IMPROVING MUSIC RECOMMENDATIONS WITH A WEIGHTED FACTORIZATION OF THE TAGGING ACTIVITY

Andreu Vall **Marcin Skowron** **Peter Knees** **Markus Schedl**
Department of Computational Perception, Johannes Kepler University, Linz, Austria
{andreu.vall, marcin.skowron, peter.knees, markus.schedl}@jku.at

ABSTRACT

Collaborative filtering systems for music recommendations are often based on implicit feedback derived from listening activity. Hybrid approaches further incorporate additional sources of information in order to improve the quality of the recommendations. In the context of a music streaming service, we present a hybrid model based on matrix factorization techniques that fuses the implicit feedback derived from the users' listening activity with the tags that users have given to musical items. In contrast to existing work, we introduce a novel approach to exploit tags by performing a weighted factorization of the tagging activity. We evaluate the model for the task of artist recommendation, using the expected percentile rank as metric, extended with confidence intervals to enable the comparison between models. Thus, our contribution is twofold: (1) we introduce a novel model that uses tags to improve music recommendations and (2) we extend the evaluation methodology to compare the performance of different recommender systems.

1. INTRODUCTION AND RELATED WORK

We provide the motivation of our work together with a review of the relevant related work, divided into three parts. First, we introduce the types of user feedback under consideration. Then, we present the family of models we use to build recommender systems. Finally, we review the evaluation methodology.

1.1 Explicit, Implicit and One-Class Feedback

The interactions between users and items provide a useful source of data to produce recommendations [16]. It is commonly accepted to distinguish between *explicit feedback* and *implicit feedback*, depending on whether the user actively provides feedback about an item or this is tracked from the user's interaction with the system [1]. Examples of explicit feedback are rating a movie, giving a "like" to a blog post, or tagging an artist, because the user actively provides an opinion. In contrast, the listening histories of users in a music streaming service are an example of implicit feedback.

The standard approach to make use of implicit feedback is to count or aggregate all the interactions for each user-item pair [5, 7, 8], yielding a user-item-count table. In structure, this is identical to an explicit feedback user-item-rating table. We henceforth refer to such data structure as *user-item interactions matrix*, regardless of the type of feedback (implicit or explicit).

In some cases a user-item interaction can express both positive and negative opinions, in other cases it only reflects positive (or active) examples. Ratings in a 1 to 5 scale conventionally range from strongly disliking an item to strongly liking it. However, tracking whether a user visited or not a website, only provides a binary feedback describing action or inaction. Binary feedback is often referred to as *one-class feedback* [12, 15, 17], and examples of it can be found both in explicit and in implicit feedback. For example, a user-item interactions matrix (be it from explicit or implicit feedback) contains intrinsically a source of one-class feedback, revealing which user-item pairs were observed and which not.

Inaction must not be confused with a negative opinion, because a user may not have interacted with an item for a variety of reasons, not necessarily because of lack of interest. Social tags also exhibit this property, and treating this correctly will be a key point of the presented model.

1.2 Matrix Factorization for Collaborative Filtering

Collaborative filtering is a widely used recommendation method which aims at recommending the most relevant items to a user based on relations learned from previous interactions between users and items [16]. The factorization of the user-item interactions matrix into latent factors matrices is a well established technique to implement collaborative recommender systems, both for explicit feedback and implicit feedback datasets [7, 10, 15]. Compared to other methods, it has the advantage of uncovering latent data structures by solving an optimization problem, instead of using problem-specific and manually-designed features.

Specific collaborative systems for implicit feedback data based on matrix factorization techniques are presented in [7, 15]. The key technique is to use appropriate weights in the low-rank approximation of the user-item interactions matrix. More specifically, even if the weighting schemes are different, both [7, 15] assign higher confidence to the

observed user-item pairs and lower (but still positive) confidence to the unobserved user-item pairs. This is important to handle the uncertainty derived from the one-class property described before. We will insist on this point later, because our improved treatment of the tagging activity will rest on the same principle.

1.3 Hybrid Recommender Systems

In collaborative filtering implementations based on matrix factorization techniques, hybrid models can be based on the simultaneous factorization of the user-item interactions matrix, together with other data for users and items [5, 13]. The motivation for that is that latent factors summarizing user and item properties should be reinforced, or better described, if other data sources related to the same users and items are involved in the optimization problem.

The tags that users assign to musical items –or other forms of textual data, like user profiles or genre annotations for the items– are an obvious example of potentially useful additional information. In this line, the research presented in [5] is a valid starting point, dealing with implicit feedback data and hybridized with user and item profiles, built on the basis of tf-idf weights calculated for each user, each item and each considered word in a dictionary.

Tagging information is an explicit source of feedback (because users actively provide it) that exhibits, at the same time, the one-class property described before; the tags assigned to musical items are only positive examples (even if the meaning of a tag is semantically negative). A particular tag may not have been applied to a musical item, but this does not imply that the tag is not suited to describe that musical item. This property of social tags is also referred to as *weak labeling* [18]. It is reasonable to assume though, that the more often a tag has been applied to a musical item, the more it should be trusted. Similarly, if a user applies a tag very often, it may be assumed that the tag is to some extent relevant for her to describe musical items. To address the uncertainty that arises from this wide range of possibilities, we propose to exploit the tagging activity with a weighted matrix factorization scheme similar to the one applied for collaborative filtering in implicit feedback datasets. Observed tags can be given higher confidence. Unobserved tags can be given lower confidence, but still positive, so that they are not ignored in the recommendation system.

1.4 Evaluation of Recommender Systems

The Netflix Prize [3] has motivated an important progress in the domain of collaborative filtering, but probably due to the specific approach considered in the challenge, research has centered on attaining maximum levels of accuracy in the prediction of ratings. However, improvements in predictive accuracy do not always translate to improved user satisfaction [14].

To make the evaluation task more similar to a real use case (although still in an off-line experiment), [9] evaluates different recommender systems on the basis of issued ranked lists of recommendations. A recommender able to rank first the relevant items should be considered better than a recommender that is not able to do so. An extension of this evaluation methodology to deal with implicit feedback datasets is proposed in [7] and applied in [8, 12]. It consists in a central tendency measure, called *expected percentile rank*, assessing how good is the recommender at identifying relevant items.

The expected percentile rank is a valid metric to measure the average behavior of a single recommender system, but in order to compare the performance of different recommender systems, considering only mean values can be inaccurate. We propose to use bootstrapping techniques to examine the distribution of the expected percentile rank and test for significant differences between models.

2. METHODOLOGY

This work is framed in the context of music streaming services in which users interact with musical items, mainly listening to music, but also through the free input of text describing them. We focus on the task of artist recommendations. The listening data is aggregated at the artist level, obtaining a user-artist-count matrix of implicit feedback. The tagging activity yields a user-artist-tag matrix of one-class feedback, processed to obtain: a user-tag-count matrix, describing how many times a user applied a tag, and an artist-tag-count matrix, describing how many times a tag was applied to an artist. The proposed model is actually flexible regarding the tagging activity data. In our experiments, we successfully use a collection of top used tags (not a complete list of all the used tags) together with weights describing the tag relevance (instead of actual counts).

2.1 Recommender System Models

We compare three recommender systems. The first is a standard collaborative filtering model for implicit feedback data. The second is a hybrid model incorporating textual data, that we modify for the specific task of using tags. Finally, we introduce a novel model, able to improve the quality of the recommendations through a weighted factorization of the tagging activity.

2.1.1 Implicit Feedback Matrix Factorization (MF)

We use the approach described in [7] to perform collaborative filtering on implicit feedback data. It consists in a weighted low-rank approximation of the user-artist-count matrix, adjusting the confidence of each user-artist pair as a function of the count. Given a system with N users and M artists, the counts for each user-artist pair are tabulated in a matrix $R \in \mathbb{N}^{N \times M}$, where users are stored row-wise and artists column-wise. A binary matrix $\widetilde{R}$ is defined, such that for each user u and each artist a

$$\widetilde{R}_{ua} = \begin{cases} 1 & \text{if } R_{ua} > 0 \\ 0 & \text{if } R_{ua} = 0 \end{cases}, \tag{1}$$

and the following weight function is defined as

$$w(\eta, x) = 1 + \eta \log(1 + x). \tag{2}$$

Other weight functions can be defined and may better suit each specific problem and distribution of the data. We choose a logarithmic relation (instead of the also common linear relation used in [5, 7, 8]) to counteract the long-tail distribution of the data, where a majority of users have a small percentage of the total observed interactions. However, the detailed optimization of this function is not within the scope of this work.

Finally, the matrix factorization consists in finding two D-rank matrices $P \in \mathbb{R}^{N\times D}$ and $Q \in \mathbb{R}^{M\times D}$ (rows are latent features for users and artists respectively) minimizing the following cost function:

$$J_{MF}(P,Q) = \sum_{ua\in R} w(\alpha, R_{ua}) \left(\widetilde{R}_{ua} - P_u Q_a^T\right)^2 + \lambda \left(\|P\|_F^2 + \|Q\|_F^2\right). \tag{3}$$

Matrix $\widetilde{R}$ is reconstructed using P and Q. $\widetilde{R}_{ua}$ 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 counts in R_{ua} according to equation (2) and it is summed over all the user-artist pairs. [1] The parameter α contributes to the weight function and is determined by grid search. A regularization term involving the Frobenius norm of P and Q is added to prevent the model from over-fitting. The regularization parameter λ is also determined by grid search.

2.1.2 Implicit Feedback Matrix Factorization with Tagging Activity (TMF)

Equation (3) is extended in [5] to incorporate textual information. We present a modification of this model to specifically deal with tags. Given a system where T tags have been used, the counts for each user-tag pair are stored in a matrix $T^U \in \mathbb{N}^{N\times T}$, where rows correspond to users and columns correspond to tags. The counts for each artist-tag pair are stored in a matrix $T^A \in \mathbb{N}^{M\times T}$, where rows correspond to artists and columns correspond to tags. The modified model factorizes together $\widetilde{R}$, T^U and T^A into three D-rank matrices $P \in \mathbb{R}^{N\times D}$, $Q \in \mathbb{R}^{M\times D}$, $X \in \mathbb{R}^{T\times D}$ (rows are latent features for users, artists and tags respectively) minimizing the following cost function:

$$\begin{aligned} J_{TMF}(P,Q,X) = &\sum_{ua\in R} w(\alpha, R_{ua}) \left(\widetilde{R}_{ua} - P_u Q_a^T\right)^2 \\ &+ \mu_1 \sum_{ut\in T^U} \left(T^U_{ut} - P_u X_t^T\right)^2 \\ &+ \mu_2 \sum_{at\in T^A} \left(T^A_{at} - Q_a X_t^T\right)^2 \\ &+ \lambda\Big(\|P\|_F^2 + \|Q\|_F^2 + \|X\|_F^2\Big). \end{aligned} \tag{4}$$

The first term is identical as in (3). The second and third terms account for the contribution of tags. X_t is the row of X corresponding to tag t. Matrices T^U and T^A are reconstructed using P, Q and X, and the squared reconstruction errors are summed over all user-tag pairs and artist-tag pairs. The parameters μ_1, μ_2 account for the contribution of each term to the cost function, and are determined by grid search. The regularization term is analogous as in (3).

This formulation modifies the one described in [5], in that it factorizes T^U and T^A using a single shared tags' factor matrix X, instead of two dedicated factor matrices. The tagging activity consists of user-artist-tag observations. Even if we use separated user-tag-count and artist-tag-count matrices as inputs for the model, the tags must be factorized in the same space of latent features.

This model factorizes the user-tag and artist-tag raw counts. If, for example, an artist-tag pair has never been observed, the model will try to fit a value of 0 counts for it. This seems an unsuited model, because we know that a tag that has not been applied may still be relevant.

2.1.3 Implicit Feedback Matrix Factorization with Weighted Tagging Activity ($WTMF$)

We introduce a novel approach to improve the hybridization with tagging activity, by using a weighted factorization scheme similar to the one used for implicit feedback data. The observed user-tag and artist-tag pairs are given high confidence and therefore have a higher contribution to the cost function. The unobserved user-tag and artist-tag pairs are given low confidence. They become less relevant in the cost function, and at the same time the model has more freedom to fit them. As the results in Section 3.3 demonstrate, this is a better approach to model the weak labeling property of social tags.

We define binary matrices $\widetilde{T}^U$ and $\widetilde{T}^A$, such that for each user u, each artist a and each tag t

$$\begin{aligned} \widetilde{T}^U_{ut} &= \begin{cases} 1 & \text{if } T^U_{ut} > 0 \\ 0 & \text{if } T^U_{ut} = 0 \end{cases} \\ \widetilde{T}^A_{at} &= \begin{cases} 1 & \text{if } T^A_{at} > 0 \\ 0 & \text{if } T^A_{at} = 0 \end{cases}. \end{aligned} \tag{5}$$

We factorize together $\widetilde{R}$, $\widetilde{T}^U$ and $\widetilde{T}^A$ 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) minimizing the following cost function:

$$\begin{aligned} J_{WTMF}(P,Q,X) = &\sum_{ua\in R} w(\alpha, R_{ua}) \left(\widetilde{R}_{ua} - P_u Q_a^T\right)^2 \\ &+ \mu_1 \sum_{ut\in T^U} w(\beta, T^U_{ut}) \left(\widetilde{T}^U_{ut} - P_u X_t^T\right)^2 \\ &+ \mu_2 \sum_{at\in T^A} w(\gamma, T^A_{at}) \left(\widetilde{T}^A_{at} - Q_a X_t^T\right)^2 \\ &+ \lambda\Big(\|P\|_F^2 + \|Q\|_F^2 + \|X\|_F^2\Big). \end{aligned} \tag{6}$$

The equation is similar to (4), but now all the terms involve a weighted factorization. Note that the second and

[1] As described in [7], this includes the zero entries of R as well.

third terms have specific weight coefficients β and γ, determined by grid search.

2.2 Parameter Estimation

Alternating Least Squares (ALS) is usually the preferred method to minimize the objective functions of models based on matrix factorization [2, 5–8, 15, 19]. ALS is an iterative method, where subsequently all but one of the factor matrices are kept fixed. This results in quadratic functions that approximate the original one. At each step, the cost value is expected to move closer to a local minimum and the process is repeated until convergence. Since the approximated functions are quadratic, the exact solution for the factors can be computed in closed form.

For each of the presented models, we provide the exact solution for the factors of each user u stored in P_u, each artist a stored in Q_a and each tag t stored in X_t. We introduce some additional notation. $R_{r_u}, R_{c_a}, T^U_{r_a}, T^U_{c_t}, T^A_{r_a}, T^A_{c_t}$ refer to the u^{th}, a^{th}, t^{th} row or column (r, c) of the corresponding matrix (R, T^U, T^A).[2] We also need to define the following matrices:

- $W^{r_u}_R \in \mathbb{R}^{M\times M}$ is a diagonal matrix with the weights computed for the u^{th} row of R in the diagonal
- $W^{c_a}_R \in \mathbb{R}^{N\times N}$ is a diagonal matrix with the weights computed for the a^{th} column of R in the diagonal
- $W^{r_u}_{T^U} \in \mathbb{R}^{T\times T}$ is a diagonal matrix with the weights computed for the u^{th} row of T^U in the diagonal
- $W^{c_t}_{T^U} \in \mathbb{R}^{N\times N}$ is a diagonal matrix with the weights computed for the t^{th} column of T^U in the diagonal
- $W^{r_a}_{T^A} \in \mathbb{R}^{T\times T}$ is a diagonal matrix with the weights computed for the a^{th} row of T^A in the diagonal
- $W^{c_t}_{T^A} \in \mathbb{R}^{M\times M}$ is a diagonal matrix with the weights computed for the t^{th} column of T^A in the diagonal

2.2.1 Solution for J_{MF}

For each user u and artist a, the latent factors are given by

$$\begin{cases} P_u = \left(Q^T W^{r_u}_R Q + \lambda I\right)^{-1} \left(Q^T W^{r_u}_R R^T_{r_u}\right) \\ Q_a = \left(P^T W^{c_a}_R P + \lambda I\right)^{-1} \left(P^T W^{c_a}_R R^T_{c_a}\right) \end{cases} \tag{7}$$

2.2.2 Solution for J_{TMF}

For each user u, artist a and tag t, the latent factors are given by

$$\begin{cases} P_u = \left(Q^T W^{r_u}_R Q + \mu_1 X^T X + \lambda I\right)^{-1} \\ \qquad\qquad \left(Q^T W^{r_u}_R R^T_{r_u} + \mu_1 X^T T^{UT}_{r_a}\right) \\ Q_a = \left(P^T W^{c_a}_R P + \mu_2 X^T X + \lambda I\right)^{-1} \\ \qquad\qquad \left(P^T W^{c_a}_R R^T_{c_a} + \mu_2 X^T T^{AT}_{r_a}\right) \\ X_t = \left(\mu_1 P^T P + \mu_2 Q^T Q + \lambda\right)^{-1} \\ \qquad\qquad \left(\mu_1 P^T T^{UT}_{c_t} + \mu_2 Q^T T^{AT}_{c_t}\right) \end{cases} \tag{8}$$

[2] T^U and T^A may be further transposed, reading T^{UT} and T^{AT}.

2.2.3 Solution for J_{WTMF}

For each user u, artist a and tag t, the latent factors are given by

$$\begin{cases} P_u = \left(Q^T W^{r_u}_R Q + \mu_1 X^T W^{r_u}_{T^U} X + \lambda I\right)^{-1} \\ \qquad\qquad \left(Q^T W^{r_u}_R R^T_{r_u} + \mu_1 X^T W^{r_u}_{T^U} T^{UT}_{r_a}\right) \\ Q_a = \left(P^T W^{c_a}_R P + \mu_2 X^T W^{r_a}_{T^A} X + \lambda I\right)^{-1} \\ \qquad\qquad \left(P^T W^{c_a}_R R^T_{c_a} + \mu_2 X^T W^{r_a}_{T^A} T^{AT}_{r_a}\right) \\ X_t = \left(\mu_1 P^T W^{c_t}_{T^U} P + \mu_2 Q^T W^{c_t}_{T^A} Q + \lambda\right)^{-1} \\ \qquad\qquad \left(\mu_1 P^T W^{c_t}_{T^U} T^{UT}_{c_t} + \mu_2 Q^T W^{c_t}_{T^A} T^{AT}_{c_t}\right) \end{cases} \tag{9}$$

2.3 Producing Recommendations

The technique employed to produce recommendations is the same for all the models. Once the factor matrices P, Q and X are learned, the user-artist preferences are predicted as $Z = PQ^T$. Note that the tags' factor matrix X is not directly involved in the prediction, although it contributed to a better estimation of P and Q. The new matrix Z is expected to be a reconstruction of $\widetilde{R}$ for the observed user-artist pairs. For unobserved entries, Z is expected to reveal potential preferences on the basis of the learned user and artist factors. The closer a predicted user-artist preference is to 1, the more confidence we have that it corresponds to an interesting artist for the user. For each user u, a recommendation list is prepared showing the artists with higher predicted preference values in Z_u.

3. EXPERIMENTAL STUDY

3.1 Dataset

We compare the different models on a dataset of Last.fm listening histories, top tags used by users and top tags applied to artists, collected through the Last.fm API.[3] The combination of the standard Taste Profile Subset[4] with the Last.fm tags dataset[5] would seem a preferable choice, but the absence of users' tagging activity makes it unsuited.

The dataset is built as a stable subset of a running crawl of Last.fm listening events. The original crawl includes only users with non-empty country information, non-empty gender information and a value in the age field between 10 and 80 years, although such filtering is actually not needed. There is no constraint on the minimum or maximum number of artists a user has listened to. However, we only include users such that at least 95% of their listened artists have a valid MusicBrainz[6] identifier, which is required to accurately crawl the artists' tagging activity. This does not bias the dataset towards popular artists, because the MusicBrainz is an open and collaborative platform, including a wide variety of artists. The users' tagging activity is fetched with the Last.fm user names.

[3] http://www.last.fm/api
[4] http://labrosa.ee.columbia.edu/millionsong/tasteprofile
[5] http://labrosa.ee.columbia.edu/millionsong/lastfm
[6] https://musicbrainz.org/

# listened artists	# users
$1-10$	64
$11-20$	84
$21-30$	122
$31-40$	77
$41-50$	96
$50-100$	466
$101-2,332$	1,993
total	2,902

Table 1: Distribution of users per number of listened artists.

The dataset includes $21,852,559$ listening events, relating to $2,902$ users and $71,223$ artists, yielding $687,833$ non-zero user-artist-count entries. This corresponds to a matrix density of roughly 0.3%. Table 1 shows the distribution of users as a function of the number of artists they listened to.

The top tags for each user (if any) are provided together with a count variable describing how many times the user applied it. The top tags applied to an artist (if any) are provided together with a percentage relative to the most frequently applied tag [11]. Because the API functions only return the top tags, we only observe a partial set of the tagging activity. In addition, although the user is presented with previously used tags, she can always input free text. To overcome these limitations, we perform regularization and simplification operations to the tag strings, namely: replacements of genre abbreviations with their extended version, spelling corrections, removal of non-alphanumeric characters and mapping of different spelling variants to a unique tag string, resulting in a unified set of tokens. After this process is applied to the fetched tags, we are left with 630 unique tags for 600 users and $12,902$ unique tags for $67,332$ artists, among which 494 unique tags are identified as identical between the user and the artist list. Note that tags were found for most of the artists, but only for 20% of the users. Probably, only a small subset of active users use the tagging functionality.

The whole matrix of user-artist counts is used, although not all users or artists have related tagging activity. Tags are a complement whenever they are available.

3.2 Evaluation Methodology

The most reliable evaluation method for a recommender system is an actual large-scale on-line experiment, where real users interact with the system [16]. This requires a complex infrastructure which, unfortunately, is not within the scope of this work. Since we only have access to historical data, we can not measure how new recommendations would be perceived by the users. Furthermore, in contrast to explicit feedback applications, accuracy metrics for predicted ratings are not meaningful for implicit feedback. Therefore, we adopt the evaluation approach proposed in [9] and adapted in [7] to deal with implicit feedback datasets in a recall-oriented setting and we additionally propose an extension to it.

The observed user-artist pairs are split into training and test sets to perform 5-fold cross validation, letting each user have approximately 80% of the listened artists in the training set and 20% in the test set. For each user-artist pair u, a assigned to the test set, a random list of artists (not including a) is drawn. The list is then ranked according to the preferences of user u, learned from the training set as explained in Section 2. Finally, a is inserted in the sorted list, and its percentile rank within the list is stored as $rank_{ua}$.[7] 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%.

After this process is done over all the splits, $rank_{ua}$ is known for all the observed user-artist pairs in the dataset. Then, following [5, 7, 8], the *expected percentile rank* is defined as the weighted average of $rank_{ua}$ with weights given by the user-artist counts:

$$\overline{rank} = \frac{\sum_{ua \in R} R_{ua} rank_{ua}}{\sum_{ua \in R} R_{ua}}. \quad (10)$$

Correctly ranking a highly relevant artist is more important than correctly ranking a less relevant artist. Likewise, failing to recommend a highly relevant artist is worse than failing to recommend a less relevant one. Values of $\overline{rank}$ close to 0% indicate that the recommender is able to correctly rank the relevant artists. Producing ranked lists uniformly at random results in an expected percentile rank of 50%. Ranking all the relevant items in the last position of the list results in an expected percentile rank of 100%.

We extend the evaluation methodology by building confidence intervals of $\overline{rank}$. This allow us to test for significant differences in the performance of models. We use basic bootstrap confidence intervals, based on the bootstrap distribution of the expected percentile rank (see [4]). For all the observed user-artist pairs in the dataset, random samples with replacement and with the same size as the dataset are drawn. For each sample of user-artist pairs, the expected percentile rank is computed. We repeat this step $1,000$ times to obtain the bootstrap distribution of $\overline{rank}$. We then build 95% confidence intervals of $\overline{rank}$ using the basic bootstrap scheme described in [4].

3.3 Model Comparison

The models are evaluated and compared for a varying number of latent factors D, and for a varying number of training iterations. On the one hand, we fix the number of iterations to 10 and evaluate the models with $5, 10, 20, 50, 100$ latent factors. On the other hand, we fix the number of factors to 10 and evaluate the models for $5, 10, 20, 50, 100$ training iterations. We choose 10 factors and 10 training

[7] Lists of any length may be prepared, and the percentile rank provides a unified scale. We use lists of 100 artists in our experiments. According to our experience, longer lists do not yield significant differences.

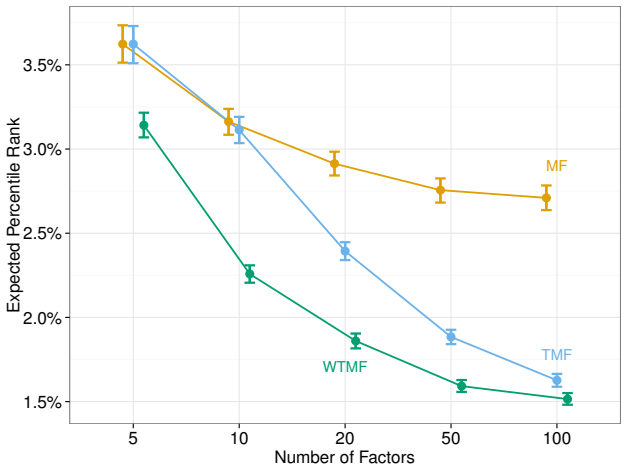

Figure 1: Model comparison for different number of latent factors. The dots correspond to $\overline{rank}$ and the error bars display 95% basic bootstrap confidence intervals. The different models are dodged to avoid overlapping. The top and center lines correspond to the baseline models. The lowest corresponds to the presented model.

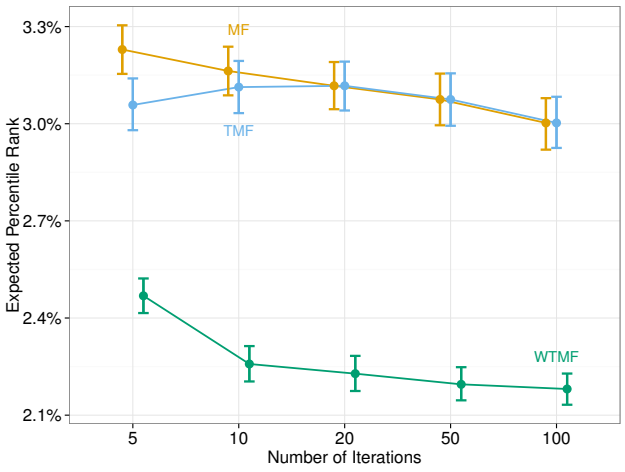

Figure 2: Model comparison for different number of training iterations. The dots correspond to $\overline{rank}$ and the error bars display 95% basic bootstrap confidence intervals. The different models are dodged to avoid overlapping. The top lines correspond to the baseline models. The lowest corresponds to the presented model.

iterations as a basic setting, because they balance well performance and computational requirements.

For each model and each combination of factors and iterations, we tune the parameters α, β, γ, μ_1, μ_2 and λ by grid search. We choose the set of values that provides lowest expected percentile rank, computed by 5-fold cross validation as described in Section 3.2. Figures 1 and 2 show the results for different number of factors and iterations respectively.

Note that all models, including the plain matrix factorization model, provide very good results, with values of expected percentile rank under 4%. This implies that, on average, the models are able to rank relevant artists among the top 4 positions of a list of 100 random artists.

The performance of TMF and WTMF improves significantly when more latent factors are used (see Figure 1). The presented model outperforms the baselines, although for 100 factors the difference between TMF and WTMF is small. We examine this case. We compute a 95% basic bootstrap confidence interval for the difference of $\overline{rank}$ and it does not include 0. We conclude that the difference in performance is still significant. For lower number of factors the differences between the presented model and the baselines are remarkable. Good performance at inexpensive computational requirements is a crucial property, especially for large-scale implementations.

Increasing the number of training iterations results in smaller improvements (see Figure 2). Our model clearly outperforms the baselines in this set of experiments too, with a difference of nearly 1% in expected percentile rank. With the basic setting of 10 factors, TMF can not fully exploit the tagging activity and performs comparably to MF. For experiments with 20 or more training iterations they perform exactly as well, because the grid search process finds that discarding the tagging activity yields best results. TMF performs slightly better than MF with 5 training iterations. The performance of TMF does not improve monotonically with more training iterations, although the model is not over-fitting. This is because after 5 iterations the cost function of TMF reaches a flat region close to a local minimum, resulting in small performance variations.

4. CONCLUSIONS AND FURTHER RESEARCH

In this paper we presented a novel model to incorporate tagging activity into implicit feedback recommender systems. Our approach proves to work better than previous hybrid models, based on experiments conducted with real data from Last.fm, a well-known music streaming service. We extended the common evaluation methodology computing basic bootstrap confidence intervals for the expected percentile rank. This allows us to test for significant differences in the performance of models.

As future work, we will evaluate the robustness of the presented model for different recommendation tasks. We are particularly interested in the task of song recommendations, but we will also experiment in fields other than music, like movies or websites. Another interesting question is the effect of the size and connectedness of the tagging data on the final quality of the recommendations. We will investigate how rich and linked together needs to be the tagging activity in order to enhance the recommendations. This could provide indications of when can the model be successfully utilized, or which kind of processing of the tag strings is required to make the tagging activity helpful.

5. ACKNOWLEDGMENTS

This research is supported by the Austrian Science Fund (FWF) under project no. P25655 and the EU FP7 through projects 601166 (PHENICX) and 610591 (GiantSteps).

6. REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, June 2005.

[2] Robert M. Bell and Yehuda Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Proc. ICDM*, pages 43–52. IEEE, 2007.

[3] James Bennett and Stan Lanning. The netflix prize. In *Proc. KDDCup*, page 35, 2007.

[4] Thomas J. DiCiccio and Bradley Efron. Bootstrap confidence intervals. *Statistical science*, pages 189–212, 1996.

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

[6] K. Ruben Gabriel and S. Zamir. Lower Rank Approximation of Matrices by Least Squares with Any Choice of Weights. *Technometrics*, 21(4):489, November 1979.

[7] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Proc. ICDM*, pages 263–272. IEEE, 2008.

[8] Christopher C. Johnson. Logistic Matrix Factorization for Implicit Feedback Data. 2014.

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

[10] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[11] M. Levy and M. Sandler. Music information retrieval using social tags and audio. *IEEE Transactions on Multimedia*, 11(3):383–395, April 2009.

[12] Yanen Li, Jia Hu, ChengXiang Zhai, and Ye Chen. Improving one-class collaborative filtering by incorporating rich user information. In *Proc. CIKM*, pages 959–968. ACM, 2010.

[13] Hao Ma, Tom Chao Zhou, Michael R. Lyu, and Irwin King. Improving recommender systems by incorporating social contextual information. *ACM Transactions on Information Systems*, 29(2):1–23, April 2011.

[14] Sean M. McNee, John Riedl, and Joseph 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.

[15] Rong Pan, Yunhong Zhou, Bin Cao, Nathan Nan Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. One-class collaborative filtering. In *Proc. ICDM*, pages 502–511. IEEE, 2008.

[16] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B Kantor, editors. *Recommender systems handbook*. Springer, 2011.

[17] Vikas Sindhwani, Serhat S. Bucak, Jianying Hu, and Aleksandra Mojsilovic. One-class matrix completion with low-density factorizations. In *Proc. ICDM*, pages 1055–1060. IEEE, 2010.

[18] Douglas Turnbull, Luke Barrington, and Gert Lanckriet. Five approaches to collecting tags for music. In *Proc. ISMIR*, volume 8, pages 225–230, 2008.

[19] Yunhong Zhou, Dennis Wilkinson, Robert Schreiber, and Rong Pan. Large-scale parallel collaborative filtering for the netflix prize. In *Algorithmic Aspects in Information and Management*, pages 337–348. Springer, 2008.