Collaborative-filtering and Content-based recommendation approaches with Context-awareness in Music Recommendation System

Project report submitted to
Indian Institute of Information Technology, Nagpur in partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology
In
Computer Science and Engineering

by

Piyush Ojha (BT17CSE084), Himanshu Gupta (BT17CSE093), Karan Kushwaha (BT17CSE097)

Under the guidance of

Dr. Milind Penurkar

Department of Computer Science and Engineering
Indian Institute of Information Technology,
***Nagpur 440006* (India)**

2017-2021

Collaborative-filtering and Content-based recommendation approaches with Context-awareness in Music Recommendation System

Project report submitted to
Indian Institute of Information Technology, Nagpur in partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology
In
Computer Science and Engineering

by

Piyush Ojha (BT17CSE084), Himanshu Gupta (BT17CSE093), Karan Kushwaha (BT17CSE097)

Under the guidance of

Dr. Milind Penurkar

Department of Computer Science and Engineering
Indian Institute of Information Technology,
***Nagpur 440006* (India)**

2017-2021

Department of Computer Science and Engineering
Indian Institute of Information Technology, Nagpur

Declaration

I, Piyush Ojha, Himanshu Gupta and Karan Kushwaha, hereby declare that our work titled "**Collaborative-filtering and Content-based recommendation approaches with Context-awareness in Music Recommendation System**" is carried out by us in the Department of Computer Science and Engineering at Indian Institute of Information Technology, Nagpur. The work is original and has not been submitted earlier whole or in part for the award of any degree/diploma at this or any other Institution / University.

Date: 30 November, 2020

Sr. No.	**Enrollment No.**	**Names**	**Signature**
1.	BT17CSE084	Piyush Ojha	
2.	BT17CSE093	Himanshu Gupta	
3.	BT17CSE097	Karan Kushwaha	

Declaration

We, **Piyush Ojha(BT17CSE084), Himanshu Gupta(BT17CSE093) and Karan Kushwaha(BT17CSE097)** understand that plagiarism is defined as any one or the combination of the following:

1. Uncredited verbatim copying of individual sentences, paragraphs or illustrations (such as graphs, diagrams, etc.) from any source, published or unpublished, including the internet.
2. Uncredited improper paraphrasing of pages or paragraphs (changing a few words or phrases, or rearranging the original sentence order).
3. Credited verbatim copying of a major portion of a paper (or thesis chapter) without clear delineation of who did or wrote what. (Source: IEEE, the institute, Dec.2004) I have made sure that all the ideas, expressions, graphs, diagrams, etc. that are not a result of my own work, are properly credited. Long phrases or sentences that had to be used verbatim from published literature have been clearly identified using quotation marks.

We affirm that no portion of our work can be considered plagiarism and We take full responsibility if such a complaint occurs. We understand fully well the guide of the thesis may not be in a position to check for the possibility of such incidences of plagiarism in this body of work.

Date: 30 November, 2020

Sr. No.	**Enrollment No.**	**Names**	**Signature**
1.	BT17CSE084	Piyush Ojha	
2.	BT17CSE093	Himanshu Gupta	
3.	BT17CSE097	Karan Kushwaha	

Acknowledgement

We are grateful to our institute for providing us this opportunity to write this dissertation. We would like to thank **Dr. Milind Penurkar**, for his supporting suggestions and supervising our project work. We thank him for believing in us and providing his invaluable advice and unconditional support which helped us in timely and successful completion of this project.

We are grateful to **Dr. O.G. Kakde**, Director, **Dr. Ashwin Kothari** (Dean Academics), Computer Science and Engineering Department, Indian Institute of Information Technology, Nagpur for giving this opportunity of research work.

Lastly, We would like to thank the people who gave their precious time for filling out the reader survey, whose data is used in this thesis.

Abstract

In recent years, the online digital music brands have simply gone far and beyond. And therefore, the need of managing and sorting through the gigantic amount of music data leads to the study and development of some automation system to help users experience the true beauty of music on their fingertips via various devices like mobile phones and iPods. Though music information retrieval (MIR) techniques are developed and evolved effectively in last ten years, the development of music recommender systems is still at an early stage.

Recommendations can be classified as personalized or non-personalized. In non-personalized type, selection of the items for a user is based on how often an item has been visited in the past by other users. However, in the personalized type, the basic objectiveis to provide the best items to the users based on their taste and preferences. Although, in many domains recommender systems gained significant improvements and provide better services for users, they still require further research to enhance the accuracy of recommendations in many aspects. In fact, the current developments in recommender systems are far from the ideal model of the recommender system. Two popular algorithms: content-based model (CBM) and collaborative filtering (CF) implement the aforementioned recommendation ideas respectively.

In this paper, we reviewed state of art in recommender systems algorithms and techniques that are researched upon already, which is necessary to identify the gaps and improvement areas. Along with that, we provide effective solutions to overcome the shortcomings of existing recommender systems using Context-awareness considering “time of play” with traditional recommendations techniques content-based and collaborative filtering.

Keywords: Collaborative-filtering, Content-based recommendation system, Context-awareness recommendation, Cold-start problem.

Table of Contents

1. Introduction

With the boom of World Wide Web (2.0) in the past decades, internet has become the major source of retrieving multimedia information such as video, books, and music etc. Information recommendation has become an important research area since the first papers on collaborative filtering published in the 1990s [1]. Extensive work has been done in both industry and academia on developing new approaches on recommendation systems over the last decades. Recently, the interests have been increased due to the abundance of practical applications such as recommendation system of books, CDs, and other products at Amazon.com, flipkart.com, and movies by Movie Lens, YTS and so on.

Music recommendation is also an area where this recommendation system is required. As the World Wide Web becomes the source and distribution channels of diverse digital music, a large amount of music is accessible to people. In this situation, music recommendation gets required for each person since it becomes a difficult and time-consuming job to search and change the music whenever he wants to. There is already a commercial product like iTunes by Apple Computer even though they have used simple rules described by the users [3].

Previously, H. Chen and A. Chen[2] presented the music recommendation system for website, and Kuo and Shan(2002)[3] proposed a personalized music filtering system considering user preference. These studies considered the user preference fixed in their recommendation models. However, a user's preference on music changes according to the context and mood of the user. It varies so dynamically that the recommendation system should consider this information.

Basically, all of these systems rely directly or indirectly on user's preferences and user history of the system.

In this paper, we tried to solve a famous problem of "Cold Start Problem" combining traditional approaches content-based and collaborative filtering with Context-awareness approach. Because all current recommendation systems fail in case of new user, new item/song or user's preferences changed. So, this problem can solve considering other way of recommendations like time of play, location, count of song views etc.

This paper is organized as follows:

1). Improved Previous/current techniques of music recommendation system.

2). Other ways of recommendations are explored.

3). Combined traditional model with Context-awareness approach to better recommendations in music system.

2. Literature Review

We have done a broad exploration on Google Scholar for music recommendation models related papers. We selected articles that provide the overview of CFM and CBM recommendation system.

We also emphasized upon considering researches taking into account intrinsic, extrinsic, and contextual aspects of the users. For instance, personality and emotional state of the listeners (intrinsic) [4] as well as their activity (extrinsic) [5] are known to influence musical tastes and needs. So are users' contextual factors including weather conditions, social surrounding, or places of interest [6]. Also, the composition and annotation of a music playlist or a listening session reveal information about which songs go well together or are suited for a certain occasion. Therefore, researchers and developers of MRS should consider incorporating techniques to predict users' mood and Time of play for the algorithm to predict the best recommendation at a point of time.

Yading Song, Simon Dixon, and Marcus Pearce [19] present a vast overview of MRS techniques. The two common approaches are Collaborative-filtering (CFM) and Content-based filtering (CBM). Xiaoyuan Su[7] provides an in-depth analysis on CFM. Also, correspondence should be addressed to Michael J. Pazzani and Daniel Billsus[8] for we drew resources for CBM in our review.

2.1. Collaborative-filtering:

Collaborative systems generate recommendations by comparing ratings of items between different users. Collaborative systems do not attempt to model the actual item being recommended; they only analyze users' responses to those items. Collaborative systems are built on the assumption that users who rate items similarly in the past will continue to rate them similarly in the future. Last.fm is an example of a prominent collaborative system.

A) Cold start problem- One of the major problems of recommender systems in general, and music recommender systems in particular is the cold start problem, i.e., when a new user registers to the system or a new item is added to the catalog and the system does not have sufficient data associated with these items/users. In such a case, the system cannot properly recommend existing items to a new user (new user problem) or recommend a new item to the existing users (new item problem. Another subproblem of cold start is the sparsity problem which refers to the fact that the number of given ratings is much lower than the number of possible ratings, which is particularly likely when the number of users and items is large. The inverse of the ratio between given and possible ratings is called sparsity. High sparsity translates into low rating coverage, since most users tend to rate only a tiny fraction of items. The effect is that recommendations often become unreliable.

2.2. Content-based approach:

Instead of comparing user ratings, content-based systems construct models of the items being recommended and compare them to the preference model of the current user. The user's preference model doesn't necessarily contain any information about specific songs that the user does or does not like. Instead, it records the user's response to each of the components in the item model. For instance, the item model may include the gender of the vocalist. If the user consistently has a positive response to songs with female vocals, the user preference model will likely indicate a positive bias towards female vocals.

B) Glass-ceiling effect- The major problem with content-based filtering is that it relies on the correctness of the item model. The system is thus limited by what it understands about the music. Regardless of how much user input is collected, the system is unable to overcome limitations in the item models.

Schedl (2018) describes this problem as the "glass-ceiling effect" of music recommendation [9]. For example, the item model may not take into account important differences between otherwise similar songs. The system may not know the difference between a hard rock song with melodic lyrics and a hard rock song with screamed lyrics. It might then repeatedly recommend songs with screamed lyrics to users who only like melodic hard rock.

2.3. Context-awareness:

Context-awareness is to use information about the circumstances that the application is running in, to provide relevant information and services to the user. The term ‘context-aware’ was introduced by Schilit and Theimer [10]. They defined ‘context’ through giving a number of examples of context-location, identities of nearby people and objects, and changes to those objects.

C) Problems or Gaps- There are many other music recommendation systems based on Context awareness, and they consider context factors such as emotional states [11,13,15,18], location [14,16], time [11,14,17], weather [11,17] and other activities [12]. But to the best of our knowledge, none of the current systems can recommend reliable music for daily routines such working, sleeping, meditation, running and studying.

So, User have to make manually playlists according to their activities and timing which time consuming and need user efforts.

But our recommendation system can detect user's activities in current time and recommend or play a suitable song automatically.

3. Work done:

3.1 Problem formulation:

Since for recommendation we need a big dataset. But for a new user we don't have a big dataset. So, problem arises when we have a small dataset of a user. Or we have less rating or no rating for a new music. This problem is called as "Cold Start Problem". There are already many researchers who is also working on this problem. But till now no reliable solution has provided. So, we try to solve this problem by combining three approach together to get better recommendation.

Let U be a new user and he opens our system, initially our system doesn't have any data of him. Then our system will apply context-awareness approach and extract his timing and other activities and based on those get similarity with user dataset and recommend a music.

So, briefly our model can be considered as a 3-layer recommendation algorithm where first layer or algorithm is content-based which classified the whole dataset according to user's rating, second is context-awareness which extract the timing of play of particular music by user. And then third algorithm i.e., collaborative-filtering find the similarity between both algorithm result and predict best music recommendations.

Working of all these algorithms as follow:

3.2 Algorithms:

3.2.1. Content-based:

1) Random Forest Technique:

It is a supervised learning and classification algorithm consisting of many decision trees.

In this approach, our Model first classifies all music songs according to the Genre (Album or Artist). If a music belongs to any Genre then in the dataset value of music for this Genre will be 1 else 0.

In this way our model classified the dataset according to corresponding Genre. Then it assigns to music a value based on rating provided by user. If the rating between 0 to 3 then it will assign it a 0 else if greater than 3, then value will be 1.

So, in this way our model can solve "**Cold start problem**" for new items.

2) Code:

~/Last year project/Music_RS_algo

```
# create classes
# 1-3 ratings -> disliked; 4-5 -> liked
ratings.loc[ratings['rating'] <= 3, "rating"] = 0
ratings.loc[ratings['rating'] > 3, "rating"] = 1


merged = pd.merge(ratings, songs, left_on='songId', right_on='songId', sort=True)
merged = merged[['userId', 'title', 'genres', 'rating']]

# separating genres column
merged = pd.concat([merged, merged['genres'].str.get_dummies(sep='|')], axis = 1)

# we still have genres and no genres listed: remove them
del merged['genres']
# del merged['(no genres listed)']

# keeping classes column in the last
cols = list(merged.columns.values)
cols.pop(cols.index('rating'))
merged = merged[cols+['rating']]

# Building model

# instantiate the matrices of features(independent and dependent variables)
X = merged.iloc[:, 2:61].values   # independent var: genres
Y = merged.iloc[:, 61].values     # dependent var: rating


# splitting train and test data
#from sklearn.cross_validation import train_test_split
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state = 0)
```

```
# Normalize the data to a common scale
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)

# Using a flexible and efficient classification model- Random Forest
# Build Binary classification model, since we have only 2 classes
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 500, criterion='entropy', random_state = 0

# fit the model to the training set
classifier.fit(X_train, Y_train)

# predict the test set results
Y_pred = classifier.predict(X_test)

# check accuracy of predictions
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test, Y_pred)    # allows the visualization of performance of the alg

totalSongIds = songs['songId'].unique()
```

```
def nonratedsongs(userId):

    ratedsongs = ratings['songId'].loc[ratings['userId'] == userId]
    non_ratedsongs = np.setdiff1d(totalSongIds, ratedsongs.values)
    non_ratedsongsDF = pd.DataFrame(non_ratedsongs, columns=['songId'])
    non_ratedsongsDF['userId'] = userId
    non_ratedsongsDF['prediction'] = 0
    active_user_nonratedsongs = non_ratedsongsDF.merge(songs, left_on='songId', right_on='songId', sort=True)
    active_user_nonratedsongs = pd.concat([active_user_nonratedsongs, active_user_nonratedsongs['genres'].str.get_dummies(sep = '|')], axis =

    del active_user_nonratedsongs['genres']
    # del active_user_nonratedsongs['(no genres listed)']
    del active_user_nonratedsongs['title']

    return (active_user_nonratedsongs)

active_user_nonratedsongsDF = nonratedsongs(15)

df = active_user_nonratedsongsDF.iloc[:, 5:].values
Y_pred2 = classifier.predict(df)

active_user_nonratedsongsDF['prediction'] = Y_pred2
recommend = active_user_nonratedsongsDF[['songId', 'prediction']]
recommend = recommend.loc[recommend['prediction'] == 1]
# recommend = recommendation.loc[recommendation['prediction'] == 1]

recommend.to_csv('../../../Dataset/Created/recommend.csv', sep = ',', index = False)

# now we can sort this data in any way to show the results to the user.
```

3.2.2. Context-awareness approach:

In this approach, Our model extract user login time or time of play of each music song. Then add these details in the old dataset by inserting new column of time.

Now, the combine dataset work as base dataset for user also.

Let a new user U enter on the system and the system doesn't have any data of user. So, the system will extract it time and other activities like it searches, location etc.

Based on these values, collect similar datasets of other users and provides recommendations.

So, in this way it solves "**Cold start problem**" for new user.

Code work flow:

- Extracted timestamp data for each user from the ratings.csv.
  - Merged the data from songs.csv and ratings.csv and saved the required columns from the merged data frame.

```
ts = ratings['timestamp']
ts = pd.to_datetime(ts, unit='s').dt.hour
```

songs['hours'] = ts

merged = ratings.merge(songs, left_on='songId', right_on='songId', suffixes=['_users', ''])

merged = merged[['userId', 'songId', 'genres', hours']]

- Separating all the genres in the Genres column to be able to process and visualize it numerically.

merged = pd.concat([merged, merged['genres'].str.get_dummies(sep='|')], axis = 1) del merged['genres']

- Building the context profile of the user.

  def activeuserprofile(userId):

  # extract user rating info

  userprofile = merged.loc[merged['userId'] == userId]

  del userprofile['userId']

  #get the preference of the user for hours of the day

  Userprofile = userprofile.groupby(['hours'], as_index = False, sort =True).sum()

  #normalizing the userprofile **dataframeuserprofile.iloc[:, 2:] = userprofile.iloc[:, 2:].apply(lambda x:(x -np.min(x))/(np.max(x) - np.min(x)), axis = 1)**

  return (userprofile)

- Evaluating user preference by merging the results of CB model over our dataset and merged dataframe evaluated above

user_pref = recommend.merge(merged, left_on='songId', right_on='songId', suffixes=['_users', ''])

- Generating the song recommendation at 18thhour of the day by performing element-wise multiplication of user Content Recommendation and the Context profile of the user generated above [activeuserprofile(userId)]

product = np.dot(user_pref.iloc[:, 4:].to_numpy(), activeuser_.iloc[2:, 1].to_numpy())

using numpy function stack () to get the results as columns

preferences= np.stack((user_pref['songId'], product), axis = 1)

- Sorting and securing the top ten recommendations

df = pd.DataFrame(preferences, columns = ['songId', 'preferences'])

res = df.drop_duplicates()res = res[res['preferences'] == 1]

result = (res.sort_values(['preferences'], ascending = False)).iloc[0:10, 0]

result - Series

Index	songId
218	6
374	8
398	10
658	16
1188	27
2152	55
2631	95

Dataset table after applying context-awareness approach

3.2.3. Collaborative-filtering:

In this approach our model first find the songs the current user has liked, then find other users that have liked similar songs, and then recommend other songs that those users have liked.

Steps to implement collaborative filtering-

a. Preprocessing data
b. Learning the latent factors
c. Calculate predictions
d. Optimization

1. **Preprocessing data**: dataset those are produced by content-based and context-awareness approach are processed together based on similarities.

2. **Learning the latent factors-** First Step of this approach is to randomly initialize some parameters. These parameters will be a set of latent factors for each user and song. We will have to decide how many to use (5)

Second step of this approach is to calculate our predictions. We can do this by simply taking the dot product of each song with each user. If, for instance, the first latent user factor represents how much the user likes rock songs and the first latent song factor represents if the song is of rock type or not, the product of those will be particularly high if either the user likes rock songs and the song is of rock type or the user doesn't like rock songs and the song is not of rock type. On the other hand, if we have a mismatch (a user loves rock songs but the song is not of rock type, or the user doesn't like rock songs and it is one), the product will be very low.

Third step is to calculate our loss. We used mean squared error to calculate loss

Step four of this approach is to optimize our parameters (that is, the latent factors), using stochastic gradient descent, such as to minimize the loss. After each epoch, the stochastic gradient descent optimizer will calculate the match between each song and each user using the dot product, and will compare it to the actual rating that each user gave to each song.It will then calculate the derivative of this value and will step the weights by multiplying this by the learning rate. After doing this lots of times, the loss will get better and better, and the recommendations will also get better and better.

At the beginning, latent factors are just random numbers that don't mean anything but by the end of training, they start to represent hidden relations in the dataset.

Next step of this approach is to use l2 regularization since our dataset is very small and we don't want to do overfitting.

Accuracy achieved =90%

ADVANTAGE:

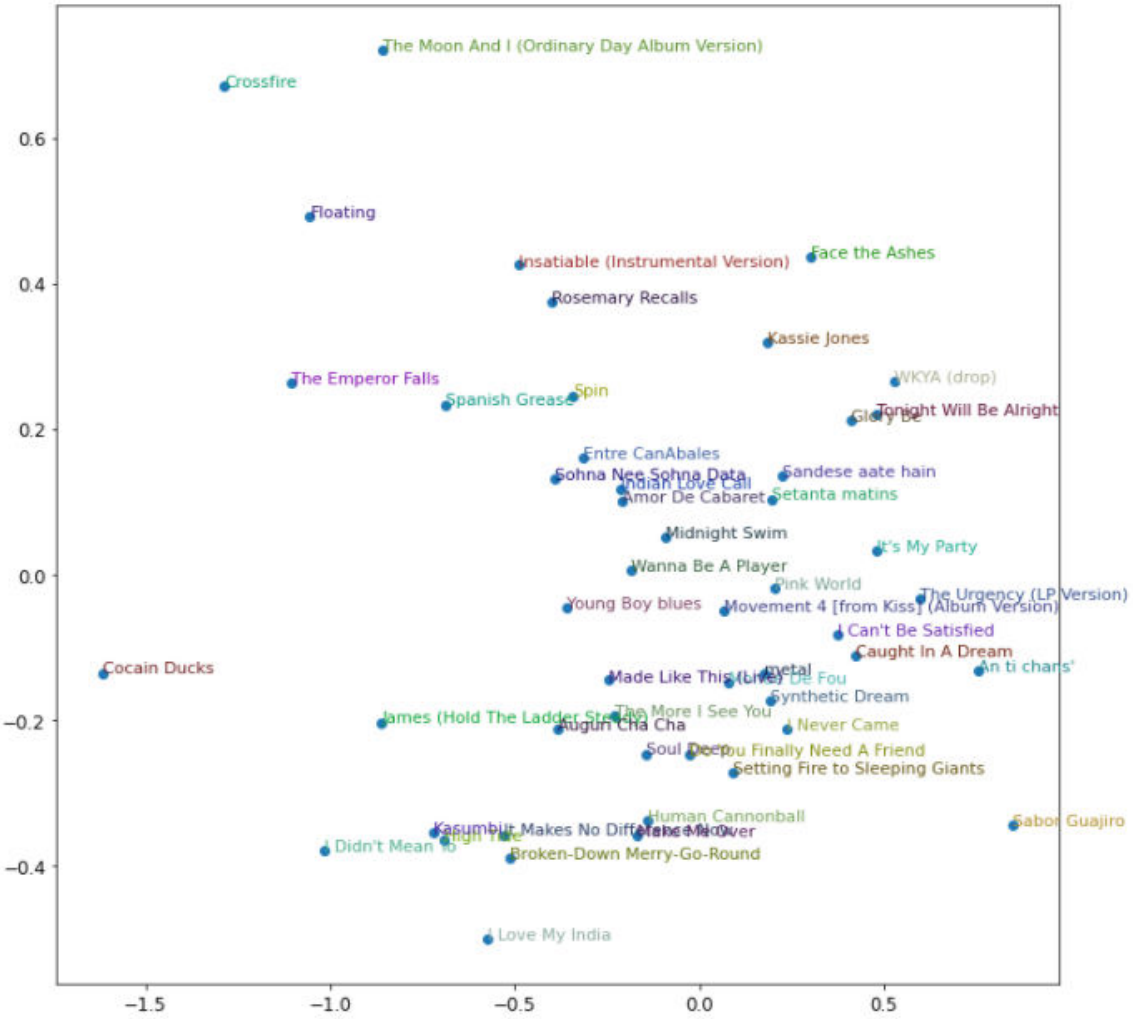
Biggest advantage of using this approach can be applied in practice even when we don't have any user data the model will improve as the user gives input.

By learning on existing data about the relations between users and songs, without having any other information, we saw that the model still gets some important features, and can isolate rock songs from pop, gospel, songs from romance, and so on.

epoch	train_loss	valid_loss	time
0	1.686957	1.558523	00:00
1	1.393135	1.047392	00:00
2	0.980252	0.919160	00:00
3	0.703032	0.911729	00:00
4	0.553132	0.909909	00:00

So, using these step wise approaches, “Cold start Problem” has solved.

Diagram showing songs which are most similar to each other:-



4. Results and Discussions

We evaluated our project on some datasets, and we got the 90% accuracy using combine model of content-based, collaborative-filtering and context-awareness. Considering context-factor as time, our model is able to solve “Cold start problem”.

5. Summary and conclusions

In this paper we have described a novel model for music recommendation that combines Context-awareness and collaborative-filtering with music content analysis, with support for a rich set of activities and music content features like rating and genre.

From this paper, It is clear that there are more future possibilities to improve this music recommendation model using emotional states, location, weather and many other factors as context factor.

References:

[1] 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. On Knowl. and Data Eng., vol. 17, pp. 734–749, June 2005.

[2] A. C. North, D. J. Hargreaves, and J. J. Hargreaves, “Uses of Music in Everyday Life,” Music Perception: An Interdisciplinary Journal, vol. 22, no. 1, 2004.

[3] D. J. Levitin and J. McGill, “Life Soundtracks: The uses of music in everyday life.” 2007.

[4] Ferwerda B, Schedl M, Tkalˇciˇc M (2015) Personality & emotional states: understanding users music listening needs. In: Extended proceedings of the 23rd international conference on user modeling, adaptation and personalization (UMAP), Dublin, Ireland

[5] T. S. Saponas, J. Lester, J. Froehlich, J. Fogarty, and J. Landay, “iLearn on the iPhone: Real-Time Human Activity Classification on Commodity Mobile Phones,” in UW CSE Tech Report, 2008.

[6] T. Brezmes, J.-L. Gorricho, and J. Cotrina, “Activity Recognition from Accelerometer Data on a Mobile Phone,” in IWANN, 2009.

[7] M. Berchtold, M. Budde, D. Gordon, H. R. Schmidtke, and M. Beigl, “ActiServ: Activity Recognition Service for mobile phones,” in ISWC, pp. 1–8, Oct. 2010.

[8] M. Khan, S. I. Ahamed, M. Rahman, and R. O. Smith, “A Feature Extraction Method for Real time Human Activity Recognition on Cell Phones,” in isQoLT, 2011.

[9] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, “Activity recognition using cell phone accelerometers,” SIGKDD Explor. Newsl., vol. 12, pp. 74–82, Mar. 2011.

[10] Y. S. Lee and S. B. Cho, “Activity recognition using hierarchical hidden markov models on a smartphone with 3D accelerometer,” in HAIS, pp. 460–467, 2011.

[11] H.-S. Park, J.-O. Yoo, and S.-B. Cho, “A Context-Aware Music Recommendation System Using Fuzzy Bayesian Networks with Utility Theory,” in FSKD, 2006.

[12] S. Dornbush, A. Joshi, Z. Segall, and T. Oates, “A human Activity Aware Learning Mobile Music Player,” in Proc. of the 2007 conference on Advances in Ambient Intelligence, 2007.

[13] S. Cunningham, S. Caulder, and V. Grout, “Saturday Night or Fever? Context-Aware Music Playlists,” in AM ’08, 2008

[14] A. Lehtiniemi, “Evaluating SuperMusic: streaming context-aware mobile music service,” in ACE, 2008.

[15] S. Rho, B. J. Han, and E. Hwang, “SVR-based music mood classification and context-based music recommendation,” in ACM MM, 2009.

[16] A. Camurri, G. Volpe, H. Vinet, R. Bresin, M. Fabiani, G. Dubus, E. Maestre, J. Llop, J. Kleimola, S. Oksanen, V. Välimäki, and J. Seppanen, “User-Centric Context-Aware Mobile Applications for Embodied Music Listening User Centric Media,” in LNICST, pp. 21–30, 2010.

[17] J.-H. Su, H.-H. Yeh, P. S. Yu, and V. S. Tseng, “Music Recommendation Using Content and Context Information Mining,” Intelligent Systems, IEEE, vol. 25, pp. 16–26, Jan. 2010.

[18] G. Reynolds, D. Barry, T. Burke, and E. Coyle, “Interacting with large music collections: Towards the use of environmental metadata,” in ICME, June 2008.

[19]