Exploiting User Actions for App Recommendations

Kai Shu*, Suhang Wang*, Jiliang Tang†, Yi Chang‡, Ping Luo§, and Huan Liu*
*Arizona State University, {kai.shu, suhang.wang, huan.liu}@asu.edu
†Michigan State University, tangjili@msu.edu
‡Jilin University, yichang@acm.org
§Chinese Academy of Sciences, luop@ict.ac.cn

***Abstract*—Mobile Applications (or Apps) are becoming more and more popular in recent years, which has attracted increasing attention on mobile App recommendations. The majority of existing App recommendation algorithms focus on mining App functionality or user usage data for discovering user preferences; while actions taken by a user when he/she decides to download an App or not are ignored. In realistic scenarios, a user will first view the description of the App and then decide if he/she wants to download it or not. The actions such as *viewing* or *downloading* provide rich information about users' preferences and tastes for Apps, which have great potentials to advance App recommendations. However, the work on exploring action data for App recommendations is rather limited. Therefore, in this paper we study the novel problem of exploiting user actions for App recommendations. We propose a new framework *ActionRank*, which simultaneously captures various signals from user actions for App recommendations. Experimental results on real-world datasets demonstrate the effectiveness of the proposed framework.**

***Keywords*-App recommendations, user behaviors, one-class collaborative filtering**

I. Introduction

Mobile Apps have become increasingly popular in recent years. With the huge amount of Apps in App markets every year, it's difficult for people to find the Apps they are interested in. Thus, App markets, such as Apple Store[1] and Google Play Store[2], categorize the Apps and recommend specific Apps for users. However, the ranking list of Apps provided by App markets are usually general for all users which cannot satisfy the personalized user tastes. Thus, a personalized App ranking list is more desirable that has cultivated numerous App recommender systems by mining user's personal interests from various sources such as reviews written by users [1], App functionalities [2] and information from Twitter [3].

Figure 1 gives a typical scenarios when a user u_1 attempts to find an App satisfying his/her tastes. User u_1 first browses the App lists and gets brief and basic information about Apps such as icon images, App genres and user ratings. If she is attracted by an App, she will *view* the detailed App description to decide whether to *get* it or not. This aforementioned process produces massive user action data such as *viewing* or *downloading*. These actions could provide useful and personalized signals about users' preferences towards Apps. First, user actions can indicate the ranking of users' preferences towards Apps (or *preference ranking signal*). For example, u_1 is more likely to prefer those Apps she has viewed, such as v_1 and v_3, rather than other Apps she does not perform actions on, such as v_2. Second, different users could take actions on Apps at different time; for example, officers may need to work at daytime and usually view or get Apps during lunch time or after work, while students are more flexible to access the platform. Moreover, Apps from different categories have distinct temporal distributions on action taken time within a cyclic period, such as daily or weekly. Thus user actions afford rich temporal information (or *temporal signal*) about user preferences. Third, the descriptions of the Apps viewed/downloaded by a user (or *content signal*) are strongly related to users' preferences. For example, u_1 views and downloads App v_1 with description d_1 containing "invincible power", she is more likely to play role-playing games. Similarly, u_1 views App v_2 with description d_2 including "wonderful sound quality", which also indicates her preference to music Apps. User actions provide strong signals for users' preferences that can be explored to build effective App recommender systems. However, the work on exploring user actions for App recommendations is rather limited.

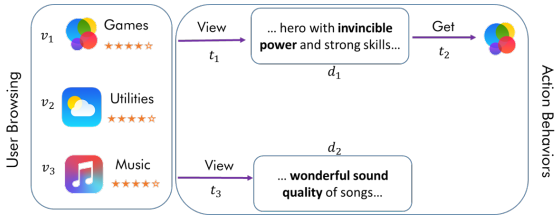


Figure 1. An example of user action behaviors in App market. During the process of user browsing App list, the user views App v_1 and downloads it, views App v_3 without downloading it, and does not perform action on App v_2.

In this paper, we study the user action driven top-k App recommendation problem by exploiting the aforementioned three signals about users' preferences. In essence, we investigate: (1) how to model these signals mathematically; (2) how to take advantage of these signals from user actions for mobile App recommendations. In an attempt to solve these two challenges, we propose a novel framework ActionRank

[1]https://itunes.apple.com/us/genre/ios/id36?mt=8
[2]https://play.google.com/store/apps

that captures various signals from user actions for personalized mobile App recommendations. The main contributions of the paper are summarized as follows:

- We provide a principled way to model signals from user actions;
- We propose a new framework ActionRank which integrates various signals from user actions into a coherent model for personalized App recommendations;
- We conduct experiments on real-world datasets to demonstrate the effectiveness of the proposed framework ActionRank.

II. Modeling User Actions for App Ranking

In this section, we discuss the details of extracting preference ranking, temporal and content signals, and how to integrate them for App personalized recommendations.

A. Modeling Preference Ranking Signal

During users' browsing process for finding their interested Apps, they can decide to take action on the Apps or not. Thus, we can learn users' preference to different Apps based on whether they take actions on the Apps or not. For simplicity, we use download action to illustrate the idea.

We denote $S \subset \mathcal{U} \times \mathcal{V}$ to be the observed actions from users to all Apps, then we define $\mathcal{V}_i^+ := \{v_j \in \mathcal{V} : (u_i, v_j) \in S\}$ which contains all the Apps that user u_i performed actions on. If an App v_j is downloaded by user u_i (i.e. $v_j \in \mathcal{V}_i^+$), then she prefers v_j over any App v_k in the set of Apps which she does not download (i.e. $v_k \in \mathcal{V} \backslash \mathcal{V}_i^+$). Thus, we represent the training set to learn the preference ranking signal as a set $D_S \in \mathcal{U} \times \mathcal{V} \times \mathcal{V}$, which is defined as follows,

$$D_S = \{(u_i, v_j, v_k) | v_j \in \mathcal{V}_i^+ \wedge v_k \in \mathcal{V} \backslash \mathcal{V}_i^+\} \tag{1}$$

To compute the preference of user u_i towards App v_j, we adopt the matrix factorization method as the basic model to learn the latent representations of users and items. Let $\mathbf{X} \in \{0,1\}^{n \times m}$ denote the user-App action matrix, it is approximated by $\hat{\mathbf{X}}$ which is the product of two low-rank matrices $\mathbf{U} \in \mathbb{R}^{n \times K}$ and $\mathbf{V} \in \mathbb{R}^{m \times K}$ which satisfies,

$$\hat{\mathbf{X}} = \mathbf{U}\mathbf{V}^T \tag{2}$$

where K is the dimension of latent factors, $\mathbf{U}$ is the user latent factor matrix with each row $\mathbf{u}_i$ being u_i's latent factor and $\mathbf{V}$ is the App latent factor matrix with each row $\mathbf{v}_j$ being the latent factor for App v_j. Thus the predicted preference of user u_i to App v_j $\hat{x}_{ij}$ can be represented by,

$$\hat{x}_{ij} = \mathbf{u}_i \cdot \mathbf{v}_j^T \tag{3}$$

The objective becomes that we want to rank the observed Apps (e.g., v_j) higher than all non-observed Apps (e.g., v_k). Similar to Bayesian Personalized Ranking (BPR) [4], we define $\hat{x}_{ijk}$, which can capture the relationship between user u_i, App v_j and App v_k, as below,

$$\hat{x}_{ijk} = \hat{x}_{ij} - \hat{x}_{ik} \tag{4}$$

which actually differentiates the ranking of Apps that have been viewed or downloaded (e.g. v_j) with the ones having no actions observed (e.g. v_k) by user u_i. Then the probability that the user u_i prefers App v_j to App v_k can be computed by $\sigma(\hat{x}_{ij} - \hat{x}_{ik})$ where σ is a sigmoid function (i.e. $\sigma(x) = \frac{1}{1+e^{-x}}$). Thus, the objective function is to minimize the negative log likelihood as follows,

$$\min_{\mathbf{U},\mathbf{V}} \sum_{(u_i,v_j,v_k) \in D_S} -\ln \sigma(\mathbf{u}_i(\mathbf{v}_j^T - \mathbf{v}_k^T)) + \varphi(||\mathbf{U}||_F^2 + ||\mathbf{V}||_F^2) \tag{5}$$

where the term $\varphi(||\mathbf{U}||_F^2 + ||\mathbf{V}||_F^2)$ is added to avoid over-fitting.

B. Modeling Temporal and Content Signals

From the preference ranking signal, we can obtain the user preference vector $\mathbf{u}_i$ via Eq. (5) for each user u_i. For each user u_i, we can extract a set of features from the temporal and content signals as $\mathbf{f}_i \in \mathbb{R}^{1 \times d}$ and we assume there is a embedding matrix $\mathbf{W} \in \mathbb{R}^{d \times K}$ that can map $\mathbf{f}_i$ to $\mathbf{u}_i$ linearly by following $\mathbf{u}_i = \mathbf{f}_i\mathbf{W}$. Given $\mathbf{u}_i$ and $\mathbf{f}_i$, the mapping function $\mathbf{W}$ can be obtained by:

$$\min_{\mathbf{W}} \sum_{u_i \in \mathcal{U}} ||\mathbf{u}_i - \mathbf{f}_i\mathbf{W}||_2^2 + \zeta||\mathbf{W}||_F^2 \tag{6}$$

where the term $\zeta||\mathbf{W}||_F^2$ is introduced to avoid over-fitting.

Next we give more details about how to construct $\mathbf{f}_i$ from the temporal and content signals.

Extracting Temporal Features

To model this non-uniformness property of user actions, we first introduce temporal state $t \in [1, T]$ to represent the hour of the day, where $T = 24$ is the total number of temporal states. Then we represent $\mathbf{f}_i^{(1)'}$ as,

$$\mathbf{f}_i^{(1)'} = (n_1, n_2, ..., n_T) \tag{7}$$

where n_t is the number of Apps taken actions on by user u_i in the t-th temporal state. For easier comparison of temporal features among different users, we further normalize the vector with the corresponding $z-$score [5] as shown below,

$$Z(n_k) = \frac{n_k - \mu(\mathbf{f}_i^{(1)'})}{\sigma(\mathbf{f}_i^{(1)'})} \tag{8}$$

where $\mu(\mathbf{f}_i^{(1)'})$ and $\sigma(\mathbf{f}_i^{(1)'})$ are the mean and standard deviation of feature vector $\mathbf{f}_i^{(1)'}$, respectively. Therefore, the normalized temporal feature vector can be represented as,

$$\mathbf{f}_i^{(1)} = (Z(n_1), Z(n_2), ..., Z(n_T)) \tag{9}$$

Table I
THE STATISTICS OF DATASETS

Datasets	ViewData	GetData
No. of users	$10,627$	$10,583$
No. of Apps	$12,730$	$5,721$
No. of actions	$1,275,806$	$288,878$
Density	9.4×10^{-3}	4.8×10^{-3}

Extracting Content Features Since the content signal is from the set of the descriptions of the Apps a user has taken actions on, to construct features from the content signal, a natural choice is to use topic models such as LDA ([6]) to learn the latent topic distributions from descriptions. In this way, all the Apps that have been visited by user u_i can be represented as a vector $\mathbf{f}_i^{(2)}$ that denotes users' preference distribution over these latent topics.

To this end, we can concatenate the above features together and generate $\mathbf{f}_i$ as follows,

$$\mathbf{f}_i := [\mathbf{f}_i^{(1)}, \mathbf{f}_i^{(2)}] \tag{10}$$

and the relative importance of temporal and content features can be optimized by using the weight matrix $\mathbf{W}$.

C. The Proposed Framework-ActionRank

With Equation 5 modeling the preference ranking signal and Equation 6 modeling the temporal and content signals, the proposed framework $ActionRank$ solve the following optimization problem to model these three signals:

$$\begin{aligned}\min_{\mathbf{U},\mathbf{V},\mathbf{W}} \sum_{(u_i,v_j,v_k)\in D_S} & -\ln \sigma(\mathbf{u}_i(\mathbf{v}_j^T - \mathbf{v}_k^T)) \\ & +\alpha \sum_{u_i \in \mathcal{U}} ||\mathbf{u}_i - \mathbf{f}_i\mathbf{W}||_2^2 \\ & +\lambda(||\mathbf{U}||_F^2 + ||\mathbf{V}||_F^2 + ||\mathbf{W}||_F^2)\end{aligned} \tag{11}$$

where $\mathbf{U} \in \mathbb{R}^{n\times K}$ and $\mathbf{V} \in \mathbb{R}^{m\times K}$ are the latent matrices of users and Apps, respectively. $\mathbf{W} \in \mathbb{R}^{d\times K}$ is a linear mapping matrix that connects the temporal and content signals with the preference ranking signal. $\mathbf{f}_i$ is the set of the features on user u_i extracted from the temporal and content signals. α is a scalar which controls the contributions from the temporal and content signals. $\lambda(||\mathbf{U}||_F^2 + ||\mathbf{V}||_F^2 + ||\mathbf{W}||_F^2)$ is introduced to avoid over-fitting problem.

Since objective function in Equation 11 is non-convex with respect to $\mathbf{U}$,$\mathbf{V}$ and $\mathbf{W}$, we use alternating least square method which optimizes one variable by fixing the other variables.

III. EXPERIMENTAL EVALUATION

In this section, we will conduct experiments on real-world datasets to demonstrate the effectiveness of the proposed framework.

A. Datasets and Experimental Settings

We use the user log data from an iPhone App company named *Limited-time Free*[3], which collects the information of those Apps for some time in Apple App Store. This platform can record two types of actions for each user[4]: 1) *ViewApp* which means that users pick up an App and view the detailed description, and 2) *GetApp* that indicates that users download some Apps after viewing the App descriptions. Users perform two types of actions in the dataset including viewing and downloading. Thus, we construct two datasets from the dataset: ViewData and GetData (see Table I).

We randomly hide a fraction of positive user-App pairs and use the remaining $x\%$ of all Apps to form a partially observed matrix for training. The hidden user-App pairs form the test set. We repeat the generating process of training/test set for three times and the average performance is reported. We use top-k evaluation metrics to measure the recommendation performance:$Precision@k = \frac{1}{|\mathcal{U}|}\sum_{u_i\in\mathcal{U}} \frac{|TopK(u_i)\cap App(u_i)|}{|TopK(u_i)|}$ and $Recall@k = \frac{1}{|\mathcal{U}|}\sum_{u_i\in\mathcal{U}} \frac{|TopK(u_i)\cap App(u_i)|}{|App(u_i)|}$. where $TopK(u_i)$ is the set of Apps recommended to user u_i that u_i has not visited in the training set. $App(u_i)$ indicates the set of Apps that have been performed actions in testing set. In our experiment, k is set to 5 and 10, respectively.

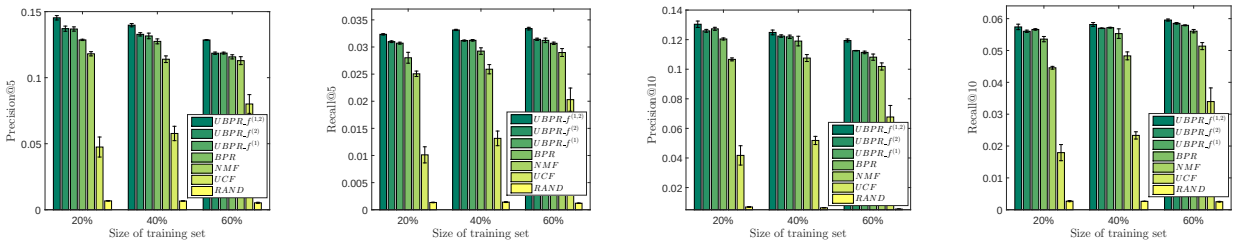
B. Recommendation Performances

The ranking methods for comparison include: **RAND:** which rank Apps randomly for users and is non-personalized. **UCF:** User-oriented collaborative filtering, which finds a set of users similar to the target user and recommends those Apps that has been viewed or downloaded by those similar users to the target user; **NMF:** Nonnegative Matrix Factorization based collaborative filtering [7]; **BPR:** Bayesian Personalized Ranking optimization for MF [4], which is the state-of-art approach for implicit feedback data; **UBPR:** which considers the temporal and content signals from user actions. Note that there are two set of features: $\mathbf{f}^{(1)}$ from the temporal signal and $\mathbf{f}^{(2)}$ from the content signal. We can incorporate either $\mathbf{f}^{(1)}$ or $\mathbf{f}^{(2)}$ or their combination. Thus, we construct three variants: $UBPR\_f^{(1)}$, $UBPR\_f^{(2)}$ and $UBPR\_f^{(1,2)}$. We have following observations:

- We can see that the ranking performance of $RAND$ is very low, which indicates our problem is really challenging. Even though the value of precision and recall for $UBPR$ is not very high, but compared with $RAND$ method, its performance is way better.
- The ranking performance of BPR is much better than UCF and NMF on both datasets (i.e. $BPR > UCF$ and $BPR > NMF$). The reason is BPR models

[3]https://itunes.apple.com/app/id440230030

[4]The user information is properly anonymized.



(a) Precision@5 (b) Recall@5 (c) Precision@10 (d) Recall@10

Figure 2. Precision@5, Recall@5, Precision@10 and Recall@10 on GetData.

Table II
THE ABSOLUTE AVERAGE IMPROVEMENTS OF OUR $UBPR$ METHODS COMPARED WITH BPR. THE RESULTS WITH BOLD FONTS AND STARS ARE STATISTICALLY SIGNIFICANT UNDER t-TEST.

Dataset	Proposed Methods	Precision@5	Recall@5	Precision@10	Recall@10	AUC
GetData	$UBPR\_f^{(1,2)}$	**13.01%***	**15.41%***	**8.26%***	**7.14%***	**7.56%***
	$UBPR\_f^{(1)}$	**6.41%***	**9.60%***	**5.70%***	**5.66%***	**5.71%***
	$UBPR\_f^{(2)}$	**6.59%***	**10.56%***	**8.10%***	**4.60%***	**5.83%***
ViewData	$UBPR\_f^{(1,2)}$	**9.05%***	**13.92%***	**5.84%***	**14.43%***	**3.47%***
	$UBPR\_f^{(1)}$	**5.72%***	**8.62%***	**4.00%***	**11.94%***	**2.38%***
	$UBPR\_f^{(2)}$	**6.03%***	**9.44%***	**3.53%***	**11.26%***	**2.47%***

the preference ranking signal from user actions, which indicates the effectiveness of preference ranking signal.

- The proposed UBPR model with either temporal or content signal is always better than BPR model (i.e. $UBPR\_f^{(1)} > BPR$ and $UBPR\_f^{(2)} > BPR$). This indicates that the temporal and content signals contain complimentary information to the preference ranking signal and thus help to improve the ranking performance. The absolute improvements are shown in Table II and is averaged over all test data. We can see that the temporal and content signals can significantly improve the performance almost in all cases.
- Moreover, the proposed UBPR model with both temporal and content signals achieves slightly better ranking performance than UBPR with either temporal or content signal (i.e. $UBPR\_f^{(1,2)} > UBPR\_f^{(1)}$ and $UBPR\_f^{(1,2)} > UBPR\_f^{(2)}$). From Table II, we can also see that the improvement value for $UBPR\_f^{(1,2)}$ is biger than either $UBPR\_f^{(1)}$ or $UBPR\_f^{(2)}$ in most of the cases with statistically significant level.

IV. CONCLUSION AND FUTURE WORK

In this paper, we proposed a user action driven framework $ActionRank$ for App personalized ranking. First, we utilize BPR to model user preferences ranking signal; Second, we capture temporal and content signals to represent users' action time habits and tastes of Apps. Third, the proposed framework incorporates these three types of signals into a coherent model. Experimental results on real-world datasets demonstrate the effectiveness of the proposed framework and the importance of user actions in App recommendations.

In the future, we will consider other approaches to model the temporal [8] and cold start scenarios [9] for recommending those long tail Apps. In addition, we will combine other explicit feedbacks such as user reviews and ratings to help improve App recommendations.

V. ACKOWLEDGMENTS

This material is based upon work supported by, or in part by, the ONR grant N00014-16-1-2257, N000141310835 and ARO (W911NF-15-1-0328).

REFERENCES

[1] B. Fu, J. Lin, L. Li, C. Faloutsos, J. Hong, and N. Sadeh, "Why people hate your app: Making sense of user feedback in a mobile app store," in *KDD'13*.

[2] P. Yin, P. Luo, W.-C. Lee, and M. Wang, "App recommendation: a contest between satisfaction and temptation," in *WSDM'13*.

[3] J. Lin, K. Sugiyama, M.-Y. Kan, and T.-S. Chua, "Addressing cold-start in app recommendation: latent user models constructed from twitter followers," in *SIGIR'13*.

[4] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *UAI'09*.

[5] M. Margulies, M. Egholm, W. E. Altman, S. Attiya, J. S. Bader, L. A. Bemben, J. Berka, M. S. Braverman, Y.-J. Chen, Z. Chen *et al.*, "Genome sequencing in open microfabricated high density picoliter reactors," *Nature'05*.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *JMLR'03*.

[7] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *NIPS'01*.

[8] H. Gao, J. Tang, X. Hu, and H. Liu, "Exploring temporal effects for location recommendation on location-based social networks," in *RecSys'13*.

[9] S. Wang, Y. Wang, J. Tang, K. Shu, S. Ranganath, and H. Liu, "What your images reveal: Exploiting visual contents for point-of-interest recommendation," in *WWW'17*.