In-app Purchase Prediction Using Bayesian Personalized Dwell Day Ranking

Shonosuke Harada
Kyoto University
Kyoto, Japan
sh1108@ml.ist.i.kyoto-u.ac.jp

Kazuki Taniguchi
CyberAgent, Inc.
Tokyo, Japan
taniguchi_kazuki@cyberagent.co.jp

Makoto Yamada
Kyoto University/RIKEN AIP
Kyoto, Japan
myamada@i.kyoto-u.ac.jp

Hisashi Kashima
Kyoto University/RIKEN AIP
Kyoto, Japan
kashima@i.kyoto-u.ac.jp

ABSTRACT

In this paper, we consider the in-app purchase prediction problem to effectively show display ads and to promote in-app purchases to users in a personalized manner. Compared to install prediction, purchase prediction is more difficult in the sense that the number of in-app purchases is much fewer than that of installations. To resolve this issue, following the idea of the Bayesian personalized ranking (BPR) framework, we consider the installation of an application as an intermediate feedback between purchase and unobserved feedback and propose install-enhanced BPR. More specifically, in our proposed method, we enhance the quality of the intermediate information by combining the dwell time on the application. Through experiments using a real-world dataset, we show that our proposed method outperforms the baselines and provides several insights on user activity.

KEYWORDS

recommender systems, game applications

ACM Reference Format:

Shonosuke Harada, Kazuki Taniguchi, Makoto Yamada, and Hisashi Kashima. 2019. In-app Purchase Prediction Using Bayesian Personalized Dwell Day Ranking. In *AdKDD '19, August 5, 2019, Anchorage, Alaska, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Mobile games attract a large number of users, as they are freely available, and there exist various types of games ranging from puzzle games to role-playing games. While most users play games without paying any money, some users purchase items in the game (e.g., swords and armors to power up soldiers in role-playing games) to enjoy the games more actively. Thus, game companies need to not only increase the number of users, but also let the users make purchases within the games. To this end, display advertisements play an important role.

Recently, with regard to the acquisition of new users, advertisers have insisted that more attention should be given to the lifetime value (LTV) of a user, which is the revenue that a user will generate in the future, than the cost per install (CPI). For advertisers, it is sometimes not sufficient to just improve the CPI because if a new user stops using an application right after installation, it is not worth the expense and only contributes to unsustainable business growth. Hence, from a marketing perspective, to achieve persistent revenue growth, LTV is becoming increasingly important for the pursuit of long-standing high-revenue users.

The long-standing high-revenue users are those who bring in revenue actively and sustainably, and finding such users is one of the most important tasks to improve LTV. In this regard, a crucial task is to figure out users who have a high potential to purchase as well as items that are more likely to be purchased, rather than finding users who have high a potential to install applications. This problem is regarded as the personalized in-app purchase prediction problem. In this problem, we face an issue regarding data sparsity that we do not have as many purchase feedbacks as we have install feedbacks. As most applications are available on the application store and the install of an application itself does not involve any cost, the number of application installs is relatively large compared with that of in-app purchases. Therefore, install feedback information can be used as intermediate feedbacks, which are ranked between in-app purchases and unobserved feedbacks. Such intermediate feedbacks are called implicit feedbacks, and some of the ranking-based recommender systems utilize them to improve their recommendations.

Bayesian Personalized Ranking (BPR) [12] is a ranking-based recommendation method, which ranks items for each user on the assumption that users prefer observed items to unobserved items. BPR has been successfully applied to various domains [3, 15] and also extended to situations such as those with multiple-type feedbacks [2, 7, 10].

In this study, based on BPR, we utilize the login information of users as the dwell time information for enhancing predictions. The use of dwell time has been shown to be effective in various recommendation domains [6, 13, 14], and we apply this idea to the mobile game domain. Specifically, we model the dwell time using Weibull distribution and incorporate it into our BPR formulation.

AdKDD '19, August 5, 2019, Anchorage, Alaska, USA

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

Through exploratory analysis and performance comparison using a real-world dataset containing in-app purchases and application install records collected by CyberAgent Inc. over a period of two years, we show that the proposed method outperforms the baselines.

In summary, our contributions in this paper are two-fold:

(1) A game-app recommendation method using user-item oriented weights based on dwell time analysis.
(2) Analysis of the dwell time in a real dataset in the game application domain on a relatively long scale, and comparison with baselines using a real-world dataset to demonstrate the effectiveness of our proposed method.

2 RELATED WORK

Recommendation problems can be categorized into two types: (1) explicit feedback settings where we can directly observe each user's preference through the rating scores given by the user. (2) implicit feedback settings where we can only observe each user's past interactions that are weakly related to the user's preference. In this paper, we consider the latter, and we review several implicit feedback models based on BPR and the variances of BPR for multi-feedback. We also review recommendation methods based on dwell time.

2.1 Implicit Feedback

While the implicit feedback setting is more practical than the explicit setting, it is more challenging in terms of handling unobserved instances, which are usually ignored in the explicit setting. One of the most popular and simplest solution to handle them is to treat all the unobserved instances as negative feedbacks [5] or to use a subset of them as negative feedbacks [12]. The key idea behind this is that users might prefer observed items than unobserved items. BPR [12] is one of the prevalent learning-to-rank methods applicable to the implicit feedback setting. It generates personalized rankings based on the assumption that users prefer observed items to unobserved items. BPR has been extended to several specific domains [3, 9]. They combine user-item pairs with domain features and show their effectiveness. Loni et al. [7] improved BPR by considering different levels of implicit feedbacks. In order to predict users' regarding items that are preferred more feedback (i.e., high score rating, purchase), additionally sampled intermediate feedbacks are used for more sophisticated rankings [2, 10].

2.2 Dwell time analysis

Dwell time, which is the duration of time one spends for an item, has been successfully applied in recommendation tasks [13, 14]. Liu et al. [6] modeled users' dwell time on web pages using Weibull distribution for the first time. Yin et al. [14] modeled users' view-vote behaviours in a joke app domain using a graphical model and converted the dwell time into a pseudo vote for mitigating data sparsity. Although their work is closely related to ours, we focus on a completely different domain, namely game applications, and demonstrate the effectiveness of dwell time in a ranking method (i.e., BPR) in this domain; we do not consider different methods of processing the dwell time. We emphasize that the employment of the dwell time significantly improved the performance of the traditional method such as matrix factorization in several domains. We are motivated to show the effectiveness of the dwell time in the game application domain and as the user-item oriented weight in the ranking method for purchase prediction.

Table 1: An illustrative example of our game application dataset.

user ID	app ID	first login date	last login date	in-app purchase
user0	app0	20XX/1/April	20XX/1/May	0
user0	app2	20XX/1/April	20XX/1/May	0
user0	app3	20XX/1/April	20XX/1/May	1
user1	app1	20XX/1/April	20XX/1/May	1
user1	app2	20XX/1/April	20XX/1/May	0
...	...	...	...	...

3 PROBLEM STATEMENT

Suppose we have a set of users $\mathcal{U}$ and another set of applications $\mathcal{A}$. Let $\mathcal{P} = \{(u, a)\} \subset \mathcal{U} \times \mathcal{A}$ be the set of in-app purchases consisting of pairs of user u and application a. (u, a) indicates that user u made at least one in-app purchase in application a. We also let $\mathcal{T} \subset \mathcal{U} \times \mathcal{A}$ be a set of user-app pairs that the users have not even installed the applications. Our goal is to find a subset of our target pairs $\mathcal{T}$ consisting of user-app pairs having high probabilities of in-app purchases if the users install the apps in the future.

In addition to $\mathcal{P}$ and $\mathcal{T}$, we assume we also have records of users who installed the apps but not made any in-app purchase for each application. We denote by $\mathcal{I} \subset \mathcal{U} \times \mathcal{A}$ the set of such user-app pairs. Note that $\mathcal{P} \cap \mathcal{I} = \phi$, $\mathcal{T} \cap \mathcal{I} = \phi$, $\mathcal{P} \cap \mathcal{I} = \phi$, and $\mathcal{P} \cup \mathcal{T} \cup \mathcal{I} = \mathcal{U} \times \mathcal{A}$ hold. We regard an install of an application by a user as an intermediate feedback that should be ranked between 'purchase' and 'no-install', and we want to exploit the implicit feedback information effectively in order to improve in-app purchases in game applications.

In this paper, we assume each instance of the data dataset consists of a user ID, an app ID, the first login date, the last login date, and whether or not the user made an in-app purchase. We show an illustrative example of the dataset in Table 1. They correspond to $\mathcal{P}$ and $\mathcal{I}$ in our notation.

In addition to the user-app pairs, we also have the first login dates and the last login dates. In this paper, we exploit them as the side information by converting them into the dwell time. The intuition behind the use of the dwell time information is that time they have spent on the apps should reflect their preference and can be useful for more precise ranking. The previous work [13, 14] also used similar features in terms of the dwell time; however, the time scale of their dwell time were relatively short, i.e. several seconds to a few minutes, while our time scale is in days and months. Also, we first apply the idea to the game-app domain to the best of our knowledge.

4 PROPOSED METHOD (DBPR)

We first review two existing approaches: the vanilla BPR [12] and the install-enhanced BPR [2]. We then propose our Bayesian Personalized Dwell Day Ranking model, which takes the dwell day information into account.

4.1 BPR

BPR [12] is a widely used learning-to-rank method for implicit feedback. The key idea is to rank the observed items higher than the unobserved items for each user. The loss function of BPR is given as

$$L = \sum_{(u,i,j)\in\mathcal{D}} \log p(\Theta \mid i >_u j) = \sum_{(u,i,j)} -\log p(i >_u j \mid \Theta)p(\Theta)$$
$$= \sum_{(u,i,j)} -\log \sigma(\hat{r}_{ui}(\theta) - \hat{r}_{uj}(\theta)) + \lambda_\Theta \Theta, \tag{1}$$

where $\mathcal{D}$, $\hat{r}_{ui}$, and Θ denote $\{(u,i,j) \mid i \in \mathcal{P}_u \wedge j \in \mathcal{I}_u \cup \mathcal{T}_u\}$, the inner product of the user latent factor p_u and item latent factor q_i, and the set of all the parameters, respectively. The traditional matrix factorization-based methods basically treat observed targets as 1 and unobserved targets as 0, when applying the least squares method; however, the problem to be solved for the implicit setting is not a regression problem but should be a classification problem, or more precisely, a ranking problem. A previous work has shown that such a formulation is more effective [12].

4.2 Install-enhanced BPR

An extension of BPR (viewBPR) was proposed to enhance BPR in an e-commerce domain by combining view and purchase feedbacks [2]. In the viewBPR model, they sampled not only the purchase feedback but also the view feedback, and used them to generate a more sophisticated ranking. The loss function of viewBPR is given as

$$L = \sum_{(u,i,j,k)\in\hat{\mathcal{D}}} -\log \sigma(\hat{r}_{ui}(\theta) - \hat{r}_{uj}(\theta))$$
$$- (1-\alpha_u)\log \sigma(\hat{r}_{ui}(\theta) - \hat{r}_{uk}(\theta))$$
$$- \alpha_u \log \sigma(\hat{r}_{uk}(\theta) - \hat{r}_{uj}(\theta))) + \lambda_\Theta \Theta, \tag{2}$$

where $\hat{\mathcal{D}}$ and α_u denote $\{(u,i,j,k) \mid i \in \mathcal{P}_u \wedge k \in \mathcal{I}_u \wedge j \in \mathcal{T}_u\}$ and a user-dependent weight, respectively. The viewBPR model assumes that a viewed item k should not be ranked higher than a purchased item i but should be ranked higher than an unobserved item j, and α_u controls the relative importance of the view information. In the e-commerce domain, they calculated the ratio of the number of views to the number of purchases. In this paper, we simply introduce the in-app purchase ratio $PR_u = \frac{|\mathcal{P}_u|}{|\mathcal{I}_u \cup \mathcal{P}_u|}$ for user u and use it as α_u.

We call this method the install-enhanced BPR. The key ideas are (1) the simultaneous sampling of purchased items, viewed items, and unobserved items to obtain a more effective loss function, and (2) controlling the relative importance of the intermediate feedbacks with the user-oriented weights.

4.3 Dwell Day BPR

Now, we introduce our model, the dwell day BPR (DBPR). In the game application domain, we can use dwell days, i.e., how long the users have enjoyed the games. Although the dwell time is considered a kind of implicit feedback and also seems promising and helpful in this domain, making use of it as additional information is not trivial as suggested in the literature [6, 13, 14]. While the previous work handled the seconds-to-minutes scale dwell time (i.e., how long they read news, or how long they read jokes), we use the days-to-months scale dwell time. However, how to handle such a dwell time is not obvious, and is highly domain dependent. We design an approach that combines the dwell day and BPR frameworks. To use the first login dates and the last login dates for the recommendations effectively, we simply process them as follows for the user-item pair $(u,i) \in \mathcal{P} \cup \mathcal{I}$:

$$\tilde{t}_{ui} = \text{Last login timestamp} - \text{First login timestamp}. \tag{3}$$

ALGORITHM 1: Normalized dwell time for game applications

> **for** *item i in* $\mathcal{A}$ **do**
> $\quad \mu_i \leftarrow \frac{1}{|(u,i)\in\mathcal{P}\cup\mathcal{T}|}\Sigma_{(u,i)\in\mathcal{P}\cup\mathcal{T}} \log(\tilde{t}_{ui})$
> $\quad \sigma_i^2 \leftarrow \frac{1}{|(u,i)\in\mathcal{P}\cup\mathcal{T}|}\Sigma_{(u,i)\in\mathcal{P}\cup\mathcal{T}} (\log(\tilde{t}_{ui}) - \mu_i)^2$
> $\quad$ **for** *user u in* $\mathcal{U}$ **do**
> $\quad\quad t_{ui} \leftarrow \exp\left(\frac{\log(\tilde{t}_{ui}) - \mu_i}{\sigma_i}\right)$
> $\quad$ **end**
> **end**

Since different game apps have different release dates, it is not easy to preprocess of dwell days for every item in order alleviate the effect of items that which have distinct release dates. We transform each $\tilde{t}_{ui}$ into *the normalized dwell time* [13] as shown in Algorithm 1. Basically, we follow the same procedure proposed as that in the previous work [13]. First, we calculate the average and standard deviation in the log scale for every item, and then all the dwell days are normalized.

Figures 1 (a), (b) , and (c) show the dwell day distribution, the normalized day distribution in our game application dataset, and the correlation distribution between the actions (i.e., purchase and install) and the normalized time, respectively.

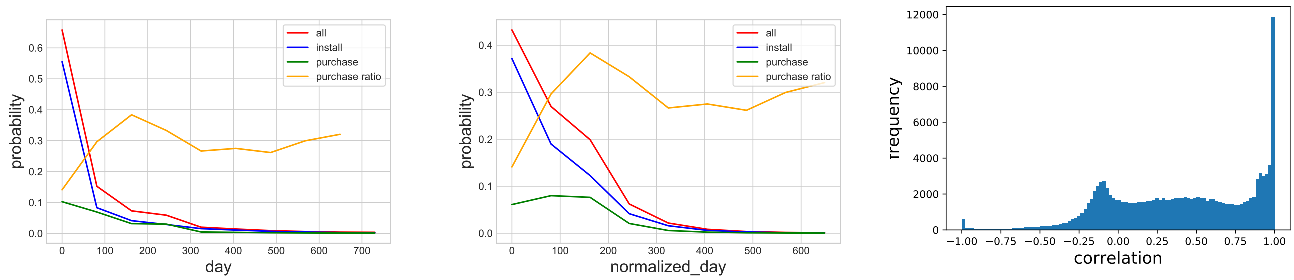
Figures 1 (a) and (b) show that as the day or normalized day increases, the ratio of the in-app purchase increases and this indicates that users tend to play items in which they have made in-app purchases longer than just installed apps. In addition, we examine the correlation between the action and the dwell time for every user. Figure 1 (c) shows the point biserial correlation, which equals to 1 when in-app purchased apps are enjoyed longer than just installed ones, and is equal to −1 when in-app purchased apps are enjoyed shorter than just installed ones. The distribution tends to the right; hence it can be seen that the dwell time and the actions of most users are correlated with their dwelling time. Here, we introduce the assumption that users are more likely to purchase something in longer dwelled applications than in shorter dwelled ones.

Based on these assumptions, we introduce user-item oriented weights using Weibull distribution for improving the in-app purchase predictions in game applications. Weibull distributions are often used for survival analysis [1], analysis of the properties of brittle materials [8], and activity analysis of website visitors [6]. Weibull distribution and its cumulative distribution function are given as

$$p(x,m,\lambda) = \frac{m}{\lambda}\left(\frac{x}{\lambda}\right)^{m-1} \exp\left((-x/\lambda)^m\right), \tag{4}$$

$$F(x) = \int_0^x p(x,m,\lambda)\,dx = 1 - \exp\left((-x/\lambda)^m\right), \tag{5}$$

where $x > 0, m > 0$ and $\lambda > 0$ indicate the random variable, the shape parameter, and the scale parameter, respectively. Each



(a) The raw dwell-day distribution (b) The normalized dwell-day distribution (c) Distribution of correlations between in-app purchases and normalized days

Figure 1: (a) The distribution of the raw dwell days. (b) Since different applications have different release dates, we normalized each item using Algorithm 1. Note that the sum of the blue line and the green line equals to the red line. The proportion of in-apps purchase increases as the normalized dwell days increase as the orange line indicates. This suggests users tend to enjoy preferable applications longer. (c) The distribution of the correlations between the in-app purchases and normalized days.

parameter is estimated using the least square method [11]; we calculate the mean rank $\hat{F}(x_i) = \frac{i}{n+1}$ for the i-th smallest value and solve the linear regression problem

$$\hat{m} = \frac{n\sum_{i=1}^{n} A(x_i) - \sum_{i=1}^{n}\log x_i \sum_{i=1}^{n} A(x_i)}{n\sum_{i=1}^{n}(\log x_i)^2 - \sum_{i=1}^{n}(\log x_i)^2}, \tag{6}$$

$$\hat{\lambda} = \exp\left(\frac{-\sum_{i=1}^{n} A(x_i) + \hat{m}\sum_{i=1}^{n}\log x_i}{n\hat{m}}\right), \tag{7}$$

where n is the sample size, i.e, $|\ (u,i) \in \mathcal{P} \cup \mathcal{T}\ |$ and $A(x_i) = \log\left[-\log(1 - \hat{F}(x_i))\right]$. Note that the other estimation approaches (e.g., maximum likelihood estimation) can be more efficient, but exploring them is beyond the scope of this paper. We have made an assumption that long-dwelling users like the dwelled items much more than short-stayed items. Hence, we aim to measure how strong they like the items by using the cumulative Weibull distribution. While in the install-enhanced BPR, a user-oriented constant value PR_u is given for every user, we provide a user-item constant value α_{uk}, to obtain a more precise purchase potential. Then, the user-item oriented weight α_{uk}, which indicates how much a user u is not likely to make an in-app purchase in the app k, is calculated by feeding the dwell day t_{uk} into $F_u(x)$, which indicates the probability that user $u$will quit or make an in-apps purchase by day x since u has installed an app k as $\alpha_{uk} = 1 - F_u(t_{uk})$. Using α_{uk}, the objective function to minimize is defined as follows:

$$\begin{aligned} L = \sum_{(u,i,j,k)\in\hat{\mathcal{D}}} & -\log(\hat{r}_{ui}(\theta) - \hat{r}_{uj}(\theta)) \\ & -\alpha_{uk}\log(\hat{r}_{ui}(\theta) - \hat{r}_{uk}(\theta)) \\ & -(1-\alpha_{uk})\log(\hat{r}_{uk}(\theta) - \hat{r}_{uj}(\theta)) + \lambda_\Theta\Theta. \end{aligned} \tag{8}$$

We employ the matrix factorization model and Stochastic Gradient Descent (SGD) for the optimization, similar to that in the previous work [2, 10]. We update each parameter with the following formulas:

ALGORITHM 2: DBPR algorithm

Initialize: Θ
\# Estimate the dwell-time distribution for each user
for *user* u *in* $\mathcal{U}$ **do**
  $x_i = \mathcal{P}_u \cup \mathcal{I}_u$
  Estimate the Weibull distribution for user u using Eqs. (6) and (7)
end
\# Train a BPR model
while *not converged* **do**
  $u \leftarrow$ draw a sample user from $\mathcal{U}$
  $i \leftarrow$ draw a sample item from $\mathcal{P}_u$
  $j \leftarrow$ draw a sample item from $\mathcal{T}_u$
  $k \leftarrow$ draw a sample item from $\mathcal{I}_u$ (**intermediate sample**)
  \# Compute dwell-day weights
  $\alpha_{uk} = 1 - F_u(t_{uk})$
  Compute the loss function using Eq. (8)
  Compute the gradients using Eq. (9)
  Update parameters (SGD)
end

$$\frac{\partial}{\partial\theta}L = \begin{cases} \tau(\hat{r}_{uij})(q_i - q_j) & \\ \ +\alpha_{uk}\gamma(\hat{r}_{uik})(q_i - q_k) & \\ \ +(1-\alpha_{uk})\gamma(\hat{r}_{ukj})(q_k - q_j) + \lambda p_i & (\theta = p_i) \\ \gamma(\hat{r}_{uij})p_u + \alpha_{uk}\gamma(\hat{r}_{uik})p_u + \lambda q_i & (\theta = q_i) \\ -\gamma(\hat{r}_{uij})p_u - \alpha_{uk}\gamma(\hat{r}_{ukj})p_u + \lambda q_j & (\theta = q_j) \\ -\alpha_{uk}\gamma(\hat{r}_{uik})p_u + \alpha_{uk}\gamma(\hat{r}_{ukj})p_u + \lambda q_k & (\theta = q_k) \end{cases}, \tag{9}$$

where $\gamma(x) = 1 - \sigma(x)$ and $\hat{r}_{uij} = \hat{r}_{ui}(\theta) - \hat{r}_{uj}(\theta)$. We summarize the algorithm of the proposed method in Algorithm 2.

5 EXPERIMENTS

5.1 Dataset

We use an anonymized dataset containing in-app purchases and application install records collected by CyberAgent over a period of 2 years. We randomly select a subset of users from among users who have installed at least 20 applications and purchased at least one item in some of them. For detailed analyses of the experimental results, we divide the test users into three groups based on their correlation coefficients. Users whose correlation coefficients are above 0.2 are regarded as "positively correlated" users, those between 0.2 and −0.2 are regarded "uncorrelated" users, and those below −0.2 are regarded as 'negatively correlated' users.

The numbers of users, items, install, in-app purchases are 140856, 706, 2119990, and 726419, respectively.

5.2 Baselines

We choose several BPR-based methods, including the state-of-the-art methods capable of handling multi-feedbacks as the baselines. As a trivial baseline, we also use the item popularity method, which predicts items based only on their popularity.

BPR (In-app purchases) [12] is the vanilla BPR, which samples in-app purchase transactions. BPR, which samples both purchased items and installed items without distinguishing them, showed poor performance in our preliminary experiment, unlike the result in the previous work [10]; we omit the results of this version of BPR.

AdaptiveBPR (ABPR) [10] is the state-of-the-art Bayesian personalized method for multi feedbacks. This method learns the confidence scores and regularizers for the immediate feedback in cases where we cannot explicitly prepare the confidence scores. It also samples positive feedbacks and immediate feedbacks. The difference from the install-enhanced BPR is in the method of constructing the loss function. While the install-enhanced BPR combines the loss functions using simultaneous pairwise ranking and updates the parameters simultaneously, ABPR does it separately. We tuned the hyperparameters, the threshold τ, and the iteration number T using the validation dataset, where we used $\tau = 150$ and T=10.

Install-enhanced BPR [2] is the state-of-the-art Bayesian personalized method for multiple feedbacks, which utilizes the install-enhanced scheme using Eq. (2). The key idea is to enhance the preference ranking by using the intermediate feedback information (e.g., view, install, etc.) for predicting positive feedbacks.

5.3 Experimental setup

For all the methods, the performance was evaluated when the the number of factors $k = 32, \lambda_\Theta = 0.01$, and learning rate 0.01. Although some works [2, 4] adopted the leave-one-out scheme, this is not a realistic scenario. Hence, we employ $Recall@k$, $Precision@k$, and $nDCG@k$ ($k \in \{5, 10\}$) as the evaluation metrics.

We use the data for which the last login dates are between 2016 and 2018 as the training data and aim to predict in-app purchases after this period. In order to ensure that the user engagements of the users applications can be observed appropriately, we kept the data in which both the first login and the last login occurs between 2016 and Feb. 14, 2018. We set the period in which a user quits using use an application for some reason as 2 weeks. Hence, if a user has never logged into an application for 2 weeks, we assume that the user must have stopped using the application for some reason and we use them as training data. We focus on effect of the dwell time, because an appropriate estimation of how long they an enjoy app is necessary. We conduct experiments with three different initial seeds and report the averaged results. For early stopping, we evaluated the model using nDCG@20 and stopped the training when the model showed the best performance for the validation dataset.

5.4 Experimental results

Table 2 shows a summary of the experimental results. Table 2 (a) shows the result all for the whole users, while Tables 2 (b), (c) and (d) show the results those for the positively correlated users, the uncorrelated users, and the negatively correlated users, respectively. It can be seen that our method outperforms the baselines in all of the performance metrics when we test them with the data of all users, which clearly demonstrates the effectiveness of our learning-to-rank method using the dwell time information. The vanilla BPR improves upn ABPR in the same way as BPR (in-app purchases+install). It is worth pointing out that the install-enhanced BPR shows better performance, which demonstrates the effectiveness of simultaneous pairwise ranking, which is consistent with the findings of previous work [2].

The results for particular classes of users show that the performance improvement by the proposed method is less significant for the negatively-correlated users who tend to quit using applications in which they have made an in-app purchase, compared with the other two user groups; however, the result is still comparable and shows that our proposed approach is promising.

6 CONCLUSION

We proposed the Bayesian personalized dwell-day ranking for mobile game applications, a novel method that considers how long users have dwelled with each item. Our proposed method is based on the assumption that the longer they enjoy certain applications, the more they like them and the greater the possibility of making in-app purchases. Our proposed method shows significant improvements over the baseline methods, especially for users whose purchases have strong positive correlations with their dwell times. Our future work will address the relatively difficult cases for the current model such as those with the negatively-correlated users.

ACKNOWLEDGMENTS

This work was supported by JSPS KAKENHI 15H01704.

REFERENCES

[1] Claudio Cardelli and Theodore P. Labuza. 2001. Application of Weibull hazard analysis to the determination of the shelf life of roasted and ground coffee. *LWT-Food Science and Technology* 34, 5 (2001), 273–278.
[2] Jingtao Ding, Guanghui Yu, Xiangnan He, Yong Li, and Depeng Jin. 2018. Sampler Design for Bayesian Personalized Ranking by Leveraging View Data. *arXiv preprint arXiv:1809.08162* (2018).
[3] Ruining He and Julian McAuley. 2016. VBPR: visual bayesian personalized ranking from implicit feedback. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI)*.
[4] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web (WWW)*. 173–182.
[5] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM)*. 263–272.
[6] Chao Liu, Ryen W. White, and Susan Dumais. 2010. Understanding web browsing behaviors through Weibull analysis of dwell time. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 379–386.
[7] Babak Loni, Roberto Pagano, Martha Larson, and Alan Hanjalic. 2016. Bayesian personalized ranking with multi-channel user feedback. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 361–364.

Table 2: Performance comparison of the proposed and baseline methods. Item popularity is a trivial baseline that predicts items based only on their popularity. Our method achieves the best performance with (a) the whole users as well as with (b) the positively-correlated users, (c) the uncorrelated users, and (d) the negatively-correlated users. Improvement is most significant for the positively correlated users.

(a) Results for all users

	recall@5	recall@10	Precision@5	Precision@10	nDCG@5	nDCG@10
Item Popularity	0.0238 ± 0.0005	0.0471 ± 0.0008	0.0133 ± 0.0002	0.0145 ± 0.0002	0.0159 ± 0.0003	0.0265 ± 0.0004
BPR [12]	0.0676 ± 0.0010	0.1182 ± 0.0036	0.0274 ± 0.0005	0.0263 ± 0.0007	0.0463 ± 0.0004	0.0667 ± 0.0020
Install-enhanced BPR [2]	0.0702 ± 0.0005	0.1213 ± 0.0006	0.0275 ± 0.0003	0.0258 ± 0.0001	0.0473 ± 0.0013	0.0674 ± 0.0014
ABPR [10]	0.0661 ± 0.0036	0.1138 ± 0.0037	0.0252 ± 0.0016	0.0238 ± 0.0042	0.0463 ± 0.0021	0.0652 ± 0.0014
DBPR (ours)	**0.0792 ± 0.0021**	**0.1293 ± 0.0015**	**0.0315 ± 0.0008**	**0.0276 ± 0.0003**	**0.0564 ± 0.0024**	**0.0761 ± 0.0015**

(b) Results for positively correlated users

	recall@5	recall@10	Precision@5	Precision@10	nDCG@5	nDCG@10
Item Popularity	0.0239 ± 0.0004	0.0483 ± 0.0006	0.0127 ± 0.0001	0.0138 ± 0.0001	0.0159 ± 0.0001	0.0268 ± 0.0002
BPR [12]	0.0701 ± 0.0015	0.1217 ± 0.0038	0.0286 ± 0.0006	0.0268 ± 0.0008	0.0485 ± 0.0013	0.0692 ± 0.0022
Install-enhanced BPR [2]	0.0748 ± 0.0002	0.1240 ± 0.0008	0.0296 ± 0.0002	0.0264 ± 0.0002	0.0506 ± 0.0015	0.0700 ± 0.0019
ABPR [10]	0.0679 ± 0.0034	0.1172 ± 0.0013	0.0268 ± 0.0018	0.0249 ± 0.0004	0.0479 ± 0.0016	0.0674 ± 0.0005
DBPR (ours)	**0.0842 ± 0.0023**	**0.1332 ± 0.0021**	**0.0339 ± 0.0009**	**0.0284 ± 0.0005**	**0.0603 ± 0.0025**	**0.0795 ± 0.0021**

(c) Results for uncorrelated users

	recall@5	recall@10	Precision@5	Precision@10	nDCG@5	nDCG@10
Item Popularity	0.0197 ± 0.0009	0.0381 ± 0.0009	0.0118 ± 0.0004	0.0130 ± 0.0003	0.0132 ± 0.0006	0.0219 ± 0.0005
BPR [12]	0.0589 ± 0.0011	0.1052 ± 0.0036	0.0234 ± 0.0006	0.0233 ± 0.0006	0.0402 ± 0.0009	0.0588 ± 0.0020
Install-enhanced BPR [2]	0.0581 ± 0.0003	0.1073 ± 0.0008	0.0224 ± 0.0001	0.0223 ± 0.0004	0.0391 ± 0.0001	0.0582 ± 0.0039
ABPR [10]	0.0592 ± 0.0021	0.1018 ± 0.0006	0.0215 ± 0.0009	0.0203 ± 0.0002	0.0410 ± 0.0018	0.0577 ± 0.0014
DBPR (ours)	**0.0663 ± 0.0022**	**0.1150 ± 0.0019**	**0.0260 ± 0.0006**	**0.0243 ± 0.0004**	**0.0470 ± 0.0021**	**0.0660 ± 0.0012**

(d) Results for negatively correlated users

	recall@5	recall@10	Precision@5	Precision@10	nDCG@5	nDCG@10
Item Popularity	0.0384 ± 0.0005	0.0749 ± 0.0014	0.0227 ± 0.0002	0.0246 ± 0.0010	0.0255 ± 0.0002	0.0424 ± 0.0007
BPR [12]	0.0864 ± 0.0068	0.1486 ± 0.0042	**0.0355 ± 0.0021**	**0.0349 ± 0.0013**	**0.0575 ± 0.0024**	0.0834 ± 0.0019
Install-enhanced BPR [2]	**0.0913 ± 0.0060**	**0.1597 ± 0.0050**	**0.0356 ± 0.0023**	**0.0355 ± 0.0008**	**0.0604 ± 0.0054**	0.0881 ± 0.0049
ABPR [10]	0.0822 ± 0.0106	0.1410 ± 0.0089	0.0307 ± 0.0031	0.0308 ± 0.0016	0.0573 ± 0.0075	0.0816 ± 0.0069
DBPR (ours)	**0.0997 ± 0.0047**	**0.1623 ± 0.0041**	**0.0394 ± 0.0020**	**0.0360 ± 0.0015**	**0.0707 ± 0.0050**	**0.0961 ± 0.0032**

[8] Chunsheng Lu, Robert Danzer, and Franz Dieter Fischer. 2002. Fracture statistics of brittle materials: Weibull or normal distribution. *Physical Review E* 65, 6 (2002), 067102.

[9] Jarana Manotumruksa, Craig Macdonald, and Iadh Ounis. 2017. A personalised ranking framework with multiple sampling criteria for venue recommendation. In *Proceedings of the 26th International Conference on Information and Knowledge Management (CIKM)*. 1469–1478.

[10] Weike Pan, Hao Zhong, Congfu Xu, and Zhong Ming. 2015. Adaptive Bayesian personalized ranking for heterogeneous implicit feedbacks. *Knowledge-Based Systems* 73 (2015), 173–180.

[11] Ivana Pobočíková and Zuzana Sedliačková. 2014. Comparison of four methods for estimating the Weibull distribution parameters. *Applied Mathematical Sciences* 8, 83 (2014), 4137–4149.

[12] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uuncertainty in Artificial Intelligence (UAI)*. 452–461.

[13] Xing Yi, Liangjie Hong, Erheng Zhong, Nanthan Nan Liu, and Suju Rajan. 2014. Beyond clicks: dwell time for personalization. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys)*. 113–120.

[14] Peifeng Yin, Ping Luo, Wang-Chien Lee, and Min Wang. 2013. Silence is also evidence: interpreting dwell time for recommendation from psychological perspective. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 989–997.

[15] Fajie Yuan, Joemon M Jose, Guibing Guo, Long Chen, Haitao Yu, and Rami S Alkhawaldeh. 2016. Joint geo-spatial preference and pairwise ranking for point-of-interest recommendation. In *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*. 46–53.