Expert Systems With Applications 55 (2016) 546–558

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Online recommender system for radio station hosting based on information fusion and adaptive tag-aware profiling

Dmitry I. Ignatov[a,*], Sergey I. Nikolenko[b,c], Taimuraz Abaev[a], Jonas Poelmans[a]

[a] Computer Science Faculty, National Research University Higher School of Economics, Kochnovskiy proezd, 3, Moscow 125319, Russia
[b] Laboratory of Internet Studies, National Research University Higher School of Economics, ul. Soyuza Pechatnikov, d. 16, St. Petersburg 190008, Russia
[c] Steklov Mathematical Institute, nab. r. Fontanka, 27, St. Petersburg 191023, Russia

ARTICLE INFO

Article history:
Received 22 July 2014
Revised 13 February 2016
Accepted 14 February 2016
Available online 18 February 2016

Keywords:
Music recommender system
Interactive radio network
Hybrid recommender system
Information fusion
Adaptive tag-aware profiling
Implicit feedback

ABSTRACT

We present a new recommender system developed for the Russian interactive radio network *FMhost*. To the best of our knowledge, it is the first model and associated case study for recommending radio stations hosted by real DJs rather than automatically built streamed playlists. To address such problems as cold start, gray sheep, boosting of rankings, preference and repertoire dynamics, and absence of explicit feedback, the underlying model combines a collaborative user-based approach with personalized information from tags of listened tracks in order to match user and radio station profiles. This is made possible with adaptive tag-aware profiling that follows an online learning strategy based on user history. We compare the proposed algorithms with singular value decomposition (SVD) in terms of precision, recall, and normalized discounted cumulative gain (NDCG) measures; experiments show that in our case the fusion-based approach demonstrates the best results. In addition, we give a theoretical analysis of some useful properties of fusion-based linear combination methods in terms of graded ordered sets.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

Music recommendation is an important and challenging direction in the field of recommender systems. It is a hard problem to extract relevant information for similarity search from large music collections, and sources of rich semantic metadata are often needed (Celma, 2010). Many recent works in this area have appeared at the International Society for Music Information Retrieval Conference (ISMIR) (Müller & Wiering, 2015) and the Recommender Systems conference (RecSys) (Werthner, Zanker, Golbeck, & Semeraro, 2015).

Recently, the focus of computer science research in music information studies has shifted from pure music information retrieval and exploration (Gleich, Zhukov, Rasmussen, & Lang, 2005; Hilliges, Holzer, Klüber, & Butz, 2006) to music recommendations (Brandenburg et al., 2009; Celma, 2010). It is not a new direction (Avesani, Massa, Nori, & Susi, 2002); however, it is now inspired by new capabilities of large online services that can provide not only millions of tracks for listening but also thousands of radio stations to choose from on a single web site. Moreover, social tagging is an important factor that paves the way to new recommender algorithms based on tag similarity (Nanopoulos, Rafailidis, Symeonidis, & Manolopoulos, 2010; Symeonidis, Ruxanda, Nanopoulos, & Manolopoulos, 2008b; Yang, Bogdanov, Herrera, & Sordo, 2012).

Despite many high quality works on different aspects of music recommendation, there are only a few studies devoted to online radio station recommender systems (Aizenberg, Koren, & Somekh, 2012; Grant, Ekanayake, & Turnbull, 2013). Several radio-like online broadcasting services, including *last.fm, Yahoo!LaunchCast*, and *Pandora*, are known for their recommender systems and work on a commercial basis (however, the latter two do not operate in Russia). There is their Russian counterpart, Yandex.Radio[1]; however, as the aforementioned services, it only provides access to radio stations with playlists, automatically composed according to a catalogue of selection criteria. Recommendation of real online radio stations is different. It presents difficulties not only due to the musical content but also because a recommender needs to find relevant dynamically changing objects while usually relying on implicit feedback only.

Thus, in this work we consider the music recommendation problem from a slightly different angle. We consider the Russian online radio hosting service *FMhost*, in particular, its new hybrid recommender system. First, we recommend radio stations, i.e.,

* Corresponding author. Tel.: +7 9263818033.
E-mail addresses: dignatov@hse.ru, dmitrii.ignatov@gmail.com (D.I. Ignatov), sergey@logic.pdmi.ras.ru (S.I. Nikolenko).

[1] http://radio.yandex.ru.

http://dx.doi.org/10.1016/j.eswa.2016.02.020
0957-4174/© 2016 Elsevier Ltd. All rights reserved.

sequences or sets of compositions, which are manually composed by real DJs and dynamically change, rather than individual tracks as most other music recommenders do. Second, as we demonstrate below, the *FMhost* service does not have enough data for reasonable SVD-based recommendations; still, recommendations have to be provided. To overcome these problems, we propose a novel recommender algorithm that combines three data sources: radio station visits, listening events for music with specific tags, and frequency of tags applied to radio stations and their content. We show experimental results on the *FMhost* dataset and propose a fusion of two different algorithms that can be tuned for specific quality metrics, e.g., precision, recall, and NDCG.

The paper is organized as follows. In Section 2, we briefly survey related work in music recommendation. Section 3 outlines the online radio service *FMhost*. In Section 4, we propose a novel recommender model, two basic recommender algorithms, a third algorithm that combines them, and describe the recommender system architecture. Section 5 provides examples, and Section 6 discusses the basic principles and problems in details. Quality of service (QoS) measurements, a comparison with an SVD-based approach, and certain insights into *FMhost* user behaviour are discussed in Section 7. In Section 8, we provide a theoretical justification of the chosen aggregation of rankings. Section 9 concludes the paper.

2. Related work

Music recommendation becomes especially important because modern systems that provide music to their users aim to take into account infrequently requested musical compositions and/or collections of compositions such as radio stations from the long tail of the distribution. Most music recommender systems work under the general principles of collaborative filtering (Koren & Bell, 2011). For instance, *last.fm* mines user tastes both explicitly, from likes with which the users mark compositions, and implicitly, from compositions that the users actually listen to. Many music recommenders also incorporate content-based features, analysing the actual composition. An interesting example of the latter is the *Pandora* service (Castelluccio, 2006; Joyce, 2006). *Pandora* decomposes a music composition into the so-called music genome; the “genes” of a composition are different for different music genres, and compositions are graded for various genes by professional musicologists. Still, even content-based systems usually employ collaborative filtering and use content features to supplement classical recommender algorithms, so basic ideas of collaborative filtering such as matrix factorization (Koren, 2008; Koren, Bell, & Volinsky, 2009; Ma, Yang, King, & Lyu, 2009; Salakhutdinov & Mnih, 2008a; 2008b) and nearest neighbors in both user-based and item-based collaborative filtering (Koren, 2010; Resnick, Iacovou, Sushak, Bergstrom, & Riedl, 1994; Said, Jain, Kille, & Albayrak, 2012; Sarwar, Karypis, Konstan, & Riedl, 2001) certainly apply; see also recent surveys on recommender systems (Bobadilla, Ortega, Hernando, & Gutiérrez, 2013; Konstan & Riedl, 2012; Lü et al., 2012; Zhou, Xu, Li, Josang, & Cox, 2012).

A widely acclaimed public contest on music recommender algorithms, KDD Cup,[2] was recently held by *Yahoo!*. In KDD Cup, track 1 was devoted to learning to predict users’ ratings of musical items (tracks, albums, artists, and genres) where the items formed a taxonomy: tracks belong to albums, albums belong to artists, where albums and tracks are also tagged with genres. Track 2 aimed at developing learning algorithms for separating music tracks scored highly by specific users from tracks that have not been scored by them. It attracted a lot of attention to the problems that are both typical for recommender systems and specific for music recommendation: scalability issues, capturing the dynamics and taxonomic properties of the items (Koenigstein, Dror, & Koren, 2011). Another major music recommender contest with open data, the Million Songs Dataset Challenge,[3] was held in 2012 by the Computer Audition Lab at UC San Diego and LabROSA at Columbia University (McFee, Bertin-Mahieux, Ellis, & Lanckriet, 2012). The core data consists of triples (*user, song, count*); it covers approximately 1.2 million users and more than 380,000 songs. While music recommender problems seldom have explicit user preferences (it is unlikely that users would rate every track they listen to), this kind of data can serve as input to one-class recommender algorithms that operate on only one kind of activity as input (Pan & Scholz, 2009; Pan et al., 2008; Sindhwani, Bucak, Hu, & Mojsilovic, 2010); these algorithms can also benefit from additional information about both users and items (Li, Hu, Zhai, & Chen, 2010; Yuan, Cheng, Zhang, Liu, & Lu, 2013).

Recent music recommendation trends reflect the advantages of hybrid approaches and call for user-centric quality measures (Celma & Lamere, 2011). For instance, the work (Hu & Ogihara, 2011) proposes a novel approach based on the so-called “forgetting curve” to evaluate “freshness” of predictions. Tags (both moderated and user-generated) turn out to be especially important for recommendations. The work (Bogdanov & Herrera, 2011) studies the problem of how much metadata one needs in music recommendation: a subjective evaluation of 19 users has revealed that pure content-based methods can be drastically improved with genre tags. Finally, the authors proposed a recommender approach that starts from an explicit set of music tracks provided by the user as evidence of his/her preferences and then infers high-level semantic descriptors for each track (Bogdanov et al., 2013). There is also a rich body of work devoted to tag suggestion and automated tag generation for music compositions since tags, especially generic tags like genres or social tags like “music for jogging” can then serve as preconstructed recommenders for the users (Bergstra, Casagrande, Erhan, Eck, & Kégl, 2006; Eck, Lamere, Bertin-Mahieux, & Green, 2007; Font, Serrà, & Serra, 2012; Zhao et al., 2010). However, in their extensive survey on music recommendation and retrieval (Kaminskas & Ricci, 2012) wrote “unfortunately, research on semantic music retrieval is still in its early stages,” and call to use tag-based approaches since “pure content analysis often fails to capture important aspects of human perception of music tracks.” Thus, in Hyung, Lee, and Lee (2014) the authors extracts keyterms from user’s personal stories and match their profiles with songs by latent semantic analysis. From graph-theoretic point of view, a quite close work is done by Lee, Kahng, and Lee (2015). Thus, the authors use so called aggregated bipartite factor-items graphs where factors should eliminate the direct usage of heterogeneous metadata reducing the number of stored graph nodes. In fact, our tag-aware profiles aggregate listened tracks considered as graph nodes into tag-based factors. However, we do it dynamically. Existing tag-aware frameworks are mostly used for building static profiles or tag recommendations per se (Kim & Kim, 2014; Nanopoulos et al., 2010); they also use different tag-propagation mechanism to cope with sparsity. In our solution we use online tag propagation for dynamic updating of users and radios profiles.

A substantial part of works on music recommendation is devoted to personalised playlist generation (Balkema & van der Heijden, 2010; Liu, Hsieh, & Tsai, 2010; Mocholí, Martinez, Martínez, & Catalá, 2012). And many online services (e.g., *last.fm* and *LaunchCast*) call their audio streams “radio stations”, however they are actually playlists from a database of tracks which are based on a recommender system rather than real predefined channels.

[2] http://kddcup.yahoo.com/.

[3] http://labrosa.ee.columbia.edu/millionsong/.

FMhost,[4] on the other hand, provides users with online radio stations in the classical meaning of this term: human DJs perform live, and a radio station actually represents a strategy or mood of a specific person (DJ) who plays his/her own tracks, performs contests etc. Thus, the problem we aim to solve differs from most of the work done in terms of music recommendation, and some of the challenges are unique. For instance, our system is scalable and it allows users to listen to music without being registered, and as a result our test dataset contains only 4266 registered users with a recorded listening history. The dataset contains 2206 radio stations with 24,803 non-zero entries in the user–station matrix; it is small and sparse, so standard recommendation techniques (e.g., SVD that we compare our algorithms to) fail to achieve good prediction quality.

3. Online service FMhost.me

3.1. A concise online broadcasting dictionary

We begin by briefly introducing some basic domain terminology. A *chart* is a track rating of a particular radio station; for example, a rock chart shows a certain number (e.g., 10) of most popular rock tracks, ranked from the most popular (rank 1) to the least popular (rank 10) according to a survey. A *live performance* (or just *live* for short) is a performance with one or several *DJs* (*disk jockeys*) assigned to it. They perform using their own PCs, and the audio stream is being redirected from them to an Icecast server and then distributed to the users. The DJs may also have their own blog for each live, where people can interact with DJs who perform live. *LiquidSoap* is a sound generator that broadcasts audio files (∗.mp3, ∗.aac etc.) into an audio stream, and *Icecast* is a retranslation server that redirects an audio stream from one source (e.g., LiquidSoap) to many receivers.

3.2. The FMhost project

FMhost is an interactive radio network. The portal allows users to listen and broadcast their own radio stations. There are four user categories in the portal: (1) unauthorized user, (2) listener, (3) Disk Jockey (DJ), and (4) radio station owner.

User capabilities vary with their status. Unauthorized listeners can listen to any station but cannot vote or become DJs and cannot use the recommender system and the rating system. Listeners can vote for tracks, lives, and radio stations. They are allowed to use the recommender system and the rating system, and they can subscribe to lives, radio stations, or DJs. They also can be appointed to a live and become a DJ.

There are three types of broadcasting: (1) stream redirection from another server, (2) AutoDJ translation, and (3) live performance. Stream redirection applies when a radio station owner has a separate dedicated server, uses *FMhost* as a broadcasting platform, and broadcasts with her own sound generator, e.g., SamBroadcaster,[5] LiquidSoap[6] etc. AutoDJ is a special option that allows the users to play music directly from the *FMhost* server. Every radio owner gets some space where she can upload tracks, and then LiquidSoap generates the audio stream and the Icecast server[7] redirects it to the listeners. Usually the owner sets a schedule for his radio station. Live performances are done by DJs. Everyone who has performed live at least once can be called a DJ. He or she can also be added to a radio station crew. Moreover, a DJ can perform lives at any station, not only on the station where he or she is included in the crew.

FMhost was the first project of its kind in Russia, starting in 2009. Following *FMhost*'s success, several radio broadcasting portals emerged, including http://myradio24.com/ and http://www.radio-hoster.ru/. In late 2011, *FMhost* was taken down for a major redesign of both codebase and recommender system architecture. In this paper, we describe the results of this upgrade.

The previous version of the recommender system experienced several problems, including tag discrepancy and personal tracks without tags at all. A survey conducted by *FMhost* with about a hundred respondents showed that more than half of them appreciated the previous version of our recommender system, and more than 80% of the answers were either positive or neutral (see Table 1 in Ignatov, Konstantinov, Nikolenko, Poelmans, & Zaharchuk, 2012); nevertheless, the new recommender model and algorithms provide even better recommendations and make even fewer prediction mistakes.

4. Models, algorithms, and recommender architecture

4.1. Input data and general structure

Our model is based on three data matrices.[8] The first matrix $A = (a_{ut})$ tracks the number of times user u visits radio stations with a certain tag t. Each radio station r broadcasts audio tracks with a certain set of tags T_r. The sets of all users, radio stations, and tags are denoted by U, R, and T respectively. The second matrix $B = (b_{rt})$ contains how many tracks with a tag t a radio station r has played. Finally, the third matrix $C = (c_{ur})$ contains the number of times a user u has visited a radio station r. For each of these three matrices, we denote by v^A, v^B, and v^C the respective vectors containing sums of elements: $v^A = \sum_{t\in T} a_{ut}$, $v^B = \sum_{t\in T} b_{rt}$, and $v^C = \sum_{r\in R} a_{ur}$. We also introduce for each matrix A, B, C the corresponding frequency of visits matrices A_f, B_f, and C_f; the frequency matrix is a result of normalization the matrix with the respective vector of visits, e.g., $A_f = a_{ut} \cdot (v^A_u)^{-1}$. Our model is dynamic: the matrices A, B, and C change after a user u visits a radio station r with a tag t, i.e., each value a_{ut}, b_{rt}, and c_{ur} is incremented by 1 after this visit.

The model consists of three main blocks: Individual-based recommender system (IBRS) model, collaborative-based recommender system (CBRS) model, and fusion recommender system (FRS) that aggregates the results of the first two. Each model has its own algorithmic implementation.

4.2. IBRS

The **IBRS** model uses matrices A_f and B_f and aims to provide a particular user $u_0 \in U$ with top N recommendations represented mathematically by a special structure $Top_N(u)$. Formally, $Top_N(u_0)$ is a triple $(R_{u_0}, \preceq_{u_0}, \text{score})$, where R_{u_0} is a set of at most N radio stations recommended to a particular user u_0, $\preceq_{u_0}$ is a well-defined preordering (reflexive, transitive, and complete) on the set R_{u_0}, and score is a function which maps each radio station r from R_{u_o} to $[0, 1]$.

The algorithm computes the l_1-norm (Manhattan) distance between a user u_0 and a radio station r: $d(u_0, r) = \sum_{t\in T} |a_{u_0 t} - b_{rt}|$. Then distances between the user u_0 and all radio stations $r \in R$ are computed, and the algorithm constructs the relation $\prec_{u_0}$ according to the following rule: $r_i \quad r_j$ iff $d(u_0, r_j) \leq d(u_0, r_i)$. The function

[4] http://host.fm/en/.

[5] http://spacial.com/sam-broadcaster.

[6] http://savonet.sourceforge.net/.

[7] http://www.icecast.org/.

[8] The data used for the experimentation is available by the following link: http://bit.ly/hostfm.

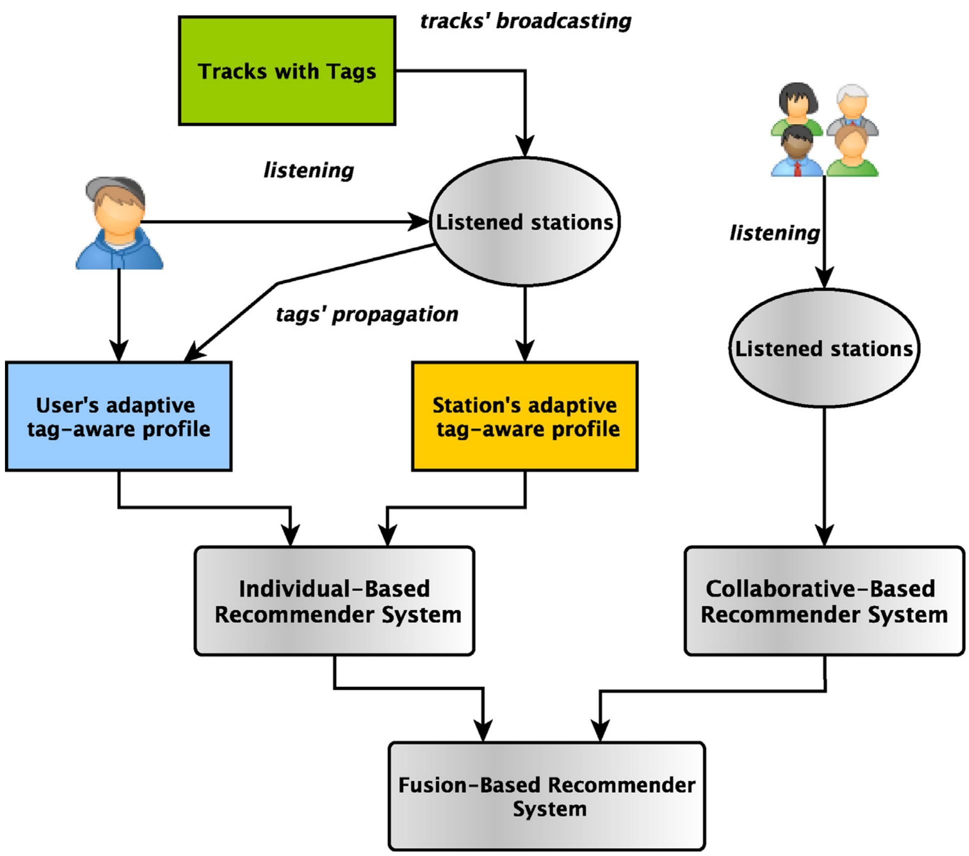
Tracks with Tags
tracks' broadcasting
listening
Listened stations
tags' propagation
listening
Listened stations
User's adaptive tag-aware profile
Station's adaptive tag-aware profile
Individual-Based Recommender System
Collaborative-Based Recommender System
Fusion-Based Recommender System

Fig. 1. The recommender system architecture.

score operates on R_{u_0} according to the following rule:

$$\mathrm{score}(r_i) = 1 - \frac{d(u_0, r_i)}{\max_{r_j \in R} d(u_0, r_j)}.$$

Finally, after selecting N radio stations with N largest values in R_{u_0}, we obtain a ranked list $Top_N(u_0)$ of radio stations recommended to the user u_0. In case there are several elements with the same score (rank) so that $Top_N(u)$ is not uniquely defined, we simply choose the first elements according to some arbitrary ordering (e.g., lexicographically by their names).

As shown on Fig. 1, our simplified model takes into account only "listened tracks" but the previously proposed one (Ignatov, Konstantinov et al., 2012) also deals with "liked tracks", "liked radio stations", and "favorite radio stations".

4.3. CBRS

The **CBRS** model is based on the C_f matrix (normalised number of times a user u has visited a radio station r). This matrix also yields a vector n^C which stores the total number of stations listened by each user $u \in U$. This vector also changes over time, and this value is used as a threshold to transform matrix C_f into a similarity matrix D via cosine similarity between users i and j:

$$sim(i, j) = \frac{c_{fi} \cdot c_{fj}}{\sqrt{\left(\sum_{r \in R} c_{fir}^2\right)\left(\sum_{r \in R} c_{fjr}^2\right)}}$$

After computing D, the algorithm constructs the list $Top_k(u_0) = (U_{u_0}, \preceq_{u_0}, \mathrm{sim})$ of k users similar to the target user u_0 who awaits recommendations. We define the set of all radio stations user u_0 has listened to by $L(u_0) = \{r | c_{fu_0r} \neq 0\}$. In a similar way, we define

$$Top_N(u_0) = (R_{u_0}, \preceq_{u_0}, \mathrm{score}), \text{ where } \mathrm{score}(r) = \max_{u \in U_{u_0}} \mathrm{sim}(u) \cdot c_{fur}.$$

Another variant is to choose the same user (for all $R \backslash L(u_0)$), who has the highest product of similarity and frequency for a certain radio station.

$$Top_N(u_0) = (R_{u_0}, \preceq_{u_0}, \mathrm{score}),$$
$$\text{where } \mathrm{score}(r) = \mathrm{sim}(u^*) \cdot c_{fu^*r}$$
$$\text{and } u^* = \arg \max_{u \in U_{u_0}, r \in R/L(u_0)} \mathrm{sim}(u) \cdot c_{fur}.$$

The weighted frequency by similarity with top-k similar user is also one of the conventional choices.

Note that the variable score takes values in the range of [0, 1]. The problem of choosing exactly the N best stations is solved in the same way as in the IBRS submodel.

4.4. FRS

After IBRS and CBRS have produced recommendations, we have two ranked lists of recommended stations $Top_N^I(u_0)$ and $Top_N^C(u_0)$ for a target user u_0 from IBRS and CBRS respectively. The **FRS** submodel proposes a simple solution for aggregating these lists into the final recommendation structure $Top_N^F(u_0) = (R_{u_0}^F, \preceq_{u_0}^F, \mathrm{score}^F)$. For every $r \in R_{u_0}^C \cup R_{u_0}^I$, the function $\mathrm{score}^F(r)$ maps r to the weighted sum

$$\beta \cdot \mathrm{score}^C(r) + (1 - \beta) \cdot \mathrm{score}^I(r),$$

where $\beta \in [0,1]$, $\mathrm{score}^C(r) = 0$ for all $r \notin R^C$, and $\mathrm{score}^I(r) = 0$ for all $r \notin R^I$. The algorithm adds the best N radio stations according to this criterion to the set $R_{u_0}^F$.

Table 1
A small example with host.fm data.

A

	t_1	t_2	t_3	t_4
u_1	20	80		
u_2	60	40		
u_3	5	5	50	40
u_4			30	70

B

	t_1	t_2	t_3	t_4
r_1	300	600	100	
r_2	700	300		
r_3			600	400
r_4			200	800

C

	r_1	r_2	r_3	r_4
u_1	10			
u_2	5	5		
u_3	1		6	3
u_4			4	6

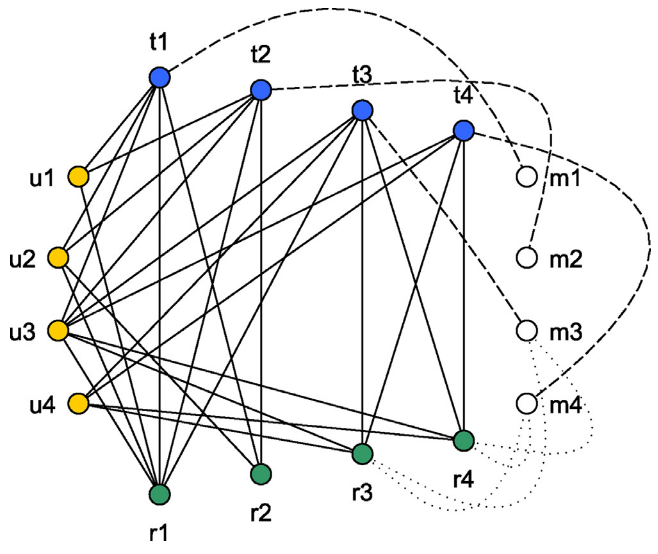

Fig. 2. A small example of host.fm radiograph.

5. How it works: explaining by example

In fact, we deal with a dynamic quadri-partite weighted graph

$$\Gamma = (U \sqcup T \sqcup R \sqcup M; E_{UT} \sqcup E_{RT} \sqcup E_{UR} \sqcup E_{RM} \sqcup E_{UM} \sqcup E_{MT}; w),$$

where U is a set of users, T is a set of tags, R is a set of radio stations, M is a set of musical tracks, $E_{UT} \subseteq U \times T$ (similarly for the rest edges), and w is a weight function that assigns to an edge e ot the graph its weight$w(e) \in \mathbb{R}_+$.

Note that this graph can be built from a collection of tuples

$$(u, t, r, m, \text{time\_stamp}),$$

where each tuple means that user u listened to music track m with tag t played by radio station r at time *time_stamp*.

A small example of this graph is shown in Fig. 2. Here, the edge weights are omitted, dotted edges show which music tracks a particular radio station plays, and dashed edges show which tags a music track has (some dotted and dashed edges are omitted).

In fact, in our model we need to explicitly operate with only E_{UT}, E_{RT}, and E_{UR}, which are defined by matrices A, B, and C respectively. However, through played tracks their tags propagate to users and radio stations, and thus the edges E_{RM}, E_{UM}, and E_{MT} are also implicitly used.

Consider a user u_1 who actually listened to only one radio station r_1 (Table 1). Let us recommend radio stations to u_1 by means of the proposed approach.

Let $\beta = 0.5$. The corresponding normalising vectors are

$$v^A = (1/100, 1/100, 1/100, 1/100),$$
$$v^B = (1/1000, 1/1000, 1/1000, 1/1000),$$
$$v^C = (1/10, 1/10, 1/10, 1/10).$$

First of all, we need to calculate distances to all the other users:

$$d(u_1, r_1) = |0.2 - 0.3| + |0.8 - 0.6| + |0 - 0.1| + |0 - 0| = 0.4,$$
$$d(u_1, r_2) = |0.2 - 0.7| + |0.8 - 0.3| + |0 - 0| + |0 - 0| = 1,$$
$$d(u_1, r_3) = |0.2 - 0| + |0.8 - 0| + |0 - 0.6| + |0 - 0.4| = 2,$$
$$d(u_1, r_4) = |0.2 - 0| + |0.8 - 0| + |0 - 0.2| + |0 - 0.8| = 2.$$

The resulting individual scores are

$$\text{score}^I(u_1, r_1) = 1 - 0.4/2 = 0.8,$$
$$\text{score}^I(u_1, r_2) = 1 - 1/2 = 0.5,$$
$$\text{score}^I(u_1, r_3) = 1 - 2/2 = 0,$$
$$\text{score}^I(u_1, r_4) = 1 - 2/2 = 0.$$

The cosine similarity of u_1 with the other users in the CBRS part is as follows:

$$sim(u_1, u_2) = \frac{10 \cdot 5}{\sqrt{10^2 \cdot (5^2 + 5^2)}} = 1/\sqrt{2} \approx 0.71,$$
$$sim(u_1, u_3) = \frac{10 \cdot 1}{\sqrt{10^2 \cdot (1^2 + 6^2 + 3^2)}} = 1/\sqrt{46} \approx 0.15,$$
$$sim(u_1, u_4) = \frac{10 \cdot 0}{\sqrt{10^2 \cdot (4^2 + 6^2)}} = 0.$$

The stations u_1 listened to are $L(u_1) = \{r_1\}$. The resulting collaborative scores are

$$\text{score}^C(r_2) = 0.71 \cdot 0.5 \approx 0.35,$$
$$\text{score}^C(r_3) = 0.15 \cdot 0.6 = 0.09,$$
$$\text{score}^C(r_4) = 0.15 \cdot 0.3 = 0.045.$$

Finally, we aggregate the scores:

$$\text{score}^F(r_2) = 0.5 \cdot 0.5 + 0.5 \cdot 0.35 = 0.425,$$
$$\text{score}^F(r_3) = 0.5 \cdot 0 + 0.5 \cdot 0.09 = 0.045,$$
$$\text{score}^F(r_4) = 0.5 \cdot 0 + 0.5 \cdot 0.045 = 0.0225.$$

The final ordering is $r_2 \preceq^F_{u_0} r_3 \preceq^F_{u_0} r_4$. It is beneficial to use the graded ordered set $Top^F_N(u_0) = (R^F_{u_0}, \preceq^F_{u_0}, \text{score}^F)$ (in the sense of Birkhoff, 1967) for detailed recommendation list exploration since one can take into account not only the order but the score values.

If radio station r_4 starts playing a track m with tag t_2 and user u_3 is listening to that station at the moment, the profiles of r_4 and u_3 become (0, 1, 200, 800) and (5, 6, 50, 40), respectively.

6. Discussion

6.1. Cold start

The cold start problem is a fundamental challenge in recommender systems domain and it touches both new users and items (Lika, Kolomvatsos, & Hadjiefthymiades, 2014).

If a new radio station starts broadcasting, CBRS component cannot help. However, since it plays some tracks with thematic tags, IBRS is able to match user profiles with the repertoire of this new station via adaptive tag-aware representation. For the CBRS component to cope with the problem of a new radio station, one can also use matrix imputation (Ocepek, Rugelj, & Bosnić, 2015).

In case of a new user the system cannot help immediately, but after some listening experience it may improve its recommendations by both IBRS and CBRS components. Before obtaining a sufficiently large user history, we suggest providing the user with text-based query search or an interactive thematic radio station catalog (taxonomy).

6.2. The evolution of tastes and repertoire

When user preferences change, it should be captured first via the IBRS component, since now tracks with different tags should become more important than previous ones. When a user changes radio stations he or she listens to, it is also accounted for by both IBRS and CBRS.

Changes of radio repertoire are captured by the IBRS component since other tags become more important as newly played tracks dominate the radio station output.

One may experiment with the forgetting parameter meaning that there is a discounting (forgetting) mechanism for tracks played reasonably far in the past.

6.3. Gray sheep feeding

The gray sheep problem is peculiar to similarity-based collaborative recommenders (Ghazanfar & Prügel-Bennett, 2014). Since the input rating matrix is sparse, a particular user may not be similar enough with the others to reveal his/her preferences. Since our FBRS has a collaborative component, CBRS, in such cases we need to rely on individual user preferences expressed by the adaptive tag-aware profiling. That is why we suggest to tune the β parameter to discover whether a user has unique individual preferences or shows conformistic behaviour.

6.4. Novelty

Another important issue for recommender systems is how to provide a user with accurate and novel recommendation at the same time (Lee & Lee, 2015). In our application, novelty is directly expressed by the presence of new radio stations in the list of recommendations. Since every radio station has tag-aware adaptive profile and contributions of the tags are normalised by their playcounts, a new radio station is not discriminated against and has the same chances to be shown among top-k station in a target user's profile (the β parameter should also be properly tuned here, perhaps with a higher contribution of the IBRS component).

Note that not all novelty-enhancing approaches can be used in our radio recommender since we use only implicit feedback; for instance, we cannot use positively co-rated items (Lee & Lee, 2015).

6.5. Explainability

Explaining the recommendations is also an important part of user satisfaction and loyalty (Hernando, JesúsBobadilla, Ortega, & Gutiérrez, 2013); tag-based explanation has been established among the most successful (Gedikli, Jannach, & Ge, 2014). In our system, IBRS component may directly benefit from the tag-based profiling. For example, a particular radio station in the final recommendation list can be annotated by its tag cloud: the more contribution a given tag gives to final score, the bigger it is displayed. Since in the CBRS component we use a user with maximal product of similarity and listening frequency, we can also annotate recommendations, e.g., "radio station r has been recommended since user u, which is most similar to you, frequently listens to it."

Table 2
Sample genre and epoch tagging from http://grooveshark.im/tag/Music.

80s:	Best of the 80's, Dexys Midnight Runners, Rick Astley
90s:	Paula Cole, Will Smith, R.E.M.
Classic Rock:	Creedence Clearwater Revival, Bad Company, Boston
Classical:	Johann Sebastian Bach, Claude Debussy, Mozart
Country:	Kenny Chesney, Brad Paisley, Toby Keith
Dance:	Kesha, Lady Gaga, Cobra Starship
Electronic:	MSTRKRFT, Danger, The Prodigy
Hip Hop:	Frank T, Organismen, 7 Notas 7 Colores
Indie Folk:	Edward Sharpe & The Magnetic Zeros, Noah and the Whale
Indie Rock:	The Bravery, Bloc Party, Arctic Monkeys
Jazz:	Charlie Haden, Louis Armstrong, Charlie Parker
Pop Rock:	Red Hot Chili Peppers, James Blunt, Pink
R&B:	Keyshia Cole, Trey Songz, Monica
Rap:	Duo Kie, Young Jeezy, 50 Cent
Reggae:	Mato Seco, Chala Rasta, Riddim

6.6. Quality of tags

The crucial factor for learning relevant dynamic profiles of users and radios is the tag quality. Many existing music resources use tags and even provide APIs to get tags for different tracks and albums or find similar ones.[9] The usual way is to use music genres or time periods as tags (Table 2).

Another tendency is to use mood-based and context-oriented tagging (Deng, Wang, Li, & Xu, 2015), such as "groovy", "cool", "lost in thought", "calm", "winter", "homework", "happy", "beautiful", "aggressive", "romantic", "untroubled", "magic", "road trip" and so on.[10] The singer or band name is also a possible tag.

A user may be subjective while tagging, there may be spelling mistakes, tag polysemia, different valid ways to write multiword tags, tags can be too specific, and so on. Hence, radio hosting owners should take care about moderated tag sources, and all tracks they play should have at least one tag shared by other tracks (i.e., not exceedingly rare).

It is important to note that even if a new song has been mistakenly assigned a "Rap" tag instead of "Reggae", it will not have too much influence on the radio station profile since the station broadcasts many other songs, most likely tagged correctly. If a user listens only to this specific song with the wrong tag, the user's profile might be biased, but again, it is virtually impossible to listen to the same song broadcasted by a single radio station all day long. Another mistake might happen when a music track is assigned by the artist name, say "Madonna", instead of the genre, "Pop". However, again since a user listens to radio stations that continuously play many songs tagged "Pop", this mistake will also correct itself.

7. Quality of service assessment

7.1. Setting and quality metrics

To evaluate the quality of the developed system, we used a variant of the cross-validation technique proposed in Ignatov, Poelmans, Dedene, and Viaene (2012). We represent the dataset as an object-attribute table (binary relation) $T \subseteq U \times I$, where uTi iff user $u \in U$ used (purchased, watched, listened etc.) item $i \in I$. To evaluate the quality of recommendations in terms of precision and recall, we split the initial user set U into training and test subsets U_{train} and U_{test}, where the test set is smaller than the training set, e.g., with a 20/80 proportion. Recommendation precision and recall are evaluated on the test set, and this part of the algorithm is similar to one step of conventional cross-validation. Then each user vector u from U_{test} is divided into two parts which consist of

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

[10] Examples are taken from http://www.stereomood.com/.

Table 3
Basic parameters of the user and radio visits datasets, with power-law fits and the corresponding p-value.

Dataset	n	$\langle x\rangle$	σ	x_{max}	$\hat{x}_{min}$	$\hat{\alpha}$	n_{tail}	p-value
User dataset	4187	5.86	12.9	191	12 ± 2	2.46(0.096)	117	**0.099**
Radio dataset	2209	11.22	60.05	1817	46 ± 11	2.37(0.22)	849	**0.629**

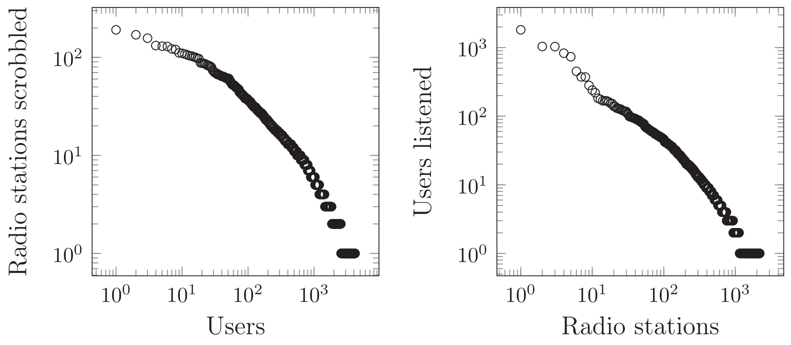

Fig. 3. Power law at FMHost. On the left, users sorted by the number of radio stations they have listened to. On the right, radio stations sorted by the number of their users. Both graphs (on log scale) show power law dependencies.

evaluated items I_{visible} and items I_{hidden} which we have intentionally hidden. Note that in the existing literature, the proportion between the size of I_{visible} and I_{hidden} is not discussed even in similar schemes (Symeonidis, Nanopoulos, Papadopoulos, & Manolopoulos, 2008a). Then, for example, a user-based algorithm makes recommendations according to similarity between users from the test and training sets. Each user from U_{test} gets recommendations as a set of fixed size $r_n(u) = \{i_1, i_2, \dots, i_n\}$. Precision and recall are defined as

$$\text{Precision} = \frac{|r_n(u) \cap u^I \cap I_{\text{hidden}}|}{|r_n(u) \cap I_{\text{hidden}}|}, \text{Recall} \quad = \frac{|r_n(u) \cap u^I \cap I_{\text{hidden}}|}{|u^I \cap I_{\text{hidden}}|},$$

where u^I is the set of all items from I used by u. These measures are calculated for each user and then averaged. The experiment can be repeated several times, e.g. 100, for different test and training set splits, and then the values are averaged again. In addition, one can select the set I_{hidden} at random, but with a specific proportion, e.g. 20%. The idea behind the method comes from traditional cross-validation, but in the case of recommender systems some modifications are needed. We used a modified version of m-fold cross-validation, which is performed by splitting the initial set into m disjoint subsets, where each subset is used as a test set and the other subsets are considered as training sets. To evaluate ranking quality based on the number of performed listenings we use the normalized discounted cumulative gain (NDCG) measure.

Before we proceed to the detailed description of the procedure, we discuss some important aspects of the *FMHost* data that we have mined.

7.2. Basic statistics

The dataset contains the following entities: users, tags, radio stations, and tracks. We have selected users with at least one tag in the profile, and then all the tags related to the selected users. At the next step, we have chosen radio stations that the selected users have listened to or with selected tags in the radio station profiles. Finally, we have chosen tracks related to at least one of the selected radio stations and to at least one tag. The resulting dataset contains 4266 users, 3618 tags, 2209 radio stations, and 4165 tracks. The corresponding matrices have the following number of nonempty entries: 38,504 in the user–tag matrix, 18,539 in the radio station–tag, 24,803 in the user–radio station, 18,781 in the track–tag, and 22,525 in the radio station–track matrix.

It is well known that many statistics of complex networks often follow the power law distribution (Clauset, Shalizi, & Newman, 2009). In order to choose which number of active users or radio stations we have to take into account for making recommendations, we performed a simple statistical analysis of the user and radio station activity. Around 20% of the users (only registered ones) were analyzed.

Table 3 shows p-values of statistical tests performed with the *Matlab* package introduced in Clauset et al. (2009). It shows that the power law does fit the radio station dataset, and the probability to make an error by ruling out the null hypothesis (no power law) is about 0.1 for the user dataset. Thus, the radio station visits dataset is more likely to follow the power law than the user visits dataset, but we should take it into account for both datasets; Fig. 3 shows how the power law actually fits our data.

This analysis implies useful consequences according to the well-known “80:20” rule $W = P^{(\alpha-2)/(\alpha-1)}$, which means that the fraction W of the wealth is in the hands of the richest P of the population. In our case, 50% of users make 80% of all radio station visits, and 50% of radio stations have 83% of all visits (which is actually a rather flat distribution compared to most services). Thus, if the service tends to take into account only active stations and users, it can cover 80% of all visits by considering 50% of their active audience. However, new radio stations still deserve to be recommended, so this rule can only be applied to the user database.

7.3. Quality assessment

For quality assessment in the IBRS subsystem, we count average precision and recall on the set $R_N \subset R$, where N is the number of randomly “hidden” radio stations. We assume that for every station $r \in R_N$ and every user $u \in U$ the algorithm does not know whether the radio stations were visited, and we change A_f and R accordingly. Then IBRS attempts to recommend the Top-N radio stations for the modified matrix A_f. Top-N average precision and recall are computed as follows:

$$\text{Precision} = \frac{1}{|U|} \sum_{u \in U} \frac{|R_u^I \cap L_u \cap R_N|}{|L_u \cap R_u^I|},$$

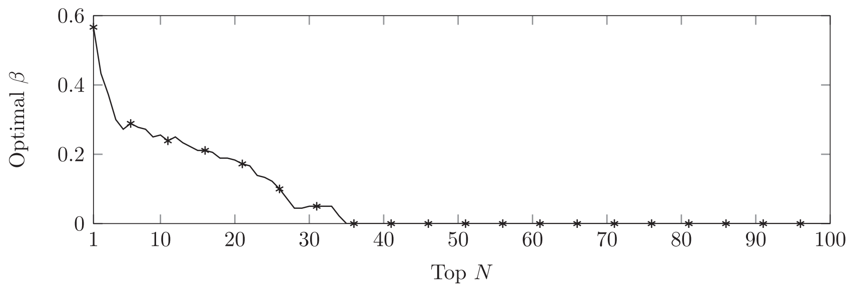

Fig. 4. Tuning the parameter β to maximize *F*-measure.

$$\text{Recall} = \frac{1}{|U|}\sum_{u\in U}\frac{|R_u^I \cap L_u \cap R_N|}{|L_u \cap R_N|}.$$

To deal with CBRS, we use a modification of the leave-one-out technique. At each step of the procedure for a particular user u, we "hide" all radio stations $r \in R_N$ by setting $c_{fur} = 0$. Then we perform the CBRS algorithm assuming that $c_{fu'r}$ is unchanged for $u' \in U/u$ and then compute

$$\text{Precision} = \frac{1}{|U|}\sum_{u\in U}\frac{|R_u^C \cap L_u \cap R_N|}{|L_u \cap R_u^C|},$$

$$\text{Recall} = \frac{1}{|U|}\sum_{u\in U}\frac{|R_u^C \cap L_u \cap R_N|}{|L_u \cap R_N|}.$$

To tune the FRS system, we can use a combination of these two procedures, looking for optimal β as

$$\beta^* = \arg\max_{\beta} \frac{2\cdot \text{Precision}\cdot \text{Recall}}{(\text{Precision} + \text{Recall})}.$$

After launching the system, we project to be able to collect enough reliable statistics in about one month of active operation in order to tune β and choose appropriate similarity and distance measures and thresholds. We suppose that the resulting system will provide reasonably accurate recommendations using only a single (last) month of user history and only 50% of the most active users. For quality assessment during the actual operation, we will compute Top-*N* precision and recall measures as well as NDCG. In addition, online surveys can be launched to assess user satisfaction with the new RS system.

7.4. Experimental results

The IBRS algorithm uses the user–tag matrix *A* and the radio station–tag matrix *B*. As a result, we have a matrix of predicted scores that contains recommendations of radio stations for users. Fig. 7 shows the precision, recall, and NDCG measure of IBRS versus the size of the recommendation list. For the first recommended item, precision is about 30% and then it rapidly drops as Top-*N* grows to 5–10 elements. Then it becomes close to 1% while *N* goes to 100. Recall for the first recommended radio station is 50%, and then the recall value slowly increases as Top-*N* grows. NDCG slowly increases while Top-*N* grows. Our *Matlab* implementation of IBRS runs for about 80s for all users.

The CBRS algorithm uses the user–radio station matrix *C*. To evaluate its quality, we used 3 × 3-fold cross-validation which we described in the beginning of Section 7. Therefore, as a result we have nine different partitions of the initial data into training and test set, and all measures are averaged by all partitions. Fig. 7 shows that CBRS precision is smaller than IBRS precision for small Top-*N* size: for the first recommended radio station it is about 7%. However, for large Top-*N* size it goes down to 2% ($N = 100$), which is better than IBRS precision. In terms of recall CBRS also loses: recall of the first recommended radio station is about 10%, and then for higher values of *N* it becomes closer to IBRS recall. NDCG of CBRS is strictly less than the NDCG of IBRS, which shows that IBRS is a better ranker than CBRS. The testing time for the entire cross-validation procedure is about 50 min.

The hybrid recommender system FRS uses output matrices with scores of recommended radio stations for IBRS and CBRS. To make the final ranking, it employs a weighted sum of the IBRS and CBRS output matrices. Parameter β is tuned for each Top-*N* size. To this end, for β from 0 to 1 with step 0.05 we have calculated the resulting weighted matrix; for this matrix, we calculate precision, recall, the F-measure, and NDCG. The procedure is repeated for *N* ranging from 1 to 100. The value of β is then tuned to maximize one of the measures. By maximizing the *F*-measure, we improve the general quality of the output but nearly ignore the ranking. Maximization of NDCG takes into account the ranking. The time needed to tune β in the range from 0 to 1 with step 0.05 in our implementation is 200s. It takes up to 3s to calculate the final recommendation matrix with the chosen β.

When we maximize the *F*-measure with respect to β, for $N = 1$ IBRS has a slightly higher weight, about 0.57, but as Top-*N* grows to 30–35 radio stations, the parameter β smoothly decreases, thus increasing CBRS's contribution; for $N > 35$ β drops to 0 (see Fig. 4). These results can be explained by the fact that IBRS provides higher precision and recall than CBRS, but as *N* grows the *F*-measure of CBRS is getting higher than for IBRS. As a result, the *F*-measure for FRS is not less than the *F*-measure of either IBRS or CBRS individually, and FRS performs efficiently for every tested recommender list size. In particular, for *N* in the order of 15–20 radio stations the *F*-measure of FRS is 2–3% higher than the *F*-measure of the best basic method (IBRS). For a larger size of the recommendation list, *N*, the FRS *F*-measure is close to CBRS.

NDCG maximisation by the parameter β prefers IBRS at first: the parameter begins with 0.6. But then in the range of *N* from 2 to 10 radio stations β decreases to 0.4 and stabilizes, oscillating for larger values of *N* around 0.40–0.41. This implies that CBRS contributes slightly more to the final recommendation (see Fig. 5). When we tune β to maximize NDCG, the weighted sum approach performed better than IBRS (which is better than CBRS with respect to NDCG) by 4–5% for all tested output sizes.

For comparison, we have also implemented a standard collaborative filtering model based on singular value decomposition (SVD) (Bell & Koren, 2007a; 2007b; Koren, 2008; Koren & Bell, 2011; Koren et al., 2009). In this model, an item's rating is approximated as a scalar product of user and item feature vectors (plus certain baseline predictors). In our setting, lacking explicit ratings, we have applied SVD to the matrix of user listenings to radio stations. Due to the power law dependencies we have discovered (see Section 7.2), we have used the logarithm of the number of listenings (plus one) in the model:

$$\log(\text{listen}_{u,r} + 1) \sim \mu + b_u + b_r + v_u^\top v_r,$$

where $\text{listen}_{u,r}$ is the number of times user *u* listened to radio station *r*, μ is the general mean, b_u and b_r are the baseline predictors

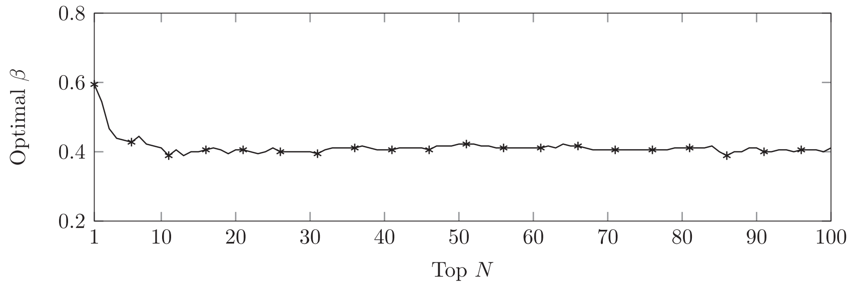

Fig. 5. Tuning the parameter β to maximize NDCG.

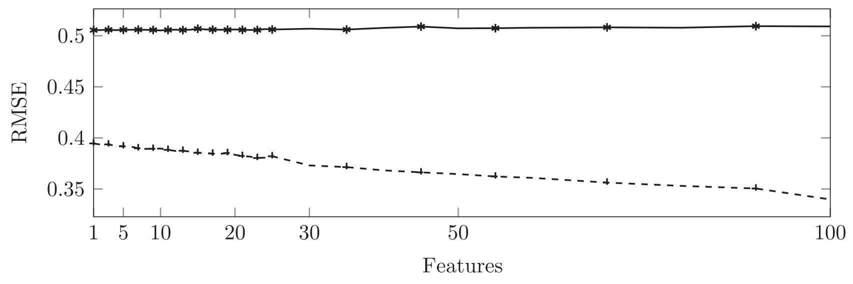

Fig. 6. Root mean squared error in SVD experiments as a function of the number of features. Solid line denotes error on the validation set; dashed line, error on the training set.

for the user u and the station r respectively, and v_u and v_r are the vectors of the user and station features respectively.

It was clear that there is not enough data and the matrix is too sparse for SVD. This was also supported by our experiments: we did not see any improvement at all as the number of SVD features grew, the quality on the validation set was stable all the way from a single feature to about 100 and then started showing clear signs of overfitting; this is depicted in Fig. 6. Results of the main experiments also supported this observation: in Fig. 7, SVD clearly loses to the methods proposed in this work.

8. Theoretical discussion

Linear combination of rankings seems to be a simple solution for their aggregation. However, so far this effective and efficient way of aggregation has been considered only as a well-chosen heuristic without proper theoretical discussion of its properties (Celma, 2010; Deng et al., 2015; Domingues et al., 2013; Kim & Kim, 2014; Wu, Chang, & Liu, 2014). We try to bridge the gap by order-theoretic treatment.

First of all we assume that every ranking of radio stations is represented by a graded linear order, $Top_N(u) = (R_u, \preceq_u, \text{score})$. We consider two rankings $Top^A_N(u) = (R^A_u, \preceq^A_u, \text{score}^A)$ and $Top^B_N(u) = (R^B_u, \preceq^B_u, \text{score}^B)$, and perfect ranking $Top^I_N(u) = (R^I_u, \preceq^I_u, \text{score}^I)$; in what follows $R[a\colon b]$ denotes the subranking of R from rank a to rank b (including both). We aim at recovering the perfect ranking (to some extent) based on the two initial ones. We do not discuss the nature of each of these two given initial ranking, but we usually assume that $(R^A_u \cup R^B_u) \cap R^I_u = R^I_u$. The resulting linear combination of rankings is $Top^C_I(u) = (R^C_u, \preceq^C_u, \text{score}^C)$, where score of $r \in R^C_u$ is equal to $\alpha \cdot score^A(r) + (1-\alpha) \cdot score^B(r)$ and $\alpha \in [0;\ 1]$; if $r \notin R^A_u$ its $score^A(r) = 0$ (similarly for R^B_u). For simplicity we assume that for every input ranking, the score value for the topmost element of R_u is N, its lower neighbor has score value $N-1$, and so on till the bottom element with its score value 1. Since we deal with radio recommender system, the size of final ranking for a user is usually small (N is about 5 or 10). The ranker is called *weak with respect to a set of items* if less than half of its elements are not among the items of perfect ranking, otherwise the ranker is called *bad with respect to set of items*.

The more elements R^C_u and R^I_u have in common and more common pairs from $\preceq^C_u$ and $\preceq^I_u$, the better ranking $Top^C_I(u)$ is. Thus, if we recover the perfect ranking, the corresponding recall, precision and NDCG measures are all equal to 1.

Property 1 (Two irrelevant tails)**.** Let $Top^A_N(u) = (R^A_u, \preceq^A_u, \text{score}^A)$ and $Top^B_N(u) = (R^B_u, \preceq^B_u, \text{score}^B)$ be weak rankings, and suppose that elements from R^A_u and R^B_u with score less than $\frac{N}{2}$:

- are not in R^I_u: $R^A_u[1 : \frac{N}{2}] \cap R^I_u = R^B_u[1 : \frac{N}{2}] \cap R^I_u = \emptyset$;
- are different in R^A_u and R^B_u: $R^A_u[1 : \frac{N}{2}] \cap R^B_u[1 : \frac{N}{2}] = \emptyset$.

Then $Top^C_N(u)$ satisfies $R^C_u = R^I_u$ for all α in the interval $(\frac{N}{2(N+1)}; \frac{N+2}{2(N+1)})$.

Proof. W.l.o.g. assume that N is even, maximal score is equal to N, and minimal is equal to 1. Then to select only the relevant items from R^A_u and R^B_u, one needs to keep their score higher than the score of all the irrelevant items, getting the following system:

$$\begin{cases} \alpha\left(\dfrac{N}{2}+1\right) > (1-\alpha)\dfrac{N}{2} \\ (1-\alpha)\left(\dfrac{N}{2}+1\right) > \alpha\dfrac{N}{2} \end{cases}$$

The solution is $\alpha \in (\frac{N}{2(N+1)}; \frac{N+2}{2(N+1)})$. □

Property 2 (Bad item)**.** *Suppose that two rankings* $Top^A_N(u) = (R^A_u, \preceq^A_u, \text{score}^A)$ *and* $Top^B_N(u) = (R^B_u, \preceq^B_u, \text{score}^B)$ *have the same items as* R^I_u *except perhaps one "bad" element in each ranking. Then:*

(1) if the "bad" element is present in only one ranking, say A, at position p: $R^B_u = R^I_u$, $R^A_u \setminus R^I_u = \{r\}$, $\text{score}^A(r) = p$, then for any $\alpha \in [0, \frac{1}{p+1})$ $R^C_u = R^I_u$;

(2) if the same "bad" element is present in both rankings: $R^A_u \setminus R^I_u = R^B_u \setminus R^I_u = \{r\}$, it is impossible to find α that will achieve $R^C_u = R^I_u$ for all possible rankings; for specific R^A_u and R^B_u, all such α must satisfy

$$\alpha\text{score}^A(r) + (1-\alpha)\text{score}^B(r) < \min_{r' \in R^I_u}\{\alpha\text{score}^A(r')$$
$$+(1-\alpha)\text{score}^B(r')\};$$

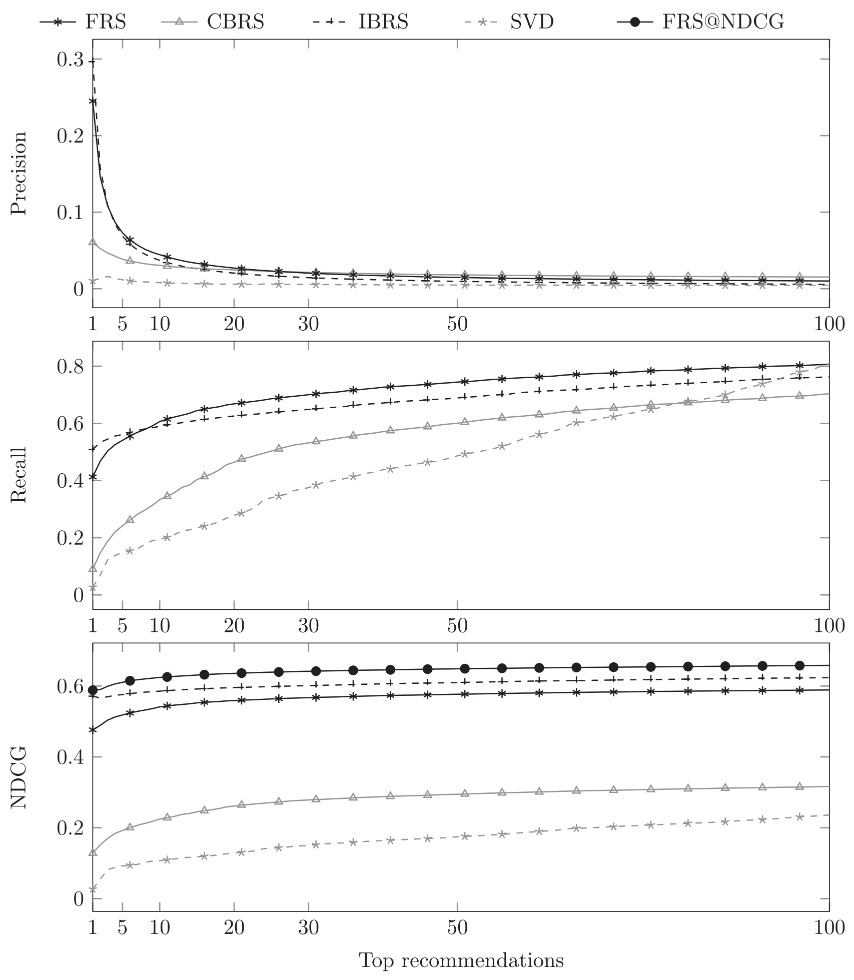

Fig. 7. Experimental results for four algorithms: IBRS, CBRS, FRS, and SVD. Graphs, top to bottom: precision, recall, and NDCG as a function of the number of top recommendations predicted for a user (averaged over all users).

(3) if "bad" elements in the two rankings are different: $R_u^A \setminus R_u^I = \{r\}$, $\mathrm{score}^A(r) = p$, $R_u^B \setminus R_u^I = \{r'\}$, $\mathrm{score}^B(r') = q$, it is impossible to find α that will achieve $R_u^C = R_u^I$ for all possible rankings; for specific R_u^A and R_u^B, all such α must satisfy

$$\alpha p < \min_{r' \in R_u^I}\{\alpha \mathrm{score}^A(r') + (1-\alpha)\mathrm{score}^B(r')\} \text{ and}$$

$$(1-\alpha) q < \min_{r' \in R_u^I}\{\alpha \mathrm{score}^A(r') + (1-\alpha)\mathrm{score}^B(r')\}.$$

Proof.

(1) To achieve $i \notin R_u^I$ we have to keep $\mathrm{score}^C(i)$ smaller than the resulting score of every relevant item, i.e., we need to keep $\alpha p < 1 - \alpha$.
(2) Again, we have to keep $\mathrm{score}^C(i)$ smaller than the resulting score of every relevant item. Obviously, for some rankings this condition will not be possible to achieve, e.g., when p takes first places in both A and B.
(3) Similar to (1), we have to keep $\alpha p < 1 - \alpha$ to remove the first bad element and at the same time $(1-\alpha)q < \alpha$ to remove the second bad element. Hence, in order for such α to exist even in case when unique relevant elements in A and B are in last place, it must hold that $\frac{q}{q+1} < \frac{1}{p+1}$, i.e., $qp < 1$, a contradiction. □

Property 3 (One irrelevant tail)**.** Let $Top_N^A(u) = (R_u^A, \preceq_u^A, \mathrm{score}^A)$ be a ranking such that all elements $r \in R_u^A$ with score $\mathrm{score}^A(r) > q$ are in R_u^I, and suppose that $Top_N^B(u) = (R_u^B, \preceq_u^B, \mathrm{score}^B)$ has all correct elements, i.e., $R_u^B = R_u^I$. Then $Top_N^C(u)$ satisfies $R_u^C = R_u^I$ for all $\alpha \in [0, \frac{1}{q+1})$.

Proof. Every irrelevant item with score $\leq q$ should be eliminated, i.e., its score in the linear combination should be less than the lowest possible score of elements from B: $\alpha q < 1 - \alpha$. □

In particular, for $\alpha \in [0, \frac{1}{N+1})R_u^A$ becomes completely irrelevant, i.e., $Top_N^C = Top_N^B$.

Corollary 1 (Bad item and irrelevant counterpart tail)**.** *Let $Top_N^A(u) = (R_u^A, \preceq_u^A, \mathrm{score}^A)$ be a ranking with one irrelevant item r: $r \notin R_u^I$, $\mathrm{score}^A(r) = p$, and let $Top_N^B(u) = (R_u^B, \preceq_u^B, \mathrm{score}^B)$ be a ranking such that all elements $r' \in R_u^A$ with score $\mathrm{score}^A(r') > q$ are in R_u^I. Then, if $1 - \frac{1}{q} < \frac{1}{p}$ then for any $\alpha \in (1 - \frac{1}{q}, \frac{1}{p})Top_N^C(u)$ satisfies $R_u^C = R_u^I$.*

Proof. Follows immediately from Properties 2 and 3: it should hold that $\alpha < \frac{1}{p}$ and $(1-\alpha)q < 1$ at the same time. □

Our ranking aggregation task is different from the one which is studied in choice theory. In choice theory several rankings are usually aggregated to fulfill special conditions for implementing fair voting scheme. In our setting the aim is to recover perfect ranking having several others. The simple way of aggregation through linear combination which minimises square difference between scoring (ranking) functions have lead us to an optimisation task which seems to be a special case of isotonic regression (Best & Chakravarti, 1990).

Problem 1 (Ranking recovery problem)**.** Ranking recovery problem for linear combination of two rankings for a particular user u is formulated as follows:

$$\arg\min_{\alpha\in[0,1]} \sum_{r\in R_u} \left(\alpha \cdot score^A(r) + (1-\alpha)\cdot score^B(r) - score^I(r)\right)^2,$$

where

$score^C(r) \le score^C(r')$ whenever $r \preceq^I_u r'$, and

$score^C(r)$ and $score^I(r)$ are the scoring functions of the combined $Top^C_N(u) = (R_u, \preceq^C_u, \text{score}^C)$ the perfect ranking $Top^I_N(u) = (R_u, \preceq^I_u, \text{score}^I)$, respectively.

Corollary 2 (Relaxed ranking recovery problem)**.** *For the relaxed version of Problem 1 below*

$$\arg\min_{\alpha\in\mathbb{R}} \sum_{r\in R_u} \left(\alpha \cdot score^A(r) + (1-\alpha)\cdot score^B(r) - score^I(r)\right)^2$$

there is a unique solution

$$\alpha = \frac{\sum_{r\in R_u}(score^B(r) - score^I(r))(score^A(r) - score^B(r))}{\sum_{r\in R_u}(score^A(r) - score^B(r))^2}$$

whenever $score^A(r) \neq score^B(r)$ *for all* $r \in R_u$.

Note that for equal rankings (scoring functions), the parameter α can be chosen arbitrarily.

9. Conclusion and further work

In this work, we have described the underlying models, algorithms, and system architecture of the new improved *FMHost* service and tested it on the available real dataset. We hope that the developed algorithms will help a user to find relevant radio stations for listening. During future optimization and tuning of the algorithm, special attention should be paid to scalability issues and user-centric quality assessment.

By using bimodal cross-validation, we have built a hybrid algorithm FRS tuned to maximize either F-measure or NDCG for various values of N and β. The FRS algorithm performs better than the three other approaches, namely IBRS, CBRS, and SVD, both in terms of F-measure and in terms of NDCG.

According to the NDCG@n (NDCG at n) measure, IBRS is strictly better than CBRS, so the former one is a better ranker. The maximal value of F-measure is obtained for the Top-N of size 15. Note that tuning of β by F-measure and NDCG results in different optimal values. Thus, in the bottom plot of Fig. 7 the NDCG curve for FRS (optimised by NDCG) is strictly higher than the remaining ones.

Surprisingly, in our experiments the state-of-the-art SVD-based technique performed poorly in comparison to our proposed algorithms. This can be explained by the small size and sparseness of our dataset. We hypothesize that the methods described in this work will suit datasets with similar properties. Matrix factorization techniques remain a reasonable tool to increase scalability, but they have to be carefully adapted and assessed, taking into account the folksonomic nature of the track tags and rather disappointing results of the SVD-based technique.

For future research, we would like to mention four possible directions.

1. One important issue is connected to the relational nature of the data (users, radio stations or tracks, and tags), which constitutes the so called *folksonomy* (Vander Wal, 2007), a fundamental data structure in resource-sharing systems with tags. As shown in Ignatov, Gnatyshak, Kuznetsov, and Mirkin (2015); Ignatov, Kuznetsov, Poelmans, and Zhukov (2013); Nanopoulos et al. (2010), this data can be successfully mined by means of triclustering to find homogeneous groups of objects (users, items, or tags). So, we also plan to build a tag-based recommender system by those means. In our task, this online multi-modal clustering and tensor factorisation can be applied to the whole dynamic hypergraph or tensor.
2. Linear combination for information fusion and recommenders' aggregation is a rather simple scheme. Research on other aggregation schemes and their theoretical properties (like robustness), e.g., usage of functional form of the scoring function or consensus schemes coming from choice theory (Domenach & Tayari, 2013), is definitely needed.
3. Useful information can come from different sources of available contextual information (Panniello, Tuzhilin, & Gorgoglione, 2014) and be exploited for online preference learning (Fürnkranz, Hüllermeier, Cheng, & Park, 2012; Siddiqui, Tiakas, Symeonidis, Spiliopoulou, & Manolopoulos, 2015).
4. A reasonable complement to adaptive tag-aware profiling to capture dynamic users' and radios' behavior in music-oriented recommender could be reinforcement learning (King & Imbrasaite, 2015). However, it should be turned first from user-music matching approach to bimodal user-radio station matching where played music is only a mediator that helps to capture user and radio similarity dynamically.

Acknowledgments

We would like to thank Rustam Tagiew and Mykola Pechenizkiy for their comments and Vasily Zaharchuk (FMhost co-founder), Rimma Ahmetsafina, Natalia Konstantinova and Andrey Konstantinov for their very important work at the previous stages of this project. Our research was done at the National Research University Higher School of Economics in 2014–2015, at the Laboratory of Intelligent Systems and Structural Analysis (Moscow) and Laboratory for Internet Studies (St. Petersburg). Dmitry Ignatov's work was supported by the Russian Foundation for Basic Research (grant # 13-07-00504); Sergey Nikolenko's work was supported by the Basic Research Program of Higher School of Economics, 2014–2015.

References

Aizenberg, N., Koren, Y., & Somekh, O. (2012). In A. Mille, F. L. Gandon, J. Misselis, M. Rabinovich, & S. Staab (Eds.), *Build your own music recommender by modeling Internet radio streams, WWW* (pp. 1–10). ACM.

Avesani, P., Massa, P., Nori, M., & Susi, A. (2002). Collaborative radio community. In *Proceedings of the second international conference on adaptive hypermedia and adaptive web-based systems, AH '02* (pp. 462–465). London, UK: Springer-Verlag.

Balkema, W., & van der Heijden, F. (2010). Music playlist generation by assimilating gmms into soms. *Pattern Recognition Letters, 31*, 1396–1402.

Bell, R. M., & Koren, Y. (2007a). Lessons from the Netflix Prize challenge. *SIGKDD Explorations, 9*(2), 75–79.

Bell, R. M., & Koren, Y. (2007b). Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Proceedings of the 7th IEEE international conference on data mining* (pp. 43–52). Omaha, Nebraska, USA: IEEE Computer Society.

Bergstra, J., Casagrande, N., Erhan, D., Eck, D., & Kégl, B. (2006). Aggregate features and AdaBoost for music classification. *Machine Learning, 65*(2–3), 473–484. doi:10.1007/s10994-006-9019-7.

Best, M., & Chakravarti, N. (1990). Active set algorithms for isotonic regression; a unifying framework. *Mathematical Programming, 47*(1–3), 425–439. doi:10.1007/BF01580873.

Birkhoff, G. (1967). *Lattice theory* (3). Providence, R.I.: American Mathematical Society.

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems, 46*(0), 109–132. http://dx.doi.org/10.1016/j.knosys.2013.03.012.

Bogdanov, D., Haro, M., Fuhrmann, F., Xambó, A., Gómez, E., & Herrera, P. (2013). Semantic audio content-based music recommendation and visualization based on user preference examples. *Information Processing and Management, 49*(1), 13–33.

Bogdanov, D., & Herrera, P. (2011). *How much metadata do we need in music recommendation? A subjective evaluation using preference sets* (pp. 97–102). ISMIR.

Brandenburg, K., Dittmar, C., Gruhne, M., Abeßer, J., Lukashevich, H., Dunker, P., et al. (2009). Music search and recommendation. In B. Furht (Ed.), *Handbook of multimedia for digital entertainment and arts* (pp. 349–384). Springer US.

Castelluccio, M. (2006). The music genome project. *Strategic Finance, 88*(6), 57–58.

Celma, Ò. (2010). *Music recommendation and discovery – The long tail, long fail, and long play in the digital music space*. Springer.

Celma, O., & Lamere, P. (2011). Music recommendation and discovery revisited. In *Proceedings of the fifth ACM conference on recommender systems, RecSys'11* (pp. 7–8). New York, NY, USA: ACM. doi:10.1145/2043932.2043936.

Clauset, A., Shalizi, C. R., & Newman, M. E. J. (2009). Power-law distributions in empirical data. *SIAM Review, 51*(4), 661–703. doi:10.1137/070710111.

Deng, S., Wang, D., Li, X., & Xu, G. (2015). Exploring user emotion in microblogs for music recommendation. *Expert Systems With Applications, 42*(23), 9284–9293. doi:10.1016/j.eswa.2015.08.029.

Domenach, F., & Tayari, A. (2013). Implications of axiomatic consensus properties. In B. Lausen, D. Van den Poel, & A. Ultsch (Eds.), *Algorithms from and for nature and life*. In *Studies in Classification, Data Analysis, and Knowledge Organization* (pp. 59–67). Springer International Publishing. doi:10.1007/978-3-319-00035-0_5.

Domingues, M. A., Gouyon, F., Jorge, A. M., Leal, J. P., Vinagre, J., Lemos, L., et al. (2013). Combining usage and content in an online recommendation system for music in the long tail. *International Journal of Multimedia Information Retrieval, 2*, 3–13.

Eck, D., Lamere, P., Bertin-Mahieux, T., & Green, S. (2007). Automatic generation of social tags for music recommendation. . In J. C. Platt, D. Koller, Y. Singer, & S. T. Roweis (Eds.), *Advances in neural information processing systems*. Curran Associates, Inc.

Font, F., Serrà, J., & Serra, X. (2012). Folksonomy-based tag recommendation for online audio clip sharing. . In F. Gouyon, P. Herrera, L. G. Martins, & M. Müller (Eds.), *Proceedings of the 13th international society for music information retrieval conference* (pp. 73–78). FEUP Edições.

Fürnkranz, J., Hüllermeier, E., Cheng, W., & Park, S. (2012). Preference-based reinforcement learning: a formal framework and a policy iteration algorithm. *Machine Learning, 89*(1–2), 123–156. doi:10.1007/s10994-012-5313-8.

Gedikli, F., Jannach, D., & Ge, M. (2014). How should I explain? A comparison of different explanation types for recommender systems. *International Journal of Human-Computer Studies, 72*, 367–382.

Ghazanfar, M. A., & Prügel-Bennett, A. (2014). Leveraging clustering approaches to solve the gray-sheep users problem in recommender systems. *Expert Systems with Applications, 41*, 3261–3275.

Gleich, D. F., Zhukov, L., Rasmussen, M., & Lang, K. (2005). The world of music: SDP embedding of high dimensional data. In *Information visualization 2005*.Interactive Poster.

Grant, M., Ekanayake, A., & Turnbull, D. (2013). Meuse: Recommending Internet radio stations. In *Proceedings of the 14th international society for music information retrieval conference, ISMIR 2013, Curitiba, Brazil, November 4–8, 2013* (pp. 281–286).

Hernando, A., JesúsBobadilla, Ortega, F., & Gutiérrez, A. (2013). Trees for explaining recommendations made through collaborative filtering. *Information Sciences, 239*, 1–17.

Hilliges, O., Holzer, P., Klüber, R., & Butz, A. (2006). Audioradar: A metaphorical visualization for the navigation of large music collections. In A. Butz, B. D. Fisher, A. Krüger, & P. Olivier (Eds.), *Smart graphics*. In *Lecture Notes in Computer Science: Vol. 4073* (pp. 82–92). Springer.

Hu, Y., & Ogihara, M. (2011). *Nextone player: A music recommendation system based on user behavior* (pp. 103–108). ISMIR.

Hyung, Z., Lee, K., & Lee, K. (2014). Music recommendation using text analysis on song requests to radio stations. *Expert Systems with Applications, 41*, 2608–2618.

Ignatov, D. I., Gnatyshak, D. V., Kuznetsov, S. O., & Mirkin, B. G. (2015). Triadic formal concept analysis and triclustering: Searching for optimal patterns. *Machine Learning, 101*(1–3), 271–302. doi:10.1007/s10994-015-5487-y.

Ignatov, D. I., Konstantinov, A. V., Nikolenko, S. I., Poelmans, J., & Zaharchuk, V. (2012). Online recommender system for radio station hosting. In N. Aseeva, E. Babkin, & O. Kozyrev (Eds.), *Bir*. In *Lecture Notes in Business Information Processing: Vol. 128* (pp. 1–12). Springer.

Ignatov, D. I., Kuznetsov, S. O., Poelmans, J., & Zhukov, L. E. (2013). Can triconcepts become triclusters? *International Journal of General Systems, 42*(6), 572–593.

Ignatov, D. I., Poelmans, J., Dedene, G., & Viaene, S. (2012). A new cross-validation technique to evaluate quality of recommender systems. In M. K. Kundu, S. Mitra, D. Mazumdar, & S. K. Pal (Eds.), *PerMIn*. In *LNCS: Vol. 7143* (pp. 195–202). Springer.

Joyce, J. (2006). Pandora and the music genome project. *Scientific Computing, 23*(10), 40–41.

Kaminskas, M., & Ricci, F. (2012). Contextual music information retrieval and recommendation: State of the art and challenges. *Computer Science Review, 6*, 89–119.

Kim, H., & Kim, H.-J. (2014). A framework for tag-aware recommender systems. *Expert Systems with Applications, 41*(8), 4000–4009. http://dx.doi.org/10.1016/j.eswa.2013.12.019.

King, J., & Imbrasaite, V. (2015). Generating music playlists with hierarchical clustering and q-learning. In *Advances in information retrieval – 37th European conference on IR research, ECIR 2015, Vienna, Austria, March 29–April 2, 2015. Proceedings* (pp. 315–326). doi:10.1007/978-3-319-16354-3_34.

Koenigstein, N., Dror, G., & Koren, Y. (2011). Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In *Proceedings of the fifth ACM conference on recommender systems, RecSys'11* (pp. 165–172). New York, NY, USA: ACM. doi:10.1145/2043932.2043964.

Konstan, J. A., & Riedl, J. (2012). Recommender systems: from algorithms to user experience. *User Modeling and User-Adapted Interaction, 22*(1–2), 101–123. doi:10.1007/s11257-011-9112-x.

Koren, Y. (2008). Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 426–434). Las Vegas, Nevada, USA: IEEE Computer Society.

Koren, Y. (2010). Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data, 4*(1), 1:1–1:24. doi:10.1145/1644873.1644874.

Koren, Y., & Bell, R. M. (2011). Advances in collaborative filtering. In F. Ricci, L. Rokach, B. Shapira, & P. B. Kantor (Eds.), *Recommender systems handbook* (pp. 145–186). Springer.

Koren, Y., Bell, R. M., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *IEEE Computer, 42*(8), 30–37.

Lee, K., & Lee, K. (2015). Escaping your comfort zone: A graph-based recommender system for finding novel recommendations among relevant items. *Expert Systems with Applications, 42*(10), 4851–4858. http://dx.doi.org/10.1016/j.eswa.2014.07.024.

Lee, S., Kahng, M., & Lee, S.-g. (2015). Constructing compact and effective graphs for recommender systems via node and edge aggregations. *Expert Systems with Applications, 42*(7), 3396–3409. http://dx.doi.org/10.1016/j.eswa.2014.11.062.

Li, Y., Hu, J., Zhai, C., & Chen, Y. (2010). Improving one-class collaborative filtering by incorporating rich user information. In *Proceedings of the 19th ACM international conference on information and knowledge management* (pp. 959–968).

Lika, B., Kolomvatsos, K., & Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems. *Expert Systems with Applications, 41*, 2065–2073.

Liu, N.-H., Hsieh, S.-J., & Tsai, C.-F. (2010). An intelligent music playlist generator based on the time parameter with artificial neural networks. *Expert Systems with Applications, 37*, 2815–2825.

Lü, L., Medo, M., Yeung, C. H., Zhang, Y.-C., Zhang, Z.-K., & Zhou, T. (2012). Recommender systems. *Physics Reports, 519*(1), 1–49. http://dx.doi.org/10.1016/j.physrep.2012.02.006.

Ma, H., Yang, H., King, I., & Lyu, M. R. (2009). Semi-nonnegative matrix factorization with global statistical consistency in collaborative filtering. In *Proceedings of the 18th ACM international conference on information and knowledge management* (pp. 767–776).

McFee, B., Bertin-Mahieux, T., Ellis, D. P. W., & Lanckriet, G. R. G. (2012). The million song dataset challenge. In WWW (Companion Volume) (pp. 909–916).

Mocholí, J. A., Martinez, V., Martínez, J. J., & Catalá, A. (2012). A multicriteria ant colony algorithm for generating music playlists. *Expert Systems with Applications, 39*(3), 2270–2278. doi:10.1016/j.eswa.2011.07.131.

Müller, M., & Wiering, F. (Eds.) (2015). Proceedings of the 16th international society for music information retrieval conference, ISMIR 2015, Málaga, Spain, October 26--30, 2015.

Nanopoulos, A., Rafailidis, D., Symeonidis, P., & Manolopoulos, Y. (2010). Musicbox: Personalized music recommendation based on cubic analysis of social tags. *IEEE Transactions on Audio, Speech & Language Processing, 18*(2), 407–412.

Ocepek, U., Rugelj, J., & Bosnić, Z. (2015). Improving matrix factorization recommendations for examples in cold start. *Expert Systems with Applications, 42*(19), 6784–6794. http://dx.doi.org/10.1016/j.eswa.2015.04.071.

Pan, R., & Scholz, M. (2009). Mind the gaps: Weighting the unknown in large-scale one-class collaborative filtering. In *Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining, KDD'09* (pp. 667–676). New York, NY, USA: ACM. doi:10.1145/1557019.1557094.

Pan, R., Zhou, Y., Cao, B., Liu, N., Lukose, R., Scholz, M., et al. (2008). One-class collaborative filtering. In *Proceedings of the 8th IEEE international conference on data mining* (pp. 502–511).

Panniello, U., Tuzhilin, A., & Gorgoglione, M. (2014). Comparing context-aware recommender systems in terms of accuracy and diversity. *User Modeling and User-Adapted Interaction, 24*(1–2), 35–65. doi:10.1007/s11257-012-9135-y.

Resnick, P., Iacovou, N., Sushak, M., Bergstrom, P., & Riedl, J. T. (1994). Grouplens: An open architecture for collaborative filtering of netnews. In *1994 ACM conference on computer supported collaborative work conference* (pp. 175–186). Chapel Hill, NC: Association of Computing Machinery, Association of Computing Machinery.

Said, A., Jain, B. J., Kille, B., & Albayrak, S. (2012). Increasing diversity through furthest neighbor-based recommendation. In *Proceedings of the WSDM'12 workshop on diversity in document retrieval (DDR'12)*.

Salakhutdinov, R., & Mnih, A. (2008a). Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *Proceedings of the 25th international conference on machine learning* (pp. 880–887).

Salakhutdinov, R., & Mnih, A. (2008b). Probabilistic matrix factorization. In *Proceedings of the 21st annual conference on neural information processing systems: Vol. 20*. Vancouver, British Columbia, Canada: Curran Associates, inc.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on world wide web* (pp. 285–295).

Siddiqui, Z. F., Tiakas, E., Symeonidis, P., Spiliopoulou, M., & Manolopoulos, Y. (2015). Learning relational user profiles and recommending items as their preferences change. *International Journal on Artificial Intelligence Tools, 24*(2). doi:10.1142/S0218213015400096.

Sindhwani, V., Bucak, S. S., Hu, J., & Mojsilovic, A. (2010). One-class matrix completion with low-density factorizations. In *Proceedings of the 10th IEEE international conference on data mining, ICDM'10* (pp. 1055–1060). Washington, DC, USA: IEEE Computer Society. doi:10.1109/ICDM.2010.164.

Symeonidis, P., Nanopoulos, A., Papadopoulos, A. N., & Manolopoulos, Y. (2008a). Nearest-biclusters collaborative filtering based on constant and coherent values. *Information Retrieval, 11*(1), 51–75.

Symeonidis, P., Ruxanda, M. M., Nanopoulos, A., & Manolopoulos, Y. (2008b). In J. P. Bello, E. Chew, & D. Turnbull (Eds.), *Ternary semantic analysis of social tags for personalized music recommendation* (pp. 219–224). ISMIR.

Vander Wal, T. (2007). Folksonomy coinage and definition. http://vanderwal.net/folksonomy.html (accessed on 12.03.2012).

Werthner, H., Zanker, M., Golbeck, J., & Semeraro, G., (Eds.) (2015). Proceedings of the 9th ACM conference on recommender systems, Recsys 2015, Vienna, Austria, September 16--20, 2015. ACM.

Wu, M.-L., Chang, C.-H., & Liu, R.-Z. (2014). Integrating content-based filtering with collaborative filtering using co-clustering with augmented matrices. *Expert Systems with Applications, 41*(6), 2754–2761. http://dx.doi.org/10.1016/j.eswa.2013.10.008.

Yang, Y.-H., Bogdanov, D., Herrera, P., & Sordo, M. (2012). Music retagging using label propagation and robust principal component analysis. In WWW (Companion Volume) (pp. 869–876).

Yuan, T., Cheng, J., Zhang, X., Liu, Q., & Lu, H. (2013). A weighted one class collaborative filtering with content topic features. In S. Li, A. El Saddik, M. Wang, T. Mei, N. Sebe, & S. Yan, et al. (Eds.), *Advances in multimedia modeling*. In *Lecture Notes in Computer Science: Vol. 7733* (pp. 417–427). Berlin, Heidelberg: Springer. doi:10.1007/978-3-642-35728-2_40.

Zhao, Z., Wang, X., Xiang, Q., Sarroff, A. M., Li, Z., & Wang, Y. (2010). Large-scale music tag recommendation with explicit multiple attributes. In *Proceedings of the international conference on multimedia, MM'10* (pp. 401–410). New York, NY, USA: ACM. doi:10.1145/1873951.1874006.

Zhou, X., Xu, Y., Li, Y., Josang, A., & Cox, C. (2012). The state-of-the-art in personalized recommender systems for social networking. *Artificial Intelligence Review, 37*(2), 119–132. doi:10.1007/s10462-011-9222-1.