Expert Systems With Applications 111 (2018) 51–60

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A user similarity-based Top-*N* recommendation approach for mobile in-application advertising

Jinlong Hu[a,b,*], Junjie Liang[a,b,c], Yuezhen Kuang[a,b], Vasant Honavar[c]

[a] School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China
[b] Guangdong Key Laboratory of Communication and Computer Network, South China University of Technology, Guangzhou 510006, China
[c] Artificial Intelligence Research Laboratory, College of Information Sciences and Technology, Pennsylvania State University, University Park, PA 16802, United States

ARTICLE INFO

Article history:
Received 24 April 2017
Revised 7 February 2018
Accepted 8 February 2018
Available online 8 February 2018

Keywords:
Neighborhood-based recommendation
User similarity
Top-*N* preference
Mobile in-application advertising

ABSTRACT

Ensuring scalability of recommender systems without sacrificing the quality of the recommendations produced, presents significant challenges, especially in the large-scale, real-world setting of mobile ad targeting. In this paper, we propose MobRec, a novel two-stage user similarity based approach to recommendation which combines information provided by slowly-changing features of the mobile context and implicit user feedback indicative of user preferences. MobRec uses the contextual features to cluster, during an off-line stage, users that share similar patterns of mobile behavior. In the online stage, MobRec focuses on the cluster consisting of users that are most similar to the target user in terms of their contextual features as well as implicit feedback. MobRec also employs a novel strategy for robust estimation of user preferences from noisy clicks. Results of experiments using a large-scale real-world mobile advertising dataset demonstrate that MobRec outperforms the state-of-the-art neighborhood-based as well as latent factor-based recommender systems, in terms of both scalability and the quality of the recommendations.

1. Introduction

In recent years, mobile applications (app), installed on the mobile devices, have become some of the most important ways for people to access information and services online. Mobile in-application (in-app) advertising plays a key role in the app ecosystem. Such advertising is not only the primary source of revenue for app publishers but also the primary means for advertisers to target and reach specific audiences. In this context, effective approaches to personalize the recommendation of products and services to customers based on their individual tastes, and targeting the ads become critically important in the mobile app ecosystem.

Collaborative Filtering (CF) (Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994) is one of the most well-known and widely used approaches in recommender systems. CF aims to predict the tastes of an individual based on the tastes of other individuals. Neighborhood-based Collaborative Filtering (NBCF) focuses on the similarity between users (user-based method) or, alternatively, between items (item-based method). The former predicts the tastes of an individual based on the tastes of other individuals who are most similar to that individual (e.g., in terms of the products that they like). The latter predicts the likely audience for a product based on the audience for other similar products. NBCF methods are very popular due to their simplicity and their ability to provide intuitive explanations of the rationale behind their recommendations, which often enhances the user experience (Ning, Desrosiers, & Karypis, 2015). NBCF methods have been used to predict the user ratings of items (e.g., Resnick et al., 1994; Xia et al., 2016) and to identify the top-*N* items to recommend to a user (e.g., (Lee, Lee, Lee, Hwang, & Kim, 2016; Liu & Yang, 2008)).

However, when the number of users and items becomes very large, some of the popular NBCF methods become impractical because of their high computational complexity. For example, when the number of user m reaches to the order of a few hundred million, the computational complexity of traditional user-based methods, which is $O(m^2 \cdot n')$ (where n' is the time needed to compute the similarity between a pair of users), will become impractical (Aggarwal, 2016b). Hence, there are growing interests in approaches to scaling up nearest-neighbor computations. Examples of such approaches include sampling based approaches in which the nearest neighbors of a user (or item) are obtained from a chosen subset of users (or items) as opposed to the entire set of users (or items)

* Corresponding author at: School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China.
E-mail addresses: jlhu@scut.edu.cn (J. Hu), jul672@ist.psu.edu (J. Liang), cskuangyuezhen@mail.scut.edu.cn (Y. Kuang), vhonavar@ist.psu.edu (V. Honavar).

https://doi.org/10.1016/j.eswa.2018.02.012
0957-4174/

(Anastasiu, Christakopoulou, Smith, Sharma, & Karypis, 2016); Clustering based methods, where the users (or items) are first grouped into a number of clusters and the nearest neighbor calculations are limited to the members of the cluster (e.g., Katukuri, Mukherjee, Konik, & Konik, 2013; Xu, Bu, Chen, & Cai, 2012; Xue et al., 2005). But the gain in the efficiency of sampling or clustering based methods comes at the expense of a loss in quality (Aggarwal, 2016b) and the tradeoff between improved efficiency and possible decrease in accuracy needs to be carefully measured (Amatriain, Jaimes, Oliver, & Pujol, 2011). More importantly, the clusters have to be recomputed or adjusted to accommodate as new data about users' actions become available over time (Agarwal, Chen, & Elango, 2010). Against this background, it is of interest to explore accurate and efficient NBCF for large-scale systems where the recommendations have to be constantly adjusted based on new data as they become available.

To address this challenge, we focus on the user-based neighborhood-based model for large-scale mobile in-app advertising system, and propose a two-stage user similarity based approach to recommend the top-*N* items (MobRec). MobRec identifies users that are most similar to given user by taking advantage of user clusters obtained in an offline clustering phase. First, MobRec groups the user behavior in the context of mobile in-app advertising into two sets of features: (a) mobile context: features that capture the contextual information about users such as routine, mobility and mobile device features that are suggestive of typical use patterns, but not user actions on specific ads, and hence reflect fairly stable user preferences to ads in specific contexts; (b) implicit feedback: a set of user actions on ads (such as clicks, downloads and installations) that are indicative of user interests as a function of incoming data which are more precise and direct. MobRec clusters the users based on the coarse and stable mobile context in an offline mode. Because of the stable nature of the mobile context, the clusters are updated only infrequently. MobRec computes the nearest neighbors of a target user in an online mode using the similarity of the target user with other users which in turn is obtained by aggregating the simirlarity with respect to the dynamic features (implicit feedback) with the stable features (mobile context) within the cluster to which the target user belongs. The resulting nearest neighbors are used with a novel preference model compute the latent preference scores of items which in turn are used to recomme to the top-N recommendations to the target user.

Our contributions could be summarized as follows:

- We design a novel, two-stage approach to recommendation consisting of an offline user clustering stage and an online top-*N* ranking stage, which improves scalability without compromising the quality of recommendations.
- We introduce a novel approach to representing the dynamicity and richness of user behaviors using mobile context that changes slowly and implicit feedback which captures the users' response to specific ads. Contextual features are used for offline clustering of users; the contextual features are used together with implicit feedback features in the online stage to enhance the accuracy of predictions.
- We make use of a novel preference model to improve the accuracy of top-*N* recommended ads by learning the latent user preference from the observed implicit feedback.
- We present results of experiments comparing our approach to recommendation with several state-of-the-art methods, including neighborhood-based models and latent factor models which clearly show that our approach is able to improve scalability without compromising the quality of the recommendations.

The rest of the paper is organized as follows: Section 2 summarizes related work. Section 3 describes our proposed two-stage approach to recommendation. Section 4 describes our experiments and results of comparison of our approach with other state-of-the-art methods. Section 5 concludes the paper.

2. Related work

There are two primary Collaborative Filtering approaches: neighborhood-based models (also referred to as memory-based methods) and the latent factor models. Neighborhood-based methods are widely used in real-world settings because they are simple, easy to implement, easy to understand and explain, and often work well in practice (Ning et al., 2015). The key idea behind neighborhood-based methods is the use of the user-item rating matrix to calculate the similarities between users or items and make recommendations to users based on the observed behaviors of similar users (Linden, Smith, & York, 2003). For example, Resnick et al. (1994) describe a user-based Collaborative Filtering method, which utilizes the explicit ratings of items provided by users to compute the similarity between users and recommend items to a user by taking a weighted average of the ratings of the user's nearest neighbors. Comprehensive surveys of neighborhood-based Collaborative Filtering methods could be found in (Aggarwal, 2016b; Ning et al., 2015). The quality of recommendations produced by neighborhood-based methods depends in part on the information and the metric used to identify the nearest neighbors of targets of recommendation. Hence, there has been much work focused on improving the quality of neighborhood-based model, including, for example: 1) predicting the rating r_{ui} of the user u for item i with algorithms that focus on similarity computation on explicit ratings, such as significant factor optimization (Ma, King, & Lyu, 2007), graph-based similarity (Jeong, Lee, & Cho, 2010) and global similarity (Ahn, 2008; Liu, Hu, Mian, Tian, & Zhu, 2014); or algorithms that focus on implicit feedback or features, such as FeatureKNN (Xia et al., 2016); 2) learning to rank methods that identify the top-*N* most relevant items for a particular user based on the estimated user preferences, such as EigenRank (Liu & Yang, 2008) and PrefKNN (Lee et al., 2016). The main drawback of such methods is their limited scalability.

Existing works on scaling up nearest-neighbor approaches are partitioned into three categories: filtering based approaches, approximate methods for nearest neighbor identification methods and sampling based approaches (Anastasiu et al., 2016). Filtering based approaches often pre-filter, using domain knowledge when available, the user or item pairs that cannot be neighbors using an inverted index data structure (Anastasiu & Karypis, 2014; Awekar & Samatova, 2009; Bayardo, Ma, & Srikant, 2007); approximate methods for identifying nearest neighbors usually rely on latent decomposition or low-rank embeddings to learn dense latent vectors representing users or items (Bell & Koren, 2007; Koren, 2010b). Sampling based approaches often make use of clustering to precompute the likely candidate nearest neighbors of the target user in an offline clustering stage. For example, Xue et al. (2005) have proposed the SCBPCC model to cluster the users based on their explicit ratings of items and use the resulting clusters to identify the candidate nearest neighbors for a target user. In contrast, Katukuri et al. (2013) use clustering algorithms to partition the set of items based on a set of contextual features. Xu et al. (2012) focus on co-clustering models to group users and items. However, such off-line clustering approaches can suffer from loss of quality, and may need to be rebuilt frequently (e.g., daily (Agarwal et al., 2010)), resulting in added computation cost.

Latent factor models, i.e., PMF (Mnih & Salakhutdinov, 2008), SVD++(Koren, 2010b), generally seek to decompose the original rating matrix into fully specified low-rank matrices (user factors and item factors) with low redundancies to provides robust estimation of the missing entries. With the implementa-

tion of alternating-least-squares factorization algorithm (Bell & Koren, 2007), the optimization process is performed alternatively on one space (user or item space) while keeping the other space fixed, thus enhancing scalability. Another notable advantage of latent factor models is their ability to leverage side information, whenever such information is available. ALS-IMF (Hu, Koren, & Volinsky, 2008), treats the ratings as "confidence values" related to the strength of indicated user preferences (implicit feedback), rather than explicit ratings of the items by users. Latent factor models are considered to be the state-of-the-art in terms of both scalability and quality of recommendations in a broad range of application scenarios (Aggarwal, 2016a).

While there has been considerable work on recommender systems in general, there has been relatively limited work on recommender systems for mobile advertising. Existing work on this topic focuses more on the mining and utilization of mobile user (or mobile ad) contextual features such as location, time, weather, temperature and their combination (e.g., (Park, Park, & Cho, 2015; Yuan & Tsao, 2003)). For example, Zhu et al. (2015) proposed a novel context-aware preference mining approach to learn the intra-user and inter-user mobile contextual features. The extracted contextual features were used to form the mobile user profile used to train recommenders using existing supervised learning techniques, e.g., regression model (Wu et al., 2016), Gradient Boosting Decision Trees (Wang et al., 2016).

Against this background, we present a stable offline neighborhood-based model based on the slowly-changing contextual features which are combined with information provided by implicit feedback by an online preference-based ranking algorithm that achieves scalability without compromising the quality of recommendations.

3. User similarity-based aggregated recommendation model

3.1. Preliminaries

We first define the basic notations used throughout this paper. Given the set of m users, $\mathcal{U} = \{u_1, \ldots, u_m\}$, and the set of n ads, $\mathcal{I} = \{i_1, \ldots, i_n\}$. All user-ad pairs can be denoted by an m-by-n matrix $\mathcal{R} = \mathcal{U} \times \mathcal{I}$, where the entry r_{ui} indicates the assigned value of implicit feedback of u to i. If r_{ui} has been observed (or known), it is represented by a rating associated with the specific behavior; otherwise, a global default rating is used. Let $\mathcal{R}_+ \subseteq \mathcal{R}$ denote a subset of user-ad pairs for which implicit feedbacks are observed and $\mathcal{R}_{u+}$ denote the observed implicit feedbacks for user u. We reserve the indexing letters u, v to indicate arbitrary users in $\mathcal{U}$ and i, j to represent arbitrary ads in $\mathcal{I}$. Let $x_u = \{x_{u1}, x_{u2}, \ldots, x_{uP}\}$ be the contextual feature vector of u, where P is the length of the contextual feature vector x_u. Without loss of generality, we constrain the entries of x_u to be binary.

3.1.1. User-based neighborhood models

User-based neighborhood models are based on the similarity between users. The basic assumption of neighborhood-based models is that if two users share similar behaviors in the past (e.g., have similar behaviors on common ads), they will have similar response in the future (e.g., have similar actions on other ads) (Goldberg, Roeder, Gupta, & Perkins, 2001). Let $U_k(u; i)$ be the set of top-k similar neighbors of u who have rated item i. Given the user-ad rating matrix $\mathcal{R}$, the predicted rating for user u to ad i is computed by:

$$\hat{r}_{ui} = d_{ui} + \frac{\sum_{v \in U_k(u;i)} (r_{vi} - d_{vi}) sim(u, v)}{\sum_{v \in U_k(u;i)} |sim(u, v)|} \tag{1}$$

where d_{ui} is a biased rating value for u to i, and $sim(u, v)$ is the similarity weight between users u and v. Specifically, when the *mean-centered* prediction (Koren, 2010a) is used, we can replace d_{ui} with the mean rating μ_u of user u. Then, the predictive function in Eq. (1) is revised as:

$$\hat{r}_{ui} = \mu_u + \frac{\sum_{v \in U_k(u;i)} (r_{vi} - \mu_v) sim(u, v)}{\sum_{v \in U_k(u;i)} |sim(u, v)|} \tag{2}$$

3.1.2. User-based preference models

In contrast to the traditional user-based neighborhood models that try to estimate the exact ratings of users, preference models focus on a function that produces a (possibly partial) ranking or ordering of items. Following (Cremonesi, Koren, & Turrin, 2010), we can specify the predictive function based on a preference model as:

$$\hat{r}_{ui} = \sum_{v \in U_k(u;i)} pref_{vi} sim(u, v) \tag{3}$$

where $pref_{vi}$ denotes the user-defined preference score for user v to ad i over all other ads associated to v.

The key components of neighborhood-based models consist of similarity computation, nearest neighbor filtering and prediction (ranking). The user-user (or item-item) similarity is computed offline to enable rapid retrieval; nearest filtering and prediction are computed online centered at the target (Aggarwal, 2016b).

3.2. Architecture

The overall structure of our two-stage recommender system (MobRec) is presented in Fig. 1. Our framework includes two major stages: offline user clustering stage and online nearest filtering and ranking stage. We first pre-process the user behavior logs and construct the slowly-changing mobile context, which is then used as the input to the clustering stage. The user clustering stage uses the k-means algorithm to create a small number of peer groups. Due to the inactive nature of mobile context, the computed similarity may persist for several hours to several days.

To retrieve the most relevant ads for a target user, the dynamic implicit feedback is integrated with contextual information to select the nearest neighbors of the target user within the closest cluster. The ranking stage takes the combination of mobile context and implicit feedback as input and sorts the predicted scores of ads with a preference-based Collaborative Filtering approach. The highest scoring ads are then presented to the target user.

The complete workflow of our algorithm (MobRec) is presented in Algorithm 1.

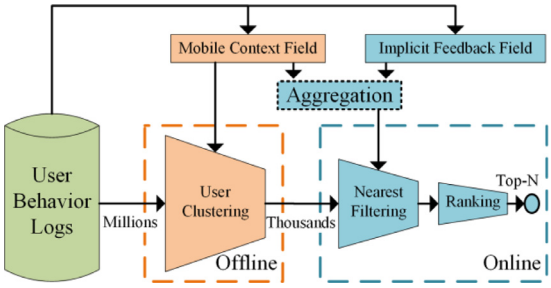

Fig. 1. The two-stage framework for a mobile in-application ad recommender system. The slowly-changing mobile context is used to cluster the users and then combined with dynamic implicit feedback to refine the nearest neighbor computation. Lastly, a preference-based ranking model is used to retrieve the top-N ads to display to the target user.

3.3. User clustering

There are two key challenges in clustering users using rating matrix: a) the incompleteness of rating matrix leads to loss of ac-

Algorithm 1 The workflow of MobRec.

Input: User's contextual feature X, implicit feedback $\mathcal{R}$, target user u, number of clusters $|C|$, elastic factor ε.
Output: top-N recommendation list for target user u.
Offline stage:
Perform user clustering using the coarse context-based similarity of users (Section 3.3).
Online stage:
1. Retrieve the closest cluster of the target user using the coarse context-based similarity.
2. Computer the aggregated similarity between target user and the users in the closest cluster (Section 3.4.1).
3. Selected the nearest neighbor of the target user.
4. Perform the top-N ranking algorithm on the nearest neighbor (Section 3.4.2) and compute the top-N recommendation list for target user u.

curacy; b) the context-dependent changes in user behavior impacts the performance of the model in the long term (Koren, 2010a; Koren & Bell, 2015). We seek to overcome the chanllenges using slowly-changing contextual features. Our approach is motivated by the following observations:

(1) Mobile context provides information about a user's long-term routine, mobility pattern, and device features.
(2) The slowly-changing nature of mobile context ensures the stability of the clusters generated using contextual information (e.g., from several hours to several days).
(3) The mobile context-based similarity could be easily combined with other similarity measures.

We compute context similarity of users using the Jaccard index (Koutrika, Bercovitz, & Garcia-Molina, 2009) due to the binary nature of our data entries. Thus, for two arbitrary users u, v, mobile context-based similarity $sim_X(u,v)$ captures the proportion of number of entries that overlap between the context feature vector x_u and x_v:

$$sim_X(u,v) = \frac{|x_u \wedge x_v|}{|x_u \vee x_v|} \tag{4}$$

We select the k-means algorithm to cluster the users into peer groups based on the similarity measure in Eq. (4). The number of clusters $|C|$ is heuristically set equal to the cubic root of the number of users. Suppose that C_1,..., $C_{|C|}$ represent the sets of users to be assigned to the respective clusters and $\overline{Y_1}, \ldots, \overline{Y_{|C|}}$ denote the corresponding centroids. Our clustering approach is summarized as follows:

1. For each $i \in \{1, \ldots, |C|\}$, initialize the centroid $\overline{Y_i}$ with the contextual features of a randomly chosen user.
2. Determine the cluster assignment C_1,..., $C_{|C|}$ by assigning each user to the cluster whose centroid is closest to the user as measured by Eq. (4).
3. Update the centroids based on the average (mean) contexts of the set of users assigned to the respective clusters and then return to step 2 unless the clusters converge.

Time complexity

The running time of k-means algorithm for each iteration is linear to the user volume m. For a given number of cluster $|C|$, the time complexity for each iteration is $O(|C|Pm)$.

3.4. Nearest neighbor filtering and ranking

Traditional neighborhood-based Collaborative Filtering models retrieve the neighbors of target user on the whole dataset (Lee et al., 2016; Resnick et al., 1994; Xia et al., 2016). However, this approach does not scale as the number of users and items grows. Our use of clustering allows us to limit the set of users to be considered to those that belong to the cluster whose centroid is closest to the target user. Suppose we denote the target user by u and its closest cluster by C_u.

3.4.1. Nearest neighbor filtering

Although mobile context captures some aspects of a user's behavior, it is too coarse to provide accurate information about the user's preferences. Hence, we integrate the information provided by dynamic implicit feedback with contextual information to refine the set of closest neighbors of a target user. Let $sim_R(u,v)$ be an implicit feedback-based similarity model (several alternative models are considered in our experiments). The resulting similarity function takes the maximum of the context-based similarity and implicit feedback-based similarity:

$$sim(u,v) = \max\left\{ \frac{sim_X(u,v) - \min\limits_v sim_X(u,v)}{\max\limits_v sim_X(u,v) - \min\limits_v sim_X(u,v)}, \frac{sim_R(u,v) - \min\limits_v sim_R(u,v)}{\max\limits_v sim_R(u,v) - \min\limits_v sim_R(u,v)} \right\} \tag{5}$$

Notie that the min-max scaling is adopted to make the two types of similarity comparable. Based on the combined similarity function, we select the k nearest neighbors of u (denoted as U_k).

3.4.2. Top-N ranking

It is common in real-world applications of recommender systems that only a few of the recommended items are presented to the users. When the set of items to be recommended are chosen among the topmost with respect to the predicted partial order, the resulting problem corresponds to the well-studied topic of top-N recommendation (Balakrishnan & Chopra, 2012), where latent user preferences are used instead of actual ratings (Lee et al., 2016; Liu & Yang, 2008). This is consistent to our mobile ads recommendation scenario, since only a tiny set of ads are presented at a time. Therefore, we use the latent preference scores to choose the top-N ads based on the observed implicit feedback $\mathcal{R}_{u+}$ of user u. Let $\mathcal{S}_{u+}$ denote the ascending order of ads based on $\mathcal{R}_{u+}$, we estimate a preference (likelihood) score θ_i for the corresponding ad i to reflect the extent to which ad i is preferred by u. We use the resulting likelihood to generate the ranking list $\mathcal{S}_{u+}$. Thus,

$$\theta^* = \arg\max_{\theta} p(\mathcal{S}_{u+}|\theta) \tag{6}$$

For simplicity, suppose given ad i, we divide the implicit feedback into three categories in terms of i. That is, $T_i = \{T_{<i}, T_{=i}, T_{>i}\}$ where $T_{<i}$, $T_{=i}$ and $T_{>i}$ denote the set of ads that are ranked lower than, equal to and higher than i respectively. We can rewrite the likelihood $p(\mathcal{S}_{u+}|\theta)$ as the product of the likelihoods of generating the correct order for any two ads. Thus $p(\mathcal{S}_{u+}|\theta)$ is replaced by:

$$p(\mathcal{S}_{u+}|\theta) = \prod_i \left(\prod_{j \in T_{<i}} p(i > j|\theta) \prod_{j \in T_{=i}} p(i > j|\theta) \right) \tag{7}$$

where $p(i > j|\theta)$ is the likelihood that ad i is preferred to ad j, which is defined as:

$$p(i > j|\theta) = \theta_i(\epsilon - \theta_j) \tag{8}$$

We introduce an elastic factor ε, which is set to shrink or augment the importance of preference associated with u, in ways that account for the noisy nature of user preferences. For example, because of the limited device screen size and complex usage scenarios (e.g. walking or taking a bus), mobile users are prone to generate misclicks (e.g. the inaccurate clicks do not result in the desired action); malicious publishers often try to generate fake clicks (Hu, Liang, & Dong, 2017). Hence, the preference information obtained from the clicks is inherently noisy. See Section 4.6. for further discussion of the effect of elastic factor.

Note that the probability scores for any two ads that lie within the same category (as described above) are identical. Hence the gradient of the log likelihood of Eq. (7) is given by:

$$\frac{\partial \log p(\mathcal{S}_{u+}|\theta)}{\partial \theta_i} = |T_{<i}|\left(\frac{1}{\theta_i}\right) + |T_{=i}|\left(\frac{1}{\theta_i} - \frac{1}{\epsilon - \theta_i}\right) - |T_{>i}|\left(\frac{1}{\epsilon - \theta_i}\right) \tag{9}$$

By setting the gradient to zero, we can obtain the optimal probability score θ_i^* as:

$$\theta_i^* = \frac{\epsilon |T_{\leq i}|}{|\mathcal{S}_{u+}| + |T_{=i}|} \tag{10}$$

We first transform the implicit feedback $\mathcal{R}_{U_k \cup \{u\}}$ into a preference matrix using Eq. (10). We then compute the predictions using the weighted sum of the neighbors' preference scores as in Eq. (3). The top-N ads are then identified based on the preference scores.

Time complexity. The time complexity of aggregated similarity calculation is $O(|C_u|n)$ (where n is the number of items). For nearest neighbor selection, we examine the members of the cluster whose centroid is closest to the target user which takes $O(|C_u|n)$ time. In the prediction phase, the computation time of both the preference matrix transformation and recommendation is $O(kn)$. Since k is typically much smaller than $|C_u|$, the overall complexity of top-N ranking stage is $O(|C_u|n)$. Since $|C_u|$ is much smaller than the user volume m, our online model is substantially more scalable than those that do not benefit from the clustering stage (Lee et al., 2016; Resnick et al., 1994; Xia et al., 2016).

4. Empirical analysis

We experimentally evaluated the proposed algorithm on a real-world dataset from one of the mobile advertising platforms in China. We proceed to describe the data set, the experimental setup, and the experimental results.

4.1. Dataset and basic settings

We collect a set of three-week user behavior logs, in which implicit feedbacks of user-ad pairs are divided into six categories, including view, click, download starts, download completes, installation starts, and installation completes. Note that the implicit feedback are inherently ordered according to user action: e.g., view < click < download < installation. The resulting information can be used directly in our proposed top-N ranking model to compute the estimated user preference (Eq. (10)). In addition, since explicit ratings are often required for Collaborative Filtering models, we empirically map the implicit feedback to explicit ratings as (view, click, download starts, download completes, installation starts, installation completes) = (1, 10, 15, 20, 25, 30). As in (Lee et al., 2016), all missing ratings are imputed with zero. In addition to the behavior logs, we use the following attributes: **Id**: unique string to identify an object (i.e., user, application or ad). **Geographical Attributes**: a set of attributes that can uncover the geographical position of a user, such as IP, city and mobile network access method. **Action time**: the timestamp of an action. **Device Attributes:** characteristics of user's mobile device, e.g., device's operating system, screen size, etc. All sensitive fields are encrypted before further processing due to privacy considerations.

We first discard users that do not meet the following criteria: (a) being active for at least seven days, (b) being associated with at least seven unique ads and (c) having clicked at least one ad. This yields a dataset of approximately 1 M users, 1 K ads and 27 M records. Since we focus on the users' mobile context that changes slowly over time, the mobile contextual features are modeled using binary values (i.e., {0,1}; with value 1 indicating the presence of the specific context feature and 0 otherwise). We capture three types of contextual features: user's long-term routine, mobility and device feature. Three of the features we used in each type is listed in Table 1, where we use the term *regular* to represent whether a behavior appears consistently for at least seven days.

Let R be the full dataset, we create multiple subset $R^c \subseteq R$ by randomly sampling a fraction of users. For example, we denote a 1% sample of R by $R^{1\%}$. Each subset is further split into a training set Tr^c and a test set Te^c with ratio of 7:3. In order to measure the performance of our algorithms, we exclude 50% of the rated ads in the test set for use as holdout data (denoted by $Te^c_{\_}$). The reported performance values represent averages across 5 such runs. All algorithms are implemented on an Apache Spark cluster (Shoro & Soomro, 2015) with eight compute nodes (Intel Xeon 2.1 GHz CPU with 10 G RAM per node) and the raw data are stored on a Hadoop Distributed File System (HDFS).

4.2. Baselines and parameter settings

We compare our model with two broad types of Collaborative Filtering models: neighborhood-based models and latent factor models. We consider two subtypes of neighborhood-based models including: neighborhood-based models without clustering and neighborhood-based models with clustering. We consider several models of each type as summarized below.

4.2.1. Neighborhood-based models without clustering

We compare the following three neighborhood-based models:

- **RatingKNN.** The well-known user-based Collaborative Filtering method (Resnick et al., 1994), which utilizes the user ratings to compute the similarity or weight between users and produce recommendations by aggregating the top-k results of the users that are most similar to the target user.
- **FeatureKNN.** A user-based recommendation model that operates in a manner similar to RatingKNN except that it uses user features instead of ratings to compute the similarity between users (Xia et al., 2016).
- **PrefKNN.** The baseline method for top-N item recommendation in a user preference-based Collaborative Filtering model by distinguishing the qualitative user preferences from the rating matrix (Lee et al., 2016).

4.2.2. Neighborhood-based models with clustering

We augment the neighborhood-based models (PrefKNN and RatingKNN model) with an offline contextual feature-based clustering stage. Additionally, one of the well-known clustering recommendation models SCBPCC (Xue et al., 2005) is also extended

Table 1
Three of the contextual features we used in each type.

Category	Feature	Description
Routine	Dawn_Reg	The user regularly uses the mobile device at dawn
	Morning_Reg	The user regularly uses the mobile device in the morning
	Noon_Reg	The user regularly uses the mobile device at noon.
Mobility	LDM_Reg	The user regularly engages in long distance travel.
	Dawn_CRM	The user regularly engages in short distance movement at dawn.
	Morning_CRM	The user regularly engages in short distance movement in the morning.
Device	Is_largeScreen	The screen size of the user device is large(> = 5 inches).
	Is_HUAWEI	The brand of the user device is HUAWEI.
	Is_Android	The operating system model of the user device is Android.

for comparison. In our experiments, we set the number of cluster $|C| = \sqrt[3]{m}$. The resulting models are:

- **RatingKNN+.** RatingKNN method (Resnick et al., 1994) is extended with offline contextual feature-based k-means clustering.
- **PrefKNN+.** PrefKNN method (Lee et al., 2016) is extended with offline contextual feature-based k-means clustering.
- **SCBPCC+.** The original clustering-based smoothing recommendation approach (SCBPCC) (Xue et al., 2005) performs k-means clustering based on the Pearson Correlation Coefficient. In our setting, SCBPCC model is modified to use an appropriate similarity measure (instead of the original Pearson Correlation (See Section 4.4 for more details). We also tune the smoothing factor λ to optimize the performance of the resulting model.

4.2.3. Latent factor models

We implement three state-of-the-art latent factor models.

- **SVD++.** SVD-based model that utilize the set of items user u rated as implicit feedback information to help indicate users' preference (Koren, 2010b).
- **ALS-PMF.** The probabilistic matrix factorization model that assumes the latent factors of users and items are subject to zero-mean Gaussian Distribution (Mnih & Salakhutdinov, 2008).
- **ALS-IMF.** A matrix factorization model in which the rating matrix is treated as a combination of binary preferences and confidence values to stress implicit feedback (Hu et al., 2008). Specifically, we assign the implicit feedback variant α as 0.01.

We implemented all neighborhood-based models. When considering the proper number of nearest neighbors k, we vary k from 10 to 1000 and pick $k = 500$ based on the performance on the training data.

For SVD++model, we borrow the implementation on Apache Spark GraphX library[1]; for ALS-PMF and ALS-IMF models, we use the implementation provided as part of the Spark Machine Learning Library[2], where the "ALS-WR" approach (Zhou, Wilkinson, Schreiber, & Pan, 2008) is incorporated to avoid the reliance on the scale of regularization parameter λ. In addition, we set the number of latent factors f to 10, the regularization parameter λ to 0.01 and the number of iteration *iter* to 10 based on performance on the training data.

In addition, we provide comparisons with a simple non-personalized algorithm, **AdPopularity**, which commits the top-N popular items on the descending order of item's population regardless user ratings as in (Cremonesi et al., 2010; Lee et al., 2016).

[1] http://spark.apache.org/docs/latest/api/java/org/apache/spark/graphx/lib/SVDPlusPlus.html.

[2] http://spark.apache.org/docs/latest/ml-collaborative-filtering.html.

4.3. Evaluation metrics

Since we are interested in the accuracy of recommendation for targeting behaviors (the non-view behaviors), we employ precision-recall curve and normalized discounted cumulative gain (NDCG) (Järvelin & Kekäläinen, 2002) to evaluate the user-based recommender systems.

Let N be the number of top ads presented to each target user, $p@N$ be the average precision value and $r@N$ be the average recall value over all target users. Let A_u be the ads preferred by u based on u's holdout data and N_u denotes the output top-N recommended list of the algorithm, then precision and recall at N are computed as:

$$p@N = \underset{u \in \mathcal{U}}{\text{mean}} \frac{|N_u \cap A_u|}{|N_u|} \tag{11}$$

$$r@N = \underset{u \in \mathcal{U}}{\text{mean}} \frac{|N_u \cap A_u|}{|A_u|} \tag{12}$$

Next, let $DCG_u@N$ denote the cumulative relevance score at N and $IDCG_u@N$ be the optimal score at N for u, then $NDCG@N$ is computed as:

$$NDCG@N = \underset{u \in \mathcal{U}}{\text{mean}} \frac{DCG_u@N}{IDCG_u@N} \tag{13}$$

where:

$$DCG_u@N = \sum_{p=1}^{n} \frac{2^{I(N_{up} \in A_u)} - 1}{\log_2(p+1)} \tag{14}$$

In Eq. (14), function $I(P) = 1$ if and only if P is true; otherwise $I(P) = 0$. Note that users without any preferred ads in the holdout set are discarded since both the precision, recall and NDCG in such cases are fixed to zero.

4.4. Evaluation for similarity measures

The first experiment investigates the effectiveness of different similarity measures for the proposed neighborhood-based Collaborative Filtering methods. Here, we implemented the popular Pearson Correlation Coefficient (PCC) (Resnick et al., 1994), cosine similarity (COS) (Adomavicius & Tuzhilin, 2005), Jaccard similarity (Jaccard) (Koutrika et al., 2009), mean squared difference (MSD) (Cacheda, Carneiro, Fernández, Ndez, & Formoso, 2011), and the combination of Jaccard and MSD (JMSD). In these experiments, the elastic factor ε_u was set to $1/\log|S_{u+}|$.

Figs. 2 and 3 show that sensitivity to the choice of the similarity measures differs across the different algorithms. Models that use clustering are conspicuously more sensitive to the choice of the similarity measureas compared to those that do not rely on clustering. Thus, MSD, JMSD and PCC suffer significant degradation (approximately 10% in terms of the NDCG score) as seen in PrefKNN vs. PrefKNN+ and RatingKNN vs. RatingKNN+. The Jaccard similarity yields the best performance overall.

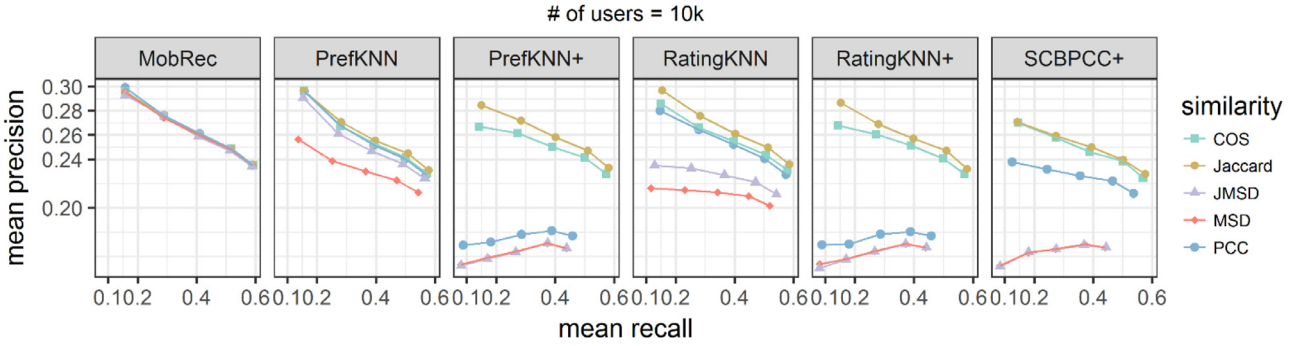

Fig. 2. Comparison of precision-recall curves on top-5 ads over different similarity models for neighborhood-based models.

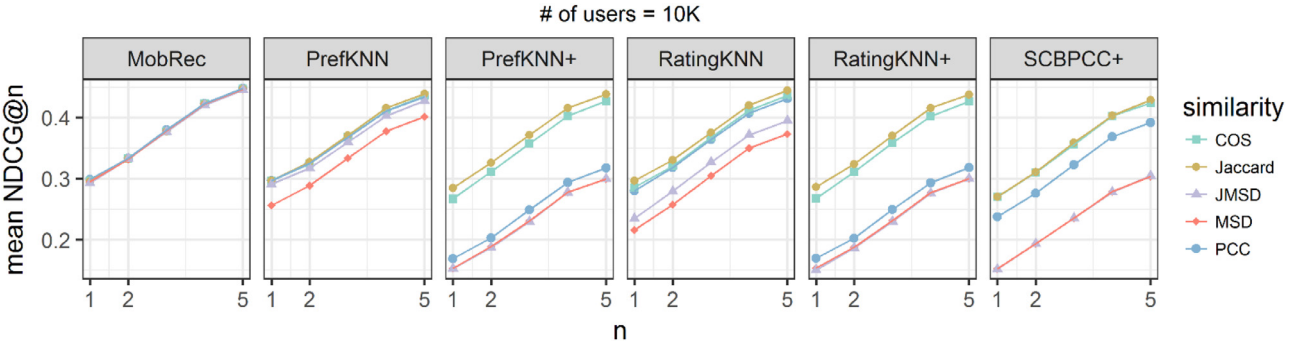

Fig. 3. Comparison of NDCG indexes on top-5 ads over different similarity models for neighborhood-based models.

Table 2
NDCG scores over different similarity models for our neighborhood-based models, where score entries are presented by the 95 percent confidence intervals.

Top-*N*	Algorithms	Similarity measures				
		COS	Jaccard	JMSD	MSD	PCC
Top-1	MobRec	0.296 ± 1.14%	0.295 ± 1.19%	0.293 ± 1.28%	0.295 ± 0.58%	0.299 ± 1.04%
	PrefKNN	0.296 ± 1.12%	0.292 ± 0.15%	0.292 ± 0.15%	0.252 ± 3.54%	0.296 ± 1.01%
	PrefKNN+	0.267 ± 1.70%	0.284 ± 0.60%	0.153 ± 2.07%	0.153 ± 0.81%	0.169 ± 1.44%
	RatingKNN	0.286 ± 1.01%	0.297 ± 1.16%	0.234 ± 2.01%	0.213 ± 2.06%	0.280 ± 1.38%
	RatingKNN+	0.268 ± 1.48%	0.286 ± 1.43%	0.150 ± 1.75%	0.153 ± 0.77%	0.169 ± 1.07%
	SCBPCC+	0.270 ± 2.27%	0.271 ± 4.85%	0.152 ± 2.69%	0.151 ± 0.81%	0.238 ± 1.49%
Top-2	MobRec	0.333 ± 1.07%	0.333 ± 1.17%	0.332 ± 0.79%	0.332 ± 0.33%	0.334 ± 0.89%
	PrefKNN	0.324 ± 0.46%	0.327 ± 0.43%	0.318 ± 0.08%	0.285 ± 2.92%	0.325 ± 0.53%
	PrefKNN+	0.311 ± 1.50%	0.326 ± 1.00%	0.187 ± 2.44%	0.188 ± 0.75%	0.203 ± 0.74%
	RatingKNN	0.321 ± 0.95%	0.330 ± 1.10%	0.277 ± 2.24%	0.253 ± 2.21%	0.318 ± 0.67%
	RatingKNN+	0.311 ± 1.53%	0.324 ± 1.41%	0.186 ± 2.47%	0.188 ± 0.80%	0.202 ± 0.75%
	SCBPCC+	0.310 ± 1.80%	0.311 ± 3.83%	0.193 ± 1.51%	0.194 ± 0.50%	0.276 ± 0.85%
Top-5	MobRec	0.447 ± 0.89%	0.447 ± 0.97%	0.445 ± 1.40%	0.447 ± 0.45%	0.448 ± 0.99%
	PrefKNN	0.435 ± 0.94%	0.439 ± 0.93%	0.430 ± 0.75%	0.399 ± 2.71%	0.434 ± 1.04%
	PrefKNN+	0.427 ± 0.77%	0.438 ± 0.66%	0.299 ± 0.98%	0.299 ± 0.35%	0.318 ± 0.36%
	RatingKNN	0.436 ± 0.31%	0.444 ± 0.86%	0.396 ± 2.77%	0.371 ± 2.35%	0.431 ± 0.33%
	RatingKNN+	0.426 ± 0.84%	0.438 ± 0.84%	0.300 ± 1.04%	0.300 ± 0.33%	0.318 ± 0.49%
	SCBPCC+	0.424 ± 1.79%	0.429 ± 3.03%	0.304 ± 1.05%	0.303 ± 0.34%	0.392 ± 0.29%

Table 2 shows that the performance of MobRec is stable over the different similarity measures. This is explained perhaps by the ability of MobRec to combine both context based and implicit feedback based information.

4.5. Evaluation for large scale dataset

Our second experiment examines the accuracy of all competing algorithms, and their scalability, using a large-scale dataset. Specifically, we fix the similarity measure to Jaccard model, the number of nearest neighbors to 500 and vary the user volume of subsampling dataset from 50 K ($R^{5\%}$) to 0.5 M ($R^{50\%}$). The elastic factor ε_u of user u in MobRec is $1/\log|S_{u+}|$.

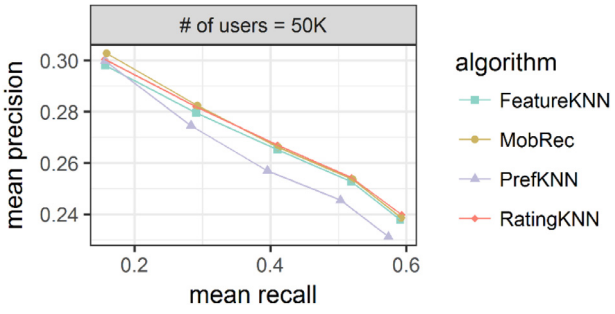

Fig. 4. Comparison of precision-recall curves on top-5 ads over different size of data for neighborhood-based models without clustering.

Table 3
NDCG scores over different size of data for all competing methods, where score entries are presented by the 95 percent confidence intervals. We use the symbol '—' to denote that the corresponding algorithm fails to process the specific data volume.

Top-*N*	Algorithms	# of users		
		50 K	0.1 M	0.5 M
Top-1	FeatureKNN	0.298 ± 0.42%	—	—
	PrefKNN	0.300 ± 0.57%	—	—
	RatingKNN	0.300 ± 0.99%	—	—
	MobRec	0.303 ± 0.32%	0.305 ± 0.26%	0.305 ± 0.32%
	PrefKN N+	0.298 ± 0.45%	0.297 ± 0.40%	0.303 ± 0.38%
	RatingKN N+	0.290 ± 0.57%	0.292 ± 0.51%	0.293 ± 0.18%
	SCBPC C+	0.295 ± 0.72%	0.298 ± 1.05%	0.290 ± 1.60%
	AdPopularity	0.296 ± 0.30%	0.295 ± 0.43%	0.299 ± 0.33%
	ALS-IMF	0.259 ± 0.34%	0.259 ± 0.35%	0.260 ± 0.21%
	ALS-PMF	0.255 ± 0.14%	0.256 ± 0.14%	0.253 ± 0.12%
	SVD++	0.262 ± 1.60%	0.276 ± 0.45%	0.264 ± 0.77%
Top-2	FeatureKNN	0.336 ± 0.29%	—	—
	PrefKNN	0.331 ± 0.55%	—	—
	RatingKNN	0.337 ± 1.00%	—	—
	MobRec	0.339 ± 0.32%	0.341 ± 0.38%	0.341 ± 0.22%
	PrefKN N+	0.327 ± 0.12%	0.327 ± 0.51%	0.331 ± 0.37%
	RatingKN N+	0.326 ± 0.36%	0.328 ± 0.50%	0.330 ± 0.23%
	SCBPC C+	0.331 ± 0.79%	0.336 ± 1.10%	0.330 ± 1.28%
	AdPopularity	0.323 ± 0.09%	0.323 ± 0.52%	0.325 ± 0.32%
	ALS-IMF	0.296 ± 0.49%	0.295 ± 0.31%	0.296 ± 0.18%
	ALS-PMF	0.288 ± 0.0.28%	0.288 ± 0.15%	0.285 ± 0.12%
	SVD++	0.299 ± 0.85%	0.306 ± 0.52%	0.295 ± 0.75%
Top-5	FeatureKNN	0.449 ± 0.22%	—	—
	PrefKNN	0.438 ± 0.79%	—	—
	RatingKNN	0.451 ± 0.88%	—	—
	MobRec	0.451 ± 0.22%	0.452 ± 0.30%	0.452 ± 0.17%
	PrefKN N+	0.437 ± 0.12%	0.435 ± 0.27%	0.439 ± 0.35%
	RatingKN N+	0.440 ± 0.36%	0.441 ± 0.37%	0.443 ± 0.17%
	SCBPC C+	0.444 ± 0.52%	0.447 ± 0.86%	0.442 ± 1.09%
	AdPopularity	0.431 ± 0.19%	0.429 ± 0.27%	0.432 ± 0.27%
	ALS-IMF	0.411 ± 0.31%	0.411 ± 0.40%	0.411 ± 0.09%
	ALS-PMF	0.397 ± 0.21%	0.397 ± 0.28%	0.392 ± 0.13%
	SVD + +	0.412 ± 0.65%	0.414 ± 0.29%	0.409 ± 0.72%

Fig. 4 reports the results for neighborhood-based models without clustering. All algorithms achieve comparable performance in the case of the top-1 ad; however, the performance of PrefKNN drops dramatically as the number of recommended ads grows. Though the performacne of MobRec, FeatureKNN and RatingKNN are similar, MobRec offers significant advantage on large-scale datasets. As shown in Table 3, all non-clustering-based neighborhood models fail when user volume grows to 0.1 M. Online runtime comparison between MobRec and neighborhood-based models without clustering is shown in Table 4. MobRec holds conspicuous advantage over other models especially when compared with RatingKNN and PrefKNN. As user volume increases, online runtime of neighborhood-based models without clustering increases dramatically. As we use offline clustering for neighbor pre-selection, online runtime is significantly reduced when the available computational resources are held constant.

Fig. 5 shows the results for clustering-based neighborhood models. The results clearly show that offline clustering dramatically enhances the scalability of neighborhood-based models, with modest drops in performance. RatingKNN + shows greater degradation compared to PrefKNN + . We find that the performance of SCBPCC + is rather unstable whereas MobRec consistently outperforms other clustering-based models by approximately 1% in terms of the NDCG score.

As shown in Fig. 6, the performance of latent factor models appear to be worse than that of neighborhood-based models. ALS-IMF consistently outperforms ALS-PMF by over 0.7% with respect to the NDCG index. In addition, AdPopularity, the non-personalized model, outperforms all latent factor models especially for the top-1 ad, suggesting that the user ratings are excessively noisy, and consequently models that directly optimize the loss on rating matrix are likely to suffer higher risk of overfitting. MobRec shows the best performance in both settings.

4.6. Impacts of elastic factor ε

The elastic factor ε can be employed to shrink or augment the confidence on the observed ratings for specific user as shown in Eq. (10), we consider three variations:

- **MobRec_1**: Confidence in user preferences are considered equal for all users, hence we set $\varepsilon_u = 1$, which is identical to the setting in PrefKNN and PrefKNN + (Lee et al., 2016).
- **MobRec_aug**: As suggested by existing works (Koren, 2010b; Liu et al., 2014), the more operations a user has performed, the greater the confidence we can have in the estimated user preferences. Hence, we specify $\varepsilon_u = \log |S_{u+}|$ to augment the confidence on longer S_{u+}.
- **MobRec**: We assume that mobile users are more prone to casual clicks or fraudulent clicks that do not faithfully depict the actual latent user preferences. Based on the finding reported in (Hu et al., 2017) that the behavior patterns of mobile users approximately follows the power-law distribution, we aim to shrink the confidence on the long-tail of the distribution. Therefore, we set $\varepsilon_u = 1/\log |S_{u+}|$.

Fig. 7(a)(b) shows that the performance of our model is sensitive to the choice of ε. MobRec outperforms MobRec_aug by over 5% and MobRec_1 by 1% with respect to NDCG scores for the top-1 ad. The observed advantage of MobRec suggests that the shrinkage strategy better matches the complexities of user behavior in the mobile ads scenario. First, mobile users with sparse behavior records are excluded in our dataset during the preprocessing stage (see Section 4.1). Second, we assume that mobile users are prone to false clicks or fraudulent clicks that do not reflect their true preferences. For example, walking, riding a bus, etc., contribute to misclicks on mobile ads. Further, the pay-per-action practice in the mobile ads market could incentivize malicious generation of fake clicks on the ads. The shrinkage strategy in our experiment is aligned with the observation made by other authors that behavior records that are neither too sparse nor too dense are likely to more accurately reflect user preferences (Tian, Zhu, Xia, Zhuang, & Zhang, 2015). In our experiments, reducing the confidence in the estimated preferences for users with intense activity shows significant positive effect on performance.

5. Conclusion

Scalability of recommender systems in real-world settings often is achieved at the expense of quality of the recommendations. In this paper, we based on a careful examination of user behaviors from a large-scale real-world mobile advertising dataset and proposed a novel two-stage approach (MobRec) that achieves scalability without compromising the quality of recommendations. The proposed approach leverages two sources of information within a neighborhood-based recommendation model: slowly-changing mobile context and dynamic implicit feedback. The two stages are: 1) an offline clustering stage which uses the mobile context to cluster users into stable clusters that group users that share similar contextual features; and 2) an online stage that focuses only on the cluster whose members are most similar to the target user to identify the nearest neighbors of target user taking into account user preferences reflected in the user feedback. The proposed model makes use of an additional parameter that ensures robust estimates of user preferences in the presence of misclicks and fake clicks. We have reported results of extensive experiments

Table 4
Online runtime over different size of data for the neighborhood-based models without clustering and MobRec, where the 95% confidence interval for the runtime entries are shown. (unit: second).

Algorithm	# of users				
	10 K	20 K	30 K	40 K	50 K
MobRec	51.6 ± 5.2	94.4 ± 12.2	138.6 ± 17.7	190.6 ± 20.9	280.4 ± 59.5
FeatureKNN	29.0 ± 1.2	103.6 ± 10.3	205.8 ± 8.6	385.6 ± 16.0	589.0 ± 18.5
PrefKNN	311.8 ± 34.7	1171.6 ± 157.2	2789.6 ± 387.1	4962.7 ± 176.2	7458.0 ± 905.4
RatingKNN	335.4 ± 101.9	1137.0 ± 81.4	2590.0 ± 101.8	4983.3 ± 462.4	7985.3 ± 4694.8

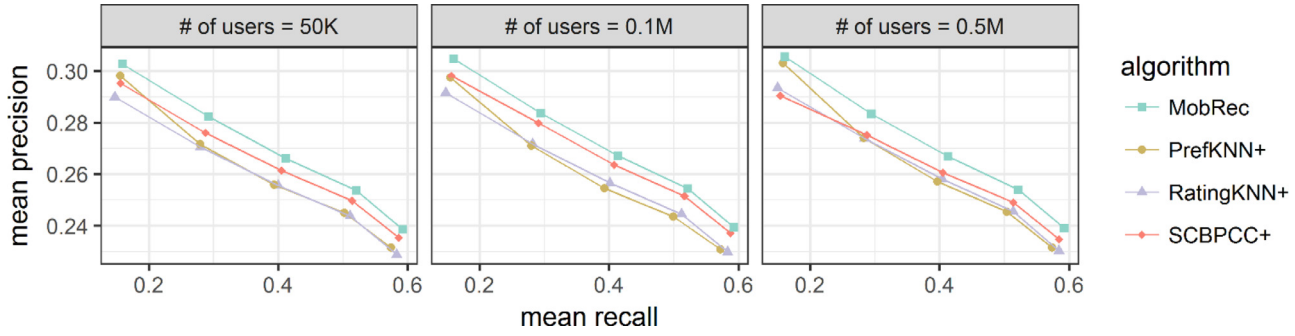

Fig. 5. Comparison of precision-recall curves on top-5 ads over different size of data for clustering-based neighborhood models.

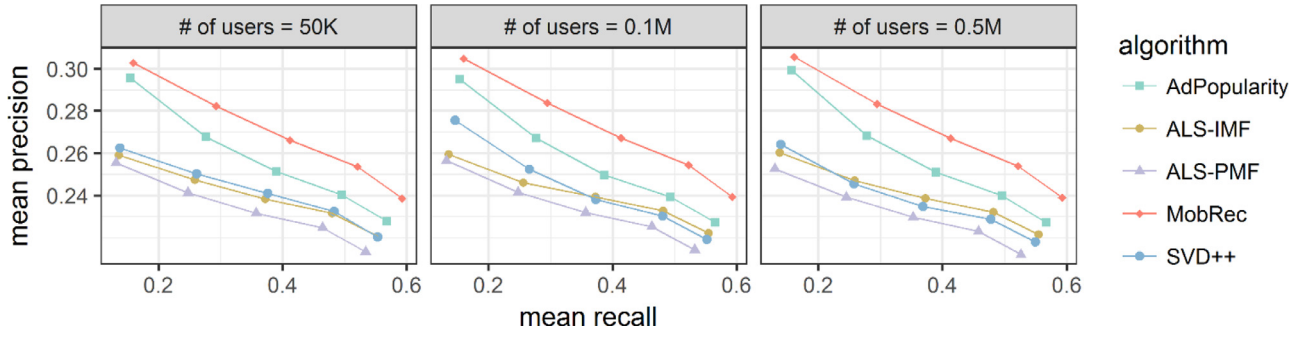

Fig. 6. Comparison of precision-recall curves on top-5 ads over different size of data for latent factor models.

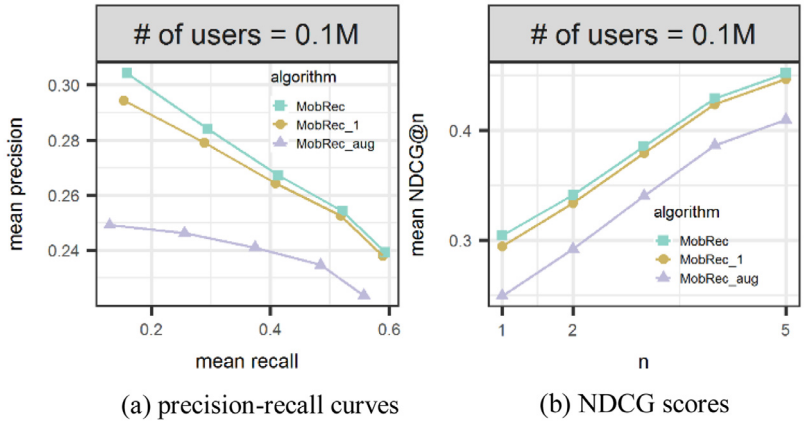

Fig. 7. Comparison results on top-5 ads over different settings of elastic factor ε.

that show that MobRec, the proposed approach, outperforms both neighborhood-based and latent factor methods in terms of run time as well as quality of results.

The work described is not without limitations. First, all of our experimental evaluation is based on a single large-scale real-world dataset. Additional studies, using additional real-world mobile advertising datasets are needed to confirm our findings. Some promising directions for further research include: (i) Principled approaches to optimizing the elastic factor ε_u; (ii) Investigation of the impact of slow drifts in mobile context on performance; (iii) Investigattion of the ways to tradeoff scalability and quality of recommendations using a combination of online and offline methods in large-scale real-world settings.

Competing interest

The authors declare that they have no competing interests.

Acknowledgements

This work is supported in part by the Science and Technology Planning Project of Guangdong Province, China [No. 2013B090500087, No. 2014B010112006], the Scientific Research Joint Funds of Ministry of Education of China and China Mobile [No. MCM20150512], and the State Scholarship Fund of China Scholarship Council [No. 201606155088] and the Edward Frymoyer Endowed Chair in Information Sciences and Technology at Pennsylvania State University [held by Professor Honavar].

References

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering, 17*(6), 734–749.

Agarwal, D., Chen, B. C., & Elango, P. (2010). Fast online learning through offline initialization for time-sensitive recommendation. In *Proceedings of the 16th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 703–712).

Aggarwal, C. C. (2016a). Model-based collaborative filtering. In *Recommender systems* (pp. 71–138). Springer.

Aggarwal, C. C. (2016b). Neighborhood-Based Collaborative Filtering. In *Recommender systems* (pp. 29–70). Springer.

Ahn, H. J. (2008). A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences, 178*(1), 37–51.

Amatriain, X., Jaimes, A., Oliver, N., & Pujol, J. M. (2011). Data mining methods for recommender systems. In *Recommender systems handbook* (pp. 39–71). Springer.

Anastasiu, D. C., Christakopoulou, E., Smith, S., Sharma, M., & Karypis, G. (2016). Big Data and Recommender Systems.

Anastasiu, D. C., & Karypis, G. (2014). L2ap: Fast cosine similarity search with prefix ʟ-2 norm bounds. In *Data Engineering (ICDE), 2014 IEEE 30th International Conference on* (pp. 784–795). IEEE.

Awekar, A., & Samatova, N. F. (2009). Fast matching for all pairs similarity search. In *Web Intelligence and Intelligent Agent Technologies, 2009. WI-IAT'09. IEEE/WIC/ACM International Joint Conferences on: 1* (pp. 295–300). IEEE.

Balakrishnan, S., & Chopra, S. (2012). Collaborative ranking. In *Proceedings of the fifth ACM international conference on Web search and data mining* (pp. 143–152). ACM.

Bayardo, R. J., Ma, Y., & Srikant, R. (2007). Scaling up all pairs similarity search. In *Proceedings of the 16th international conference on World Wide Web* (pp. 131–140). ACM.

Bell, R. M., & Koren, Y. (2007). Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on* (pp. 43–52). IEEE.

Cacheda, F., Carneiro, V., Fernández, D., Ndez, D., & Formoso, V. (2011). Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. *ACM Transactions on the Web, 5*(1), 161–171.

Cremonesi, P., Koren, Y., & Turrin, R. (2010). Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems* (pp. 39–46).

Goldberg, K., Roeder, T., Gupta, D., & Perkins, C. (2001). Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval, 4*(2), 133–151.

Hu, J., Liang, J., & Dong, S. (2017). iBGP: A Bipartite Graph Propagation Approach for Mobile Advertising Fraud Detection. *Mobile Information Systems, 2017*, 1–12.

Hu, Y., Koren, Y., & Volinsky, C. (2008). Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining* (pp. 263–272).

Järvelin, K., & Kekäläinen, J. (2002). Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS), 20*(4), 422–446.

Jeong, B., Lee, J., & Cho, H. (2010). Improving memory-based collaborative filtering via similarity updating and prediction modulation. *Information Sciences, 180*(5), 602–612.

Katukuri, J., Mukherjee, R., Konik, T., & Konik, T. (2013). Large-scale recommendations in a dynamic marketplace. *Workshop on Large Scale Recommendation Systems at RecSys*.

Koren, Y. (2010a). Collaborative filtering with temporal dynamics. *Communications of the ACM, 53*(4), 89–97.

Koren, Y. (2010b). Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data (TKDD), 4*(1), 1–24.

Koren, Y., & Bell, R. (2015). Advances in collaborative filtering. In *Recommender systems handbook* (pp. 77–118). Boston, MA: Springer.

Koutrika, G., Bercovitz, B., & Garcia-Molina, H. (2009). FlexRecs: Expressing and combining flexible recommendations. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data* (pp. 745–758).

Lee, J., Lee, D., Lee, Y. C., Hwang, W. S., & Kim, S. W. (2016). Improving the accuracy of top-N recommendation using a preference model. *Information Sciences, 348*, 290–304.

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

Liu, H. F., Hu, Z., Mian, A., Tian, H., & Zhu, X. Z. (2014). A new user similarity model to improve the accuracy of collaborative filtering. *Knowledge-Based Systems, 56*, 156–166.

Liu, N. N., & Yang, Q. (2008). Eigenrank: A ranking-oriented approach to collaborative filtering. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 83–90). ACM.

Ma, H., King, I., & Lyu, M. R. (2007). Effective missing data prediction for collaborative filtering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 39–46).

Ning, X., Desrosiers, C., & Karypis, G. (2015). A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook* (pp. 37–76). Boston, MA: Springer.

Park, H.-S., Park, M.-H., & Cho, S.-B. (2015). Mobile information recommendation using multi-criteria decision making with Bayesian network. *International Journal of Information Technology & Decision Making, 14*(02), 317–338.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work* (pp. 175–186).

Mnih, A., & Salakhutdinov, R. R. (2008). Probabilistic matrix factorization. In *Advances in neural information processing systems* (pp. 1257–1264). 2–1.

Shoro, A. G., & Soomro, T. R. (2015). Big Data analysis: Apache Spark perspective. *Global Journal of Computer Science and Technology, 15*(1).

Tian, T., Zhu, J., Xia, F., Zhuang, X., & Zhang, T. (2015). Crowd fraud detection in internet advertising. In *Proceedings of the 24th International Conference on World Wide Web* (pp. 1100–1110). International World Wide Web Conferences Steering Committee.

Wang, Y., Feng, D., Li, D., Chen, X., Zhao, Y., & Niu, X. (2016). A mobile recommendation system based on logistic regression and Gradient Boosting Decision Trees. In *Neural Networks (IJCNN), 2016 International Joint Conference on* (pp. 1896–1902). IEEE.

Wu, T.-J., Fang, S.-H., Wu, Y.-B., Wu, C.-T., Haung, J.-W., & Tsao, Y. (2016). A study of mobile advertisement recommendation using real big data from AdLocus. In *Consumer Electronics, 2016 IEEE 5th Global Conference on* (pp. 1–2). IEEE.

Xia, Z., Wen, M. S., Wang, L., Ma, Q., Chen, D. T., & Chen, P. J. (2016). Building a Bi-directed Recommendation System for Mobile Users and App-install Ad Campaigns. *TargetAd - 2nd International Workshop on Ad Targeting at Scale at WSDM*.

Xu, B., Bu, J. J., Chen, C., & Cai, D. (2012). An exploration of improving collaborative recommender systems via user-item subgroups. In *Proceedings of the 21st international conference on World Wide Web* (pp. 21–30).

Xue, G. R., Lin, C. X., Yang, Q., Xi, W. S., Zeng, H. J., & Yu, Y. (2005). Scalable collaborative filtering using cluster-based smoothing. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 114–121).

Yuan, S.-T., & Tsao, Y. W. (2003). A recommendation mechanism for contextualized mobile advertising. *Expert Systems with Applications, 24*(4), 399–414.

Zhou, Y., Wilkinson, D., Schreiber, R., & Pan, R. (2008). Large-scale parallel collaborative filtering for the netflix prize. In *International Conference on Algorithmic Applications in Management* (pp. 337–348). Springer.

Zhu, H., Chen, E., Xiong, H., Yu, K., Cao, H., & Tian, J. (2015). Mining mobile user preferences for personalized context-aware recommendation. *ACM Transactions on Intelligent Systems and Technology (TIST), 5*(4), 58.