AMADEUS - A Contextual Content-based Music Recommender System

Gerardo Vitagliano, Solange Nunes
Universitá degli Studi di Napoli Federico II, Universidade de Coimbra
ger.vitagliano@studenti.unina.it, uc2011160060@student.uc.pt

Abstract

In this paper, an experimental system for music recommendation is proposed. The recommendations are computed based on clustering of high-level content features. The similarity measure is computed using a simple Bayesian network with decision and utility nodes. A contextual pre-filtering and a contextual post-weighting are also considered for improved performances. Results provide a general insight into the quality of the recommendation system, and on the influence of the contextual information both before and after a similarity score is computed. Pre-filtering is shown to be effective in reducing computational performance and provide useful recommendation diversity. The use of post-weighting alone marks no great difference in the outcome of the recommendation, while using both at the same time resulted in a sensible loss of effective recommendations.

1 Introduction

Ever since the very beginning of human civilization, consumption of music has always been a part of common daily life. Thanks to an exponential technological progress, the last decade saw a radical change both in the quantity and the quality of the music available to a generic user. The huge amount of music a single person can choose to listen with little to no effort gave birth to a new class of systems designed for the purpose of recommending music that can fit one's personal tastes or desires. Some of them have found commercial success, such as Pandora (music genome project)[1] and Spotify's Discover Weekly feature [2] which aim to suggest new, potentially unknown songs that can be liked by the user.

The task of music recommendation has been approached with two main strategies: performing a content-based analysis or using collaborative filtering techniques. The former computes item similarity and then, based on the user's previously listened/liked items, performs its prediction. The latter is also defined a content-agnostic algorithm since rather than extracting the features of each item is based on predicting user behavior: if a user X has liked items A, B and C, and a user Y has liked items A and B, it predicts Y to like C as well. Some research has been also conducted on a possible hybrid approach [Burke, 2002], although the most successful recommendation systems use a collaborative filtering algorithm [Song *et al.*, 2012]. A useful addition to recommendation algorithms can be contextual information: in fact, the importance of context while evaluating music quality was shown by [Lesaffre *et al.*, 2006]. Recommender systems which include contextual features are also called Contextual-Aware recommendation systems (CARS), where the context can be either used to extend usual algorithms by filtering/weighting data or be a core part in the algorithm itself, for example in the data model. An analysis of Context-Aware Recommender Systems performance with respect to pre- and post-filtering models can be found in [Panniello *et al.*, 2009]. This article has the following structure: the rest of this section will briefly review literature's approach to content-based recommendation, as well as introduce time contextual filtering techniques. Section II and III will introduce the architecture and the actual implementation of "Amadeus", our contextual content-based recommendation system. In Section IV the obtained results for three "listening cases" will be presented and a final conclusion is discussed in Section V.

1.1 Content-based systems

Content-based systems can be based on various sources of information related to the item: regarding specifically music recommendation, mainly two approaches have been extensively focused on in literature.

Given the nature of music, the most low-level descriptors for a song can be considered spectral based features: for example, the system proposed in [Çataltepe and Altinel, 2007] makes frequency features extraction and clustering to perform recommendation. A different method aims at labeling the music either with emotion tags or with genre information. For example commercial applications such as Last.fm[3] or Pandora[4] have followed such a way. An example in literature of such a system can be found in [Trohidis *et al.*, 2008]. Amadeus has an approach that mediates these two, by making use of features extracted from the content of the song itself, but on a

[1] https://www.pandora.com/about/mgp
[2] https://www.spotify.com/
[3] www.last.fm
[4] www.pandora.com

higher level than frequency descriptors.

1.2 Contextual filtering

Taking into account contextual information while developing a recommender system has been proved to be a useful addition. Amongst the various contexts that can be considered for music recommendation, time stands out as one of the most easily retrieved yet still with a decisive influence on user preferences, as it is shown in [Baltrunas and Amatriain, 2009]. As such, Amadeus implements time information in the decision process in order to adjust its recommendations. Regarding possible approaches to include such information, following the review of contextual filtering approaches done in [Panniello *et al.*, 2009], the context can intervene in a content-based recommendation in three different moments: before the recommendation is computed (pre-filtering), as a part in the actual data model (contextual modeling) or to adjust the score obtained after a first recommendation is performed (post filtering or weighting). Amadeus' algorithm can implement both a pre-filtering and a post-weighting over the computed similarity score. The result section will highlight whether using or not both these functionality does grant better performance.

2 Architecture

The first part of the section introduces the core concepts of Amadeus's recommendation algorithm, while specific details are left to following subsections. Amadeus is a content-based recommendation system: the underlying mechanism to compute song suggestion makes use of feature extraction, clustering, and time-contextual information.

2.1 Creation phase

At first, to be performed once, there is a "creation phase": during this phase, Amadeus's knowledge base is created. The starting point is an initial set of song of which Amadeus has knowledge of (from here onwards, "Dataset songs"). From these songs, high-level features are extracted and a clustering is performed, thus cluster centroids are saved.

2.2 Recommendation phase

The heart of the system is the "recommendation phase", which takes as input a set of songs representing the starting point for a recommendation. These songs, from now on referred to as "listened songs" (even though the choice process can be more than a simple listening), represent the user's preferences and their features are extracted. After averaging over the features of all the listened songs, the system computes the probability of a possible user recommendation to lie in each of the clusters, based on the inverse distance of the average feature vector with respect the cluster centroids.
These probabilities are stored in a node of a simple Bayesian network, modeled in figure 1. This is composed of a decision node representing each dataset song, the aforementioned probability node storing the cluster prediction scores for the listened songs, and a utility node that combines these two to have a similarity score. The utility values corresponding to each cluster given a dataset song inside of the utility

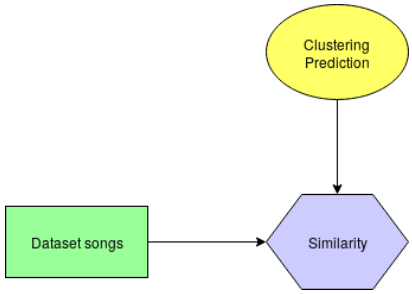

Figure 1: Amadeus Bayesian model

node are obtained with the same function used for the listened songs.The similarity for each dataset song is computed, and the 10 highest scoring songs are presented to the user as possible matches for his musical taste.
To obtain better results and performances, Amadeus makes use of contextual information regarding time, such as the hour of the day, weekday or season of the year. The contextual information steps in the recommendation phase in two different moments: before the similarity score is computed (pre-filtering) or after (post-weighting). The pre-filtering has the aim of reducing computational effort by filtering out songs for which a given feature (different for each contextual dimension) doesn't pass a certain threshold.The post-weighting readjusts the similarity scores to increase contextual inference in the recommendation process.
Regarding usefulness on the recommendation process of the contextual filtering, the result section will compare four different cases: the case in which no filtering is performed, the case in which both are used, and the cases in which respectively only one of the two is taken into account. Figure 2 holds a complete resume of the recommendation phase. The next subsections explore the details of the steps taken in the described phases.

2.3 Feature Extraction

Although the general algorithm used by Amadeus remains unchanged given any set of features extracted, the system makes use of high-level features: the ones used are the same that can be extracted from the Million Song Dataset (note) [5] (more details in the implementation section) information fields. Specifically, these are the following[6]:

1. Key: The key of the song
2. Loudness: The overall loudness of the song, measured in dB
3. Mode: The melodic mode of the song, e.g. major/minor
4. Tempo: The beats per minute of the song
5. Time signature: The number of beats per bar (4/4, 3/4 and so on)
6. Year: The year the song was composed (0 for missing information).

[5] https://labrosa.ee.columbia.edu/millionsong/
[6] https://labrosa.ee.columbia.edu/millionsong/pages/field-list

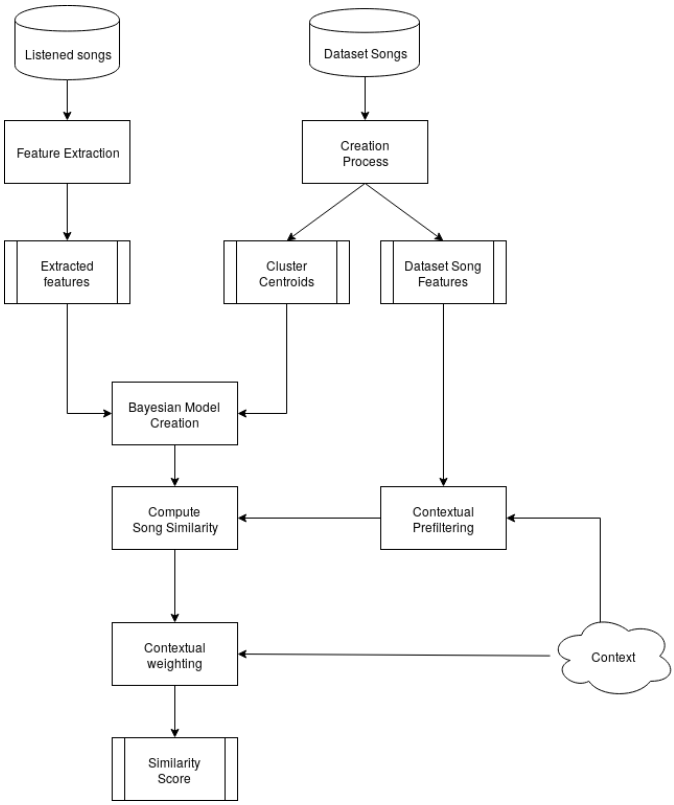

Figure 2: Amadeus Recommendation Phase

7. Energy: perceived energy of the music (modeled as a floating point number)
8. Danceability: An algorithmic estimation of such a quality (modeled as a floating point number, 0 for missing information).

Additionally, the key and the mode are scaled by their confidence level.

2.4 Clustering

In order to compute the best number of clusters a series of tests has been performed using the Silhouette metric [Rousseeuw, 1987]. This metric is used to compute, for each item, how much is he close to the centroid of the cluster it has been assigned to. The global result is obtained as the average silhouette of every item, and it's a real number in the range $[-1, 1]$, with 1 meaning well-formed compact clusters.
The two different clustering techniques taken into account were radial clustering and k-means with a chosen k ranging from 3 to 9 and 10: the results of the test can be seen in the following table:

As it can be noted from the results, the best silhouette score is obtained with a Radial clustering, which automatically computes the number of clusters to three. Also, k-means clustering with $k = 3$ has a similar performance, with three being by far the best number of clusters to have well dense clusters. However, such a small number of clusters can have a significant impact on the system performances, for example without allowing necessary diversity in the recommendation phase. Given these considerations, the number of clusters chosen to perform the algorithm is 8, which seems to be adequate enough to keep a good silhouette score and allow more sparsity of the recommendation space.

	# of clusters	Silhouette
Radial clustering	3	0.749
K-Means-3	3	0.736
K-Means-4	4	0.495
K-Means-5	5	0.451
K-Means-6	6	0.459
K-Means-7	7	0.418
K-Means-8	8	0.428
K-Means-9	9	0.391

Table 1: Different clustering silhouette scores

2.5 Contextual information

The time contextual information taken into account for the system has three domains: time of the day, weekday and season of the year.

- Time=$\{Morning, Afternoon, Evening, Night\}$
- Weekday=$\{Working, Weekend, Holiday\}$
- Season=$\{Winter, Spring, Summer, Autumn\}$

For the time of the day variable, the hours of a day have been equally divided into four six-hours periods: morning refers from the hours 06-12; afternoon refers to the hours 12-18; evening refers to the hours 18-24 and night to 00-06. For the Weekday variable, a working day is considered every day from Monday to Thursday, not including Portuguese national holidays; a weekend day is considered every Friday and Saturday, not including holidays, and holidays are all the remaining days of a year, which is to say Sundays and national holidays. Lastly, the Season of the year is computed accordingly to Northern Emisphere solar equinoxes and solstices.

Pre-filtering

Each contextual variable maps to a feature in the feature vector: time of the day relates to tempo, getting lower as it gets night; weekday relates to loudness, low for working days, medium for holidays and high for weekend days; season relates to mode, brighter for spring/summer and more sad for autumn/winter. Given these relationships, the values of each dataset song feature are normalized in an interval $[0, 1]$ in order to set relative thresholds to perform the filtering. The thresholds over which an item is filtered out chosen for the prefiltering are resumed in this table:

With these parameters, the resulting dataset after filtering contains roughly 65% of the original songs.

Post-weighting

The weights by which scale song similarity values are computed based on the same criteria seen for the prefiltering. After normalization of the features, three multipliers $\alpha_1, \alpha_2, \alpha_3$ are obtained by multiplied the relevant feature for the time variable by a Gaussian function with $\sigma = 1$ and mean centered on a specific value for each context, following the same

	Morning	Afternoon	Evening	Night
Tempo	<0.3	<0.4	>0.6	>0.7
	Working day	Weekend	Holiday	
Loudness	>0.6	>0.3	<0.5	
	Winter	Spring	Summer	Autumn
Mode	>0.6	<0.3	<0.4	>0.75

Table 2: Tresholds for contextual prefiltering: time/tempo, weekday/loudness, season/mode

	Morning	Afternoon	Evening	Night
Mean tempo	0.7	0.6	0.4	0.3
	Working day	Weekend	Holiday	
Mean loudness	0.5	0.8	0.65	
	Winter	Spring	Summer	Autumn
Mean mode	0.3	0.6	0.7	0.2

Table 3: Gaussian means used to scale utilities in post-weighting

logic used for the pre-filtering. The following table shows the center values for each context/feature.

After the three multipliers have been computed, a final value α is obtained averaging the three and used to finally scale similarity scores of dataset songs.

3 Implementation

A python implementation for the agent has been developed, and its source code is publicly available[7]. This brief section will describe the main implementation details. The datasets used to develop and test the implemented code are the "cal500" dataset and the "Million Song Subset"[8], containing respectively 500 and 10000 songs. These consist of a set of HDF5 files (.h5) containing relevant metadata extracted from songs. The fields included are compatible with the Million Song Dataset format, of which these two dataset can be considered a small subset. The creation phase of Amadeus takes as input a .txt list of the h5 files containing the necessary fields and outputs (mainly) two python *.pckl files: one which contains the relevant extracted features and one containing the cluster centroid for the chosen clustering algorithm. The current implementation of the clustering phase outputs eight different clustering centroids files, each one corresponding to one of the different clustering algorithms whose performance were analyzed in section II. Additional files include a list of the names of the song used to print out information to the final user. Once these files are created, the recommendation phase will make use of them as a knowledge base, and the original .h5 files are no longer needed.

[7]https://github.com/Mozzers/amadeus

[8]https://labrosa.ee.columbia.edu/millionsong/pages/additional-datasets

3.1 Agent

The agent itself has been deployed as a python library file that includes all the necessary functions in order to perform the recommendation as it has been described. It includes a feature extraction module, used in the feature extracting phase, to get the information regarding the fields of the h5 data. It implements the various clustering algorithms using the scikit-learn functions and contains the necessary functions used to perform data manipulation. The function $dist2prob()$, for example, is used to calculate, given a feature vector, its probability to be part of the various clusters. This is done by first computing the Euclidean distance from each centroid, and later normalizing these values such as the total sum can be equal to 1. The function $norm()$, instead, is used to perform the normalization needed in the contextual pre/post processing: it takes as input a set of features and it scales them such as the greatest value is set to 1 and the rest accordingly. This is useful to set relative thresholds and means of the Gaussian functions used in the data processing. Two different functions have been implemented to get the value of the context: $getContext()$ and $askContext()$. While the former actually gets from the running system the real-time context, the latter is used for experimentation purposes and asks for command-line user input values.

Bayesian Model - Similarity Function

The simple Bayesian network has been implemented using the library pyBN in the function $createModel()$. Although it was not strictly necessary, given its simplicity, it was chosen to do so in order to leave possible space for a future implementation of a contextual modeling, perhaps adding nodes which can map time context probabilities to clusters. Given this, and also to avoid having nodes with N variables for N songs in the dataset (with obvious scalability issues) the similarity computation has been implemented as a function, $computeUtility()$, which is used on each song for which a recommendation utility score is desired. This implementation, coupled with the contextual pre-filtering, saves memory and computation performances reducing the actual amount of comparison needed.

4 Results

The task of music recommendation is by no means easy to evaluate: musical tastes, and thus the performance of the system lies on a subjective scale. Given these premises, a quantitative, exact evaluation is not to be expected. An example set of experimentations performed will be reported and qualitative considerations will be drawn, analyzing the results. A common human classification for music is the "genre" label, assigned with respect to more or less commonly known high-level parameters such as rhythm, the instruments used, harmonies etc. Some attempts to automatically classificate music into genres have been done, for examples in [George *et al.*, 2001]. Keeping such knowledge into consideration a plausible approach for a qualitative measurement of Amadeus

performance is the following: definition of a set of songs commonly belonging to the same genre and comparison of the results with the expected genre recommended.

4.1 Experimental settings

Three "playlists" of fifteen songs each have been composed, to be found in table 4 at the end of the document, composed of songs from three different genres: rock music, orchestral music, and jazz music. To compare how differently context influences the final recommendation outcome, the same context has been used for all three playlists, specifically a time setting of $Evening, Holiday, Winter$. As per the design of the system, in this setting generally slow, medium loud, minor mode songs should be preferred. For each playlist first suggestions without contextual filtering are computed (From now on referred to "simple" suggestions), then applying only prefiltering and only post-weighting, and finally using the contextual information both before and after the similarity count. This allows two different analysis dimensions: $intra - playlist$ and $inter - playlist$. The first one will be focused on how the contextual information influences the obtained results on a single playlist, while the second on how the system performs given different musical genres. For brevity, the analysis will take into account only intra-playlist results from a single playlist and inter-playlist results from the simple case. The full set of results is, however, can be found in the appendix.

4.2 Inter-playlist analysis

As can be noted from the tables reported in the appendix, the results of the jazz recommendation are highly on point: in fact 10 out of 10 songs that Amadeus recommended are undoubtedly blues/swing songs that, although not in line with some of the more acoustic "listened" songs, can be definitely considered belonging to the same category of music. Some problems arise when evaluating the rock playlist recommendation, in fact, the exact same songs as the jazz case are recommended. These are completely out of genre with respect to the listened samples: probably the average feature vector extracted from the rock song is really close to the one of the jazz songs. The orchestra songs recommendations stand somehow midway between these two cases: half of them can be considered classical/chamber music, while the rest fall in a modern ambient music which is not related to the listener's preferences. It is to be noted, though that there is no common song to the other sets, which means the corresponding clusters are not overlapping or at least have enough distance.

4.3 Intra-playlist analysis

For the sake of analysis, the results of the jazz playlist, which is the one that gave the best results without any filtering, will be analyzed. The contextual prefiltering showed almost no difference in the given recommendations except for four songs which were changed: Wartime Blues, 34 Blues, Street Car Blues and Stop That Things. Probably the major influence on this effect was the tempo of the songs, not in line with the time of the day associated with the context (evening, slower tempo). The use of the sole post-filtering showed no difference with respect to the simple case, while interesting results can be noted when combining the two different approaches. The recommended songs in this last case are completely different from the ones saw before, with only two or three falling certainly in the jazz/blues genre. These results proves that, given correct starting recommendations, the use of contextual information does not provide major additional insight into the recommendation process. Moreover, an abuse of contextual information may lead to loss of pertinence with the original data, as it is shown in the last recommendation phase. On a final remark, performing pre-filtering shows significant advantages in the computational effort: in fact, rather than performing calculations the full dataset, only roughly the 65% of it is taken into account. This can be of extreme importance in large-scale applications.

5 Conclusions and future work

The presented work surely shows many simplifications and opportunities to improve. This section outlines the major directives of a possible future work and the findings on which prospective developments should focus. The recommendation results highlight three major issues:
(1) An insufficient number/quality of the features extracted
(2) Potential drawbacks of small-scale datasets (3) Need for extensive research on proper context integration.

The influence of the feature extraction phase can be seen in the low amount of clusters predicted by meanshift clustering and indicated by the silhouette values as well as in the failure to efficiently classify and differentiate rock and jazz music. Improvements could be obtained with the use of low-level features extracted from the spectral content of samples, such as MFCC [Logan and others, 2000]. A larger dataset, on the other hand, can surely help having a bigger variety of the data and better clusters. Amadeus architecture can be integrated with larger scale datasets without conceptual changes to its core algorithm. A possible next step can be implementing the full (or at least a major part of) Million Song Dataset. Finally, the role of context needs further research, mainly to map the correct relationships between extracted features and contextual information. This is, however, a challenge which requires a proper user validation with fixed metrics, such as the one proposed by [Bogdanov *et al.*, 2010].

References

[Baltrunas and Amatriain, 2009] Linas Baltrunas and Xavier Amatriain. Towards time-dependant recommendation based on implicit feedback. In *Workshop on context-aware recommender systems (CARS09)*, 2009.

[Bogdanov *et al.*, 2010] Dmitry Bogdanov, Martín Haro, Ferdinand Fuhrmann, Emilia Gómez, and Perfecto Herrera. Content-based music recommendation based on user preference examples. *Copyright Information*, page 33, 2010.

[Burke, 2002] Robin Burke. Hybrid recommender systems: Survey and experiments. 12, 11 2002.

[Çataltepe and Altinel, 2007] Zehra Çataltepe and Berna Altinel. Music recommendation based on adaptive feature

and user grouping. In *Computer and information sciences, 2007. iscis 2007. 22nd international symposium on*, pages 1–6. IEEE, 2007.

[George *et al.*, 2001] Tzanetakis George, Essl Georg, and Cook Perry. Automatic musical genre classification of audio signals. In *Proceedings of the 2nd international symposium on music information retrieval, Indiana*, 2001.

[Lesaffre *et al.*, 2006] Micheline Lesaffre, Marc Leman, and Jean-Pierre Martens. A user-oriented approach to music information retrieval. In *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum fr Informatik, 2006.

[Logan and others, 2000] Beth Logan et al. Mel frequency cepstral coefficients for music modeling. In *ISMIR*, 2000.

[Panniello *et al.*, 2009] Umberto Panniello, Alexander Tuzhilin, Michele Gorgoglione, Cosimo Palmisano, and Anto Pedone. Experimental comparison of pre-vs. post-filtering approaches in context-aware recommender systems. In *Proceedings of the third ACM conference on Recommender systems*, pages 265–268. ACM, 2009.

[Rousseeuw, 1987] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[Song *et al.*, 2012] Yading Song, Simon Dixon, and Marcus Pearce. A survey of music recommendation systems and future perspectives. 06 2012.

[Trohidis *et al.*, 2008] Konstantinos Trohidis, Grigorios Tsoumakas, George Kalliris, and Ioannis P Vlahavas. Multi-label classification of music into emotions. In *ISMIR*, volume 8, pages 325–330, 2008.

A Playlists and results

Jazz playlist	
Song	Author
Are you there (with another girl)	Burt Bacharach
Everyday I have the blues	Lightinin' Hopkins
Heartbreaker	Ray Charles
You can't hurry love	Phil Collins
Tenderly	Billy May
Please come home for Christmas	Lou Ann Barton
Jingle Bells Rock	Bobby Russel
Dios Bendiga	Andy & Lucas
Harlem Shuffle	Righteous Brothers
When the saints go marching in	Precious Bryant
Lonesome Road	Dean Elliott
Let it be me	Ben E. King
Apertura in Jazz	Piero Umiliani
Please Remember Me	B.B. King
Bus Station Blues	Little Freddie King

Rock playlist	
Song	Autor
Immigrant Song	Led Zeppelin
Start me up	The Rolling Stones
War?	System of a Down
Sweet child o' Mine	Guns 'N Roses
Three little birds	Bob Marley & The Wailers
Crazy	Aerosmith
Lose this skin	The Clash
Strange days	The Doors
Snow (Hey Ho)	Red Hot Chili Peppers
Vodoo Chile Blues	Jimi Hendrix
With a little help from my friends	Santana
Heart-shaped Box	Nirvana
Hit the lights	Metallica
A new hope	Blink-182
God save the queen	Sex Pistols

Orchestra playlist	
Song	Author
Two Elegiac Melodies op. 34: 2 mvt.	Neville Marriner
Bach -Concerto in C Major: 1 mvt	Lionel Rogg
The Imperial March	John Williams
Beethoven Symphony No. 7: 1 mvt	H. Von Karajan
Sporco ma distinto	Ennio Morricone
Allegro io son	Juan Florz
Bach Suite No.1	John Williams
Beethoven Piano Sonata no.28: 3 mvt	Solomon
Hary Janos Suite: 6 mvt.	Neeme Jarvi
Haendel 25 Variations and fugue on a theme	Solomon
Bach Air Suite no.3	Mariano Yanani
Mozart Requiem Mass K626	N. Scphiller
Chinoiserie	Lincoln Center Orchestra
Hungarian Dances no. 12	Hans Swarowsky
Donizetti, Le ore trascorrono	Roberto Devereux

Table 4: Listening playlists

Jazz simple recommendations		Jazz contextual pre-filtering recommendation	
Song	Author	Song	Autor
Writing Paper Blues	Blind Willie McTell	Writing Paper Blues	Blind Willie McTell
Long Lonesome Blues	Blind Lemon Jefferson	Long Lonesome Blues	Blind Lemon Jefferson
Mississippi Boweavil Blues	Charley Patton	Mississippi Boweavil Blues	Charley Patton
Got The Blues	Blind Lemon Jefferson	Got The Blues	Blind Lemon Jefferson
Ma Rainey's Black Bottom	Ma Rainey	Ma Rainey's Black Bottom	Ma Rainey
Moon Going Down	Charley Patton	Moon Going Down	Charley Patton
Wartime Blues	Blind Lemon Jefferson	Stop That Thing	Sleepy John Estes
34 Blues	Charlie Patton	Fixin' To Die Blues	Bukka White
Street Car Blues	Sleepy John Estes	Sleepy Man Blues	Bukka White
Stop That Thing	Sleepy John Estes	Kyrie Eleison	Jane Winther

(a) jazz1

Table 5: Jazz playlist recommendation(1)

Jazz contextual post-weighting recommendations		Jazz combined filtering/weighting recommendations	
Song	Author	Song	Author
Ma Rainey's Black Bottom	Ma Rainey	Bourre des Monts d'Aubrac	Jean Segurel
Moon Going Down	Charley Patton	Canon In D Major (J. Pachelbel)	Mariano Yanani
Long Lonesome Blues	Blind Lemon Jefferson	Rollercoaster Ride	Aaron Watson
Mississippi Boweavil Blues	Charley Patton	Tumba Tumba	Charanga Forever
Got The Blues	Blind Lemon Jefferson	(The Grave Prelude)	Mobb Deep
Writing Paper Blues	Blind Willie McTell	Redwing	Hem
Stop That Thing	Sleepy John Estes	Skeletons (Live Acoustic Version)	Rickie Lee Jones
Wartime Blues	Blind Lemon Jefferson	Someday (I Will Understand)	Britney Spears
34 Blues	Charlie Patton	Chime	Shapeshifters
Fixin' To Die Blues	Bukka White	5 4 3 2 1	Rob Mullins

(a) jazz2

Table 6: Jazz playlist recommendation(2)

Rock simple recommendations		Rock contextual pre-filtering recommendation	
Song	Author	Song	Autor
Writing Paper Blues	Blind Willie McTell	Writing Paper Blues	Blind Willie McTell
Mississippi Boweavil Blues	Charley Patton	Mississippi Boweavil Blues	Charley Patton
Got The Blues	Blind Lemon Jefferson	Got The Blues	Blind Lemon Jefferson
Long Lonesome Blues	Blind Lemon Jefferson	Long Lonesome Blues	Blind Lemon Jefferson
Moon Going Down	Charley Patton	Moon Going Down	Charley Patton
Ma Rainey's Black Bottom	Ma Rainey	Ma Rainey's Black Bottom	Ma Rainey
Wartime Blues	Blind Lemon Jefferson	Stop That Thing	Sleepy John Estes
34 Blues	Charlie Patton	Fixin' To Die Blues	Bukka White
Street Car Blues	Sleepy John Estes	Sleepy Man Blues	Bukka White
Stop That Thing	Sleepy John Estes	Please Send Me Someone To Love	Percy Mayfield

(a) rock1

Table 7: Rock playlist recommendation(1)

Rock contextual post-weighting recommendations	
Song	Author
Moon Going Down	Charley Patton
Mississippi Boweavil Blues	Charley Patton
Ma Rainey's Black Bottom	Ma Rainey
Writing Paper Blues	Blind Willie McTell
Long Lonesome Blues	Blind Lemon Jefferson
Got The Blues	Blind Lemon Jefferson
Stop That Thing	Sleepy John Estes
Wartime Blues	Blind Lemon Jefferson
Fixin' To Die Blues	Bukka White
34 Blues	Charlie Patton

Rock combined filtering/weighting recommendations	
Song	Author
ABCD Medley	Laurie Berkner
Turning Away	Mary Black
Why don't you (...) anything ?	Aphasia
Why don't you (...) night ?	Zuzu Bollin
Ooh Wee Baby	Doyle Bramhall
What To Do Crying(...) The Game	Bobby Vee
Morghe Eshgh	Mohammad Nouri
Long Summer Days	The Moody Blues
Bodhrn Solo	Mick Moloney
Come On (Pt. III)	Stevie Ray Vaughan

(a) rock2

Table 8: Rock playlist recommendation(2)

Orchestra simple recommendations	
Song	Author
Stille nacht, heilige nacht	The American Boychoir
Sirne	Marc Perrone
Airs and Dances Suite No. 2: III. Aria	Sir Neville Marriner
Kyrie Eleison	Jane Winther
Someone to Watch Over Me	Frank Chacksfield
All Purpose Experiment	Controlled Dissonance
When Your Ready	Dabbler
Liquid Frequencies (liquid Soul Mix)	liquid soul-freq
You Make Me Feel So Good	Simon Harris
Test 72	Mush

Orchestra contextual pre-filtering recommendation	
Song	Autor
Kyrie Eleison	Jane Winther
All Purpose Experiment	Controlled Dissonance
When Your Ready	Dabbler
Liquid Frequencies	liquid soul, freq
You Make Me Feel So Good	Simon Harris
Test 72	Mush
Chinz Ninja	Smart Alex
Mui Mal Animal	L.A.V.I
La vida	Carlos Jimenez
Om	Jane Winther

(a) orchestra1

Table 9: Orchestra playlist recommendation(1)

Orchestra contextual post-weighting recommendations	
Song	Author
Test 72	Mush
All Purpose Experiment	Controlled Dissonance
Liquid Frequencies (liquid Soul Mix)	liquid soul-freq
Kyrie Eleison	Jane Winther
You Make Me Feel So Good	Simon Harris
Airs and Dances Suite No. 2: III, Aria	Sir Neville Marriner
Chinz Ninja	Smart Alex
Someone to Watch Over Me	Frank Chacksfield
La vida	Carlos Jimenez
Sirene	Marc Perrone

Orchestra combined filtering/weighting recommendations	
Song	Author
Damisela Encantadora	Percy Faith and Orchestra
Never Alone	Open Hand
Down On Me	Janis Joplin
Ain't Got Nothin' But The Blues	Robben Ford
Think That Love Was In Your Heart	Glen Ricks
Don't Go	Bert Kaempfert
Intro	The Germs
Lingan	Atman
What Is A Woman For?	The Edgar Broughton Band
Get It Started	2Fresh

(a) orchestra2

Table 10: Orchestra playlist recommendation(2)