B.Comp. Dissertation

Serendipitous Mobile App Recommendation

By

Su Sumei

Department of Computer Science

School of Computing

National University of Singapore

2017/18

B.Comp. Dissertation

Serendipitous Mobile App Recommendation

By

Su Sumei

Department of Computer Science

School of Computing

National University of Singapore

2017/18

Project No: H0791510
Advisor: Dr Sugiyama Kazunari
Deliverables:
    Report: 1 Volume

Abstract

Recommender systems are getting more and more popular in different areas due to the information overload in the Internet. Good recommender systems help user to filter out relevant items based on their interests and preferences. In app domain, however, current recommender systems mostly focus on recommending apps that the user has downloaded, rated or reviewed, which indicates user's preferences. The apps that the user has never experienced will not feature in the recommendation list. This affects the overall user satisfactions about the recommender systems and it is hard for user to discover new apps that he or she has little knowledge on. Therefore, introducing serendipity into mobile app recommender systems is important. In this report, we will study a serendipitous mobile app recommendation approach which is adapted from a serendipitous recommender system for scholarly papers as baseline for comparison and another serendipitous approach using *k*-means clustering algorithm.

Subject Descriptors:

H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval - Information filtering

Keywords:

Recommender systems, Mobile apps, Serendipity

Acknowledgement

I would like to express my deepest gratitude to my FYP supervisors, Dr Kazunari Sugiyama and Prof Min-Yen Kan. Without them, I could not make this progress so far. Dr Sugiyama has led me into recommender system research area, especially mobile app recommendation. He gave me many valuable suggestions on my work, my progress and my report as well. More importantly, he kept me on the right track and gave me support when I was lost. Prof Kan gave me many valuable suggestions on research methods and report writing.

Thanks to their support, I have much confidence on continuing to work on this project.

Table of Contents

Chapter 1

Introduction

With the popularity of smart mobile devices (*i.e.*, smartphones and tablets) and the wide user penetration of mobile Internet, the number of mobile applications (hereafter, "app") has grown explosively in the app market. Google Play[1] and Apple App Store[2] are two main platforms for app distribution. According to statista [29, 30], as of September 2017, 3.3 million apps are available to Android users in Google Play. On the other hand, as of January 2017, iOS users can choose among 2.2 million apps from Apple App Store. Furthermore, new apps are released rapidly with a rate of more than 5,600 apps daily in Google Play Store and more than 1,600 in Apple App Store [1]. The sheer number of applications makes it difficult for users to find and decide what kind of apps are relevant to their information needs. Therefore, as one of solutions, recommender systems have been developed to alleviate the problem of information overload and help users to find relevant apps.

Recommender systems have been employed and utilized in a variety of areas including music, movies, books, news, search queries, and scholarly papers. They help to filter out irrelevant information and identify items of interests to target users. Typically, recommender systems suggest a list of recommendation and are classified into three techniques: collaborative filtering (CF) [23, 13, 40], content-based filtering (CBF) [14, 31], and hybrid of both CF and CBF [34, 11]. CF has been widely used and predicts what kind of items target users will like based on

[1]https://play.google.com/store

[2]https://www.apple.com/lae/ios/app-store/

their similarity to other users by collecting a large amount of users' preferences or behaviors. Another technique, CBF produces recommendations by comparing the representation of items to the profile of the target user constructed upon the user's interests. However, both CF and CBF mostly focus on generating relevant recommendations that each user's interests.

Recently, some works suggest that, beyond predictive accuracy, there are some other critical metrics that contribute to the quality of recommender systems such as privacy [3], diversity [42], novelty [26], and serendipity [12]. Serendipity increases the quality of recommender systems under the assumption that user would like to explore something unexpected and surprising when using recommender systems.

Recently, more and more researchers have focused on developing serendipitous recommendations. Current recommender systems mostly aim at providing relevant recommendations and are measured by users' ratings or downloaded items, while serendipitous recommendation is antithetical to the traditional recommendation since it focuses on producing a list of items that users never expect and will not discover themselves. In addition, serendipitous recommendation prefers long-tail items compared to the traditional recommendation. Furthermore, introducing serendipity into recommender systems helps to avoid obvious recommendations in CF [15] and solve over-personalization problems in CBF [17]. Suppose a user who employs a music recommendation system to discover some musics relevant to his interests. After he listens to a few musics from a singer, if the system is using CBF, the songs from the same singer or the songs with similar style would be recommended. If the system employs CF, some songs listened by those who have tastes similar to the target user would be listed for recommendation. Both situations will mostly provide items that the user has already known or consumed. This is not ideal for user's satisfaction and experience with the system.

In order to make serendipitous recommendation, we first give a definition to "serendipitous recommendation" as Bhandari *et al.* [7] defined: producing something diverse and novel. Then we propose two methods that are derived from traditional systems to increase the novelty and diversity of the recommendation list. First, we integrate Twitter information to construct dissimilar user profiles in mobile app domain to provide serendipitous recommendation. Second, we leverage k-means clustering algorithm to make user clusters and mobile app clusters for further process of recommendation.

This paper is organized as follows: in Chapter 2, we review related works on mobile app recommendation and serendipitous recommendation. In Chapter 3, we formally propose dissimilar users approach and k-means clustering approach in detail. In Chapter 4, we present our dataset, experimental results, and evaluation for our proposed methods. In Chapter 5, we conclude this paper with a summary and future work.

Chapter 2

Related Work

2.1 Mobile App Recommendation

The tremendous increase of mobile apps make it a big challenge for the users to select apps relevant to their interests. Therefore, many recommender systems have been proposed and developed in mobile app domain. Given the app domain characteristics, some works collect and utilize additional information from mobile device to recommend more relevant apps to the users. Xu *et al.* [35] investigated the diverse usage behaviors of individual apps by using anonymized network measurements from a tier-1 cellular carrier in the U.S. Yan and Chen [36], Costa-Montenegro *et al.* [10], and Baeza-Yates *et al.* [6] gathered internal information and analyzed the usage patterns of each user by monitoring user's interactions and behaviors to improve recommendation accuracy, while Zheng *et al.* [40] and Davidsson and Moritz [11] utilized external information like user's locations or GPS to construct context-aware app recommendation. Kim *et al.* [19] instead analyzed semantic relations among apps consumed by users and made recommendation based on the similarity of the semantic relations. Yin *et al.* [37] considered factors that invoke a user to replace an old app (already in use) with a new app and concluded that app adoption was the result of the contest between the actual satisfactory value each owned app had and the tempting value a new app under consideration had. Lin *et al.* [21] exploited app-related information on Twitter to resolve cold-start issue in app recommendation, and their subsequent work proposed an app recommendation system by leveraging the version features of apps [22]. Recently, Cao *et al.* [9] also proposed a version-sensitive mobile app recommendation framework to improve the recommendation performance and solve in-matrix and out-of-matrix cold-start problems by considering the progression of versions. Wang *et al.* [33] employed social influence for app recommendation. Zhu *et al.* [41] and Liu *et al.* [24] integrated user's privacy preference into app recommendation as apps could have permissions to access user's sensitive information like contacts and locations. While the above works focused on the accuracy of the app recommendation, Bhandari *et al.* [7] worked on factor of serendipity and proposed a graph-based app recommendation method. Cao *et al.* [8] integrated both numerical ratings and textual content from multiple platforms to alleviate data sparsity and cold-start issues and improve accuracy and effectiveness of the recommendation.

2.2 Serendipitous Recommendation

While most of the aforementioned works aim at recommending most relevant items that are similar to those consumed and highly rated by the target user, much less recommendation approaches focus on serendipity. Ziegler *et al.* [42] introduced an intra-list similarity metric using a taxonomy-based classification to assess the overall topical diversity of recommendation lists and proposed Topic Diversification Algorithm to decrease the intra-list similarity. Zhang and Herley [38] worked on the intra-list diversity and made trade-offs between user's interest and the diversity of the top-N list. They modeled the competing goal of maximizing the intra-list diversity and maintaining acceptable similarity to the user query as a binary optimization problem. Andre *et al.* [5] defined the potential for serendipity as search results that are interesting but not highly relevant in Web search. In their another work, they took a broader view of serendipity, focusing more on human having prepared mind and infrastructure to support discovery, innovation, and creativity rather than creating the insight only [4]. Lathia *et al.* [20] discovered that temporal diversity is an important facet of recommender systems, by showing how collaborative filtering data changes over time and performing a user survey. Then they proposed set methods to maximize temporal recommendation diversity without compromising much accuracy. Kawamae [18] assumed that the degree of user's surprise is proportional to the estimated search time that the user would like to spend on finding the item by themselves. Following this, their recommender system hypothesized that the innovators' recent purchased items will surprise other users more than other items. Sugiyama and Kan [32] proposed an approach to recommend serendipitous scholarly papers by gathering preference information from dissimilar users and co-authors to construct serendipitous target user profile and comparing similarity between user profile and candidate papers to generate serendipitous recommendation. Adamopoulos and Tuzhilin [2] proposed a new concept of unexpectedness as recommending to users those items that depart from the consideration set of each user (those user expected from the system) and a method to produce unexpected recommendations based on the utility theory of economics. Zhang *et al.* [39] evaluated the Auralist framework for music serendipitous recommendation. The Auralist framework interpolated ranking of the output from three algorithms: item-based collaborative filtering algorithm, Listener Diversity that promotes the artists that have diverse listeners, and Declustering algorithm that recommends artists outside the artist clusters, while Schedl *et al.* [28] also proposed a music recommendation model that incorporated different factors including serendipity. In addition, they proposed the use of social media mining techniques to estimate popularity and hotness in a geo-aware manner. Lu *et al.* [25], instead of maximizing the metric of area under the ROC curve (AUC) in traditional personalized ranking approach, added a popularity weight to AUC to make serendipitous recommendation. Jenders *et al.* [16] combined a traditional cosine-based similarity and a topic unexpectedness model to recommend serendipitous and interesting news to a target user. In mobile app domain, Bhandari *et al.* [7] proposed a item-item similarity graph-based serendipitous recommendation approach.

Chapter 3

Proposed Method

Our work builds on Sugiyama and Kan's [32] dissimilar user approach in app domain but incorporates with Twitter app-related information to construct user profile. We also implements k-means clustering algorithm for serendipitous mobile app recommendations. Instead of looking for apps within the same cluster, we recommend apps from neighborhood clusters for serendipitous purpose.

In both methods, we use the meta-information about the apps to represent the apps. The meta-data includes app ID, title of the app, app description, app categories, and comments or reviews of apps obtained after preprocessing. We use app ID as an index to keep track of the apps, and app title, app description and comments (hereafter, "app document") to construct feature vectors of apps. For each app document d_{app}, we transfer it into a feature vector $f^{d_{app}}$:

$$f^{d_{app}} = (\omega_{t_1}^{d_{app}}, \omega_{t_2}^{d_{app}}, ..., \omega_{t_m}^{d_{app}}), \tag{1}$$

where m is the number of distinct d terms in app document and $t_k (k = 1, 2, 3, ..., m)$ denotes each term. Using TF-IDF scheme [27], each element $\omega_{t_k}^{d_{app}}$ of Equation (1) is defined as follows:

$$\omega_{t_k}^{d_{app}} = \frac{tf(t_k, d_{app})}{\sum_{s=1}^{m} tf(t_s, d_{app})} \cdot \log \frac{N_{d_{app}}}{df(t_k)},$$

where $tf(t_k, d_{app})$ is the frequency of term t_k in the app document, $N_{d_{app}}$ is the total number of candidate apps to recommend, and $df(t_k)$ is the number of app documents in which term t_k presents. We will use the app feature vector defined by Equation (1) to represent the apps in the following two approaches.

3.1 Dissimilar User Method Incorporated with Twitter

Our dissimilar user approach consists of the following three steps:

1. We construct app feature vectors $f^{d_{app_a}}(a = 1, 2, ..., N_{d_{app}})$ and pseudo-documents introduced by Lin *et al.* [21] from Twitter followers for all the candidate apps using Equation (1) and combine them as tuple to represent the candidate apps (app representation P_{app_a}).

2. We then construct a basic user profile P_u based on the app-related information on Twitter for the target user u that we are making recommendations to. Then, we construct serendipitous user profile P_u^{srdp} for further use.

3. Finally, we compute cosine similarity between $P_{app_a}(a = 1, 2, ..., N_{d_{app}})$ and P_u^{srdp} to generate the recommendation list in order of high similarity.

3.1.1 Incorporate Twitter IDs into App Representation

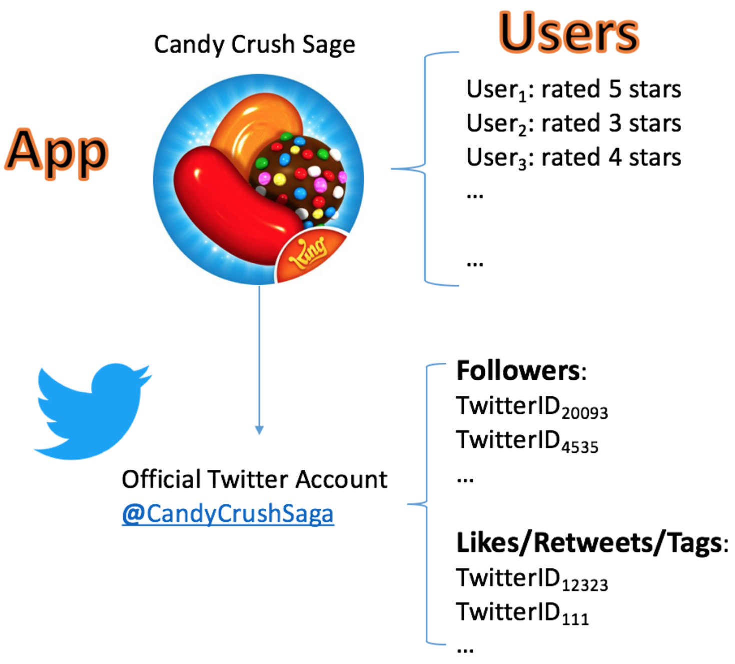


Figure 3.1: Pseudo-document makes use of the IDs of the twitter users who follow the app's official account and show interest in app-related content.

To represent an app, we borrow the concept from Lin's *et al.* [21] work of "pseudo-document" which contains the IDs of Twitter users who are interested in this app. In the work [21], only Twitter IDs that follow the apps and write tweets that contain app links are used. Here, we adjust the method and take more information from Twitter, including those Twitter IDs who write tweets containing app name as tags and and those who like the tweets containing app link of Google Play and the tags of the app names. Figure 3.1 illustrates the relationships among the app, its related Twitter IDs, and its users. We assign different weights to IDs from different source to indicate the different level of preferences. We then use both Twitter IDs and app feature vector to represent candidate apps. We assume that app a has a set of Twitter-followers $\{t_1, t_2, ..., t_m\}$ and a set of related Twitter users $\{t_{m+1}, t_{m+2}, ..., t_n\}$ who show preferences (likes or retweets the content containing app name or uses the tag of the app name) on app a. Then the pseudo-document of an app a can be represented as follows:

$$m_a = (\omega_1(t_1, t_2, ..., t_m), \omega_2(t_{m+1}, t_{m+2}, t_n)),$$

where ω_1 and ω_2 represent a different weights for direct Twitter followers and other Twitter users of app a.

Hence, the app a can be represented by combining the app feature vector and the pseudo-document:

$$P_{app_a} = (f^{d_{appa}}, m_a),$$

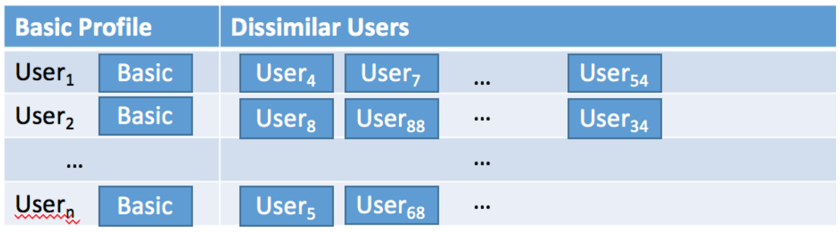
3.1.2 Basic User Profile Construction

In order to construct basic user profile, we use Lin *et al.* [21]'s pseudo-document method and take advantages of app-related information on Twitter to partially represent the apps. For each app user, we use the apps that the user likes (rated 3 and above) to construct the user profile. We also take into consideration the factors of different ratings and time decay, therefore, we assign different weights to different apps to give more emphasis on high ratings and recent ratings. We define user profile P_u as follows:

$$P_u = \sum_{j \in W} \omega_{rated} \times \omega_{decay} \times P_{app_j},$$

where ω_{rated} is defined as proportional to the rating given by User u to app j, and ω_{decay} is defined as $\omega_{decay} = \omega_0 e^{-\lambda t}$ where t is the time that has passed since the rating was given.

3.1.3 Serendipitous User Profile Construction



Basic Profile		Dissimilar Users			
User_1	Basic	User_4	User_7	...	User_{54}
User_2	Basic	User_8	User_{88}	...	User_{34}
...				...	
User_n	Basic	User_5	User_{68}	...	

Figure 3.2: Serendipitous user profile constructed from dissimilar user profiles.

Users with different interests may be able to produce interesting and surprising recommendations for the target user. In this approach, we construct serendipitous user profile by utilizing the profiles from maximally dissimilar users. Figure 3.2 illustrates how the user serendipitous profile will be constructed. In this way, we introduce different interests into a target user profile. We define serendipitous user profile, which is used for serendipitous recommendations later, as follows:

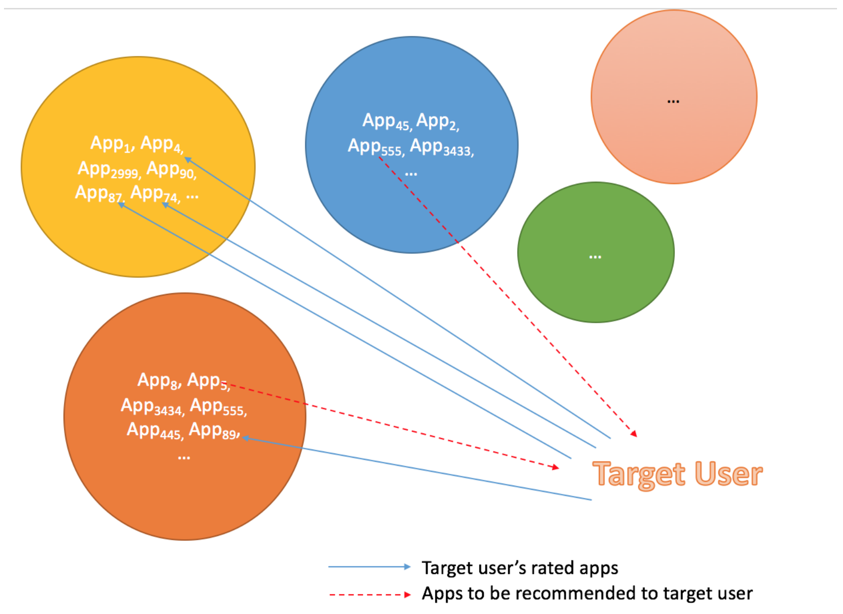
$$P_u^{srdp} = P_u + \sum_{v=1}^{N_{du}} \left(\frac{1}{sim(P_u, P_v) + k} \times P_v \right),$$

where P_u and P_v are the basic user profiles of user u and user v $(v = 1, 2, ..., N_{du})$ who are dissimilar users from the target user u. $sim(P_u, P_v)$ is the cosine similarity between user u and user v and we use the reciprocal of $sim(P_u, P_v) + k$ as the weighting factors of the basic profile from dissimilar user v, and k is a value that can set to place the bounded limit of the dissimilarity.

3.1.4 Similarity Calculation

To make recommendations, we compute the cosine similarity $sim(P_{app_a}, P_u^{srdp})$ where P_{app_a} is the app representation for each app a $(a = 1, 2, ..., N_{app})$ in the candidate list, and P_u^{srdp} is the serendipitous user profile defined in Section 3.1.3. We then generate the recommendation list in the order of the high similarity.

3.2 Serendipitous K-means Clustering Algorithm



App1, App4, App2999, App90, App87, App74, ...
App45, App2, App555, App3433, ...
...
...
App8, App5, App3434, App555, App445, App89, ...
Target User
Target user's rated apps
Apps to be recommended to target user

Figure 3.3: Clustering of candidate apps and apps to be recommended to target user

In this approach, we first construct k clusters of mobile apps by comparing the similarity among apps which is calculated using both liked user information and app document information. Then, for each target user, we recommend apps from the clusters which he never has downloaded an app so far. If there is multiple clusters that are new to the target user, the apps from the neighborhood clusters of the clusters the user is familiar to will be recommended in higher ranking. Figure 3.3 illustrates the clusters of apps and apps from neighborhood clusters will be recommended to the target user.

3.2.1 Clustering of Mobile Apps

In this approach, we use k-means clustering algorithm to group apps into k clusters. We previously define an app as a feature vector $f^{d_{app_i}}$, which contains all the meta-information of the app, however we also want to integrate the information that which users like this app. We utilize the users that like this app to represent a part of information of the app for later distance calculation. Therefore, we define a user vector for app i (app_i) as follows:

$$u^{d_{app_i}} = (u_1, u_2, ..., u_m),$$

where u_j ($j = 1, 2, ..., m$) represent the user IDs who like the app i. Then the app can be represented by combining user vector $u^{d_{app_i}}$ and feature vector $f^{d_{app_i}}$ into a tuple (referred to as "app representation" or "app" in the following algorithm):

$$P^{app_i} = (u^{d_{app_i}}, f^{d_{app_i}})$$

We also define the distance between two apps as follows:

$$D(P^{app_i}, P^{app_j}) = sim(u^{d_{app_i}}, u^{d_{app_j}}) + sim(f^{d_{app_i}}, f^{d_{app_j}}), \tag{3}$$

where sim is defined as cosine similarity.

The followings are the detailed steps in our serendipitous k-means clustering algorithm:

1. Define k, the number of clusters that will be generated by running this algorithm. Input all the candidate apps $P^{app_1}, P^{app_2}, ..., P^{app_n}$.

2. Randomly select k apps $C_1, C_2, ..., C_k$ as centroids of the k clusters.

3. Repeat the following steps until convergence:

   (a) For each app P^{app_q}:

      i. Find the nearest centroid C_r by calculating $min_q D(P^{app_q}, C_r)$

      ii. Assign the app P^{app_q} to cluster r

   (b) For each cluster $r = 1, 2, ..., k$:

      i. Calculate new centroid C_r, where C_r denotes all P^{app_q} that are assigned to the cluster r.

   Stop when none of the cluster assignment changes.

After clustering, we have k different clusters of apps with k centroids ($C_1, C_2, ..., C_k$) which are divided by both the common user groups and the content similarity. Based on the clusters and target user's preference, we can select serendipitous choices for recommendations.

3.2.2 Selecting Candidate Apps from Neighborhood Clusters

To recommend serendipitous apps for the target user, we first calculate the distance between each pair of clusters by calculating the distance of their centroids, and then define neighborhood of a cluster to be the three clusters that have the smallest distance to the target cluster.

For each target user, we analyze the apps liked by the target user to be distributed into a few clusters. There will be two cases:

Majority cluster: If the majority of the apps liked by the target user fall into cluster r with centroid C_r, then we explore the neighbors of cluster r. We select apps within cluster r neighbors which have the smallest distance to the centroid of r. All the apps within the neighboring clusters will be given a rank, defined as $1/(D(P^{app_i}, C_r) + k)$ where P^{app_i} is each app in the neighborhood clusters and C_r is the centroid of cluster r.

Outliers: There may be a few apps (less or equal to 2) liked by the target user fall into different clusters. Then, we explore the clusters themselves and choose the apps within the clusters but have the largest distance from their centroids. The rank of each app i within the clusters will be given by $1/(D(P^{app_i}, C_r) + k)$ for each app i with the clusters and their centroid C_r.

Finally, we combine the list of the selected candidate apps in the order of higher rank to generate a serendipitous recommendation list.

Chapter 4

Experiment

We will implement the approaches mentioned above and evaluate the effectiveness of the systems in future work.

4.1 Experimental Dataset

We have crawled 12,225 apps from Google Play[1] which will be used as our experimental dataset.

[1]https://play.google.com/store

Chapter 5

Conclusion

5.1 Contributions

In this paper, we have proposed two approaches to serendipitous mobile app recommendation. One is adapted from dissimilar user approach from Sugiyama and Kan's work [32], but we integrate the app-related information on Twitter to achieve serendipitous recommendation. Another approach, k-means clustering-based algorithm is novel since no work has employed k-means clustering algorithm to achieve serendipitous recommendation.

5.2 Future Work

In future work, we will implement the proposed approaches and verify the effectiveness of our proposed approaches described in Chapter 3.

References

[1] 42Matters. *App Market Stats.* [Online; accessed 24-October-2017]. 2017. URL: `https://42matters.com/stats`.

[2] P. Adamopoulos and A. Tuzhilin. "On Unexpectedness in Recommender Systems: Or How to Better Expect the Unexpected". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 5.4 (2014).

[3] E. Aimeur et al. "Alambic: a Privacy-Preserving Recommender System for Electronic Commerce". In: *International Journal of Information Security* 7.5 (2008), pp. 307–334.

[4] P. Andre, J. Teevan, and S. T. Dumais. "Discovery is Never by Chance: Designing for (Un)serendipity". In: *Proceedings of the 7th ACM Conference on Creativity and Cognition (C&C'09).* 2009, pp. 305–314.

[5] P. Andre, J. Teevan, and S. T. Dumais. "From x-rays to Silly Putty via Uranus: Serendipity and its Role in Web Search". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2009).* 2009, pp. 2033–2036.

[6] R. Baeza-Yates et al. "Predicting The Next App That You Are Going To Use". In: *Proceedings of the 8th ACM International Conference on Web Search and Data Mining (WSDM 2015).* 2015, pp. 285–294.

[7] U. Bhandari et al. "Serendipitous Recommendation for Mobile Apps Using Item-Item Similarity Graph". In: *Information Retrieval Technology.* Ed. by R. E. Banchs et al. Vol. 8281. Oxford: Springer, Berlin, Heidelberg, 2013, pp. 266–290.

[8] D. Cao et al. “Cross-Platform App Recommendation by Jointly Modeling Ratings and Texts”. In: *ACM Transactions on Information Systems (TOIS) - Special issue: Search, Mining and their Applications on Mobile Devices* 35.4 (2017).

[9] D. Cao et al. “Version-sensitive Mobile App Recommendation”. In: *Information Sciences: an International Journal* 381.C (2017), pp. 161–175.

[10] E. Costa-Montenegro, A.B. Barragans-Martinez, and M. Rey-Lopez. “Which App? A recommender system of applications in markets”. In: *Expert Systems with Applications (ESWA): An International Journal* 39.10 (2012), pp. 9367–9375.

[11] C. Davidsson and S. Moritz. “Utilizing Implicit Feedback and Context to Recommend Mobile Applications from First Use”. In: *Proceedings of the 2011 Workshop on Context-awareness in Retrieval and Recommendation (CaRR 2011).* 2011, pp. 19–22.

[12] M. Ge, C. Delgado-Battenfeld, and D. Jannach. “Beyond Accuracy: Evaluating Recommender Systems by Coverage and Serendipity”. In: *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys’10).* 2010, pp. 257–260.

[13] D. Goldberg et al. “Using Collaborative Filtering to Weave an Information Tapestry”. In: *Communications of the ACM - Special issue on information filtering* 35.12 (1992), pp. 61–70.

[14] C. Basu and H. Hirsh and W. Cohen. “Recommendation as Classification: Using Social and Content-based Information in Recommendation”. In: *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98).* 1998, pp. 714–720.

[15] J. L. Herlocker et al. “Evaluating Collaborative Filtering Recommender Systems”. In: *ACM Transactions on Information Systems (TOIS)* 22.1 (2004), pp. 5–53.

[16] M. Jenders et al. “A Serendipity Model for News Recommendation”. In: *Advances in Artificial Intelligence.* Ed. by S. Holldobler, R. Penaloza, and S. Rudolph. Vol. 9324. Lecture Notes in Computer Science. Springer, Cham, 2015.

[17] J. Kamahara et al. “A Community-Based Recommendation System to Reveal Unexpected Interests”. In: *Proceedings of the 11th International Multimedia Modelling Conference (MMM’05)*. 2005, pp. 1550–5502.

[18] N. Kawamae. “Serendipitous Recommendations via Innovators”. In: *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR’10)*. 2010, pp. 218–225.

[19] J. Kim et al. “Recommendation algorithm of the app store by using semantic relations between apps”. In: *The Journal of Supercomputing* 65.1 (2013), pp. 16–26.

[20] N. Lathia et al. “Temporal Diversity in Recommender Systems”. In: *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR’10)*. 2010, pp. 210–217.

[21] J. Lin et al. “Addressing Cold-start in App Recommendation: Latent User Models Constructed from Twitter Followers”. In: *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR’13)*. 2013, pp. 283–292.

[22] J. Lin et al. “New and Improved: Modeling Versions to Improve App Recommendation”. In: *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR’14)*. 2014, pp. 647–656.

[23] G. Linden, B. Smith, and J. York. “Amazon.com recommendations: item-to-item collaborative filtering”. In: *IEEE Internet Computing* 7.1 (2003), pp. 76–80.

[24] B. Liu et al. “Personalized Mobile App Recommendation: Reconciling App Functionality and User Privacy Preference”. In: *Proceedings of the 8th ACM International Conference on Web Search and Data Mining (WSDM 2015)*. 2015, pp. 315–324.

[25] Q. Lu et al. “Serendipitous Personalized Ranking for Top-N Recommendation”. In: *Proceedings of the 2nd Workshop on Context-awareness in Retrieval and Recommendation (CaRR 2012)*. 2012, pp. 258–265.

[26] J. Oh et al. “Novel Recommendation Based on Personal Popularity Tendency”. In: *Proceedings of the 11th IEEE International Conference on Data Mining (ICDM ’11)*. 2011, pp. 507–516.

[27] G. Salton and M.J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

[28] M. Schedl, D. Hauger, and D. Schnitzer. “A Model for Serendipitous Music Retrieval”. In: *Proceedings of the 2nd Workshop on Context-awareness in Retrieval and Recommendation (CaRR 2012)*. 2012, pp. 10–13.

[29] statista. *Number of available applications in the Google Play Store from December 2009 to September 2017*. [Online; accessed 24-October-2017]. 2017. URL: `https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/`.

[30] statista. *Number of available apps in the Apple App Store from July 2008 to January 2017*. [Online; accessed 24-October-2017]. 2017. URL: `https://www.statista.com/statistics/263795/number-of-available-apps-in-the-apple-app-store/`.

[31] K. Sugiyama and M.-Y. Kan. “Scholarly Paper Recommendation via User’s Recent Research Interests”. In: *Proceedings of the 10th ACM/IEEE Joint Conference on Digital Libraries (JCDL’10)*. 2010, pp. 29–38.

[32] K. Sugiyama and M.-Y. Kan. “Serendipitous Recommendation for Scholarly Papers Considering Relations among Researchers”. In: *Proceedings of the 11th ACM/IEEE Joint Conference on Digital Libraries (JCDL’11)*. 2011, pp. 307–310.

[33] Q. Wang et al. “A Novel APP Recommendation Method Based on SVD and Social Influence”. In: *Proceedings of the 15th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP 2015)*. 2015, pp. 269–281.

[34] W. Woerndl, C. Schueller, and R. Wojtech. “A Hybrid Recommender System for Context-aware Recommendations of Mobile Applications”. In: *Proceedings of the 23rd IEEE International Conference on Data Engineering Workshop (ICDEW’07)*. 2007, pp. 871–878.

[35] Q. Xu et al. “Identifying Diverse Usage Behaviors of Smartphone Apps”. In: *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference (IMC’11)*. 2011, pp. 329–344.

[36] B. Yan and G. Chen. “AppJoy: Personalized Mobile Application Discovery”. In: *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services (MobySys’11)*. 2011, pp. 113–126.

[37] P. Yin et al. “App Recommendation: A Contest between Satisfaction and Temptation”. In: *Proceedings of the 6th ACM International Conference on Web Search and Data Mining (WSDM’13)*. 2013, pp. 395–404.

[38] M. Zhang and N. Herley. “Avoiding Monotony: Improving the Diversity of Recommendation Lists”. In: *Proceedings of the 2008 ACM Conference on Recommender Systems (RecSys’08)*. 2008, pp. 123–130.

[39] Y. C. Zhang et al. “Auralist: Introducing Serendipity into Music Recommendation”. In: *Proceedings of the 5th ACM International Conference on Web Search and Data Mining (WSDM’12)*. 2012, pp. 13–22.

[40] V.W. Zheng et al. “Collaborative filtering meets mobile recommendation: a user-centered approach”. In: *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI-10)*. 2010, pp. 236–241.

[41] H. Zhu et al. “Mobile App Recommendations with Security and Privacy Awareness”. In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD’14)*. 2014, pp. 951–960.

[42] C. N. Ziegler et al. “Improving Recommendation Lists through Topic Diversification”. In: *Proceedings of the 14th International Conference on World Wide Web (WWW’05)*. 2005, pp. 22–32.