slot. Our approach uses the method in [13] for tensor decomposition.

Specifically, our initial tensor is composed of time slots, users, and apps. The elements in the tensor are the frequency of users using a certain app in a certain time slot. The time slot for constructing the tensor is the same as the time slot for constructing the app contextual factor model. We use Figure 6 to illustrate the principle of this process. We can extract the latent features of each user, time slot and preference by decomposing the tensor χ into three matrices A, B, and C and a core tensor G, as shown in Equation 9.

$$\chi \approx G \times_1 A \times_2 B \times_3 C = \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} g_{pqr} a_p \circ b_q \circ c_r \quad (9)$$

Here, $A \in \mathbb{R}^{I\times P}$, $B \in \mathbb{R}^{J\times Q}$, and $C \in \mathbb{R}^{K\times R}$ are the factor matrices. $G \in \mathbb{R}^{P\times Q\times R}$ is the core tensor.

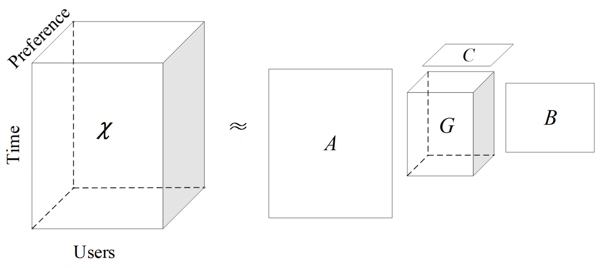


Fig. 6 User preference tensor factorization

4.5 App recommendation list generation model

The app recommendation list generation model recommends to the target users apps that meet their preference and current contextual information. We know the target user's current contextual information and the installed apps, and we use the following steps to obtain the app recommendation list.

1. According to the app similarity model, we look for apps that are similar to the target user's installed apps according to the contextual factor to form a functionally sensitive candidate app set that conforms to the target user's functional preferences for the apps. We use λ as the criterion for whether two apps are similar. If the similarity between an app and the target user's installed apps is greater than λ, then the app is selected into the functionally sensitive candidate app set. In this paper, λ is set to 0.6, which is determined experimentally in Section 5.5.5.
2. Then, we find the apps from the functionally sensitive candidate app set that match the target user's preferences, according to the preference prediction model of the target user in this time slot to form the time-sensitive candidate app set. This app set conforms to the target user's time preferences for the apps.

Algorithm 4: App ranking

```
Input: The target user $u_i$'s current contextual
       information, current time slot $t_{u_i} \in T$ and
       current location $l_{u_i} \in L$; the final candidate
       app set $A$; the app contextual factor of all apps
       in $A$.
Output: Ranking of apps in the recommendation list
1 for $a_m \in A$ in the recommendation list do
2     extract $a_m$'s app contextual factor, which is in the
      target user $u_i$'s current time slot $t_{u_i}$;
3     for the selected app contextual factor do
4         extract the $k$-th confidence ellipse that $l_{u_i}$ is
          located in;
5         calculate the recommendation rate $R_{im}$ of $a_m$
          recommended to $u_i$ with Equation 10;
6     end
7 end
8 Sort all recommendation rates of target apps in
  descending order.
```

3. We leverage the target user's current location contextual information to match the apps in the time sensitive candidate app set to form the final candidate app set. In this process, if the app contextual factor of the target app covers the target user's current contextual information, the app is selected as the final app to be recommended. The final candidate app set conforms to the target user's spatiotemporal and functional preferences for the apps.
4. Then, we rank the apps in the final candidate app set to form the app recommendation list. We use Algorithm 4 to rank apps in the final candidate app set. In Algorithm 4, the recommendation rate R_{im} is calculated as Equation 4.

$$R_{im} = \frac{\alpha_k^m \sum_{a_n \in AS} S_{mn}}{d_{u_i,a_m}} \tag{10}$$

where α_k^m is the proportion of the k-th confidence ellipse of a_m; AS is the set of installed apps in u_i' mobile device; S_{mn} is the similarity between a_m and a_n; d_{u_i,a_m} is the distance between u_i's current location l_{u_i} and the a_m's k-th confidence ellipse's center point, and $d_{u_i,a_m} = \sqrt{(\mu_k - l_{u_i})^2}$.
5. Finally, we chose the top-N apps to form the recommendation list.

5 Experiments

In this section, we perform experiments on the recommendation performance of PCMARA. Particularly, we will focus on the following issues: (1) Compared with benchmark methods, what are the advantages of PCMARA in the recommendation effect? (2) What are the advantages of the proposed app contextual factor and app ranking algorithm for improving app recommendation?

5.1 Experimental dataset

Our experimental data are the users' app usage log files from real-world scenarios in three cities, Beijing, Shanghai and Guangzhou. The datasets contain a total of 5860 users, 701 apps, and 137,047 user-app usage log records. Each app is used at least 10 times, and each user has at least 10 app usage logs. Each user has an average of 23.39 logs. The detailed information of the dataset is shown in Table 3.

Table 3 The data structure of the dataset

Dataset	Beijing	Shanghai	Guangzhou
Duration	168 Hours	168 Hours	168 Hours
App	637	667	629
User	1972	1921	1967
Record	45,707	46,973	44,367
Location	20,582	21,693	20,117

5.2 Benchmark methods

We compare PCMARA with the following recommendation methods:

– User-based collaborative filtering (**UserCF**). UserCF is a user vector similarity oriented collaborative filtering approach, where user similarity is calculated based on the users' rating data. We adapt UserCF to our problem by using frequency information of users use apps to mine user preferences and generate the recommendation list. We leverage the frequency information to find the users similar to the target user and recommend apps that similar users have used and that the target user has not used to target users.
– Item-based collaborative filtering (**ItemCF**). ItemCF is an app vector similarity oriented collaborative filtering approach. We adapt ItemCF to our problem with the following method. First, we matricize the dataset according to the frequency of users using apps to form a user-app usage matrix. Then, we extract the app vectors from the matrix and use the cosine similarity coefficient to build the app similarity model. Finally, we leverage the app similarity model to generate an app recommendation list for target users.

– Matrix factorization (**MF**). The MF recommendation approach is a classic model-based method in the recommendation domain. We input the user-app usage matrix and optimize the following equation to obtain a nonempty matrix that contains the same information as the original matrix.

$$\min_{u_i,a_j} \sum_{i=1}^{n}\sum_{j=1}^{m}(r_{ij}-u_i a_j)^2 + \frac{\lambda_u}{2}\sum_{i=1}^{n}||u_i||^2 + \frac{\lambda_v}{2}\sum_{j=1}^{m}||a_j||^2 \tag{11}$$

In Equation 11, r_{ij} is the actual usage frequency information of app j by the user i, and u_i, a_j are the latent vectors of the target matrix. The elements in the target matrix are the preference rating of target users for the apps. We obtain an app recommendation list according to the target matrix.

– Singular Value Decomposition (**SVD**). SVD approach is a matrix factorization-based (MF-based) recommendation method. The principle of SVD is the same as MF. Different from the MF method, to reduce the disturbance of matrix filling, SVD adds feedback bias for the latent vectors of users and items so that the model is more in line with the recommended reality. The aim of SVD is to compute the factors for the users and items. We input the user-app frequency rating matrix and optimize the following equation to obtain a nonempty matrix.

$$\min_{b_i,b_u,q_i,p_u} \sum_{(u,i)\in K}(r_{ui}-\mu-b_i-b_u-q_i\cdot p_u)^2 + \lambda(\sum_u(b_u^2+||p_u||^2)+\sum_u(b_i^2+||q_i||^2)) \tag{12}$$

In Equation 12, μ is the rating baseline, and b_u and b_i are the biases of the user u and app i, respectively. q_i and p_u are the latent vectors of the target users and apps of the target matrix, respectively. The actual rating of app j by the user i is the r_{ui}. The elements in the target matrix are the preference ratings of target users for the apps. We obtain an app recommendation list according to the target matrix.

– Tensor factorization (**TF**). The aim of the model is to compute the factors for the user $U^{n\times d}$, item $M^{r\times d}$ and context $C^{l\times d}$ matrices using historical usage data. the TF approach is also an MF-based recommendation method. The principle of TF is the same as MF. Similar to the above two methods, we use the frequency of a target user using an app according to certain contextual information to denote the app usage information of a target user and construct a user-app-context tensor. Then, we use the TF method to obtain a recommendation list. Here, the elements in the tensor are the app usage information of a user using an app at a certain time.

5.3 Evaluation measures

In our experiments, we split the experimental data into training data and test data. We use the training data to train PCMARA and use the apps that users actually use in the test dataset as the ground truth.

We recommend an app list for target users, which leads to two evaluation metrics: precision and recall. We recommend a top-N recommendation list to users, and the length of the recommendation list is N, so we choose precision@N and recall@N to measure the proposed method. However, precision and recall are two indicators with a relationship that when one falls, it leads to the other rising. To synthesize considering the two indicators, we use the F_α-measure to measure the recommendation quality.

We use TP to denote the intersection between the recommendation list and ground truth; we use N to denote the length of the recommendation list, and we use M to denote the length of the ground truth. The precision and recall are defined as follows:

$$Precision = \frac{TP}{N} \tag{13}$$

$$Recall = \frac{TP}{M} \tag{14}$$

$F_\alpha - measure$ balances the precision and recall, which is defined as:

$$F_\alpha - measure = (1+\alpha^2)\frac{Precision * Recall}{\alpha^2 Precision + Recall} \tag{15}$$

Here, we choose α=1, which indicates the recall and precision has the same importance.

The PCMARA uses an app ranking algorithm to generate an app recommendation list, so we use the MAP metrics to measure the effect of the proposed app ranking algorithm. The MAP (mean average precision) calculates the average recommendation precision of each recommended app in the recommendation list. If the value of the MAP metrics is higher, it indicates that the sorted recommendation list is similar to the app actually used by the target user. The MAP is calculated as follows:

$$MAP = \frac{\sum_{u=1}^{U} AveP(u)}{U} \quad (16)$$

$$AveP(u) = \frac{\sum_{n=1}^{N} (p(n) \times rel(n))}{number\ of\ relevance\ app} \quad (17)$$

In Equation 16, U is the target users set. $AveP(u)$ is the precision of the recommendation list for the target user u, and the calculation method is shown in Equation 17. In Equation 17, N is the length of the recommendation list for the target user u. $p(n)$ is the precision of the top-n recommended apps in the recommendation list. $rel(n)$ indicates whether the n-th recommended app is relevant to the app actually used by the target user. If the n-th recommended app is the same as the app actually used by the target user, $rel(n) = 1$; otherwise, the value of $rel(n)$ is 0. In our dataset, a user only uses one app in a spatiotemporal context, so the value of "number of relevant apps" in Equation 17 is 1. When the effect of the ranking algorithm is good, it will cause an increase in the value of the molecular part of Equation 17, which in turn leads to a higher value of MAP. Therefore, we can use the MAP metric to measure the ranking algorithm of the model.

5.4 Experiment set

5.4.1 Experiment 1: recommendation performance

We apply PCMARA and five benchmark methods to the dataset and generate a recommendation list. Then, we compare the recommendation effect on the three metrics of precision, recall and F-measure with different parameters.

5.4.2 Experiment 2: similarity model performance

PCMARA uses the app contextual factor to build a new app similarity model. Therefore, we use the method of controlling variables to verify the performance of the proposed app similarity model on improving the recommendation effect. We extract the app vectors from the user-app usage matrix to calculate the app similarity and leverage the traditional cosine similarity to build a traditional app similarity model. We replace the proposed app similarity model with the traditional app similarity model and name it PCMARA-S. We compare the recommendation effects of PCMARA and PCMARA-S to verify the performance of the proposed app similarity model.

5.4.3 Experiment 3: ranking algorithm performance

PCMARA uses a novel app ranking method to generate an app recommendation list. We use the MAP@N metric to compare the ranking results of the benchmark method with PCMARA to verify the improvement effect of the proposed ranking algorithm on recommendation.

5.4.4 Experiment 4: The performance of different time slots

PCMARA takes into account both the spatiotemporal context information of target users and the app contextual factor when the target users use the app. We compare the recommendation performance of PCMARA with the benchmark methods in different time periods to verify the effect of the app contextual factor on improving app recommendation performance.

5.4.5 Experiment 5: The influence of λ

PCMARA uses λ to determine whether two apps are similar. We need to verify the influence of λ on recommendation performance. We implement a comparison of the recommendation performance for precision@3 on datasets of three cities.

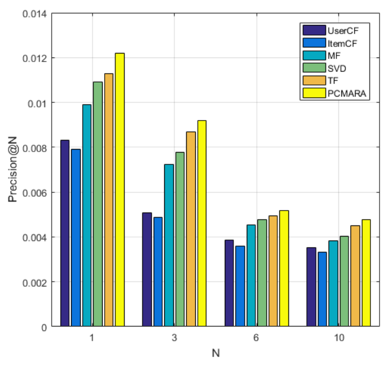
5.5 Performance comparisons

5.5.1 Results of experiment 1

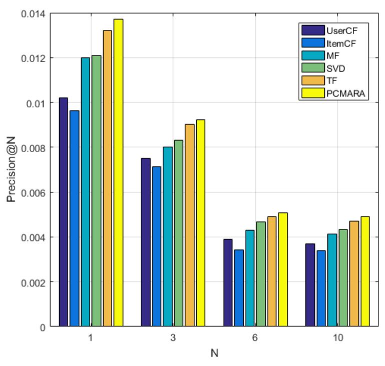
In this section, we present the results of experiment 1 with different parameter values in Figure 7.

We compared the precision@N, recall@N, and F-score of PCMARA and benchmark methods in three cities. The abscissa in the figure is the length of the recommendation list. Figure 7(a) - 7(c), Figure 7(d) - 7(f) and 7(g) - 7(i) show the precision, recall and F-measure of the PCMARA and benchmark methods' recommendation results in three cities, respectively.

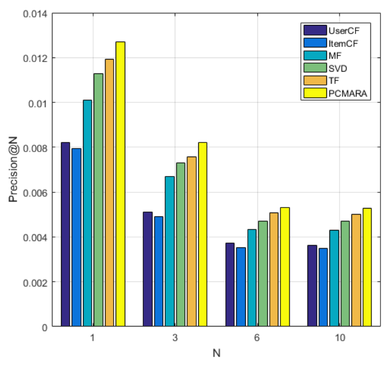
The experimental results show that the performance of UserCF and ItemCF is poor. This is because, first, the two CF-based methods use user-app interaction information to mine user preference information while ignoring the users' context information when using the app, which is an important factor that affects the user's app preference. Second, the matrix composed of users and apps is extremely sparse, which leads to the inability of the two models to accurately mine user app usage preferences. In addition, the performance of ItemCF is the worst of all the methods because there are other problems in ItemCF. In particular, because



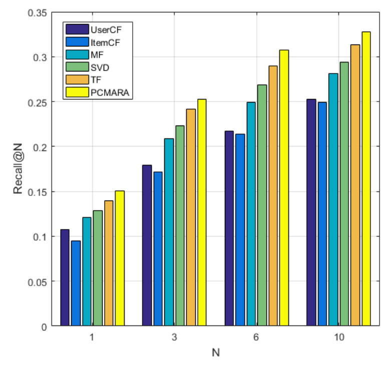
(a) Beijing



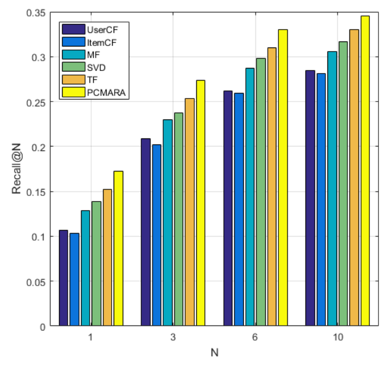
(b) Shanghai



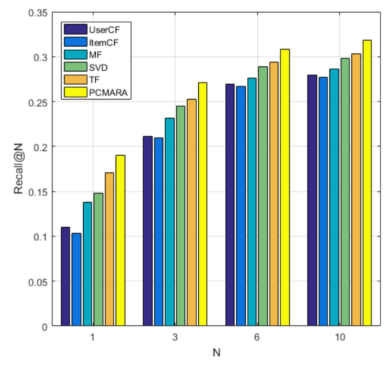
(c) Guangzhou



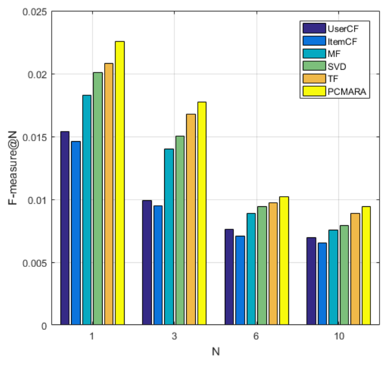
(d) Beijing



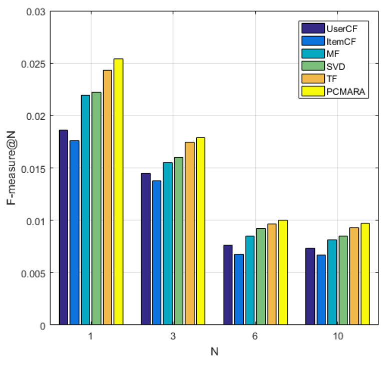
(e) Shanghai



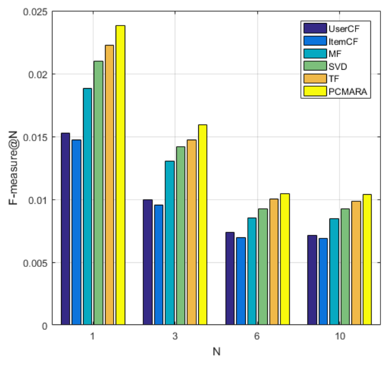
(f) Guangzhou



(g) Beijing



(h) Shanghai



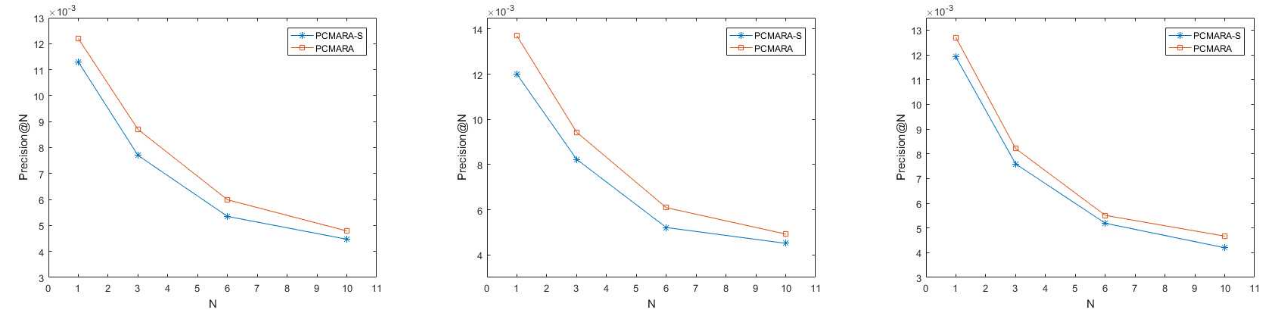
(i) Guangzhou

Fig. 7 The comparison of recommendation performance between PCMARA and benchmark methods

of the functional exclusivity of apps, only app vectors are used to build the app similarity model which leads to app recommendation lists with similar functions that will not be selected by users, resulting in a poor recommendation effect.

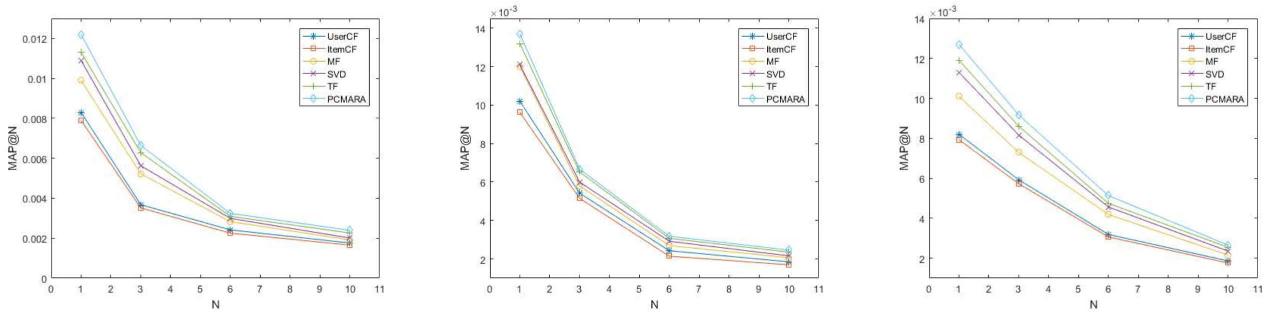
MF, SVD and TF all show relatively good recommendation performance. Among them, the common point of MF and SVD is that they use the idea of matrix factorization to supplement the user-app sparse matrix so that nonsparse vectors can more accurately reflect user preferences. The recommendation performance of TF is always better than that of MF and SVD. This is because TF builds user-app-time context information in the three-dimensional tensor model, which not only considers the interaction information between the user and the app but also considers the context in which the user uses the app. The information is integrated into the model building process, which effectively mines the user's app usage preferences under specific time conditions.

PCMARA always outperforms the benchmark method in terms of precision, recall and the comprehensive evaluation F-score metric. The main reason is that PCMARA incorporates the time and geographic informa-



(a) Beijing (b) Shanghai (c) Guangzhou

Fig. 8 The comparison of recommendation performance between PCMARA and PCMARA-S



(a) Beijing (b) Shanghai (c) Guangzhou

Fig. 9 The comparison of ranking performance between PCMARA and benchmark methods

tion of the users and apps when users are using the app into the construction of the user preference model, effectively mining the users app preferences. In addition, we designed a novel app similarity calculation method, which can effectively use app contextual factors to establish a similarity model. In the process of generating a recommendation list for the target user, PCMARA not only combines the user's current temporal and spatial context information but also considers the contextual characteristics of the app itself, which effectively overcomes the functional exclusivity problem of the app.

5.5.2 Results of experiment 2

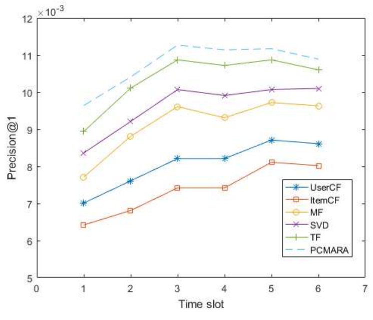
We present the results of experiment 2, as shown in Figure 8. We chose precision@N to measure the effect of the proposed similarity model and the baseline similarity model. As seen in Figure 8, the recommendation effect of PCMARA is superior to that of the PCMARA-S method in the datasets of three cities.

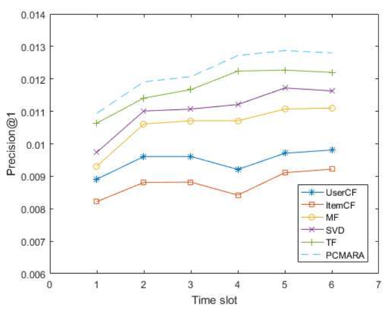
The reasons for the experimental results are as follows. First, the app contextual factor affects the app usage preferences of target users. The target users are more inclined to choose those apps that have appropriate app contextual factors. When constructing the app similarity model, the app contextual factor is incorporated in the model construction and will enable PCMARA to achieve better performance in mining the users' app usage preferences. Second, PCMARA leverages the app functional complementarity to design the similarity model. The recommended apps that have functional complementarity with the target users are more likely to meet the target users' app usage needs and preferences. Therefore, these apps are more favored by the target users.

5.5.3 Results of experiment 3

We present the results of experiment 3, as shown in Figure 9. We chose MAP@N to measure the performance of the proposed ranking algorithm and the benchmark methods. As seen in Figure 9, the recommendation effect of PCMARA using the app ranking algorithm is superior to that of the benchmark methods in the datasets of three cities.

The reasons for the experimental results are as follows. First, when designing the app ranking algorithm, we not only considered the influence of the target users' spatiotemporal context information when using the app on the users app usage preferences but also considered the influence of the coverage of the target app contextual factor on the target users' app usage preferences, which leads to PCMARA achieving better performance in mining users' app usage preferences. Second, the ap-


(a) Beijing


(b) Shanghai

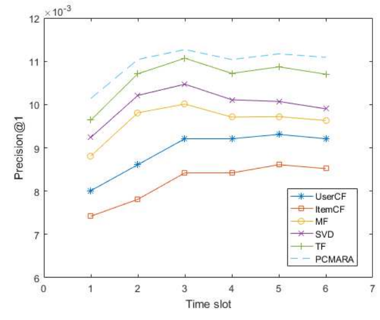

(c) Guangzhou

Fig. 10 The comparison of recommendation performance between PCMARA and benchmark methods in different time slots

p ranking algorithm of PCMARA uses Equation 10 to quantitatively calculate the functional complementarity relationship between the target app and the apps installed by the target user, which provides accurate user preference calculation criteria for the PCMARA model so that PCMARA can mine the users' app usage preferences more accurately.

5.5.4 Results of experiment 4

We present the results of experiment 4, as shown in Figure 10. We chose precision@N to measure the recommendation performance of PCMARA and benchmark methods in different time slots. During the model design process of PCMARA, a day is divided into 6 time slots, and the abscissa in Figure 10 represents different time slots. As seen in Figure 10, the recommendation effect of PCMARA is superior to that of the benchmark methods in the datasets of three cities. Moreover, the recommendation performance of PCMARA in the time range of 7:00-19:00 has a great advantage over the benchmark methods.

The reasons for the experimental results are as follows. The recommendation results of UserCF, ItemCF, MF and SVD are unacceptable. This is because these four methods do not consider the spatiotemporal context information of target users using apps when constructing the model, which makes these methods unable to effectively recommend apps for target users in different time conditions. TF is relatively good in recommendation performance because TF considers the time contextual information and enriches the information dimensions required for the model construction. PCMARA performs best on recommendation performance in different time slots. PCMARA takes into account the time, geographic information and app contextual factor when the target users use apps to provide a more personalized and reasonable app recommendation list for target users in different time conditions.

5.5.5 Results of experiment 5

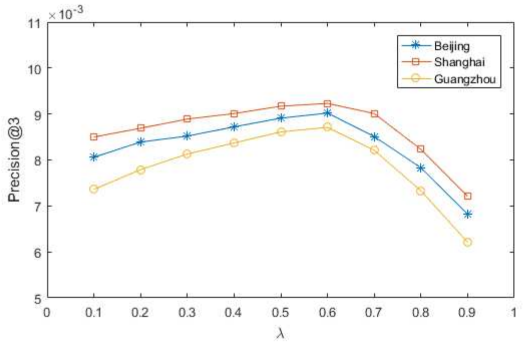


Fig. 11 The influence of parameter λ

In this section, we present the results of experiment 5 with the datasets of three cities to determine the parameter values of λ in the app recommendation list generation model of PCMARA. The abscissa of the Figure 11 is from 0.1 to 0.9, which means that the value of λ is from 0.1 to 0.9 in PCMARA in the three datasets. As seen from the result, the performance of the recommendation shows a trend of first increasing and then decreasing as λ increases. When the value of λ is 0.6, the recommendation performance of PCMARA is the best. Hence, we choose the value of λ as 0.6 for PCMARA.

6 Conclusion and discussion

This paper proposes a Personalized Context-aware Mobile App Recommendation Approach, called PCMARA. The novelty of PCMARA is that (1) it uses the app contextual factor to model the function of the target app, (2) it creatively constructs the app functional similarity model and (3) it provides an app recommendation list for target users that meets the preferences of target users' current contextual information. The experimental results show that PCMARA has excellent performance in app recommendations under contextual conditions.

One of the most outstanding features that differentiates our approach from existing works is that PCMARA incorporates the app contextual factor in the modeling. The app contextual factor reflects the app functional feature. The approach of considering the app contextual factor can better help the app recommender system mine the users' app usage preferences to improve the performance of the recommender system.

In our future work, we aim to optimize the app contextual factor construction method by deploying the method on a distributed computing platform to improve the speed of the approach. The proposed approach considers the contextual information of users and apps, and the sequential information of users using apps is also a feature that affects the app recommendation performance. Therefore, we will consider incorporating user sequence features into the construction of the recommendation model to improve the recommendation performance.

Acknowledgements This work is supported by Tianjin "Project + Team" Key Training Project under Grant No. XC202022, the National Nature Science Foundation of China under Grant No. 61702368, and the Natural Science Foundation of Tianjin under Grant No. 18JCQNJC00700.

Conflict of interest

The authors declare that they have no conflict of interest.

Human participants or animals

This article does not contain any studies with human participants or animals performed by any of the authors.

Authorship contributions

All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Ke Zhu, Wenguang Zheng and Xu Jiao. The first draft of the manuscript was written by Ke Zhu and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

References

1. Bhandari U, Sugiyama K, Datta A, et al (2013)Serendipitous Recommendation for Mobile apps Using Item-Item Similarity Graph, in: AIRS, pp. 440-451
2. Bohmer M, Ganev L, Kruger A, et al (2013) AppFunnel: a framework for usage-centric evaluation of recommender systems that suggest mobile applications, in: IUI, pp. 267-276
3. Cao D, Nie L, He X, et al (2017) Version-sensitive mobile app recommendation, Information Sciences, vol. 381, 161-175
4. Cao H, Bao T, Yang Q, et al (2010) An effective approach for mining mobile user habits, in: CIKM, pp. 1677-1680
5. Chen N, Hoi S C, Li S, et al (2016) Mobile app Tagging. in: WSDM, pp. 63-72
6. Davidsson C, Moritz S (2011) Utilizing implicit feedback and context to recommend mobile applications from first use, in: CaRR, pp. 19-22
7. DEMPSTER, A. P.(1977) Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, no. 39, pp. 1–38
8. Frey R M, Xu R, Ammendola C, et al (2017) Mobile recommendations based on interest prediction from consumer's installed apps insights from a large-scale field study. Information Systems, pp. 152-163
9. Han Y, Park S, Park S, et al (2017) Personalized app recommendation using spatio-temporal app usage log, Information Processing Letters, vol. 124, 15-20
10. Hao Y, Wang Z, Xu X (2016) Global and personal app networks: characterizing social relations among mobile apps, in: ICSC, pp. 227-234
11. He J, Liu H (2017) Mining Exploratory Behavior to Improve Mobile app Recommendations. ACM Trans. on Information Systems, vol. 35, no.4. 1-37
12. Hu J, Liang J, Kuang Y, et al (2018) A user similarity-based Top-N recommendation approach for mobile in-application advertising. Expert Systems With applications, pp. 51-60
13. Jiao X, Xiao Y, Zheng W, et al (2019) A novel next new point-of-interest recommendation system based on simulated user travel decision-making process. Future Generation Computer Systems, no. 100, pp. 982-993
14. Jiao X; Xiao Y; Zheng W, et al (2019) R2SIGTP: A novel real-time recommendation system with integration of geography and temporal preference for next point-of-interest. in: WWW, pp. 3560–3563
15. Karatzoglou A, Baltrunas L, Church K, et al (2012) Climbing the app wall: enabling mobile app discovery through context-aware recommendations. in: CIKM, pp. 2527-2530
16. Kim J, Kang S, Lim Y, et al (2013) Recommendation algorithm of the app store by using semantic relations between apps. The Journal of Supercomputing, vol 65, no. 1, pp. 16-26
17. Liang T, He L, Lu C, et al (2017) A Broad Learning approach for Context-Aware Mobile application Recommendation. in: ICDM, pp. 955-960
18. Lin J, Sugiyama K, Kan M, et al (2014) New and improved: modeling versions to improve app recommendation. in: SIGIR, pp. 647-656
19. Liu B, Wu Y, Gong N Z, et al (2016) Structural Analysis of User Choices for Mobile app Recommendation. ACM Trans. on Knowledge Discovery From Data, vol. 11, no. 2, pp. 17:1-17:23

20. Liu C, Wu X (2016) Large-scale recommender system with compact latent factor model. Expert Systems With applications, vol. 64, pp. 467-475
21. Matthias B, Gernot B, Antonio K (2010) Exploring the design space of context-aware recommender systems that suggest mobile applications, in: RecSys
22. Palomba F, Salza P, Ciurumelea A, et al (2017) Recommending and localizing change requests for mobile apps based on user reviews, in: ICSE, pp. 106-117
23. Rodriguez A, Laio A (2014) Clustering by fast search and find of density peaks. Science, vol. 344, no.6191, pp. 1492-1496
24. Rodriguezhernandez M D, Ilarri S (2016) Pull-based recommendations in mobile environments. Computer Standards and Interfaces, pp. 185-204
25. Sun Z, Ji Z, Zhang P, et al (2017) Automatic labeling of mobile apps by the type of psychological needs they satisfy. Telematics and Informatics, vol. 34, no. 5, pp. 767-778
26. Wu X, Zhu Y (2016) A Hybrid approach Based on Collaborative Filtering to Recommending Mobile apps, in: ICPADS, pp. 8-15
27. Xiao Y, Ai P, Wang H, et al (2015) ENRS: An effective recommender system using bayesian model. in: DASFAA, pp. 531–535
28. Xiao Y, Wang G, Hsu C, et al (2018) A time-sensitive personalized recommendation method based on probabilistic matrix factorization technique. Soft Computing, no. 22, pp. 6785-6796
29. Xu Y, Zhu Y, Shen Y (2019) Leveraging app usage contexts for app recommendation:a neural approach. World Wide Web, no. 22, pp. 2721–2745
30. Yang B, Wu C, Sigg S, et al (2016) CoCo (Context vs. Content): Behavior-Inspired Social Media Recommendation for Mobile apps. in: GLOBECOM, pp. 1-6
31. Yankov D, Berkhin P, Subba R, et al (2013) Interoperability ranking for mobile applications, in: SIGIR, pp. 857-860
32. Yao Y, Zhao W X, Wang Y, et al (2017) Version-Aware Rating Prediction for Mobile app Recommendation. ACM Trans. on Information Systems, vol. 35, no.4, pp. 1-33
33. Yezheng L, Fei D, Yuanchun J, et al (2016) A Novel Apps Recommendation Algorithm Based on Apps Popularity and User Behaviors, in: DSC, pp. 584-589
34. Zheng V W, Cao B, Zheng Y, et al (2010) Collaborative filtering meets mobile recommendation: a user-centered approach, in: AAAI, pp. 236-241
35. Zheng X, Ding W, Xu J, et al (2014) Personalized recommendation based on review topics. Service Oriented Computing and Applications, vol. 8, no. 1, pp. 15-31
36. Zhou Y, Huang C, Hu Q, et al (2018) Personalized learning full-path recommendation model based on LSTM neural networks. Information Sciences, pp. 135-152
37. Zhu H, Chen E, Xiong H, et al (2014) Mobile app Classification with Enriched Contextual Information. IEEE Trans. on Mobile Computing, vol. 13, no. 7, pp. 1550-1563
38. Zhu H, Chen E, Yu K, et al (2012) Mining Personal Context-Aware Preferences for Mobile Users, in: ICDM, pp. 1212-1217
39. Zhu K, Xiao Y, Zheng W, et al (2021) A Novel Context-Aware Mobile Application Recommendation Approach Based on Users Behavior Trajectories. IEEE ACCESS, vol. 9, pp. 1362-1375
40. Zhu K, Zhang L, Pattavina A, et al (2017) Learning Geographical and Mobility Factors for Mobile application Recommendation, IEEE Intelligent Systems, vol. 32, no. 2, 36-44
41. Zhu N, Cao J (2017) GTRM: A Top-N Recommendation Model for Smartphone applications. in: ICWS, pp. 309-316

Figures

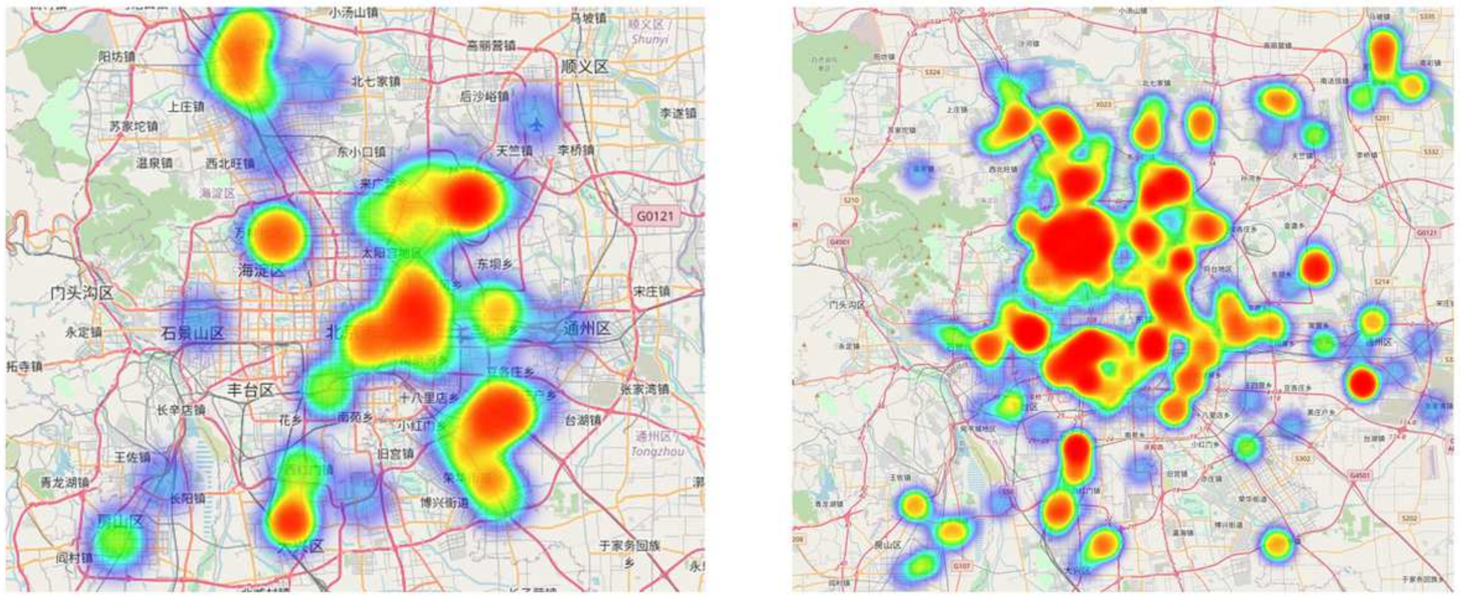


Figure 1

Two App usage location heat maps of a travel app and an electronic payment app, Monday from 8:00 to 11:00, at Beijing city. Note: The designations employed and the presentation of the material on this map do not imply the expression of any opinion whatsoever on the part of Research Square concerning the legal status of any country, territory, city or area or of its authorities, or concerning the delimitation of its frontiers or boundaries. This map has been provided by the authors.

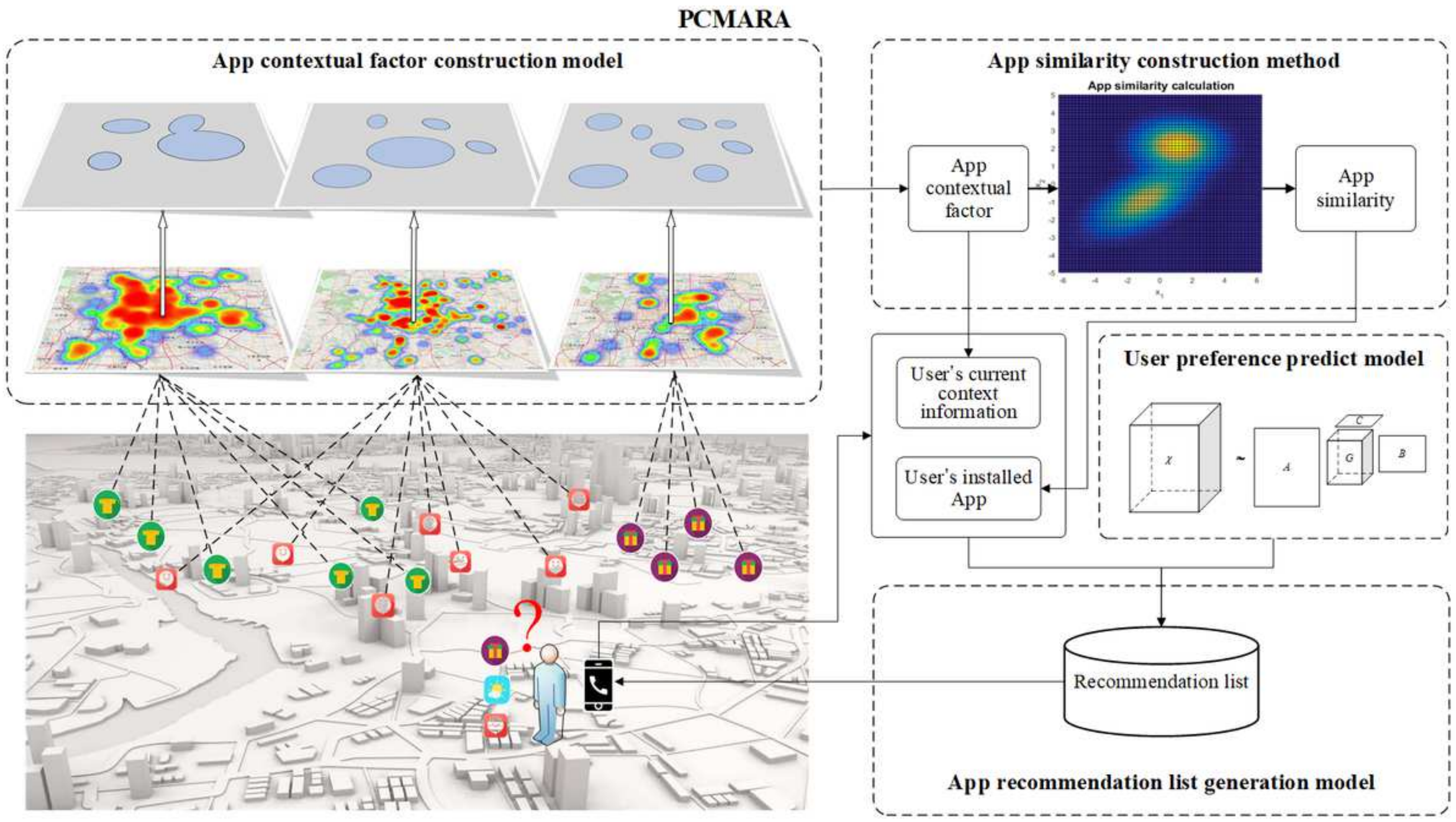

PCMARA
App contextual factor construction model
App similarity construction method
App similarity calculation
App contextual factor
App similarity
User's current context information
User's installed App
User preference predict model
Recommendation list
App recommendation list generation model

Figure 2

The architecture of PCMARA. Note: The designations employed and the presentation of the material on this map do not imply the expression of any opinion whatsoever on the part of Research Square concerning the legal status of any country, territory, city or area or of its authorities, or concerning the delimitation of its frontiers or boundaries. This map has been provided by the authors.

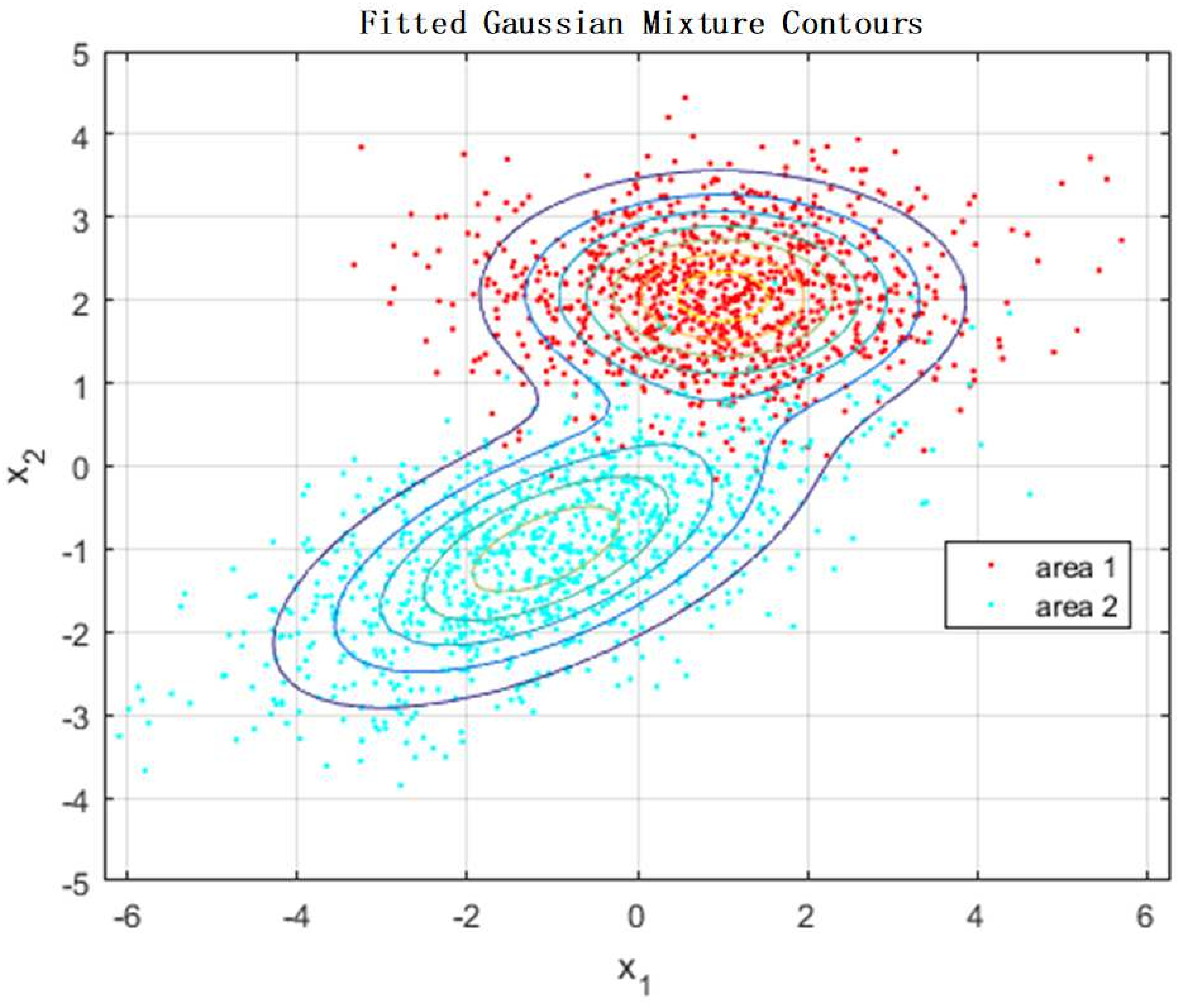


Figure 3

Gaussian mixture contours

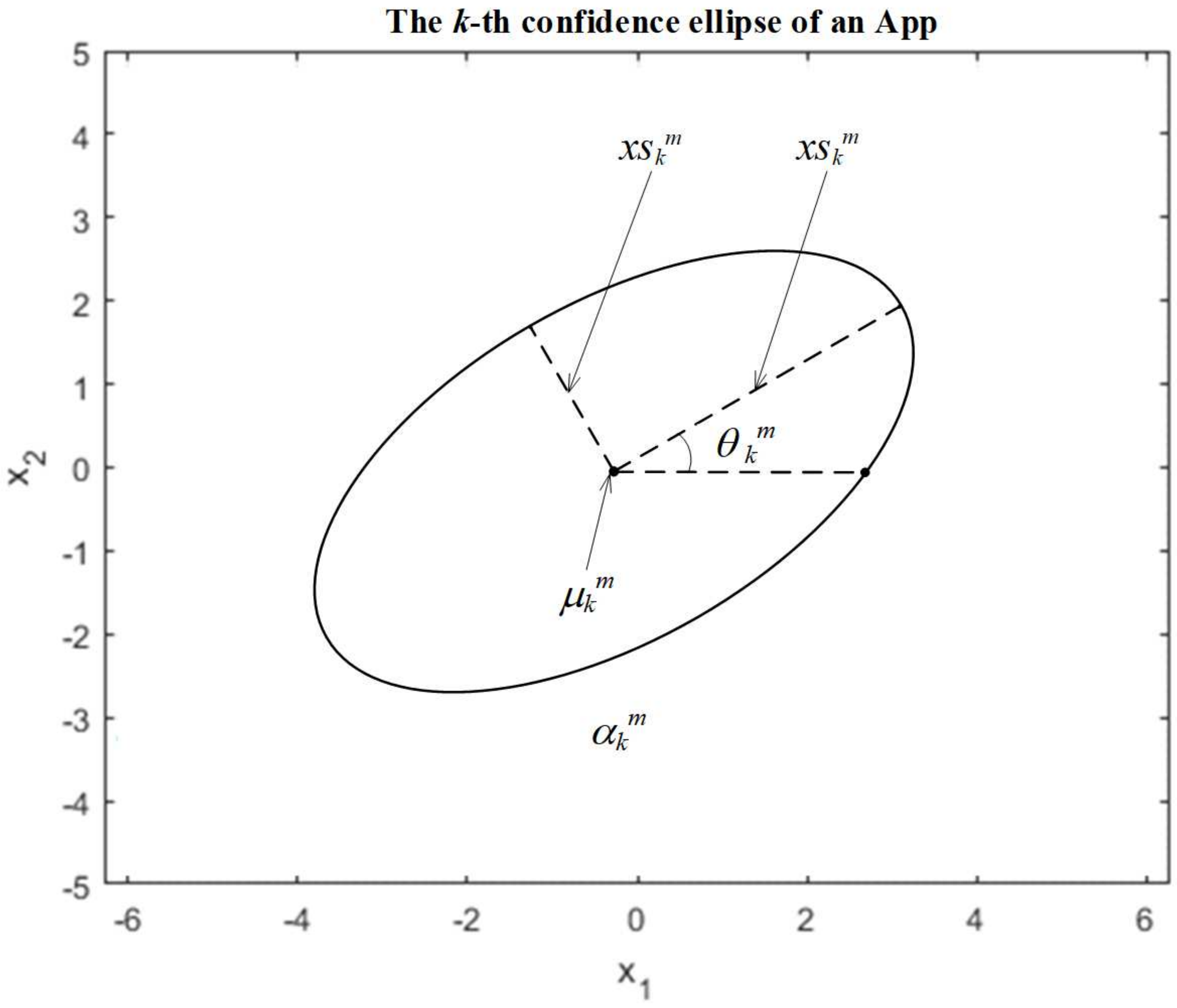


Figure 4

The K-th confidence ellipse of an App

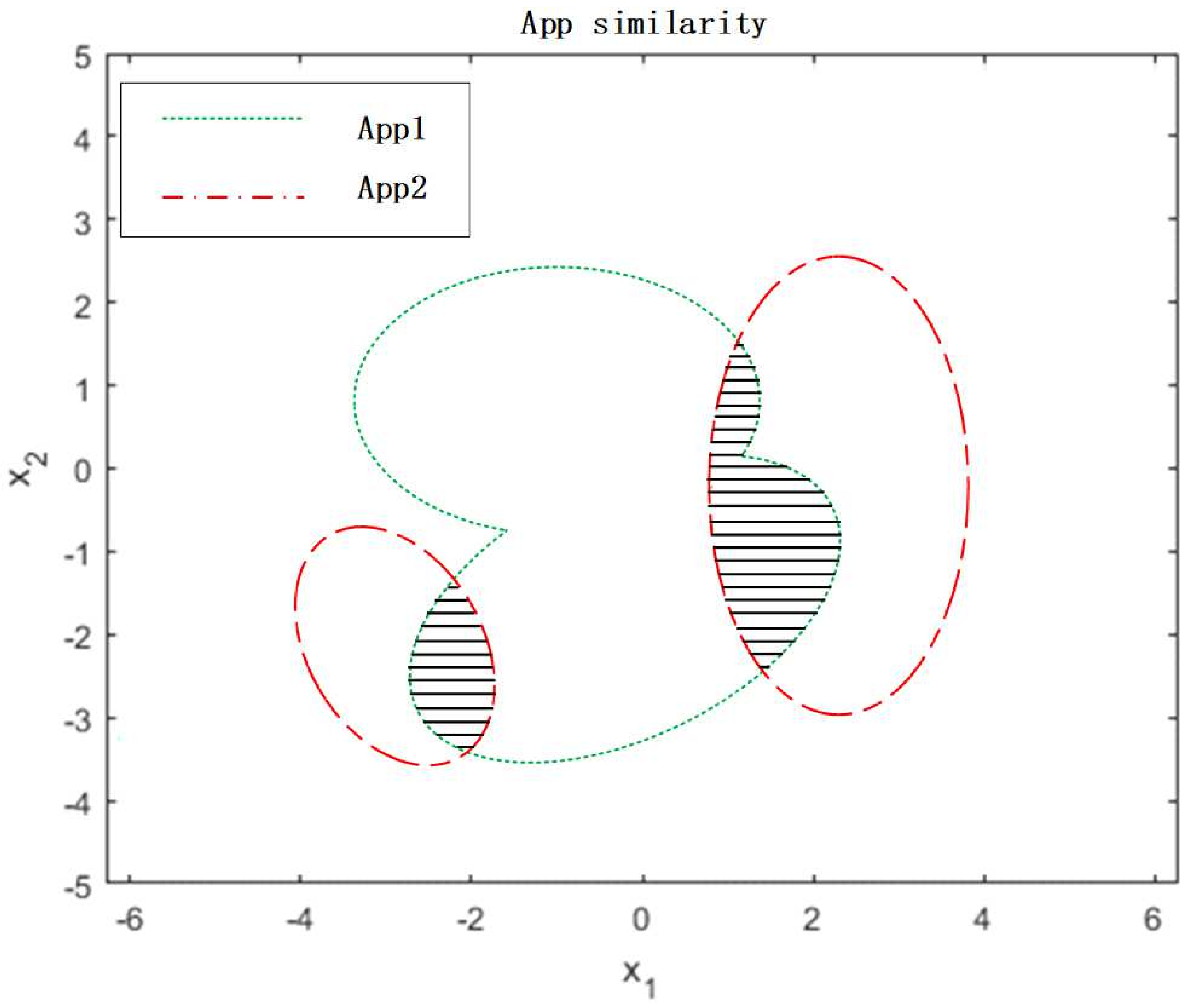


Figure 5

The similarity between two apps

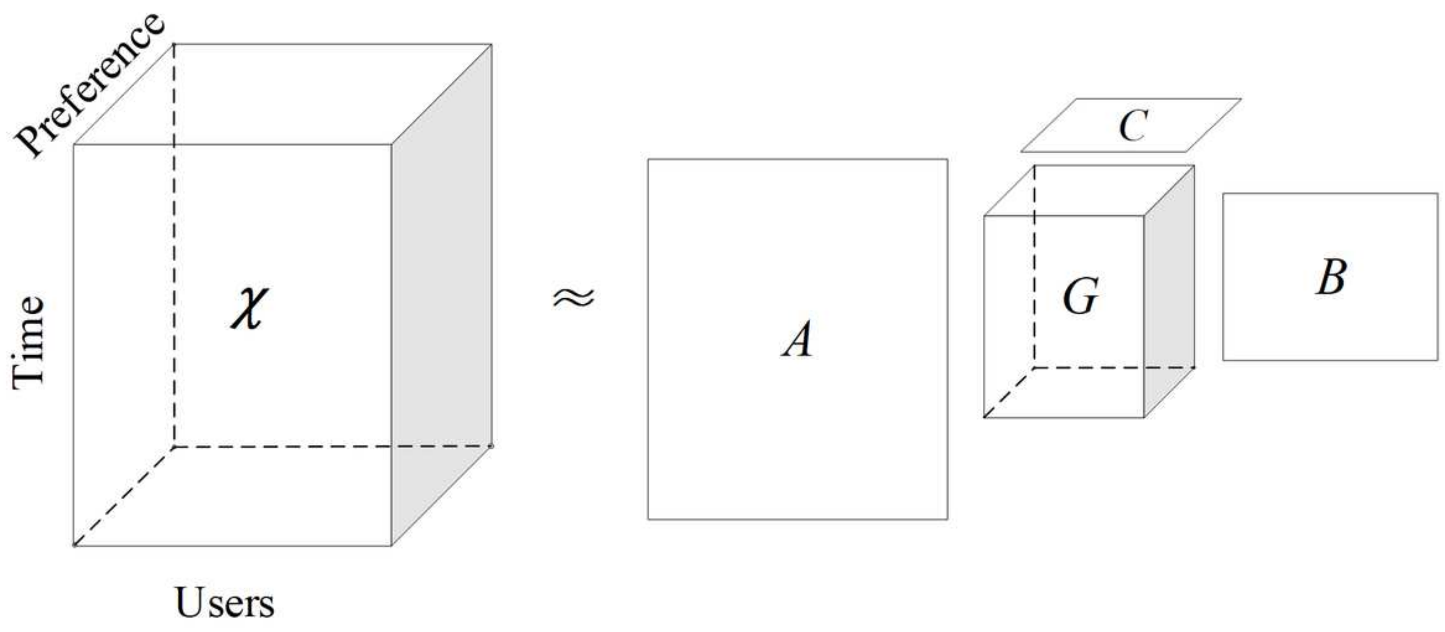


Figure 6

User preference tensor factorization

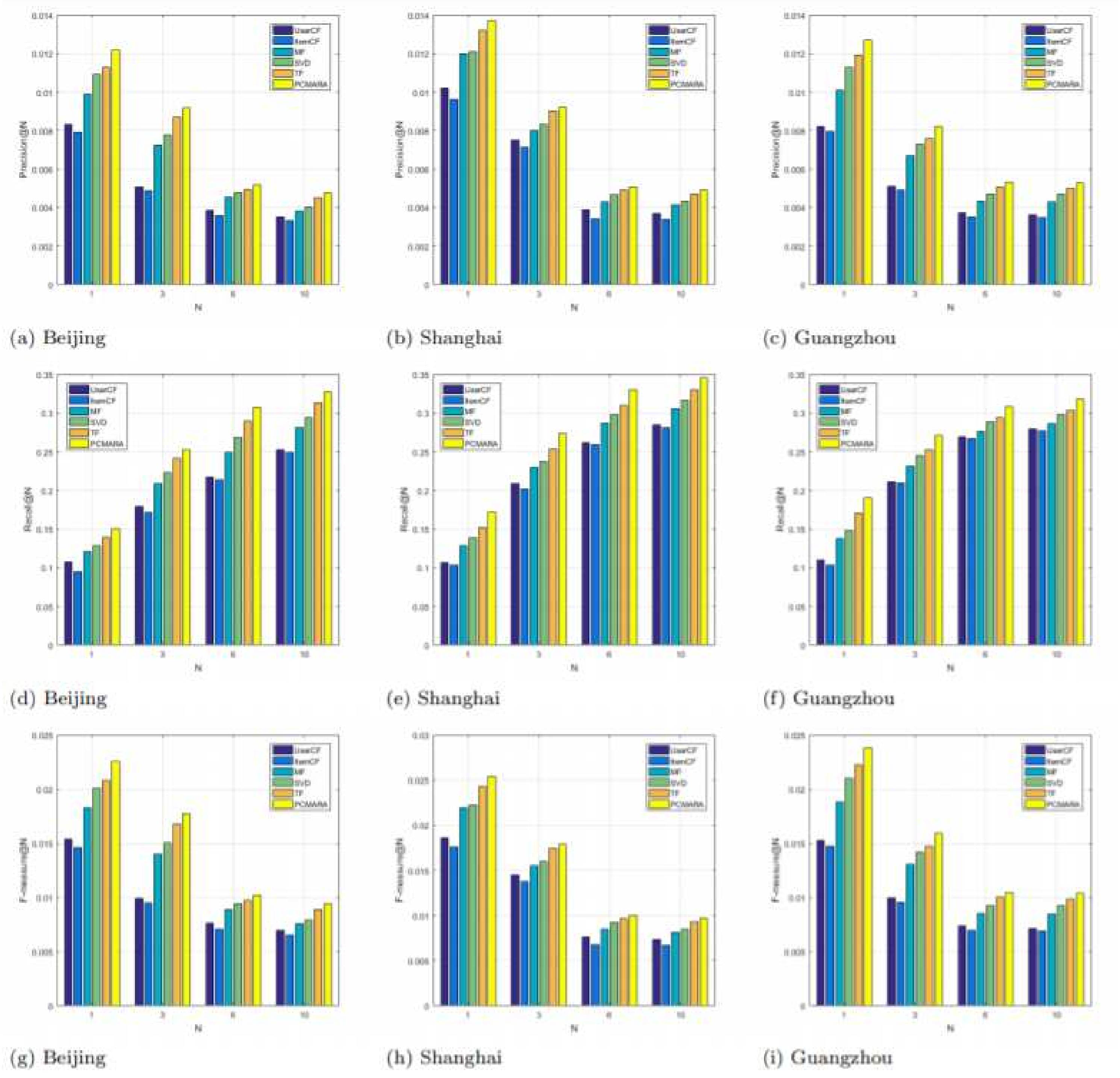


Figure 7

The comparison of recommendation performance between PCMARA and benchmark methods

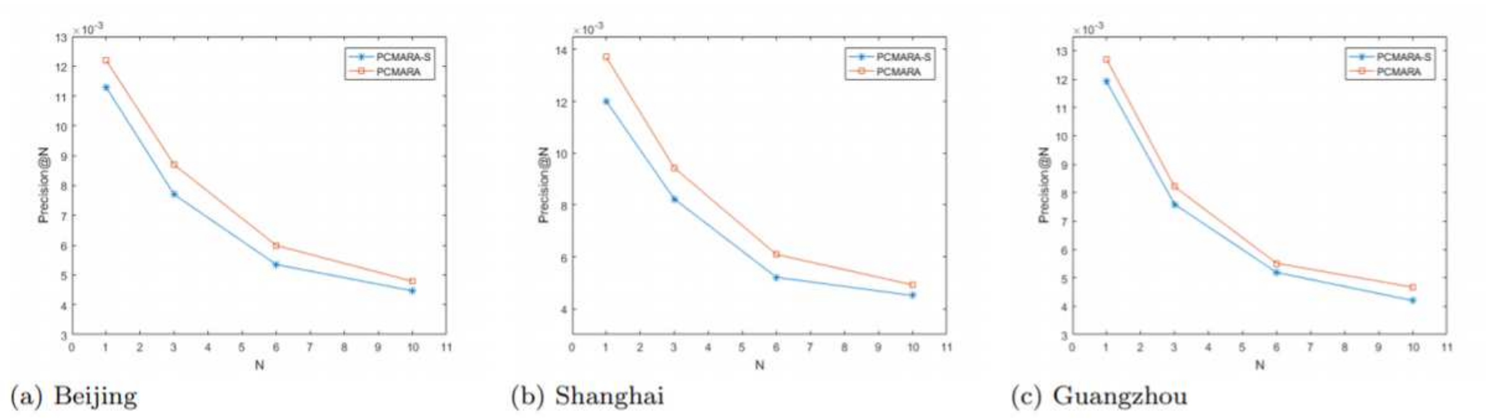


Figure 8

The comparison of recommendation performance between PCMARA and PCMARA-S

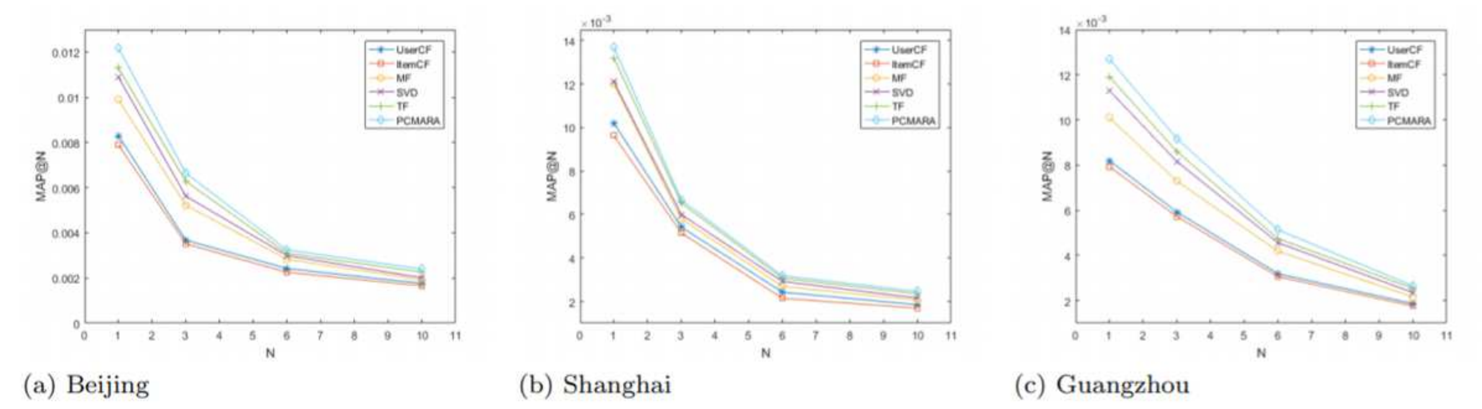


Figure 9

The comparison of ranking performance between PCMARA and benchmark methods

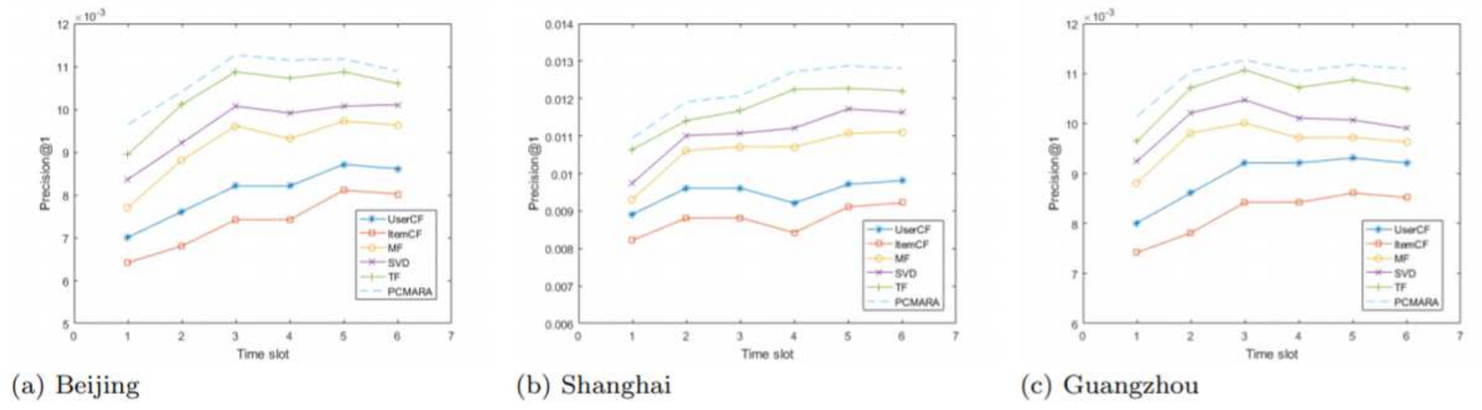


Figure 10

The comparison of recommendation performance between PCMARA and benchmark methods in different time slots

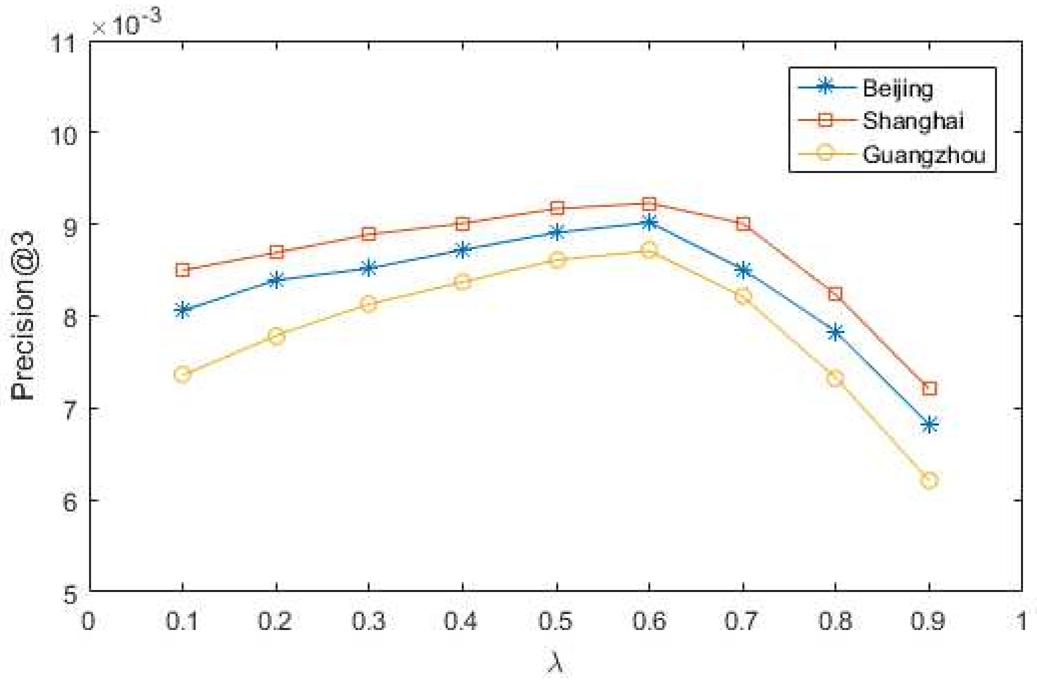


Figure 11

The influence of parameter λ