Using Contextual Information in Music Playlist Recommendations

Anna GATZIOURA[a,1] and Miquel SÀNCHEZ-MARRÈ[a]
[a]*Dept. of Computer Science,*
Knowledge Engineering and Machine Learning Group (KEMLG)
Universitat Politècnica de Catalunya · BarcelonaTech

Abstract. Recommender Systems have become a fundamental part of various applications supporting users when searching for items they could be interested in, at a given moment. However, the majority of Recommender Systems generate isolate item recommendations based mainly on user-item interactions, without taking into account other important information about the recommendation moment, able to deliver users a more complete experience. In this paper, a hybrid Case-based Reasoning model generating recommendations of sets of music items, based on the underlying structures found in previous playlists, is proposed. Furthermore, the described system takes into account the similarity of the basic contextual information of the current and the past recommendation moments. The initial evaluation shows that the proposed approach may deliver recommendations of equal and higher accuracy than some of the widely used techniques.

Keywords. Hybrid Recommender System, Set of items recommendation, Case-Based Reasoning, Graph-based Similarity, Context, Playlist Recommendations, Music Recommender Systems.

1. Introduction

The majority of current recommender systems treat the recommendation problem at two dimensions, namely users and items, capture their interactions through ratings and based on those predict the most relevant items for a specific user. They usually focus on single item recommendations and tend to ignore other information related to the recommendation moment, like time, location, and joint item selections that in some application domains may highly affect the recommendation results.

The aim of the implemented system is the automatic generation and recommendation of music playlists, more specific, the recommendation of *sets of music items* being able to complete a started playlist. Therefore our emphasis is on the item co-occurring patterns and the playlist generation moment rather than on the users that performed those, as users attitudes may vary depending on their actual needs. In addition, apart from the characteristics of each item, the recommended set must have some characteristics as a whole, like *variety* and *coherence*, meaning that the same song should not be repeated and the changes between items should be smooth [2][11].

[1] Corresponding Author, A. Gatzioura, Dept. of Computer Science, Universitat Politècnica de Catalunya, Campus Nord, C/ Jordi Girona 1-3, 08034, Barcelona, Catalonia; E-mail: gatzioura@cs.upc.edu.

In this work, we propose an extended model of our previous MusCBR recommender [7] that first evaluates the contextual similarities of the recommendation moments in order to capture possible special characteristics, and then searches for the most adequate playlists. In the next section we present a short overview of music and playlist recommendation techniques and the term of context, while in the third section the description of our framework can be found. Following the comparison of our model with commonly used techniques and some of our future objectives are presented.

2. Related Work

2.1. Music RSs

Music recommender systems have their basis both in fields of recommender systems (RSs) and Music Information Retrieval (MIR) therefore have to deal with the common limitations of both areas [9]. Traditional MIR techniques use content-based (CB) audio related techniques that apart from the general limitations of content-based systems, like *overspecialization* and *limited diversity*, require a deeper knowledge of the application domain [4][14]. On the other hand, Collaborative Filtering (CF) techniques focus on user ratings to predict the rating a user would assign to a previously unknown song, based on the behavior of similar users, thus heavily suffer from *cold-start* and *long tail* effect limitations [12].

More than recommending single songs, increased focus has been placed lately on the construction and recommendation of music playlists, sets of music items designed to be consumed as a sequence, to enable users in organizing their music libraries and having a more complete experience [13]. However, as there are no solid methods combining user's perception of music with sound characteristics, it is difficult to specify the exact characteristics that the items of a playlist should have in order to compose a satisfactory result for the listener [11]. The authors in [13] categorize playlist generation algorithms mainly into *constraint satisfaction* methods that generate playlists based on some user entered criteria, *similarity heuristics* that given a seed music item and using some similarity function aim to identify the most similar ones, and *machine learning* approaches that use a set of playlists to train a model and based on it to recommend playlists. Markov models and association rules, or sequential patterns, are also being used to capture song co-occurrences. CF approaches treating playlists as users or CB approaches using musical features can be used. Among the major limitations of these methods are their computational cost and their performance dependency on the used data. Due to the long tail effect in the music domain, popularity-based techniques may also be of high accuracy [3].

2.2. Context

Incorporating contextual parameters into the recommendation problem has been found to improve recommendation accuracy, and possible users' access to more long tail items [5]. Initially context was defined as any information referring to the user's location, the people and the resources around him/her and the changes in those. In music recommendations contextual information can be classified mainly into primary (environmental-related) like location, time and weather, and secondary (user-related) like activity and emotional parameters [8].

A context-aware recommendation process refers to the estimation of user contextual characteristics, and based on those, on the generation of the most relevant recommendations, and it generally takes three forms. *Contextual pre-filtering* selects, according to a specific context, the data that will be further used for the recommendations. In *contextual post-filtering* an initial recommendation set is generated using the traditional recommendation approaches and is then adjusted based on the actual context. Finally, in *contextual modelling* contextual information is used to transform items into a different dimension [1]. In order to improve the predictive ability and also support long tail recommendations, Domingues *et al*. in [6] present an interesting contextual modeling approach that exploits multidimensional data with two dimensional recommenders. The additional dimensions are modeled as "virtual items" that along with the regular items are used for building the recommendation model. On the other hand, Cremonesi *et al*. [5] propose the use of association rules to identify the most significant correlations among item and contextual characteristics in order to filter the set of predictions generated by a traditional recommendation method.

3. Method Description

In this work, we propose an extension of our previous model [7] that first evaluates contextual information in order to better capture the characteristics of playlists generated at a specific moment.

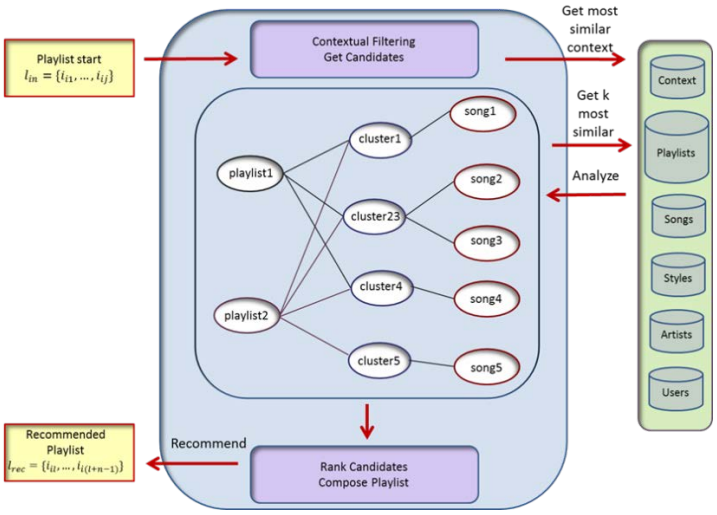

Figure 1. Recommendation process overview.

As the scope is to generate recommendation of joint music items that would fit well into a started playlist, we model entire playlists as cases. After a contextual pre-filtering the system follows the general CBR cycle, thus in order to generate recommendations for a new list we first find the most similar past lists in order to build the new solution according to their structures [10]. Finally, in order to generate the recommendation list, our algorithm sorts the items found in the most similar lists based on the aggregated similarity of the cases in which they appeared.

3.1. Contextual Filtering

In the implemented system, first the contextual information about the playlists is evaluated through a contextual pre-filtering process in order to first identify a set of candidate cases able to address more complicated queries in terms of user experience, as well as to fasten their retrieval. Among the contextual parameters available about playlists, are the user who constructed them and the time moment at which this happened. After analyzing song distributions in playlists per month and per hour, (based on the hypothesis that music preferences are associated with the weather that changes during the year, and the hour of the day that affects the activities performed), we have found that, indeed, the hour of the day seems to influence the style of the generated playlists. Therefore, the initial contextual clustering is done based on the time that playlists were constructed. When a new playlist is introduced, its context is evaluated and the candidate cases are retrieved from the most similar contextual cluster.

3.2. Candidate Cases

After the contextual pre-filtering we search among the candidate cases for the most similar to the new case. Given a database of z distinct songs (items), $I = \{i_1, \dots, i_z\}$ and a set of previously reproduced playlists $L = \{l_1, \dots, l_k\}$ where each playlist $l_. \in L$ is described as the set of songs from the set *I* of which it consists, like $l_. = \{i_1, i_2, \dots, i_j\}$. The purpose of the system is, given a new initiated list $l_{in} = \{i_{i1}, \dots, i_{ij}\}$ to find and recommend the set of n music items $l_{rec} = \{i_{il}, \dots, i_{i(l+n-1)}\}$ to complete this list. Therefore, we look for the lists, whose *global similarity*, fulfills the following equation,

$$l' \in L: \forall l \in L, l' = argmax\ Sim(l, l_{in}) \qquad (1)$$

The global similarity among two playlists is computed as the aggregated pairwise similarity of the songs (being referred to as *local similarity*) in them. Given two playlists, a new $l_N = \{i_{N1}, \dots, i_{Nn}\}$ and a retrieved $l_R = \{i_{R1}, \dots, i_{Rm}\}$ with their i-*th* item being i_{Ni} and i_{Ri} correspondingly, that have local similarity $sim(i_{Ni}, i_{Ri})$ and where $maxsim(i_{Ni}, i_{Rm})$ is the maximum local similarity among i_{Ni} and all the songs in the retrieved list, the global similarity $Sim(l_N, l_R)$, of these lists is calculated as,

$$Sim(l_N, l_R) = \sum_{i=1}^{l} maxsim(i_{Ni}, i_{Rm}) / n_N \qquad (2)$$

3.3. Item Descriptions

Every song can be treated as the node of a graph connected with edges to a set of metadata expressing its characteristics (category, tempo, lyrics language, artist etc.), thus it can be represented as $i = \{t_1, t_2, \dots, t_n\}$, $t_i \in T$. Depending on the *level of abstraction* that we would like to use, the quantity of item characteristics used in the item description may vary and we may refer as *music items* to songs, artists, or song styles. Songs are connected through the tags they have in common and the playlists of which they form part. As our scope is to capture the specifications of the songs selected together more than the exact songs, instead of looking for the songs that maximize (1) we focus on labeled styles or artists as this information is more abstract but specific enough to capture the tendencies in playlists.

Given two songs i_a and i_b with $n(a \cap b)$ number of characteristics in common, $n(a \backslash b)$ tags associated only with i_a and $n(b \backslash a)$ associated only with i_b, their (local) similarity is calculated based on the density of their common characteristics like,

$$sim(i_a, i_b) = 1 - log_2\left(1 + \frac{n(a\backslash b)+n(b\backslash a)}{n(a\cap b)+n(a\backslash b)+n(b\backslash a)}\right) \quad (3)$$

4. Evaluation

In order to evaluate the performance of the proposed framework we used a personal database with more than 3000 songs and playlists made of them, divided into a training (80%) and a testing part. Further, from the testing lists, we hide a number of items and evaluate the ability of the various methods to identify those items. We focused on the recommendations of *item clusters*, items having the same style, artist or both, as many times users select only few items from a cluster with specific characteristics simply because they are not aware of the rest.

In Tables 1-3 we present the average precision values (the number of correct recommendations, over the total number of recommendations) achieved by *CMusCBR*, our contextual recommender, *MusCBR*, the hybrid recommender in [7], as well as *ARs-based (AR)*, *Latent Dirichlet Allocation (LDA)*, two *Popularity-based* approaches, based on users' favourite artists (*popA*) and styles (*popSt*), *Collaborative Filtering (CF)* and *Content-based (CB)*. As it can be seen, the proposed methodology performs better than the compared methods for all the levels of abstraction tested.

Table1: Average precision for the recommendation of complete music styles

List length	4	6	8	10	12
AR	0.055	0.063	0.052	0.052	0.035
LDA	0.049	0.061	0.068	0.081	0.094
PopA	0.039	0.054	0.058	0.067	0.064
PopSt	0.043	0.048	0.053	0.064	0.064
CB	0.042	0.050	0.055	0.061	0.056
CF	0.045	0.058	0.068	0.08	0.079
MusCBR	0.078	0.078	0.076	0.086	0.083
CMusCBR	**0.097**	**0.090**	**0.090**	**0.096**	**0.098**

Table2: Average precision for artist recommendations

List length	4	6	8	10	12
AR	0.052	0.087	0.108	0.115	0.096
LDA	0.034	0.050	0.075	0.087	0.093
PopA	0.044	0.073	0.098	0.115	0.117
CB	0.029	0.045	0.059	0.066	0.072
CF	0.048	0.078	0.102	0.118	0.119
MusCBR	0.095	0.103	0.115	0.127	0.131
CMusCBR	**0.096**	**0.108**	**0.124**	**0.139**	**0.135**

Table3: Average precision for the recommendation of music styles

List length	4	6	8	10	12
AR	0.261	0.32	0.363	0.367	0.372
LDA	0.202	0.345	0.167	0.264	0.317
PopA	0.34	0.346	0.336	0.334	0.337
PopSt	0.38	0.402	0.394	0.394	0.395
CB	0.289	0.207	0.156	0.122	0.082
CF	0.185	0.259	0.292	0.315	0.345
MusCBR	0.386	0.39	0.397	0.396	0.395
CMusCBR	**0.392**	**0.403**	**0.4**	**0.401**	**0.401**

5. Conclusions

In this paper, a hybrid CBR approach for music playlists generation and recommendations has been presented. The implemented system first evaluates the context of the recommendation moment, and then, it identifies and recommends sets of music items that are more probable to fit into a started playlist. In order to overcome the semantic gap of music recommender systems, the proposed system combines Case-Based Reasoning with a graph-based model that connects songs through the tags that define their style and the playlists that they form part of, in order to capture their similarity and co-occurring patterns. The initial experimentations have shown that adding a contextual pre-filtering based on the playlists' creation time leads to improved recommendations' accuracy and computational performance when compared to our previous and various commonly used methods.

Our intention is to extend this framework in order to provide additional support to long tail recommendations, as items' novelty and diversity that are considered of high importance for the user experience. In addition, the possible incorporation of more contextual parameters is being evaluated. Finally, testing on large scale data is planned to take place in the near future.

References

[1] G. Adomavicius, and A. Tuzhilin, "Context-aware recommender systems," Recommender systems handbook, pp. 191-226, Springer US, 2015.

[2] C. Baccigalupo and E. Plaza, "Case-Based Sequential Ordering of Songs for Playlist Recommendation," Advances in Case-Based Reasoning 4106, p. 286-300, 2006.

[3] G. Bonnin, and D. Jannach, "A comparison of playlist generation strategies for music recommendation and a new baseline scheme," Workshops at the 27th AAAI Conference on Artificial Intelligence, 2013.

[4] M. A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney, "Content-Based Music Information Retrieval: Current Directions and Future Challenges," Proceedings of the IEEE, 96. pp. 668 – 696, 2008.

[5] P. Cremonesi, P. Garza, E. Quintarelli, and R. Turrin, "Top-n recommendations on unpopular items with contextual knowledge," In 2011 Workshop on Context-aware Recommender Systems, Chicago, 2011.

[6] M. A. Domingues, A. M. Jorge, and C. Soares, "Using contextual information as virtual items on top-n recommender systems," arXiv preprint arXiv: 1111.2948, 2011.

[7] A. Gatzioura, and M. Sànchez-Marrè, "A Case-Based Reasoning Framework for Music Playlist Recommendations," Procc. of the 4th IEEE International Conference on Control, Decision and Information Technologies (CoDIT'17), April 2017.

[8] M. Kaminskas, and F. Ricci, "Contextual Music Information Retrieval and Recommendation: State of the Art and Challenges," Computer Science Review 6, pp. 89-119, 2012.

[9] E. Y. Kim, M. E. Schmidt, R. Migneco, G. B. Morton, P. Richardson, J. Scott, A. J. Speck and D. Turnbull, "Music Emotion Recognition: A State of the Art Review," Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010), pp. 255-266, 2010.

[10] R. López de Mántaras, "Case-Based Reasoning," Machine Learning and its Application, vol. 2049, pp.127-145, 2001.

[11] F. Maillet, D. Eck, G. Desjardins, and P. Lamere, "Steerable Playlist Generation by Learning Song Similarity from Radio Station Playlists," Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009), pp. 345-350, 2009.

[12] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," pp. 1-35, Springer US, 2011.

[13] M. Schedl, P. Knees, B. McFee, D. Bogdanov, and M. Kaminskas, "Music recommender systems," Recommender Systems Handbook, pp. 453-492, Springer US, 2015.

[14] B. Shao, D. Wang, T. Li and M. Ogihara, "Music Recommendation Based on Acoustic Features and User Access Patterns," IEEE Transactions on Audio, Speech, and Language Processing 17:1602 – 1611, 2009.