Listener-Inspired Automated Music Playlist Generation

Andreu Vall
Department of Computational Perception
Johannes Kepler University
Altenberger Str. 69
A-4040 Linz, Austria
andreu.vall@jku.at

ABSTRACT

The objective of this PhD research is to deepen the understanding of how people listen to music and construct playlists. We believe that further insights into such mechanisms can lead to enhanced music recommendations. We research on the exploitation of user-generated data in the context of on-line music services, since it constitutes a rich and increasing source of information of user behavior. The research carried out so far has centered on the scenario of producing a single artist recommendation. Concretely, in this paper we show how to mitigate the cold-start problem for new artists, elaborating on our findings on the combined effect of users' listening histories and users' tagging activity. As future research, we will investigate how improved techniques to exploit user-generated data can also be applied to the task of producing sequential recommendations, like playlists. We are particulary interested in creating music playlists similarly as users would do, and in finding mechanisms to make such music streams adapt to users' feedback on-line.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering; H.5.5 [**Sound and Music Computing**]: Modeling

Keywords

Music recommendation, playlist generation, recommender systems, cold-start problem, collaborative filtering

1. INTRODUCTION

Given the amount of digitized music available, music recommendations and automatically generated playlists assist users in navigating, organizing and exploring music collections.

The interaction of users with on-line music streaming services provides an increasing amount of data, mostly in the form of implicit feedback (e.g., the number of times a user listened to an artist). Such feedback is easy to collect in comparison to elicited user ratings (i.e., explicit feedback), and has been proven powerful in collaborative filtering applications [12, 24]. Part of this PhD research is devoted to understanding and improving the techniques required to exploit implicit feedback datasets for the task of music recommendation (including automated playlist generation). For the task of recommending a single artist, we show how to mitigate the cold-start problem for new artists by extending state-of-the-art collaborative filtering models for implicit feedback with additional tagging activity.

As future work, we want to investigate the automated generation of music playlists. As already suggested in [2, 7, 21, 29], user-generated data can be successfully exploited for this task. We will explore how our extensions to state-of-the-art collaborative filtering models can be translated into sequential recommendations as well.

The remainder of this paper is organized as follows. In Section 2 we review the relevant work related to (1) the cold-start problem in collaborative filtering, and (2) the automated generation of playlists. We elaborate on our findings to mitigate the cold-start problem in Section 3. We outline future directions for music playlist generation in Section 4.

2. RELATED WORK

2.1 Hybrid Collaborative Filtering for Implicit Feedback

In the context of music streaming services, users' listening activity represents an abundant source of data. This is commonly referred to as *implicit feedback*, because the users do not provide it actively. In contrast, the extensively studied *explicit feedback* is actively provided by users (e.g. ratings, or "likes"). Collaborative filtering methods for explicit feedback, based on the factorization of the user-item-rating matrix [16], can be applied to implicit feedback datasets as well. In order to do this, the factorization models need to be adapted, weighting the observations according to their relative importance in the dataset [12, 24].

Hybrid models can be used to mitigate cold-start situations in collaborative filtering [1]. Models based on matrix factorization can fuse together the user-item feedback matrix and additional sources of data [9, 19]. The decomposition into latent factors is then reinforced, and the lack of user-item feedback is compensated with external data.

Evaluation has often focused on optimizing the accuracy achieved for predicting ratings. However, accuracy may not

RecSys'15, September 16–20, 2015, Vienna, Austria.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3692-5/15/09 ...$15.00.
DOI: http://dx.doi.org/10.1145/2792838.2796548.

be enough to satisfy users [22]. Alternatively, [15] evaluates the performance of different recommender systems on the basis of ranked lists of recommendations. This evaluation methodology is adapted in [12] to deal with implicit feedback and it is further applied in [17].

2.2 Automated Music Playlist Generation

Automated playlist generation can be based on different sources of data, e.g., musical features extracted from the audio signal [23], social web data [11], or usage data, like users' listening histories [6] or manually created playlists [21].

The context and application of playlists is an important aspect too, because ultimately, the music recommendations have to fit the users' needs. We find examples of application-specific automated playlist generators for background music while doing sports [20], while driving [3], or for discovery and exploration of new music [28, 29].

Manually generated playlists are organized following an internal logic and coherence [8]. In automated music playlist generation, we find three main approaches to favor the generation of coherent sequences. First, by imposing smooth transitions between tracks [10, 14, 25]. Second, by learning from manually generated playlists [2, 7]. Third, by using exploration-exploitation techniques to build playlists that adapt to users' feedback on-line [18, 28, 29].

For a complete survey on automated generation of music playlists please refer to [5].

3. HYBRID COLLABORATIVE FILTERING WITH TAGS

We provide a concise description of our collaborative filtering model hybridized with tagging activity and show how to evaluate the quality of its recommendations. For a complete derivation of the model, please refer to our previous work [27]. For the task of artist recommendation, we show how the proposed model mitigates the cold-start problem for new artists and provide a qualitative example.

3.1 Proposed Model

Our model naturally extends the ones described in [9, 12]. The main contribution is that we identify tags as a source of positive-only feedback (even if the meaning of a tag is semantically negative). Therefore, we use a weighting scheme for the tags similar to the one used for implicit feedback, which proves to be beneficial.

Given a system with N users and M artists, the counts for each user-artist pair are stored in a matrix $R \in \mathbb{N}^{N\times M}$. Each row of R corresponds to a user, and each column to an artist. Assuming that users have tagged artists using T different tags, two matrices are defined. The counts for each user-tag pair are stored in a matrix $T^U \in \mathbb{N}^{N\times T}$, with rows relating to users and columns relating to tags. The counts for each artist-tag pair are stored in a matrix $T^A \in \mathbb{N}^{M\times T}$, with rows relating to artists and columns relating to tags.

We define binary matrices $\widetilde{R}$, $\widetilde{T}^U$ and $\widetilde{T}^A$, such that all non-zero entries in the matrices become 1, indicating that a user-artist, user-tag or artist-tag pair has been observed. All zero entries remain as zeros. A weight function is defined:

$$w(\eta, x) = 1 + \eta \log(1 + x). \tag{1}$$

We jointly factorize $\widetilde{R}, \widetilde{T}^U, \widetilde{T}^A$ into three D-rank matrices $P \in \mathbb{R}^{N\times D}$, $Q \in \mathbb{R}^{M\times D}$ and $X \in \mathbb{R}^{T\times D}$ (rows are latent features for users, artists and tags respectively) such that the following cost function is minimized:

$$\begin{aligned} J(P,Q,X) = &\sum_{ua\in R_{tr}} w(\alpha, R_{ua}) \left(\widetilde{R}_{ua} - P_u Q_a^T\right)^2 \\ &+ \mu_1 \sum_{ut\in T^U} w(\beta, T^U_{ut}) \left(\widetilde{T}^U_{ut} - P_u X_t^T\right)^2 \\ &+ \mu_2 \sum_{at\in T^A} w(\gamma, T^A_{at}) \left(\widetilde{T}^A_{at} - Q_a X_t^T\right)^2 \\ &+ \lambda\Big(\|P\|_F^2 + \|Q\|_F^2 + \|X\|_F^2\Big). \end{aligned} \tag{2}$$

Matrices $\widetilde{R}$, $\widetilde{T}^U$ and $\widetilde{T}^A$ are reconstructed using combinations of P, Q and X. In the first term of equation (2), $\widetilde{R}_{ua}$ is the entry of $\widetilde{R}$ corresponding to user u and artist a. P_u is the row of P corresponding to user u, and Q_a is the row of Q corresponding to artist a. The squared reconstruction error is weighted using a function of the actual number of counts in R_{ua} according to equation (1), and summed over all the user-artist pairs in the training matrix R_{tr}.[1] The second and third terms can be described analogously, with the difference that the squared reconstruction errors are summed over all user-tag pairs and artist-tag pairs respectively. Specific parameters α, β and γ are used for each term, and are determined by grid search. A regularization term involving the Frobenius norm of P, Q and X is added to prevent the model from over-fitting. The regularization parameter λ is also determined by grid search.

The cost function in equation (2) is minimized using Alternating Least Squares, similarly as in [9, 12]. The details can be found in [27].

Once the model is trained, the expected binary preference for all the user-artist pairs is predicted as $Z = PQ^T$. Note that the tags' factor matrix X is not directly involved in this product, although it is important to learn the users' and artists' factor matrices P and Q.

3.2 Evaluation Methodology

We follow the evaluation methodology proposed in [15] and extended in [12] to deal with implicit feedback datasets.

The observed user-artist pairs are split into training and test sets to perform 5-fold cross validation. Each user has approximately 80% of the listened artists in the training set and 20% in the test set. For each observed user-artist pair u, a in the test set, a random list of artists (not including a) is drawn. The list is then ranked according to the learned preferences of user u. Finally, a is inserted in the sorted list, and its percentile rank within the list is stored as $rank_{ua}$. If a is ranked among the top positions of the list, then its percentile rank is close to 0%. If it is ranked in last positions, then its percentile rank is close to 100%.

This process is performed over the 5 splits, so that $rank_{ua}$ is available for all the observed user-artist pairs in the dataset. The *expected percentile rank* is defined as a weighted average of $rank_{ua}$:

$$\overline{rank} = \frac{\displaystyle\sum_{ua\in R} R_{ua} rank_{ua}}{\displaystyle\sum_{ua\in R} R_{ua}}. \tag{3}$$

[1] Including also the zero entries of R_{tr}, as described in [12].

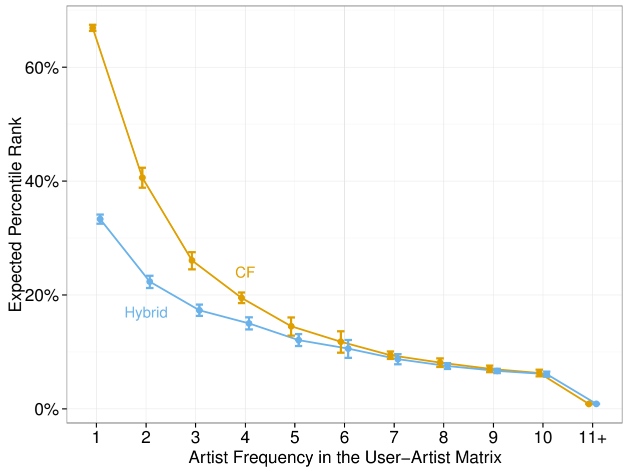

Figure 1: Performance of *CF* and *Hybrid* as a function of the number of observed occurrences of the artists in the dataset. The dots correspond to the expected percentile rank and the error bars display 95% basic bootstrap confidence intervals. The different models are dodged to avoid overlapping. The top line correspond to the baseline model. The lower corresponds to the proposed model.

3.3 Cold-Starting Collaborative Filtering

We compare the proposed model that uses implicit feedback and tags (i.e., *Hybrid*), with its simpler version, where plain collaborative filtering (*CF*) on implicit feedback data is considered.

We use a dataset of Last.fm listening events, top tags used by users and top tags applied to artists, collected through the Last.fm API.[2] The dataset includes 2,902 users, 71,223 artists and 687,833 observed user-artist pairs. We additionally obtain 630 unique tags for 600 users (20% of all users) and 12,902 unique tags for 67,332 artists (95% of all artists). We will focus on showing how the proposed model mitigates the cold-start problem for new artists.

We explore the effect of the number of observed occurrences of an artist in the dataset on the performance of the recommender systems. If an artist has only been listened by few users, plain collaborative filtering has little information on the relations between this artist and the users. Then, hybridizing with tags will prove advantageous.

Each user-artist pair in the dataset is assigned to a subset, defined by the number of occurrences of the artist in the dataset. Equation (3) is computed for each subset, summing only over the corresponding user-artist pairs. Figure 1 shows the expected percentile rank for *CF* and *Hybrid* for each subset. Artists with a single occurrence in the dataset are poorly recommended by *CF*. Thanks to the tags, *Hybrid* has more information about them and produces better recommendations. The more often an artist is observed in the dataset, the smaller the difference of performance becomes.

We provide an example to give a qualitative explanation of the effect of using tags. We train *CF* and *Hybrid* using 80% of the data. Table 1 shows the predicted preference Z_{ua} of a selected user u for four different artists he or she listened to, but belong to the 20% of data withheld for test. *CF* is not able to identify that the first and the fourth artist were interesting for the user. The reason is that these artists were not observed at all in the training set. For the second and third artists, *CF* predicts low values. *Hybrid* is able to identify the artists without listening examples and predicts higher values for the other two. Comparing the artist tags in the user's training data with the artist tags of these four artists, we find several coincidences, like *catalan*, *indie pop* or *pop surrealista* (surreal pop). Therefore, *Hybrid* is able to identify the relations.

Artist Name	Z_{ua}^{CF}	Z_{ua}^{Hybrid}
Feliu Ventura	0.00	0.61
Joan Colomo	0.35	0.75
Manos de Topo	0.23	0.64
Mazoni	0.00	0.69

Table 1: Prediction of *CF* and *Hybrid* for the preference Z_{ua} of a selected user for artists known to be relevant for the user (but hidden from the training).

4. FUTURE WORK ON AUTOMATED MUSIC PLAYLIST GENERATION

As next steps within this PhD research, we will explore how the described techniques to exploit user-generated data can be translated into automated playlist generation. In this line, the modeling of radio streams presented in [2] and the adaptive learning approach taken in [29] are interesting stepping stones. However, since both are based on plain collaborative filtering, they can not provide sensible estimates for artists in the long tail or for which little listening information is available. We will explore how our hybrid model can enhance their performance, probably mitigating the cold-start problem.

We are particularly interested in the adaptive learning capabilities of the reinforcement learning models described in [18, 28, 29]. Although implicit feedback can be used to estimate the expected user preferences, these models update their knowledge of the current user needs on the basis of elicited explicit user feedback. We consider that this does not reflect a realistic use case, where ideally, the model should learn with minimal effort from the user. We will investigate whether similar reinforcement learning approaches can be developed, updating the user's current needs on the basis of skipping behavior only (see e.g., [13, 23]).

The evaluation of the automated generation of playlists is a crucial aspect per se. We aim at generating playlists in a similar way as listeners would do. For this reason, we need to evaluate our models by means of user studies and, if possible, by means of actual large-scale on-line experiments. However, user studies also have limitations, and large-scale on-line experiments may be too complex in the context of a PhD research. Therefore, we will also need to investigate on improved off-line evaluation methods, probably in line with the approaches described in [7, 21], where actual manually generated playlists are used as a reference.

Websites, movies or videos are usually not organized in playlists, but for example, in the case of book recommendations, framing the recommendation as a sequential problem

[2] http://www.last.fm/api

has been proven beneficial in [26]. We will evaluate whether our model is also valid to produce recommendations in fields other than music. If so, the planned work for automated music playlist generation will also be explored for those fields.

Manually preparing playlists has been found to be a complex task. It involves the engagement of the creator and may require their creativity [8]. The development of algorithms able to imitate creative processes, is probably one of the most interesting challenges in automated playlist generation. We will research the field of computational creativity (see e.g., [4]) to explore how we can make computers generate playlists as listeners would do.

5. ACKNOWLEDGMENTS

This research is supported by the Austrian Science Fund (FWF): P25655.

6. 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 TKDE*, 17(6):734–749, 2005.

[2] N. Aizenberg, Y. Koren, and O. Somekh. Build your own music recommender by modeling internet radio streams. In *Proc. WWW*, pages 1–10. ACM, 2012.

[3] L. Baltrunas, M. Kaminskas, B. Ludwig, O. Moling, F. Ricci, A. Aydin, K.-H. Luke, and R. Schwaiger. Incarmusic: Context-aware music recommendations in a car. In *E-Commerce and Web Technologies*, pages 89–100. Springer, 2011.

[4] M. A. Boden. *The creative mind: Myths and mechanisms.* Routledge, London; New York, 2004.

[5] G. Bonnin and D. Jannach. Automated generation of music playlists: Survey and experiments. *ACM Computing Surveys*, 47(2):1–35, 2014.

[6] K. Bosteels, E. Pampalk, and E. E. Kerre. Evaluating and analysing dynamic playlist generation heuristics using radio logs and fuzzy set theory. In *Proc. ISMIR*, pages 351–356. Citeseer, 2009.

[7] S. Chen, J. L. Moore, D. Turnbull, and T. Joachims. Playlist prediction via metric embedding. In *Proc. SIGKDD*, pages 714–722. ACM, 2012.

[8] S. J. Cunningham, D. Bainbridge, and A. Falconer. ”More of an art than a science”: Supporting the creation of playlists and mixes. In *Proc. ISMIR*, 2006.

[9] Y. Fang and L. Si. Matrix co-factorization for recommendation with rich side information and implicit feedback. In *Proc. HETREC*, pages 65–69. ACM, 2011.

[10] A. Flexer, D. Schnitzer, M. Gasser, and G. Widmer. Playlist generation using start and end songs. In *Proc. ISMIR*, pages 173–178, 2008.

[11] N. Hariri, B. Mobasher, and R. Burke. Context-aware music recommendation based on latent topic sequential patterns. In *Proc. RecSys*, pages 131–131. ACM Press, 2012.

[12] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proc. ICDM*, pages 263–272. IEEE, 2008.

[13] Y. Hu and M. Ogihara. NextOne player: a music recommendation system based on user behavior. In *Proc. ISMIR*, pages 103–108, 2011.

[14] P. Knees, T. Pohle, M. Schedl, and G. Widmer. Combining audio-based similarity with web-based data to accelerate automatic music playlist generation. In *Proc. Int. Workshop on Multimedia IR*, pages 147–154. ACM, 2006.

[15] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proc. SIGKDD*, pages 426–434. ACM, 2008.

[16] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[17] Y. Li, J. Hu, C. Zhai, and Y. Chen. Improving one-class collaborative filtering by incorporating rich user information. In *Proc. CIKM*, pages 959–968. ACM, 2010.

[18] E. Liebman and P. Stone. DJ-MC: a reinforcement-learning framework for a music playlist recommender system. In *Proc. RLDM*, 2013.

[19] H. Ma, T. C. Zhou, M. R. Lyu, and I. King. Improving recommender systems by incorporating social contextual information. *ACM TOIS*, 29(2):1–23, 2011.

[20] N. Masahiro, H. Takaesu, H. Demachi, M. Oono, and H. Saito. Development of an automatic music selection system based on runner’s step frequency. In *Proc. ISMIR*, pages 193–8, 2008.

[21] B. McFee and G. R. Lanckriet. The natural language of playlists. In *Proc. ISMIR*, pages 537–542, 2011.

[22] S. M. McNee, J. Riedl, and J. A. Konstan. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In *Proc. CHI’06 Extended Abstracts*, pages 1097–1101. ACM, 2006.

[23] E. Pampalk, T. Pohle, and G. Widmer. Dynamic playlist generation based on skipping behavior. In *Proc. ISMIR*, volume 5, pages 634–637, 2005.

[24] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *Proc. ICDM*, pages 502–511. IEEE, 2008.

[25] T. Pohle, E. Pampalk, and G. Widmer. Generating similarity-based playlists using traveling salesman algorithms. In *Proc. DAFx*, pages 220–225. Citeseer, 2005.

[26] G. Shani, R. Brafman, and D. Heckerman. An MDP-based recommender system. *Journal of Machine Learning*, 6:1265–1295, 2005.

[27] A. Vall, M. Skowron, P. Knees, and M. Schedl. Improving music recommendations with a weighted factorization of the tagging activity. In *Proc. ISMIR*, 2015.

[28] X. Wang, Y. Wang, D. Hsu, and Y. Wang. Exploration in interactive personalized music recommendation: a reinforcement learning approach. *ACM TOMM*, 11(1), 2014.

[29] Z. Xing, X. Wang, and Y. Wang. Enhancing collaborative filtering music recommendation by balancing exploration and exploitation. In *Proc. ISMIR*, pages 445–450, 2014.