Explicit Modelling of the Implicit Short Term User Preferences for Music Recommendation

by

Kartik Gupta, Noveen Sachdeva, Vikram Pudi

in

2018 European Conference on Information Retrieval (ECIR 2018)
(*ECIR-2018*)

Grenoble, France

Report No: IIIT/TR/2018/-1

Explicit Modelling of the Implicit Short Term User Preferences for Music Recommendation

Kartik Gupta[1], Noveen Sachdeva[1], and Vikram Pudi[1]

[1]IIIT-H, Hyderabad, India
{kartik.gupta, noveen.sachdeva}@research.iiit.ac.in, vikram@iiit.ac.in

Abstract. Recommender systems are a key component of music sharing platforms, which suggest musical recordings a user might like. People often have implicit preferences while listening to music, though these preferences might not always be the same while they listen to music at different times. For example, a user might be interested in listening to songs of only a particular artist at some time, and the same user might be interested in the top-rated songs of a genre at another time. In this paper we try to explicitly model the short term preferences of the user with the help of Last.fm tags of the songs the user has listened to. With a session defined as a period of activity surrounded by periods of inactivity, we introduce the concept of a subsession, which is that part of the session wherein the preference of the user does not change much. We assume the user preference might change within a session and a session might have multiple subsessions. We use our modelling of the user preferences to generate recommendations for the next song the user might listen to. Experiments on the user listening histories taken from Last.fm indicate that this approach beats the present methodologies in predicting the next recording a user might listen to.

Keywords: Recommendation Systems, User Modelling

1 Introduction

Today with an increase in the use of digital platforms like Last.fm [9] for music sharing which have large databases, it has become a common practice to use recommendation systems to suggest musical recordings (songs) a user might listen to next. The next section describes the various types of techniques used for music recommendation such as collaborative filtering [22, 2], content based systems [4, 1, 6], sequence based systems [16, 24, 13, 3]. In addition to this, hybrid systems [23, 18, 5] are discussed which combine two or more techniques to enhance the performance of the recommender systems.

We feel the above systems are not fully able to capture the short-term preferences of the users in terms of features and the features which a user might give importance to might be much more in number than what is considered by these systems. For example, if a user wants to check out all the songs of a particular artist then content-based systems using the acoustic features to recommend songs might not perform very well but the systems which consider the metadata of the song might do good. To fully capture the preference of the users in terms of the attributes of the songs, we use the Last.fm[9] tags of each song which can be retrieved using their public API. These tags are able to describe a large variety of features of the song which may or may not be derivable from the audio of the song. For example tags like 'Guitar Solo' might be derivable from the audio of the song but there are tags like 'heartbreak' which might not be derivable from the audio.

Often users have breaks in their listening activity and the periods of activity on music sharing platforms surrounded by periods of inactivity are called sessions. Often people might have different preferences during different sessions and recommendation based on the sessions where the user preference was different might be rendered useless. There are recommender systems which take into account the sessions of a user and recommend music based on the sessions which had the same songs as the current ongoing session [15, 3]. In our work, we try to model the user preferences based on the tags of the songs that the user listens to. We introduce the concept of subsession which we define as the part of a session during which the preferences of the user remains constant. We then recommend a set of possible next songs user might listen to using based on the current subsession. The preference of the user is modelled by the Last.fm tags of the songs user has listened to in a subsession. Hence in this paper, we make the following contributions:

– We propose a model for capturing the short term preferences of the user by introducing the notion of a subsession.
– A simple recommendation system is built and tested using our model and we show that it indeed performs better than the current state of the art.
– We also test our model in predicting the immediate next set of preferences that the user might have and present the results for the same.

2 Related Works

We here present the previous works done to model the user preference for recommendation systems:

2.1 Collabrative Filtering

Collaborative filtering is a very widely used approach and recommends items to users based on the items selected by other similar users. It does not take into account the short-term preference of the user while recommending an item to the user. Similar is the item level collaborative filtering [2], wherein two items selected by the same user are considered to be similar.

Matrix factorization [22]brings about an improvement in collaborative filtering by finding out the latent user and item vectors and using them to find the probability if the user will like that item or not. Further improvements have been made to matrix factorization such as BPR [17] which is optimized for the task of ranking items for each user.

Session-based collaborative filtering [15] works by dividing the user activity into sessions and considering only songs of the active session to find similar sessions and recommend items to the user. This was further improved by another system by also taking into account the temporal context of the session and also apply topic modelling at the session level giving the role of a document to the session and songs acting as words [14].

In [12] the authors have tried to model the context of the user with the help of inputs from the user device such as the location, motion, calender etc and apply collaborative filtering on the context level. Their approach assumes an implicit correlation between the external environment and the songs the user might like at that particular time. However, we feel this might not always be the case. Sometimes a user might want to listen to something totally different than what their external environment represents.

2.2 Content Based Reccomendation

Content-based filtering is gaining popularity in recommender systems. In these systems, the items are recommended on the basis of their content [4, 1]. The content of a song usually considered are the audio features of the song such as the lyrics, instruments and the tempo of the song. Some of them might also use the metadata of the song such as the artist info or the tags associated with the song. These may or may not be machine derivable. If the content of a song is similar to the ones the user likes, then that song is recommended to the user. However, in most of the current systems, the parameters on which the interests of the user are calculated are limited and hence might not able to fully capture the interests of the user. For example, there are systems which recommend songs based on the melody of the song and hence assuming that melody is one of the features which a user might give importance to, however, this might not always be the case [7].

Liang [6] uses a very different approach and have trained the neural network to generate a latent factor vector for each song which is highly correlated to semantic tags and then used collaborative filtering to recommend songs to the user, however the neural network is trained for a small number of tags (561) which again might not be able to capture the complete preference of the user. Also, they recommend the songs based on the overall interest of the user rather than the short term preferences which our approach takes into account.

2.3 Sequence Based Recommendation

These recommender systems have started to gain popularity just recently and perform better than most of the non sequence based techniques such as collaborative filtering. Such systems base their recommendation on the sequence of items the user has already selected. Recommendation was first modeled as a sequence prediction problem by Brafman [13]. Initially, Markov chains were used to predict the recommendation and some of the improvements such as having a personal Markov chain model for each user [16]. Later on, with the popularity of neural networks, recurrent networks were successfully applied to next item prediction problem [24]. Sequential recommender systems have been extensively combined with other types of modelling to give out better results and we present some of them under the next subsection.

2.4 Hybrid Recommender Systems

There have been systems which have employed the techniques of content based as well as collaborative filtering based recommender systems. One such system was proposed by Yoshii [23] wherein they take into account the rating of the item as well as the content features. The content features are modelled using the polyphonic timbres in the song our approach is indeed a hybrid approach but in our case instead of taking into account the content of not only one item but the features of the songs recently heard by the user which might be important to the user at that point of time.

In [3] the authors combine the concept of sessions and sequence-based recommender systems. They apply sequence-based approach on the sessions rather than entire user history. We take this one step further and say that not even all items of the session are important and only last few items are important while making the prediction of the next item, and the number of few items on which the next item might depend is not constant.

In [8] Hariri. applies topic modelling to the sequence of the songs heard by the user and tries to predict the topic of the next song the user might be interested in. They learn the transitions between different topics from the playlists. Though this approach seems similar to what we propose in this paper but is indeed different. Our model allows explicit specification of the point in the history to which the current preferences of the user correspond to and hence allowing us to model the next song prediction problem as an information retrieval problem. Also, our model does not take into account the sequence but the group of items which were heard together while the user preference did not change much and we claim that its the items of the group which matters but not the sequence in which the items occurred.

3 Method

3.1 User Preference and Subsession

In this section, we present definitions of the terms used and formalization of the problem statement that we tackle in our paper.

Definition 1: Session *A session is a period of activity surrounded by periods of inactivity of at least x minutes, where x is the given threshold.*

There is no constraint on the duration of a session, however the gap between the end and start of two different sessions should at least be x minutes for them to be considered as different sessions.

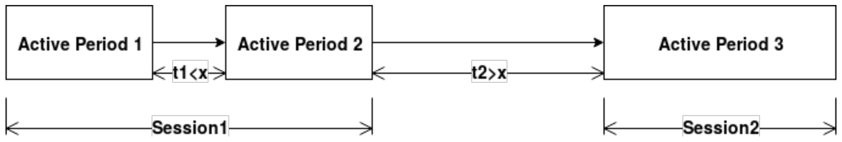

Fig. 1. Division of User Activity into Sessions

In Figure 1 the user activity is divided into three periods. The time difference between the end of period 1 and start of period 2, t_1 is less than x minutes , and hence are considered to be the part of a single session, The time difference between the end of period 2 and the start of period 3, t_2 is more than x minutes and hence this marks a session boundary putting period 3 in a different session.

Definition 2: Preference Tag Set *Considering each song as a transaction and each tag of that song as an item, the set of items with a minimum support p, in q consecutive transactions, is called as the Preference Tag Set and is denoted by Tags(q).*

Note that in the above definition *item* means a tag of a song, however in rest of the paper it means a song itself. We have presented the above definition for an easy understanding of the reader. In the above definition we model the preference of the user over the q consecutive songs listened by the user. The tags present in the preference tag set can be assumed to be related to the features of the items which matter most to the user. The tags which are associated to some of the items among the q items but not present in the preference tag set are assumed to be given less importance by the user and hence we can skip them while deciding the next items to recommend.

Definition 3: Subsession Seed *A window w of l consecutive items in a user session with a preference tag set $Tags(w)$, for a given fixed l.*

Definition 4: Subsession *A window m ending with a given subsession seed w, where $Tags(w) = Tags(m)$ and m and w occuring in the same session.*

Note that for all practical purposes, two consecutive subsessions with same preference tag sets would be merged together.

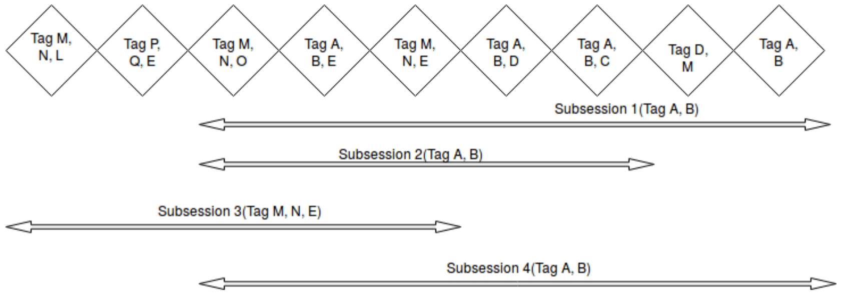

Fig. 2. Division of User Activity into Subsessions

We describe the subsession mining process with the help of example given in figure2. In the example, we keep the minimum support for a tag to be in the preference tag set to be 0.51 and the length of subsession seed as 5. A value of 0.51 as minimum support was chosen so that the tags associated to only the majority of songs can make it to the preference tag set. In the figure if we see the subsession 1, the seed will be the last 5 items for which the preference tag set will be $[A, B]$ as only these correspond to more than 51% of the items.Only the tags A and B occur more than or equal to 3 times and hence make it to the preference tag set of that subsession seed. To find the subsession with the seed of last 5 items, we increase the window to contain two more items towards the left(less recently heard songs) as the preference tag set for those 7 items and the preference tag set of the seed remain the same. We cannot expand it any further as the preference tag then changes. Note that a subsession seed always ends the subsession ie: A subsession will never include items which occurred after the subsession seed in the user history. For $Subsession2$ the seed is the 5 items before the last two items. If we try to increase the window towards less recently heard songs we see the preference tag set changes hence this subsession is equal to its subsession seed. Now since the preference tag set of $Subsession1$ and $Subsession2$ we combine them into $Subsession4$. If we consider $Subsession3$, the preference tag set $[M, N, E]$ for that is different and cannot be combined with $Subsession4$ and hence is kept separate. As is clear, we allow for overlapping subsessions.

This way all the subsession seeds are considered with a hop size of 2 and subsessions corresponding to them are found.

Below we present the problem statements we tackle in this paper.

Predicting Next Song *Given the q most recently listened songs by the user, predict the top k candidate songs which the user is most likely to listen to after these q songs.*

Once we find an appropriate session and subsession we can recommend items which the user might like after the given items.

Predicting Next User Preference *Given the m most recent subsessions of the user, predict the preference tag set the user can have during the next subsession*

3.2 Using Subsessions to Recommend Music

As we have defined subsessions in the previous section we here explain how we use subsession to find the top k candidates songs which the user might listen to next. Given q most recently heard songs heard by the user, we model the active subsession. Then we find the similar subsessions in the training data to the active subsession. The recommendation uses a n similar subsessions of the active subsession. While mining each subsession we also store the user from whose log the subsession is mined and we define that as the *Owner* of the subsession.

There are two matrices in our approach. One is the subsession-song matrix and the other one is the subsession-tag matrix. One of the axes of these matrices are the subsessions and the other axis are songs and tags respectively. The entries of these matrices are often called as ratings. The columns of these matrices tell what all songs were present in a subsession and what tags were present in the preference tag set of the subsession. The ratings in the subsession-song matrix are 1 if the song was present in the subsession and 0 otherwise. Similarly the ratings in the subsession-tag matrix are 1 if the tag was present in the preference tag set in that subsession and 0 if otherwise. Below we present the similarity measure Tanimoto coefficient [20] to find the similarity between two subsession-item vectors S_i and S_j

$$sim(S_i, S_j)_{item} = \frac{\sum_{k=1}^{t} r_{ik}.r_{jk}}{||S_i|| + ||S_j|| - \sum_{k=1}^{t} r_{ik}.r_{jk}} \tag{1}$$

where item corresponds to songs or tags and t is the total number of items, and r_{ik} denotes the rating of item k in subsession i

$$||S_i|| = \sum_{k=1}^{t} r_{ik}.r_{ik} \tag{2}$$

We use the Tanimoto coefficient instead of the generally used cosine similarity because we wanted the total number of items which are present in only one set (either S_i or S_j) to decrease the similarity value which is not taken care of in the cosine similarity. Cosine similarity only accounts for the number of similar items in the two sets. For example: If two sets have 5 items each and 3 items are common, the cosine similarity would give us the similarity value 0.6 whereas our measure would give us a value of 0.428 which tells us that there is a large number of items which are not common in both the sets.

To calculate the similarity between any two given subsessions we find a weighted sum of similarity between subsession-tag interaction vector of the two subsessions, similarity between subsession-song interaction vector of the two subsessions and Kronecker delta function having $Owner_i$ and $Owner_j$ as inputs.

$$Sim(S_i, S_j) = w_1 * sim(S_i, S_j)_{tags} + w_2 * sim(S_i, S_j)_{songs} + w_3 * \delta_{Owner_i, Owner_j} \tag{3}$$

where

$$\delta_{a,b} = \{ 1 \text{ , if } a = b, 0, \text{if } a \neq b. \tag{4}$$

The reason we use the songs contained in a subsessions to calculate similarity is that for the same value of the preference tag set there might be a very large number of subsessions and hence the existence of common songs in two subsessions is used to increase the rank of subsession having common songs with the active subsession.

Once we have the n nearest neighbour subsessions we need to find the top k songs which occur in these subsessions. The score $Score_i^A$ for item i is then calculated to predict if it is likely to occur in the given active subsession A.

$$Score_i^A = \sum_{j=0}^{n} r_{ji}.Sim(S_A, S_j) \tag{5}$$

Based on the scores for each item obtained from the above equation, we choose top k candidates and recommend them to the user.

4 Experiments

In this paper, we have tackled two tasks. Section 4.1 presents some statistics about the dataset, and 4.2 talks about the mining of subsessions from the dataset. Section 4.3 presents the methodology and results for the next preference set prediction task. Section 4.4 onwards we present the bassline models and the results for our approach for the next song prediction task.

4.1 Dataset

The dataset was a subset taken from the Last.fm dataset [10]. The original dataset consisted of the entire listening history of about 1000 users until May, 5th 2009. Each log in the dataset consisted of user id, song name, artist name and time stamp. We performed experiments on a subset consisting of 6 month histories of all the users. The original dataset did not include tags for each song but we could retrieve it using the Last.fm public API. We only considered the users who had some activity in the period considered and the number of such users was 759. Below we present some dataset statistics.

Description	**Value**
Total Logs	3553321
Total Users	759
Total Sessions	110410
Total Unique Songs	386046
Average Songs Per Session	32.18
Average logs per user	4681.58

Description	**Value**
Total Subsessions	498800
Subsessions of len 5	294410
Subsessions of len 7	111694
Subsessions of len 9	43052
Subsessions of len 11 to 19	42471
Subsessions longer than 19	7173

Table 1. Dataset And Subsession Statistics

4.2 Sessions and Subsessions

The entire log for each of the user was divided into sessions with the difference between the start and end of two sessions $x = 120$ minutes. The minimum length for a session was kept as 5 songs. If we found a session which had a length less than 5 songs we discarded it since it won't have any valid subsessions. Once we had all the sessions for each user they were chronologically sorted and 70% of the first occurring sessions were put in the train data and the last 30% of the sessions put into the test data. For the models which don't require the history to be divided into sessions, the session boundaries were discarded while training and testing.

For each session in the training set, all the valid subsessions were found keeping the length of subsession seeds to 5 and min support to 0.51 for preference tag set. To predict the next song, we find the n most similar subsessions to the active subsessions and provide k highest ranked songs to the user. For our experiments we found $n = 50$ and the weights for the similarity measure between two subsessions $w_1 = 0.25$, $w_2 = 0.5$, $w_3 = 0.25$ giving the best results.

4.3 Next Preference Set Prediction

As per our knowledge, this kind of task has never been taken up in the past and our modelling of the user preference allows us to be the first ones to do this. In this problem we treat each subsession as a basket of tags and given the latest sequence of baskets from the user history, this problem reduces to the next basket prediction task. We use the neural network model described in [19]. The number of subsessions while predicting the preference tag for the next subsession was kept as 5. For each tag, a distributed representation [21] was computed based on the tags appearing together in a preference tag set of subsessions in the training dataset. The neural network implemented predicts the tags which might occur in the preference tag set of the next subsession. We report $precision@k$ and $recallk@$ for this task.

Metric	**k=5**	**k=10**	**k=15**	**k=20**	**k=25**
$Precision$(%)	76.67±2.45	65.71±2.14	57.45±2.24	50.89±1.98	45.69±1.83
$Recall$(%)	28.11±1.16	43.82±1.57	54.11±1.59	61.42±2.08	66.87±2.13

A high precision for low values of k shows that most of the tags predicted by the model were indeed present in the actual preference tag set and a high recall for higher values of k show that we were indeed able to predict of the tags of the preference tag set successfully.

4.4 Baseline Models

We use the following baseline models to compare with our model. For POP, BPR-MF, RNN we use the implementation provided by the authors of [11]. BPR-MF and RNNs were run 5 times each and we report the mean values for them. We refer to our model as SBRS(Subsession Based Recommender System).

1. POP: Users are recommended the most popular items in the training set.

2. BPR-MF: It is a matrix factozisation based state of the art model which ranks items differently for each user [17]. The original implementation was used from MyMediaLite, using all the default values except the number of features which we kept 100 for our experiments as they gave the best results. The user histories in the training data were used to build the user profiles.
3. Session Based Collaborative Filtering(SSCF): In this method, the similar sessions are found in the training data and songs listened to in those sessions are ranked based on the number of similar sessions they occurred in and are then recommended to the user. For our experiments, we find the similar sessions based on the last 5 songs heard by the user and the number of similar sessions to consider were kept to 100 as they gave the best results.
4. RNN: In this method, the sequence of items occurring together are fed to a recurrent neural network and the target being the next item in the sequence. All the sequences in the training data are used to learn the model and to get the next recommendation, all the items heard by the user until that point are fed to the network. The Categorical Cross Entropy loss function was used with the number of hidden units 100, learning rate of 0.1. The implementation we used uses mini match stochastic gradient descent method and we kept the batch size to 20.

We built an *oracle system* which magically knows the most listened songs for each user from the user history in the test data and recommends them to the user each time. Note that it always recommends the same items to the user. This is an upper bound on the performance of the systems which always provide a constant recommendation to the user as most of the matrix factorisation based methods do. The systems which take into account the recently listened songs by the user to generate recommendations can beat this oracle. In addition to the above baselines, we also compare our model with this oracle.

4.5 Testing

We iterate through entire user histories of the user in the test data to predict the next song given all the songs heard by the user till then in the test history. We report the Hit Ratio $HR@k$ [15] by varying k from 10 to 100.

Model	**k=10**	**k=20**	**k=30**	**k=40**	**k=50**	**k=60**	**k=70**	**k=80**	**k=90**	**k=100**
POP	0.85	0.97	1.24	1.69	2.14	2.31	2.58	2.83	2.98	3.08
BPR-MF	7.34	8.13	8.56	8.98	9.27	9.36	9.72	9.87	10.03	10.16
Oracle	13.22	19.48	23.95	27.56	30.48	**32.79**	**34.93**	**36.80**	**38.55**	**40.07**
SSCF	13.69	17.12	19.66	21.30	22.34	23.22	23.93	24.46	24.88	25.24
RNN	14.42	16.26	16.74	17.09	17.38	18.02	18.88	19.10	19.76	20.43
SBRS	**19.15**	**26.14**	**28.83**	**30.35**	**31.40**	32.19	32.81	33.32	33.77	34.16

Table 2. Results

5 Results And Conclusions

As is clear from the results table our model outperforms the baseline models by a huge margin. It even outperforms the oracle up to $k = 50$ which we think is an achievement and as is expected, the models which take limited user history into account while making recommendations perform better as they implicitly do consider the short term preference of the user, though only our model beats the oracle. The strength of our approach is that it models the recommendation task as a problem of retrieving same instances of user preferences in the training data and recommend items to the user based on those past instances. The results for the next user preference prediction task are encouraging enough to apply this kind of modelling to other domains as well to understand user behaviour and preferences.

This indeed looks like a promising and simple approach and can be applied to a diverse set of fields and tasks such as trends in user preferences and recommendation in other multimedia domains.

6 Acknowledgements

We would like to thank Alastair Porter for his valuable feedback on the initial drafts of this paper and improving the quality of this paper.

References

1. A. V. D. Oord, S. Dieleman, and B. Schrauwen. Deep content-based music recommendation. In NIPS , pp. 2643−2651, 2013.
2. Badrul Sarwar, George Karypis, Joseph Konstan and John Riedl: Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th International Conference on World Wide Web, WWW 01, pp. 285−295, New York, NY, USA, 2001
3. B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk: Session- based recommendations with recurrent neural networks. In Pro- ceedings of ICLR16, 2016
4. Brian McFee, Luke Barrington, and Gert R. G. Lanckriet.:Learning content similarity for music recom- mendation. IEEE Transactions on Audio, Speech Language Processing , 20(8), 2012
5. Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D. and Sartin, M.: Combining Content-Based and Collaborative Filters in an Online Newspaper, SIGIR 99 Workshop on Recommender Systems: Algorithms and Evaluation . Berkeley, CA.
6. Dawen Liang, Minshu Zhan, and Daniel PW Ellis: Content-aware collaborative music recommendation using pre-trained neural networks. In Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Malaga, Spain, October 26-30, 2015.
7. Fang-Fei Kuo and Man-Kwan Shan: A personalized music filtering system based on melody style classification. In 2002 IEEE International Conference on Data Mining, 2002. Proceedings., pp. 649−652, 2002

8. Hariri, N., Mobasher, B., Burke, R.: Context-aware music recommendation based on latenttopic sequential patterns. Sixth ACM conference on recommender systems, pp. 131−138. Dublin (2012)
9. Last.fm: http://www.last.fm, Retrieved date: 2017/04/23
10. Last.fm-dataset-1k-users. http://www.dtic.upf.edu/ocelma /MusicRecommendationDataset/lastfm-1K.html. Retrieved date: 2017/04/23
11. R. Devooght and H. Bersini.: Long and Short-Term Recommendations with Recurrent Neural Networks. Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization (2017), pp. 13−21
12. Reena Pagare, Intekhab Naser, Vinod Pingale, and Nayankumar Wathap: Enhancing collaborative filtering in music recommender system by using context based approach.
13. R. I. Brafman, D. Heckerman, and G. Shani: Recommendation as a stochastic sequential decision problem. In ICAPS, pp. 164−173, 2003.
14. Ricardo Dias and Manuel J Fonseca.: Improving music recommendation in session-based collaborative filtering by using temporal context. In International Conference on Tools with Artificial Intelligence, pp. 783−788. IEEE, 2013.
15. S. E. Park, S. Lee, and S. g. Lee.: Session-based collaborative filtering for predicting the next song. In 2011 First ACIS/JNU International Conference on Computers, Networks, Systems and Industrial Engineering, pp. 353−358, 2011.
16. S.Rendle, C. Freudenthaler and L. Schmidt-Thieme: Factorizing personalized markov chains for next-basket recommendation. In WWW, pp. 811−820. ACM, 2010.
17. S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme: Bpr: Bayesian personalized ranking from implicit feedback. In Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence, pp. 452−461. AUAI Press, 2009.
18. Suglia, Alessandro and Greco, Claudio and Musto, Cataldo and de Gemmis, Marco and Lops, Pasquale and Semeraro, Giovanni: A Deep Architecture for Content-based Recommendations Exploiting Recurrent Neural Networks, In Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization, pp. 202−211. ACM, 2017.
19. S. Wan, Y. Lan, P. Wang, J. Guo, J. Xu, and X. Cheng: Next Basket Recommendation with Neural Networks, in Poster Proceedings of RecSys 2015.
20. Tanimoto T. An elementary mathematical theory of classification and prediction. In Internal IBM Technical Report, 1958.
21. Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A Neural Probabilistic Language Model. JMLR, 3, 2003.
22. Y. Hu, Y. Koren, and C. Volinsky.: Collaborative filtering for implicit feedback datasets. In ICDM, pp. 263−272, 2008.
23. Yoshii, Kazuyoshi and Goto, Masataka and Komatani, Kazunori and Ogata, Tetsuya and Okuno, Hiroshi G: Hybrid Collaborative and Content-based Music Recommendation Using Probabilistic Model with Latent User Preferences., Proceedings of the International Conference on Music Information Retrieval 2006
24. Y. Zhang, H. Dai, C. Xu, J. Feng, T. Wang, J. Bian, B. Wang, and T. Liu: Sequential click prediction for sponsored search with recurrent neural networks. In AAAI, pp. 1369−1375, 2014.