International Journal of Engineering Research and Development
e-ISSN: 2278-067X, p-ISSN: 2278-800X, www.ijerd.com
Volume 9, Issue 3 (December 2013), PP. 52-60

Enhancing collaborative filtering in music recommender system by using context based approach

Prof.Reena Pagare[1], Intekhab Naser [2], Vinod Pingale [3], Nayankumar Wathap[4]
[1]*Professor, Department of Computer, MIT College of Engineering, Pune, India*
[2,3,4]*B.E Student, Department Of Computer, MIT College of Engineering, Pune, India*

Abstract:- Recommender Systems analyse some user and item interactions to help users by recommending the most relevant, feasible and appropriate items from a wide range and pool of items and resources. We propose to enhance the collaborative filtering methodology in recommender systems by considering the content as well as the context based approach towards recommendation. The rating of items and the contextual information of users are expected to enhance the relevance constraint by taking into account the user's mood and activity implicitly by using respective API's to capture and consider the contextual features of the user. The methodology and technique of reduction based approach and user based rating prediction will be used to accomplish the desired results for the proposed recommender system.

Keywords:- Recommender system, collaborative filtering, context.

I. INTRODUCTION

Recommender Systems make use of community opinions to help users identify useful items from a considerably large search space [8]. The technique used by many of these systems is collaborative filtering (CF) [9], which analyses past community opinions to find correlations of similar users and items to suggest personalized items to querying users. Since collaborative-filtering methods only require the information about user interactions and do not rely on the content information of items or user profiles, they have more broad applications [10], [11], [12], and more and more research studies on collaborative filtering have been reported [13], [14], [15]. These methods filter or evaluate items through the opinions of other users [16]. They are usually based on the assumption that the given user will prefer the items which other users with similar preferences liked in the past [17]. However, existing collaborative-filtering methods often directly exploit the information about the users' interaction with the systems. In other words, they make recommendations by learning a ―*user– item*‖ dualistic relationship [2]. These kinds neglect the user interests, their behaviour or the current mood and activity of the user which comes under the contextual information of user.

Mobile devices are becoming smarter and more popular including smart phones and tablets, and accordingly access to multimedia data in real time in various environments is also getting easier. By utilizing the appropriate and proper communication services, the users can acquaint themselves and acquire broadly distributed documents, music or videos they are concerned about. As per the use, feasibility and memory or capacity requirements, the popularity of music multimedia data is more than the other types. The feasibility of viewing documents or videos on mobile phones owing to the small size of their screens is not good, and the overhead of retrieving the huge data size of videos is also more. In sync with the advancement in compression techniques for music, the memory requirement and data storage space for music is significantly reduced, and the Circulation of music data is further more facilitated. This leads to ensure that users can obtain and avail their favourite music directly on Web and the mundane task of going to music stores can be eliminated. Accordingly, helping users find music they like in a large archive has become an attractive but challenging issue over the past few years. Moreover relying on the traditional recommender systems for music recommendation may not be feasible and may give less relevant and inappropriate results. In addition, user preferences may vary in accordance with various contexts including location, season, state of movement, and environmental condition. For example, someone jogging might prefer hip-hop to classical music. A survey showed that activity (a type of context information) significantly affects a listener's mood [18]. This finding delivers an important message that context information is an important element for a music recommender to consider in selecting music to suit the listener's mood [1]. Such context information can be provided by the new generation of smart phones by utilizing the respective Application Programming Interfaces (API's) provided by android operating system for the relevant context information. Considering these aspects, in our approach we propose a strategy of music recommendation by applying the conceptual aspects of reduction based approach [7] and user based rating prediction [7]. Through this approach we expect to get more refined and relevant results for music recommendation.

II. RELATED WORK

A. **Traditional Recommender Systems**

The traditional recommendation systems generally use collaborative filtering technique for recommending relevant items to users. In this methodology, a matrix is generated with the users and items as members where for each item there is a rating given by users on an appropriate scale according to the likes or dislikes of the user of that item. Then by consideration of the ratings given by the users, an average rating for an item is calculated by all the users who rated the item. Finally the top items with maximum average rating are recommended to the current active user. In this approach, the current active user's context and his particular likes are not considered in the recommendation process.

B. Ubiquitous Recommendation System

The ubiquitous recommender system is a system that also considers context information of users. In this system the musical acoustic features are extracted from the music database and the context of the user is determined. In offline pre-processing, two-stage clustering of music data is carried out and a pattern database is created. In online prediction, musical snippets are generated. Then these features are used to group and categorize musical genres into some classic rock, Latin, simple classical, vocal opera, rock and jazz. According to the current context of the user, the evaluation is done to determine the type of user and the kind of music to be recommended. Finally an appropriate recommendation list of songs is generated and provided to the user.

C. iExpand Method

In this method, user's latent interests are considered by implicitly evaluating the user's liked or rated items and observing the characteristics of those items. According to which, a consideration of user's latent interests are calculated and more relevant traces to the most appropriate recommendation of items are provided to the user.

D. Comparison with Former Recommender Approaches

In our proposed system, we enhance the approaches adopted by the traditional systems and also introducing certain modifications by collaborating the ubiquitous system approach and an i-expand similar method approach in our proposed system for music recommendation system.

In our approach, we use the current context of the active user and then compare this contextual information with a log of user context information to fetch similar users to the current user. Again, users similar to the active user are found by calculating the similarity constraint with the application of *Pearson Correlation* (PC) similarity [7] measure. The list of users fetched from both the context and behaviour are used to get relevant list of songs from these users, and then according to the ratings to the songs, the top rated songs from the refined list will be recommended to the current active user.

As of the approaches mentioned formerly, the traditional approach of collaborative filtering can be used after some modifications in the multi-dimensional matrix in the system. The characteristics of the ubiquitous recommendation system are profound in our system when the contextual information of the user comes under consideration. The extraction of behaviourally similar users to the active user by using the concept of normalized ratings and *Pearson Correlation* (PC) similarity factor is slightly similar to the previous method. That is, our approach is a refined and idiosyncratic system dealing with the recommendation of the most relevant songs by considering intuitively the user interests and mood by the method of capturing the user's context.

III. BLOCK DIAGRAM

The block diagram shown in Fig.1 is our proposed recommender system which illustrates the modules of our system in appropriate sections considering the relations between them and an abstraction of the work flow. Broadly categorizing the system, aspects of consideration includes the musical content feature extraction which pre dominantly is obtained from million song dataset along with last.fm data from the respective websites[4][5], the user context information extraction and the processing for recommendation. Rating matrix includes the popularity of any particular music and the context log includes the behaviourally similar user's context data to the active or current user. Explanation of all these modules can be briefly summarized as follows:

A. Rating Matrix:

Rating matrix is similar to the matrix in any collaborative filtering methodology, where we have ratings given by users for items (songs). The parameters in the matrix include songs and users, where the entries in the matrix are the rating given by the users for the songs. According to the ratings for the songs, the categorization is done.

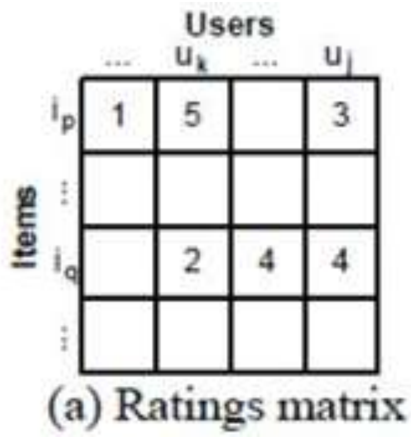

Fig.1 Rating matrix in which ratings are represented by the (user, item, rating triple**)** [3]

B. Context Log:

Context log consists of the log of history of users who previously listened to songs in a particular context [1]. Consideration of songs and the context in which those songs were listened by a respective user is stored as data. Moreover, whenever any user listens a song, his/her context along with the song and user -id are fetched and inserted into the context log.

Abbreviation	**Context dimension**	**Possible values**
LN	Location	Indoor, outdoor
M	Motion	Stop, slow, middle, fast
C	Calendar	Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec.
T	Time	Morning, afternoon, evening
LT	Light	Bright, moderate, dim, dark.
H	Humidity	High, moderate, low
AT	Air temperature	High, moderate, low

Table.1 Context information dimensions and value ranges.

Name	LN	M	C	T	LT	H	AT	Song1	Song2
Rohit	Indoor	Stop	Apr	Evening	Dim	Moderate	Low	4	2
Kamran	Outdoor	Fast	Sep	Morning	Bright	Low	Moderate	3	5
Jim	Outdoor	Middle	Mar	Evening	Dark	High	Low	0	1
Aditi	Indoor	Slow	Jun	Afternoon	Moderate	High	High	5	1

Name	LN	M	C	T	LT	H	AT	Song1	Song2
David	Outdoor	Fast	Nov	Evening	Dim	Moderate	Moderate	2	4

Table.2 Context log example

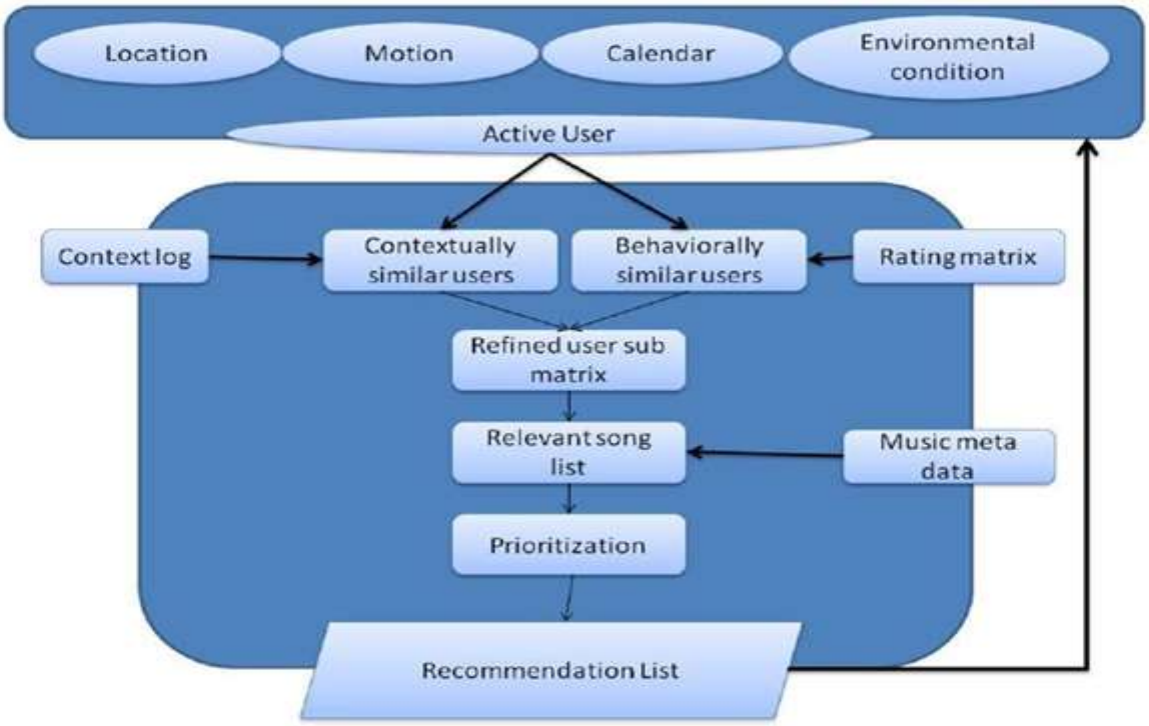

Fig 2. Block Diagram of work flow: The context information of user is evaluated for refinement in recommendation.

C. Active User Context Extraction: This module deals with the current context of an active user in which the user is in the present situation and what kind of song user may want to listen in that respective context. This information include the user's current location, the present form of motion or rest, calendar consideration which can influence the season and occasions in a particular region, the time of day and also the environmental conditions around the user including light, humidity and air temperature. This contextual information of the user can significantly affect the user's mood which enhances their preference and choice of song. This information about the user's context can be extracted by using relevant android API's (application programming interfaces) [6].

D. Contextually Similar Users: Users similar to the active user with respect to the current context of the active user from the context log are fetched and the consideration of further collaborations is carried out. This can be done by using Reduction-Based Approach [19]. The reduction-based approach reduces the problem of multidimensional recommendations to the traditional two-dimensional *User* × *Song* recommendation space.

To see how this reduction can be done, consider the basic two-dimensional rating estimation function that, given existing ratings *D* (i.e., *D* contains records *_user, content, rating* for each of the user specified ratings), can calculate a prediction for any rating,

$$R^{D}_{User\times Content} : U \times C \rightarrow rating$$

A three-dimensional rating prediction function supporting time can be defined similarly as

$$R^{D}_{User\times Content\times Time} : U \times C \times T \rightarrow rating$$

where *D* contains records as *user, content, time, rating* for the user-specified ratings. Again the three-dimensional prediction function can be reduced and expressed through a two-dimensional prediction function as follows:

$$R^{D[Time=t](User,Content,rating)}_{User\times Content}(u, c)$$

where *D*[*Time* = *t*](*User*, *Content*, *rating*) denotes a rating set obtained from *D* by selecting only those records where *Time* dimension has value *t* and keeping only the corresponding values for *User* and *Content* dimensions as well as the value of the rating itself. In other words, if we treat a set of three-dimensional ratings *D* as a relation, then *D* [*Time* = *t*](*User*, *Content*, *rating*) can be explained as simply another relation obtained from *D* by performing two relational operations: selection followed by projection.

The above three-dimensional reduction-based approach can be extended to a general method reducing an arbitrary *n*-dimensional recommendation space to an *m*-dimensional one (where $m < n$). However, in most of the applications we have $m = 2$ because traditional recommendation algorithms are designed for the two-dimensional case.

We will refer to these two dimensions on which the ratings are projected as the *main* dimensions. Usually these are *User* and *Song* dimensions. All the remaining dimensions, such as *Time*, will be called *contextual* dimensions since they identify the context in which recommendations are made (e.g., at a specific time). We reiterate that the segments define not arbitrary subsets of the overall set of ratings *D*, but rather subsets of ratings that are selected based on the values of attributes of the *contextual* dimensions or the combinations of these values. For example the *Evening* segment of *D* contains all the ratings of songs listened in evening: *Evening* = {*d* ∈ *D*|*d.Time.evening* = *yes*}. Similarly, *Outdoor-Evening* segment contains all the song ratings listened outdoor in the evenings: *Outdoor- Evening* = {*d* ∈ *D*| (*d.Location.place* = *outdoor*) ∧ (*d.Time.evening* = *yes*)}.

By applying the above strategy in our multi-dimensional search space, we can eventually reduce it to a two-dimensional matrix with the users, songs as items and the ratings given by to those songs by the respective users. Then we can apply the basic collaborative filtering approach on it to get the set of contextually similar users. The traditional CF approach computes the rating of item *i* by user *u* as

$$r_{u,i} = k \sum_{u' \in \hat{U}} sim(u, u') \times r_{u',i}.$$

Various approaches have been used to compute similarity measure *sim*(*u*, *u_*) between users in collaborative recommender systems. In most of these approaches, *sim*(*u*, *u_*) is based on the ratings of items that *both* users *u* and *u_* have rated. The two most popular approaches are the *correlation based* [19]

$$sim(x, y) = \frac{\sum_{s \in S_{xy}} (r_{x,s} - \bar{r}_x)(r_{y,s} - \bar{r}_y)}{\sqrt{\sum_{s \in S_{xy}} (r_{x,s} - \bar{r}_x)^2 \sum_{s \in S_{xy}} (r_{y,s} - \bar{r}_y)^2}},$$

and the *cosine-based* approach [19]

$$\cos(X, Y) = \frac{X \cdot Y}{\|X\|_2 \times \|Y\|_2} = \frac{\sum_{s \in S_{xy}} r_{x,s} r_{y,s}}{\sqrt{\sum_{s \in S_{xy}} r^2_{x,s}} \sqrt{\sum_{s \in S_{xy}} r^2_{y,s}}}$$

where *rx*,*s* and *ry*,*s* are the ratings of song *s* assigned by users *x* and *y* respectively,

Sx y = {*s* ∈ *Items* |*rx*,*s* _= *ε* ∧ *ry*,*s* _= *ε*} is the set of all items co-rated by both users *x* and *y*, and *X* ・ *Y* denotes the dot-product of the rating vectors *X* and *Y* of the respective users. From the above considerations, it is possible to reduce the multidimensional context information of user to a two dimensional matrix with only users, songs and the respective ratings. And we can extract the contextually similar users with maximum relevancy to the active user's context

E. Behaviourally Similar Users:

Users similar to the active user with respect to the likes or ratings gives by the current active user from the rating matrix are fetched and the consideration of further collaborations is carried out. This can be achieved by applying the *User-based Rating Prediction.[7]*

User-based neighbourhood recommendation methods predict the rating r_{ui} of a user *u* for a new song *i* using the ratings given to *i* by users most similar to *u*, called nearest-neighbours. Suppose we have for each user =*u* a value w_{uv} representing the preference similarity between *u* and *v*. The *k*-nearest-neighbours (*k*-NN) of *u*,

denoted by *N*(*u*), are the *k* users *v* with the highest similarity w_{uv} to *u*. However, only the users who have rated song *i* can be used in the prediction of r_{ui}, and we instead consider the *k* users most similar to *u* that *have rated i*. We write this set of neighbours as *Ni*(*u*).
The rating given by the neighbours to *i* can be estimated as the average rating r_{ui} as:

$$\hat{r}_{ui} = \frac{1}{|\mathcal{N}_i(u)|} \sum_{v \in \mathcal{N}_i(u)} r_{vi}.$$

To account for the fact that the neighbours can have different levels of similarity, we weigh the contribution of each neighbour by its similarity to *u*. However, if the sum of these weights is not equal 1, the predicted ratings may go out of the range of allowed values. Consequently, it is customary to normalize these weights, such that the predicted rating becomes

$$\hat{r}_{ui} = \frac{\sum\limits_{v \in \mathcal{N}_i(u)} w_{uv} r_{vi}}{\sum\limits_{v \in \mathcal{N}_i(u)} |w_{uv}|}.$$

In the denominator of, $|w_{uv}|$ is used instead of *wuv* because negative weights can produce ratings outside the allowed range. Also, w_{uv} can be replaced by $w\alpha_{uv}$, where $\alpha > 0$ is an amplification factor [20]. When $\alpha > 1$, as is it most often employed, an even greater importance is given to the neighbours that are the closest to *u*.

The similarity between two user *u* and user *v*, would then be computed as *Cosine Vector* (CV) (or *Vector Space*) similarity [7]:

$$CV(u,v) = \cos(\mathbf{x}_u, \mathbf{x}_v) = \frac{\sum\limits_{i \in \mathcal{I}_{uv}} r_{ui} r_{vi}}{\sqrt{\sum\limits_{i \in \mathcal{I}_u} r_{ui}^2 \sum\limits_{j \in \mathcal{I}_v} r_{vj}^2}},$$

where I_{uv} once more denotes the songs rated by both *u* and *v*. A problem with this measure is that is does not consider the differences in the mean and variance of the ratings made by users *u* and *v*

Another popular measure where the ratings are compared and the effects of mean and variance have been removed is the *Pearson Correlation* (PC) similarity:

$$\mathrm{PC}(u,v) = \frac{\sum\limits_{i \in \mathcal{I}_{uv}} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum\limits_{i \in \mathcal{I}_{uv}} (r_{ui} - \bar{r}_u)^2 \sum\limits_{i \in \mathcal{I}_{uv}} (r_{vi} - \bar{r}_v)^2}}.$$

Note that this is different from computing the CV similarity on the *Z*-score normalized ratings, since the standard deviation of the ratings is evaluated only on the common items I_{uv} to both the user u and user v, not on the entire set of songs rated by them, i.e. I_u and I_v.

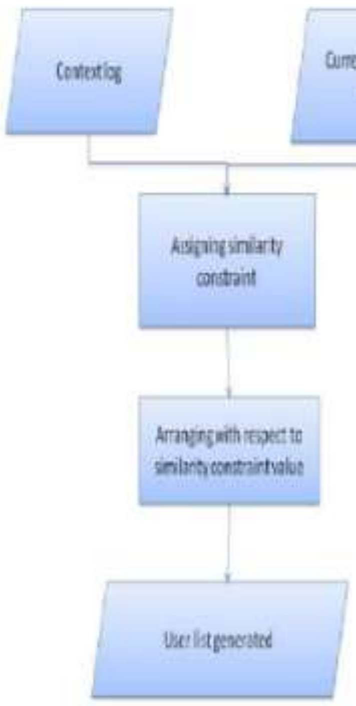
Context log
Current context of user
Assigning similarity constraint
Arranging with respect to similarity constraint value
User list generated

Fig.3 Contextually Similar Users

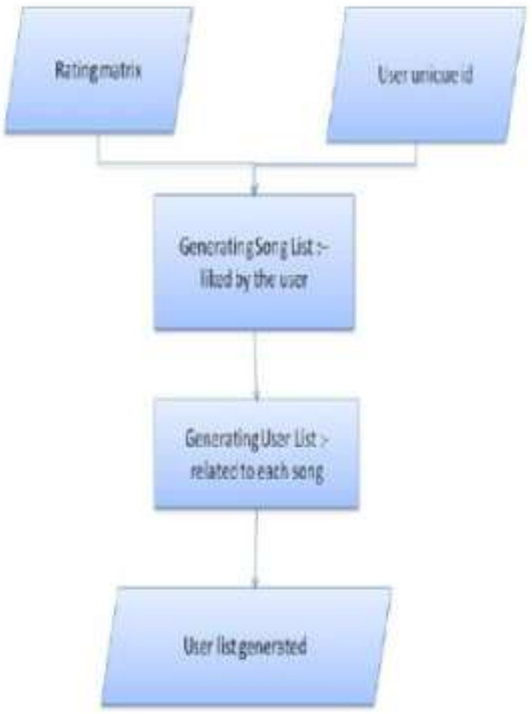
Rating matrix
User unique id
Generating Song List :- liked by the user
Generating User List :- related to each song
User list generated

Fig.4 Behaviourally Similar Users

A. Contextually Similar User:

The contextually similar users are fetched from the context log by the following sequence of steps:
1)The context log and the current context of the user are evaluated to check for similarity constraints.
2)The similarity constraints are assigned to users as a unit of degree of similarity with the current context.
3)According to the relevancy of higher degree of similarity of users with the current user, a list is generated and fetched of similar users.

B. Behaviourally Similar Users:

The behaviourally similar users are fetched from the rating matrix by the following sequence of steps:
1)The rating matrix will be checked for the unique user id of our current active user.
2)From the rating matrix, the songs rated or liked by the user will be fetched.
3)This will be done by applying one of the various techniques for similarity calculation.
4)Now the users who liked or rated these songs will be evaluated and a list of these behaviourally similar users will be generated and fetched.

IV. CONTROL FLOW THROUGH FLOWCHART

The control flow of these sub systems can be explained by referring to the flow chart above. As the user logs into the system, the inputs taken and fetched by the system concerning the user include the current context of the active user along with the unique user id of the user. Then the similar users pertaining to the user on account of the user's context as well as the behaviour of the user from previous experience or past history according to the ratings given by the user are evaluated to get a contingent of relevant similar users.

A. Refined User Sub-matrix:

Similar users with respect to the current context as well as with respect to the behaviour of the active user are generated in the previous module. Again for refinement, the intersection of the sets from the contextually similar user's module and the behaviourally similar user's module will be taken. These relevant and similar users along with the songs will give us a refined sub-matrix of users along with the ratings for required songs.

 $A \cap B=\{x: x \in A \wedge x \in B\}$

B. Relevant Song list:

The contingent of users subsequently generated in the previous module is evaluated to acquire and fetch the songs rated by these users. These highly refined users will eventually generate significantly relevant list of songs. The next module will refine this list further by application of some more appropriate constraint.

C. Music Feature Extraction:

The musical content feature extraction is pre dominantly done and is obtained from million song dataset along with last.fm data from the respective websites [4] [5]. These features help in identifying and releasing the required list of songs according to the requirements and respective constraints.

D. Prioritized Song List:

The users in the previous modules will be evaluated to generate a prioritized list of most relevant songs by assigning the rating points to the songs as a parameter for priority. The songs generated and fetched from these users are prioritized according to the rating, which means the popularity constraint of the songs is also being considered.

E. Recommendation:

The recommendation list of these highly refined, closely relevant and significantly appropriate songs is generated by tuning through all the above modules of the respective system. These songs are arranged and the top *n* entries from the list are finally provided as a recommendation to the active user.

V. CONCLUSION

The possibility of irrelevant recommendations due to only popularity consideration in the traditional approaches is very much reduced by the application of context based approach. We focused on the contextual similarity as well as the behavioural similarity of the user with other users to be able to extract implicitly relevant songs to the user for recommendation. For the contextual similarity consideration, we used Reduction Based Approach and for the behavioural similarity consideration, we used User Based Rating Prediction. With the combination of above approaches, we have increased the relevancy constraint of the system for recommendation.

VI. FUTURE WORK

The advancements in the application programming interfaces will help to enhance the extraction of relevant contextual information about the user. As more and more appropriate context data of the user will be available to the system the feasibility of more improvised and optimal list of songs for recommendation can be generated. Also connection to various social networks will boost the efficiency of our system concerning the consideration of the user's likes and dislikes.

Our system is not very concretely dependent on the data base. So with little modification in our system we can develop recommendation for a wide range of items and products. The portability, feasibility and the application of our system can be further enhanced by developing the system for various other operating systems.

VII. REFERENCES

[1] Ja-Hwung Su, Hsin-Ho Yeh, Philip S.Yu and Vincent S. Tseng, ―Music recommendation using content and context information mining‖, IEEE Intelligent Systems, IEEE Computer Society, 2010.

[2] Qi Liu, Enhong Chen, Hui Xiong,Chris H. Q. Ding, and Jian Chen, ―Enhancing Collaborative Filtering by User Interest Expansion via Personalized Ranking‖, IEEE Transactions on Systems, man, and cybernetics—PART B: CYBERNETICS, VOL. 42, NO. 1, FEBRUARY 2012.

[3] Mohamed Sarwt, Justin J. Levandoski, Ahmed Eldawy and Mohamed F. Mokbel, ―LARS*: An Efficient and Scalable Location-Aware Recommender System‖, TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 6, NO. 1, NOVEMBER 2012.

[4]Million song dataset: applications and methods http://labrosa.ee.columbia.edu/millionsong/ [5]Last.fm: Music recommendation service (2013). http://www.last.fm/

[6]Android.com Website: Application programming interfaces. http://developer.android.com/index.html

[7] Francesco Ricci, Lior Rokach, Bracha Shapira and Paul B. Kantor‖ Recommender Systems Handbook‖, Springer New York Dordrecht Heidelberg London, LLC 2011.

[8] G. Linden et al, ―Amazon.com Recommendations: Item-to-Item Collaborative Filtering,‖ *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, 2003.

[9] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, ―GroupLens: An Open Architecture for Collaborative Filtering of Netnews,‖ in *CSWC*, 1994

[10]F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, ―Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation,‖ IEEE Trans. Knowl. Data Eng., vol. 19, no. 3, pp. 355–369, Mar. 2007.

[11] Y. Ge, H. Xiong, A. Tuzhilin, K. Xiao, M. Gruteser, and M. J. Pazzani, ―An energy-efficient mobile recommender system,‖ in Proc. 16th ACM SIGKDD Int. Conf. KDD, 2010, pp. 899–908.
[12] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, ―Evaluating collaborative filtering recommender systems,‖ ACM Trans. Inf. Syst. (TOIS), vol. 22, no. 1, pp. 5–53, Jan. 2004.
[13] S. Funk, Netflix Update: Try This at Home, 2006. [Online]. Available: http://sifter.org/~simon/journal/20061211.html
[14] Y. Koren, ―Factorization meets the neighborhood: A multifaceted collaborative filtering model,‖ in Proc. 14th ACM SIGKDD Int. Conf. KDD, 2008, pp. 426–434.
[15] Y. Koren, ―Collaborative filtering with temporal dynamics,‖ in Proc. 15th ACM SIGKDD Int. Conf. KDD, 2009, pp. 447–456.
[16]J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, ―Collaborative filtering recommender systems,‖ in Proc. Adapt. Web, Lecture Notes in Computer Science, 2007, pp. 291–324.
[17]G. Adomavicius and A. Tuzhilin, ―Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions,‖ IEEE Trans. Knowl. Data Eng., vol. 17, no. 6, pp. 734–749, Jun. 2005.
[18]G. Reynolds e 1. t al., ―Interacting with Large Music Collections: Towards the Use of Environmental Metadata,‖ Proc. IEEE Int'l Conf. Multimedia and Expo, IEEE Press, 2008, pp. 989–992.
[19]Gediminas Adomavicius, Ramesh Sankaranarayanan, Shahana Sen And Alexander Tuzhilin, ―Incorporating Contextual Information in Recommender Systems Using a Multidimensional Approach‖, ACM Transactions on Information Systems, Vol. 23, No.1, January 2005.
[20] Breese, J.S., Heckerman, D., Kadie, C.: Empirical nalysis of predictive algorithms for collaborative filtering. In: Proc. of the 14th Annual Conf. on Uncertainty in Artificial Intelligence, pp. 43–52. Morgan Kaufmann (1998).