IJCTA, 9(10), 2016, pp. 4575-4582

A Novel Music Recommendation to Address Long Tail Problem

***M. Sunitha *Dr. T. Adilakshmi**

Abstract : Recommender systems are the tools used in the era of Information overloading to provide filtered and user interesting information. Most of the commonly used Recommendation systems are Amazon book recommendation, Pandora music recommendation etc. In this paper we proposed a Music recommendation system which will address long tail problem by considering sessions. Long tail means the items which are not popular. Sessions are formed by using the timestamp. The proposed algorithm is implemented on Last.fm bench mark dataset. Experimental results prove that the proposed method is capable of recommending items from long tail and thereby improving the Precision of the proposed method.

Keywords : Recommender systems, Information overloading, long tail

1. INTRODUCTION

With the advancements in the E-commerce and Internet, millions of items are available at the finger tips of the users, which is known as information overloading. Because of this information overloading it is very difficult for the users to find interesting information. Recommender systems play a vital role to solve information overloading by filtering the items and providing valuable recommendations to the users.

Recommendation systems can be used for book recommendation, movie recommendation, event recommendation, Music recommendation etc.

Nowadays, users have large number of choices of which music to listen to. We have many music portals and applications which allows users to listen almost everything on the Internet. But all these applications will provide non personalized music i.e. most popular items or recently played items etc. We proposed a personalized music recommendation system which will provide recommendations based on users history and interest. The proposed system also addresses one of the major challenges of recommender systems, long tail problem.

In literature the recommender systems are broadly categorized as Collaborative filtering (CF) approach and content based approach.

CF is based on the idea that, if users have same taste in the past, they will also have same taste in the future. CF is a memory based method that finds the users most similar to the target user in order to recommend items.

Content based method uses the content of the item such as pitch, rhythm etc. in order to recommend items.

The rest of the paper is organized as follows. Section 2 explains Related work. Section 3 discusses about the proposed algorithm. Section 4 describes conclusion and future directions of research.

2. RELATED WORK

2.1. Formulating Recommendation problem

Let U = {$u1, u2, \ldots$ um} be the set of all users, and let I = {$i1, i2, \ldots$ in} be the set of all possible items that can be recommended.

* Department of Computer Science & Engineering, Vasavi College of Engineering Ibrahimbagh, Hyderabad-31, AP, India. sashu2006@gmail.com t_adilakshmi@rediffmail.com

Iui represents the set of items rated by user ui Note that Iui $\subset$ I, and it is possible that Iui be null also. Now the recommendation problem is to find the subset of items which the user will like most and which are not already rated by the user as shown in the fig. 2.1.2. Recommendation component will find the items in recommendation list by considering various features of users, items and both.

The item space I and user space U are very large and are represented by user-item rating matrix as shown in fig 2.1.1.

Item /User	***Item$_1$***	***Item$_2$***	***… …***	***Item$_n$***
User$_1$	S_{11}	S_{12}	… …	S_{1n}
User$_2$	S_{21}	S_{22}	… …	S_{2n}
… …	… …	… …	… …	… …
User$_m$	S_{m1}	S_{m2}	… …	S_{mn}

Fig. 1. User-Item Matrix.

Where S_{ij} represents the rating of user U_i for item I_j. As the item space and user space is very large, the user-item rating matrix is very sparse. Sparsity is one of the important challenge of Recommendation system. The other challenges faced by recommendation systems are Scalability, Cold-start problem and long tail problem.

Scalability is the ability of a recommender system to work well with increased size of data.

Cold-start problem is not able to provide recommendations to new users and new items.

Long tail means 90% of the items are listened by 10% of users and remaining 10% of items are listened by 90% of users i.e. popularity bias. This issue is addressed in this paper and results show an improvement over traditional method.

Fig. 2. Structure of a recommendation system.

2.2. User – based Collaborative Filtering

In the user based collaborative filtering approach, user ratings for each item will be used to predict the set of items which will be interesting for the user. As shown in the Fig.2.2.1. Alice has rated the items from I1 to I4 and his rating for I5 is predicted based on the user most similar to Alice. Let User2 is most similar to Alice, so in user based CF will use his ratings in order to predict Alice rating for I5.

2.3. Item – based Collaborative Filtering

In the item based CF approach, item-profiles are built instead of user profiles. Similarities between any two given items can be measured by proximity measures like Euclidean distance, Tanimoto coefficient and Log likelihood similarity. As given in fig. 2.2.1. I1 and I5 are more similar to each other i.e. these two items got same ratings. So Alice can use rating of I1 to predict missing rating for I5.

	I1	*I2*	*I3*	*I4*	*I5*
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

Fig. 3 Sample user-item ratings.

3. PROPOSED METHOD

This section consists of long tail problem definition, the proposed approach to address long tail problem, provide novel items as recommendations to users and evaluation measures used to evaluate proposed approach.

3.1. Long tail problem

The Long Tail is the set of items which are not very popular. Most of the items belong to long tail and very few items belong to heavy tail as shown in Fig. 3.1.1. Nowadays, we are moving towards the Hit vs. Niche paradigm. The problem, though, is to filter and present the right artists to the user, according to her musical taste even though they does not belong to heavy tail.

Chris Anderson introduces in his book, "The Long Tail", a couple of important conditions to exploit the content available in niche markets. These are:

1. make everything available
2. help me find it (Anderson, 2006).

The first condition is already met as everything is available on the internet. Second condition can be satisfied by using Recommender systems which presents users with items from long tail.

Fig. 4. long tail VS heavy tail.

3.2. Algorithm to address long tail

The long tail problem is addresses by using frequency distribution of the items. The following are the steps to present novel recommendations to users from long tail.

1. Items are divided into long tail and heavy tail based on the average frequency of each item. If any items average frequency is less than the average frequency of all items then it is added to the long tail otherwise it is added to the heavy tail.
2. Items are separated into long and heavy tail. User based model is used to find the user clusters for long tail and heavy tail by using the pseudocode sown in fig.3.2.1. Similarly Item based model is used to form item clusters by using the pseudocode shown in fig 3.2.2 for long tail and heavy tail.
3. Each test user is mapped to the most similar user cluster in long tail as well as heavy tail. Test user is also mapped to the most similar item cluster in long tail as well as heavy tail by hiding some of his items.
4. Test user will be presented with recommendations from long tail and heavy tail by considering user model and item model.

1. **Algorithm** Threshold_clusters()
2. Begin
3. Initialize the threshold value to th_cutoff
4. For each user in $u_1, u_2 \ldots\ldots\ldots\ldots u_n$
5. Assign u_1 to cluster C_1
6. For each user u_i in $u_2 \ldots\ldots\ldots\ldots u_n$
7. begin
8. Find the similarity of each user u_i with C_1
9. Assign u_i to C_1 if the sim(u_i , C_1) d" th_cutoff
10. Otherwise create a new cluster C_2
11. end
12. Return the clusters $C_1, C_2 \ldots\ldots\ldots\ldots C_k$
13. End

Fig. 5. Pseudocode for user-based model.

1. **Algorithm** Threshold_Itemclusters()
2. Begin
3. Initialize the threshold value to th_cutoff
4. For each item in $i_1, i_2 \ldots\ldots\ldots\ldots i_n$
5. Assign i_1 to cluster C_1
6. For each user i_i in $i_2 \ldots\ldots\ldots\ldots i_n$
7. begin
8. Find the similarity of each item i_i with C_1
9. Assign i_1 to C_1 if the sim(i_i , C_1) *d*" th_cutoff
10. Otherwise create a new cluster C_2
11. end
12. Return the clusters $C_1, C_2 \ldots\ldots\ldots\ldots C_k$
13. End

Fig. 6. Pseudocode for item-based model.

The similarity measures used in user model and item model are Euclidean distance and Cosine similarity as defined below between two vectors *p* and *q*.

$$d(p, q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \ldots + (q_n - p_n)^2} = \sqrt{\sum_{i=1}^{n} (q_1 - p_1)^2}$$

$$\text{Cos(A,B)} = \text{A.B}/||\text{A}||\,||\text{B}||$$

Where A and B are the vectors. Cosine similarity defines the cosine angle between two vectors A and B.

3.3. Sessions in long tail problem

The proposed method in this paper also considers the context information while recommending items. User logs are divided into four sessions based on the timestamp i.e. the time at which user listens to a particular item. The basic idea behind the sessions is users will have different preferences at different times of the day. User may prefer devotional items in the early morning and melody songs at night etc. Each session is separately handled to find the items in long tail and heavy tail. User based model and item based model are applied to each session for the items in long tail and heavy tail. The test user will be presented recommendations based on the session and from both long tail and heavy tail.

3.4. Performance measures

Any data mining task must be evaluated by using some evaluation measures. In this research work the evaluation measures used to evaluate proposed recommender systems are Precision , Recall and F-score.

Precision reflects the percentage of items listened out of total recommendations as shown in the equation given below

$$\text{Precision} = \frac{TP}{TP + FP}$$

Recall on the other hand measures how many item that recommendation system able to recommend out of total relevant items as shown in the equation given below

$$\text{Recall} = \frac{TP}{TP + FN}$$

The harmonic mean of Precision and Recall is measured by using F-measure given below

$$\text{F} - \text{measure} = 2 * \frac{(\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})}$$

4. RESULTS

The proposed approach is implemented on bench mark data set obtained from Last.fm. The data considered to carry out the experiment is 50 user's listening history from January 2008 to December 2008 i.e. 1 year data. The total number of records in the data considered are 2,60,564. Each record consists of the following fields.

User id (User_000004) –a user-id of the format user_000004.

Date–Time (2009-04-09T12:49:50Z) – Time of activity is recorded known as timestamp

AlbumId(078a9376-3c04-4280-b7d720e158f345d) – A unique identifier is Attributed to each Album.

Album name (Frightened Rabbit) – An album to which that song belongs to.

Trackid(5ca13249-26da-47bd-bba7-80c2efebe9cd) – A unique identifier is attributed to each track / song.

Track name (Old Old Fashioned) – The songs which the user listened to.

User logs are considered to form user-item matrix as shown in the table 4.1. It consists of number of times user preferred a particular item *i.e*. frequency of each item.

	Item 1	***Item 2***	**….**	***Item n***
User 1	2	0	….	0
User 2	0	4	….	1
….	….	….	….	….
Userm	0	0	….	1

Fig. 7. user-item matrix.

Experiment is carried with sessions and without sessions.

4.1. Without Sessions

The first step in this experiment is that items are divided into long tail and heavy tail. The details of users and no. of items in long and heavy tail are given below.

No. of heavy tail items : 3561
No. of long tail items : 11435
No .of Item Clusters : 395
Precision : 0.0243
Recall : 1.0
F-Score : 0.024

User model and item model as shown in pseudocode fig. 3.2.1. & fig 3.2.2. used to form user clusters and item clusters. Each test user is mapped to most similar user cluster and item cluster in both long tail and heavy tail. Test user is presented with recommendations by repeating the experiment at various threshold values.

4.2. With Sessions

Sessions are formed by considering timestamp as given below.

Session S1 = timestamp 0 to 6
Session S2 = timestamp 6 to 12
Session S3 = timestamp 12 to 18
Session S4 = timestamp 18 to 24

The experiment is repeated for various threshold values. Precision, Recall and F-score are calculated as shown in table 4.2.1 & table 4.2.2. These values are compared with the proposed method by considering long tail and heavy tail.

Table 4.2.1. P, R, F-score with only sessions for User model

Sessions	*No. of Users*	*No. of Songs*	*No. of User Clusters*	*P*	*R*	*F-Score*
S1	33	2056	22	0.037	1.0	0.068
S2	34	2220	23	0.027	1.0	0.051
S3	38	5231	27	0.016	1.0	0.031
S4	38	4855	27	0.011	1.0	0.021

Table 4.2.2. P,R,F-score with only sessions for Item model

Sessions	*No. of Users*	*No. of Songs*	*No.of Song Clusters*	*P*	*R*	*F-Score*
S1	33	2056	44	0.037	1	0.068
S2	34	2220	44	0.027	1	0.052
S3	38	5231	73	0.016	1	0.031
S4	38	4855	70	0.011	1	0.022

4.3. Results for sessions with long tail problem

Precision for sessions with long tail is compared with only sessions by using Item model as shown in Fig.4.1. As shown in the graph the proposed approach to address long tail improves precision over with sessions. F-score which is harmonic mean of Precision and Recall also improved for the proposed approach as shown in Fig.4.2.

Table 4.3. No.of of items, users in long and heavy tail with session

Sessions	*No. of Users*	*No. of Songs*	*No. of Song Clusters*
S1(H)	27	494	1
S1(L)	33	1562	20
S2(H)	29	512	5
S2(L)	34	1993	35
S3(H)	38	1099	34
S3(L)	38	4132	300
S4(H)	34	1144	2
S4(L)	37	3711	3

Table 4.4. P,R,F-score with sessions and long tail for Item model

Sessions	*Precision*	*Recall*	*F-Score*
S1	0.088	1.0	0.080
S2	0.097	1.0	0.088
S3	0.085	1.0	0.075
S4	0.084	1.0	0.078

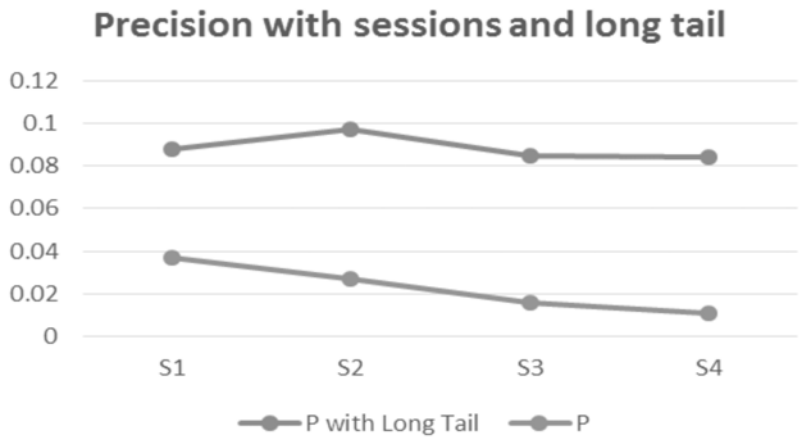

Fig. 8. Precision comparison with Session and long tail.

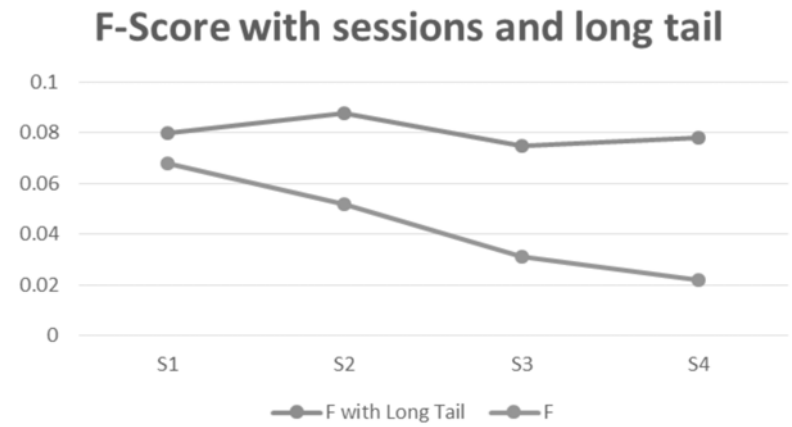

Fig. 9 F-Score comparison with Session and long tail.

5. CONCLUSION & FUTURE DIRECTIONS

Music recommendation system to address long tail problem has been proposed in this paper. The proposed method divides items into long tail and heavy tail. User and item similarities are used to form user clusters and item clusters. Then the test user will get recommendations from most similar cluster in long tail and heavy tail. The proposed approach is able to present novel items to the users even though they are not popular. The results show that the proposed approach improves the performance of the recommendation system.

Recommendation system proposed in this paper can be extended to include other parameters of users and items as music is a diverse thing to recommend.

6. REFERENCES

1. Jeyasree, S. and T.K. Thivakaran, 2014. Precise 11. Wang, F., S. Ma, L. Yang and T. Li, 2006. Recommendation System for the Long Tail Problem Recommendation on item graphs. In ICDM, Using Adaptive Clustering Technique, International pp: 1119-1123.
2. Song Jie Gong, 2009. A Collaborative Filtering Recommendation Algorithm Based on User 14. Truong K.Q., F. Ishikawa and S. Honiden, 2007. Clustering and Item Clustering, pp: 697-712.
3. C. Anderson. The Long Tail: Why the Futhure of Business is Selling Less of More. Hyperion, 2006
4. Hervas-Drane, A., 2007. Word of Mouth and Recommender Systems: A Theory of the Long Tail, working paper, Harvard Business School
5. P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In RecSys, pages 39–46, 2010.
6. M. R. McLaughlin and J. L. Herlocker. A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In SIGIR, pages 329–336, 2004.
7.] F. Fouss, A. Pirotte, J. Renders, and M. Saerens. A novel way of computing dissimilarities between nodes of a graph, with application to collaborative filtering and subspace projection of the graph nodes. In ECML, 2004.
8. Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In KDD, pages 426–434, 2008
9. Last. FM – A popular music web portal http://www.last.fm