Improving Context-Aware Music Recommender Systems: Beyond the Pre-filtering Approach

Martin Pichl, Eva Zangelere and Günther Specht
Databases and Information Systems
Department of Computer Science
University of Innsbruck
{firstname.lastname}@uibk.ac.at

ABSTRACT

Over the last years, music consumption has changed fundamentally: people switch from private, mostly limited music collections to huge public music collections provided by music streaming platforms. Thus, the amount of available music has increased dramatically and music streaming platforms heavily rely on recommender systems to assist users in discovering music they like. Incorporating the context of users has been shown to improve the quality of recommendations. Previous approaches based on pre-filtering suffered from a split dataset. In this work, we present a context-aware recommender system based on factorization machines that extracts information about the user's context from the names of the user's playlists. Based on a dataset comprising 15,000 users and 1.8 million tracks we show that our proposed approach outperforms the pre-filtering approach substantially in terms of accuracy of the computed recommendations.

CCS CONCEPTS

•**Information systems** → **Personalization;** *Clustering and classification;* **Recommender systems; Music retrieval; Collaborative filtering;**

KEYWORDS

Recommender Systems, Context, Personalization, User Modeling

ACM Reference format:
Martin Pichl, Eva Zangelere and Günther Specht. 2017. Improving Context-Aware Music Recommender Systems: Beyond the Pre-filtering Approach. In *Proceedings of ICMR '17, June 6–9, 2017, Bucharest, Romania, ,* 8 pages.
DOI: http://dx.doi.org/10.1145/3078971.3078982

1 INTRODUCTION

Recently, we are facing a fundamental change in the way people consume music: more and more people switch from private, mostly limited music collections to public music streaming collections containing several millions of tracks [23]. People increasingly do not store music locally on CDs and hard drives anymore. Instead, they access millions of tracks offered by cloud-based streaming services using various devices. To increase usability, streaming platforms heavily rely on recommender systems to help users in discovering music they like. Previous research has shown that the context of a user (i.e., occasion, event or emotional state) plays an important role for providing personalized music recommendations [20, 22]. Kamalzadeh et al. [16] showed that people listen to different music during different activities and found that people organize tracks in their music collections by the intended use (i.e., working or exercising). This finding is backed up by Cunningham et al. [10], who found that people create playlists that are intended for certain activities.

Over the last years, data for quantitatively validating these studies became available: music streaming platforms provide means for "social playlist generation"—playlists that are shared among friends or to the public. Particularly public playlists serve as an essential new data source for music recommender systems. For Spotify[1], a popular music streaming service, all user-created playlists are public by default[2] and thus can be crawled using the Spotify API[3]. Pichl et al. [27] propose an approach for clustering contextually similar playlists by exploiting the names of these playlists. The clusters are then leveraged in a collaborative filtering recommender system (CF) with pre-filtering [2], hence CF is applied to each cluster individually. Thus, the recommender system is applied to different parts of the dataset in isolation, a method that has drawbacks: the user profiles are split up among the different clusters and thus, there is no holistic view on the user. In addition, recommendation accuracy substantially varies among clusters, as these are different in size.

In this work, we follow up and complement the research of Pichl et al. [27] by utilizing their proposed playlist aggregation pipeline to implement a novel recommender system to overcome the drawbacks of contextual pre-filtering. Particularly, we are interested in how contextual clusters may be leveraged for music recommendations while ensuring that the drawbacks of the pre-filtering approach can be avoided. Therefore, we propose to make use of Factorization Machines (FM) [28] that are directly able to incorporate the contextual clusters extracted from the names of playlists for the computation of recommendations.

In several empirical experiments using k-fold cross-validation we show that our proposed factorization machine-based recommender system outperforms context-agnostic recommender systems, pre-filtering context-aware recommender systems as well

ICMR '17, June 6–9, 2017, Bucharest, Romania

ACM ISBN 978-1-4503-4701-3/17/06…$15.00
DOI: http://dx.doi.org/10.1145/3078971.3078982

[1] http://www.spotify.com
[2] https://developer.spotify.com/web-api/working-with-playlists/#public-private-and-collaborative-status
[3] http://developer.spotify.com/web-api/

as classification-based context-aware recommender systems substantially in terms of recall, precision and the F_1-measure. Our experiments show that factorization machines are particularly capable of tackling the major issue of the pre-filtering approach (i.e., splitting up the dataset). To foster reproducibility and repeatability, we make both our code and data used publicly available by publishing our recommender system and the evaluation framework utilized in this paper on GitHub[4].

The remainder of this paper is structured as follows. In the next section, we focus on related work before presenting our recommendation approach in Section 3. After that, we introduce the reader to our conducted experiments aiming to benchmark different recommendation systems including our proposed recommender system. In the subsequent sections, we present the results of the experiments and discuss them in Section 5. Finally, we wrap up our work in Section 6.

2 RELATED WORK

We classify related work into two main fields of research: context-aware music recommender systems and approaches concerned with leveraging new data sources for music recommendations.

It is widely agreed upon the fact that the user's context improves personalized recommendations [2]. This is why we can see a shift from purely content- or CF-based approaches towards more user-centric approaches incorporating the user's context [33]. In the field of music recommender systems, studies showed that users often seek for music that matches their current context (i.e., occasion, event or emotional state) [20, 22]. As for the different types of contexts, Kaminskas and Ricci [18] distinguish environment-related context (location, time, weather), user-related context (activity, demographic information, emotional state of the user) and multimedia context (text or pictures the user is currently reading or looking at). Examples for contextual information that is leveraged for music recommendations are emotion and mood [4, 7, 11, 31], the user's location [3, 9, 17, 19] or recommending music fitting to documents on the web a user reads at the moment [8]. As for the integration of contextual information into a recommender system, Adomavicius et al. [2] classify approaches modeling the user's context into contextual pre-filtering, contextual post-filtering and contextual modeling approaches. We consider the approach presented in this work as a contextual modeling approach as we do not filter the input or output data of the system.

As for music recommender systems based on novel publicly available data, Zangerle et al. [36] propose a music recommender system based on association rules computed based on user listening behavior extracted from #nowplaying tweets (tweets in which users state which musical track they are listening to at the moment). Moreover, context-aware approaches for music recommendations that are based on information extracted from public data sources have been proposed. Schedl and Schnitzer exploit #nowplaying tweets enhanced with acoustic features extracted from 7digital[5] and extract context information about these tracks by utilizing a web search on the track and artist [34]. In [35], Schedl et al. explore the use of geospatial information for a set of collaborative filtering approaches. Furthermore, also LastFM has been utilized for analyzing the listening behavior of users [12, 32]. Pichl et al. [27] extract contextual information from the names of playlists of Spotify users and incorporate these in the process of recommending tracks. The work presented in this paper builds upon this approach and aims to address the problems of the pre-filtering approach (as proposed by Pichl et al.) by using factorization machines. To the best of our knowledge, this is the first factorization machine-based recommendation approach for integrating contextual clusters derived from playlist names into a music recommender system.

[4]https://github.com/dbis-uibk/MusicRecommenderEvaluator/
[5]http://www.7digital.com

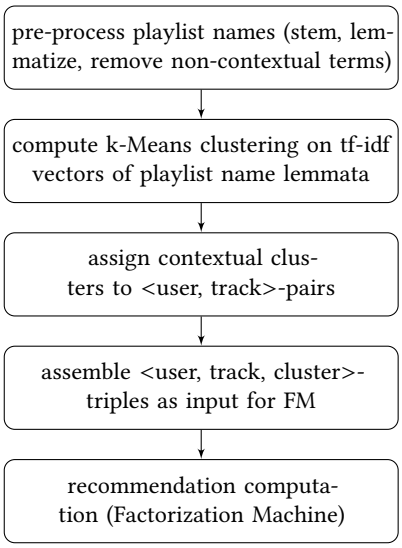


Figure 1: Pipeline for Computing Recommendations

3 METHODS

In this section, we present our proposed recommendation algorithm. First, we introduce the approach taken for computing clusters of contextually similar tracks. In a next step, we present the proposed recommendation framework, which leverages the information provided by these contextual clusters. Figure 1 depicts the overall workflow for the computation of music recommendations utilizing contextual clusters.

As the approach taken for computing contextual clusters relies on the work of Pichl et al. [27], we naturally utilize the same dataset for evaluating our approach (and comparing it to the original approach). This dataset contains 143,528 unique playlists created by 15,345 unique users who listened to 1,878,457 tracks in the form of $<user, track, artist, playlist>$-quadruples.

3.1 Playlist Aggregation and Cluster Generation

In a first step, we compute clusters of contextually similar playlists based on the context information extracted from the names of playlists. Therefore, we follow the method introduced by Pichl et al. [27], which we will shortly sketch in the following. As depicted in Figure 1, we firstly stem all playlist names and lemmatize the tokens in a first step. In a next step, we remove non-contextual terms

such as genre, artist and track names as well as general stop words, as these do not contain any contextual information. We use the resulting bags of lemmata describing each playlist to compute the term frequency-inverse document frequency (tf-idf) [15] for each bag of lemmata representing a playlist name. Using tf-idf, we represent each playlist as a vector containing the tf-idf weights. This allows us to compute playlist similarities by computing the pairwise cosine similarity of the playlist vectors. Using the computed similarities, we span a distance matrix and finally find contextually similar playlists by applying k-Means to the playlists in the matrix. As we evaluate our approach using the same dataset as Pichl et al. [27], we set the number of clusters to $k = 23$, as proposed in the original approach. In the next step, we integrate the contextual clusters in the recommendation computation as presented in the following section.

3.2 Recommendation Computation

Our proposed recommendation approach aims to provide track recommendations for a given user in a given context. Particularly, we aim to model users by the tracks they listened to and enrich this information with the contexts in which each individual user has listened to those tracks. For the given input dataset, we assume that by adding a track to a playlist, the user expresses some preference for the track. For means of simplicity, we will describe a user-track interaction extracted from a playlist as "a given user listened to a given track". Furthermore, we infer from previous findings [10, 16], that user create playlists to listen to the contained tracks in the context specified by the playlist name.

The initial input dataset contains $<user, track, playlist>$-triples. We transform this dataset into a set of $<user, track, context\_cluster>$-triples by applying the clustering method presented in Section 3.1 and assigning each user-track pair with one of the 23 contextual clusters in which the given user has listened to the given track. By adding a fourth factor *rating* to the dataset, we transform the recommendation computation task into a rating prediction task: for each unique $<user, track, context\_cluster>$-triple, the *rating* r_{ijk} is 1 if the user u_i has listened to the track t_j in cluster c_k. Our dataset does not contain any implicit feedback by users (i.e., play counts, skipping behavior, session durations or dwell times during browsing the catalog). Therefore, we cannot estimate any preferences towards an item a user not listened to as proposed by [13]. Thus, for each $<user, track, context\_cluster>$ combination for which we cannot obtain a rating for, we assume the rating to be $r = -1$ (as proposed by [13]). The rating r_{ijk} for each user u_i, track t_j and cluster c_k can now defined as stated in Equation 1. Although there is a certain bias towards negative values as some missing values might be positive, Pan et al. [26] found that this method for rating estimation works well.

$$r_{ijk} = \begin{cases} 1 & if\ u_i\ listened\ to\ t_j\ in\ c_k \\ -1 & otherwise \end{cases} \tag{1}$$

To get a better understanding of the resulting dataset, we depict a sample of the dataset in Table 1. Based on this dataset, we train a classifier that decides whether a user has listened to a track in a contextual cluster or not. For this computation, we require a given user, track and cluster as input.

As for the actual computation of recommendations, we opt for factorization machines (FM) [28, 29], as these can be considered as state-of-the-art recommendation approach and have been shown to perform well for recommender systems [30]. FMs are a generalization of factorization models and allow to model interactions of input variables in a lower-dimensional space (i.e., interactions are mapped onto a latent features-space of lower dimension). As we aim to exploit the interaction effects of users, tracks and clusters with this recommender system, we chose to utilize a FM of the order $d = 2$ modeling all single and pairwise interactions between input variables as depicted in Equation 2.

$$\hat{r}_{FM} = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{m} \sum_{j=i+1}^{m} \langle \vec{v}_i, \vec{v}_j \rangle x_i x_j \tag{2}$$

Equation 2 shows that a FM computes rating predictions by modeling a global bias (w_0), the influence of the user, track as well as the clusters ($\sum_{i=1}^{m} w_i x_i$) along with the quadratic interaction effects of those ($\sum_{i=1}^{m} \sum_{j=i+1}^{m} \langle \vec{v}_i, \vec{v}_j \rangle$). However, instead of learning all weights $w_{i,j}$ for the interaction effects, a FM relies factorization to model the interaction as the inner product of low dimensional vectors ($\langle \vec{v}_i, \vec{v}_j \rangle$) [29].

To estimate the performance of the presented recommender systems we conduct a set of experiments as described in the following section.

4 EXPERIMENTS

In this section, we introduce the experiments conducted to evaluate the proposed approach and the baseline approaches aiming at answering our research questions. We start with a description of the dataset used for the evaluation before focusing on the experimental setup and the evaluation measures.

4.1 Dataset

For our experiments, we apply the proposed clustering method on the initial dataset and reshape the input dataset into a set containing $<user, track, context\_cluster, rating>$-quadruples. We assign each track in a playlist with a rating value as described in Section 3.2. The rating indicates whether a certain user listened to a certain track in a certain cluster ($r = 1$) or not ($r = -1$). A fragment of the dataset is shown in Table 1. This excerpt shows that user 872 has listened to track 250246 in contextual cluster 0, whereas user 911 has listened to track 250246 in context 2. This dataset forms the foundation for our experiments, which are presented in the next section.

User	Track	Contextual Cluster	r
872	309275	0	1
872	309275	1	-1
911	250246	0	-1
911	250246	0	-1
911	250246	2	1

Table 1: Dataset Fragment

4.2 Baseline Recommender Systems

We compare our proposed FM approach to three baseline recommender systems: a CF-based system, a SVD-based system and a classification-based system. To incorporate context information in the CF- and SVD-based baseline approaches, we apply pre-filtering [2], where the computation of recommendations (CF or SVD) is performed on each contextual cluster individually. I.e., we compute the recommendations on a sub-dataset of the dataset restricted to a certain cluster. The classification-based system uses the computed contextual clusters as an input feature to the classifier. With those systems, we benchmark classical CF, approaches facilitating latent features (considered as state-of-the art in recent years) and a classification-based approach against our proposed factorization machine-based recommender system.

The first recommender system to benchmark is a collaborative filtering approach [1]. The idea behind CF is to recommend items the k-nearest neighbors of a user interacted with. For determining the nearest neighbors, we compute pairwise user similarities by computing the Jaccard Coefficient [14] of the set of tracks each of the two users listened to. Thus, we measure the number of commonly listened tracks in relation to the tracks both users listened to as depicted in Equation 3, where we denote S_i as the set of tracks a user i has listened to.

$$Jaccard_{i,j} = \frac{|S_i \cap S_j|}{|S_i \cup S_j|} \tag{3}$$

The second baseline recommender system is based on singular value decomposition (SVD) [21]. SVD predicts ratings by extracting a number of latent features from the user-item matrix R. In our setting, this is a sparse matrix containing all the binary ratings r_{ij} (cf. Equation 1) of all users u_i and the tracks t_j they listened to. These latent features, characterizing types of tracks, are computed by factoring the user-item matrix R into two matrices U and V, which represent the user and item factors. Hence, R is the cross product of U and V ($R = UV'$). We approximate U and V by minimizing the error to the known ratings r_{ij} using stochastic gradient descent optimization (SGD) [21].

Thirdly, we aim to compare our proposed approach with a classification-based recommendation approach as the performed recommendation computation can also be considered as a one-class classification problem [26]. Therefore, we implement a random forest classifier [24] as it has two main advantages: firstly, we only have to tune one parameter: the number of trees [25]. Secondly, all trees can be computed in parallel and the algorithm scales linearly with the number of trees.

Furthermore, we compare all recommender system to a random-choice baseline. The assumption behind this baseline is that the fundamental chances of guessing whether a track was listened by a user ($r = 1$) or not ($r = -1$) is 50%. Thus, the random baseline for the *precision* measure is 0.5. The same holds for RMSE and MAPE, where the random baseline is also 0.5. For the *recall* measure we cannot state a single baseline value, as recall is dependent on the number of recommendations n as explained in Section 4.4 and shown in Equation 8.

A detailed description of the evaluation is given in the next section.

4.3 Experimental Setup

To evaluate the performance of the different recommender systems, we conduct a 5-fold cross-validation. Therefore, we randomly split the dataset into five folds of equal size. Subsequently, we utilize four folds as training data and the remaining fold as test data. This process is repeated 5 times such that every fold serves as test data once. Due to the random selection of data for the folds, each fold contains an arbitrary number of relevant and irrelevant items. The relevant items are tracks a user has listened to within a certain cluster, whereas the latter are items a user did not listen to at all within a cluster.

For assessing the rating prediction performance of the different recommender systems, we compute the predicted rating $\hat{r}$ for each track in the current test set. Using the predicted ratings $\hat{r}$ as well as the actual ratings r in the test set, we compute the evaluation measures as described in Section 4.4. These evaluation measures are computed for each fold separately and before computing the measures, we perform a min-max scaling. For the results in Section 5, we compute the average across all folds.

For evaluating the top-n recommendations performance, we sort the result by the predicted rating $\hat{r}$ and subsequently use the top-n recommended tracks for the evaluation. We compare $\hat{r}$ to the actual rating r for the current user, track and cluster in the test set. For this comparison, we assume all track recommendations with $\hat{r} \geq 0.5$ as relevant for the user in the given context and hence, $\hat{r} = 1$.

As for the learning method utilized for the FM, we make use of Markov Chain Monte Carlo (MCMC) inference as proposed by Rendl et al. [28]. Generally, we tuned each of the recommender systems (except the random baseline), using k-fold cross-validation. For the random forest classifier, we train the random forest classifier with 1,000 trees. In preliminary experiments, we found that this is a sufficient number of trees to get stable results. Similarly, in our preliminary experiments we found that for CF, $n = 30$ and for SVD, $k = 50$ are suitable parameter options.

4.4 Evaluation Measures

In this section, we elaborate on the evaluation measures used for assessing the performance of the different recommendation algorithms.

For assessing the rating prediction task, we compute the different widely used error measures: root mean square error (RMSE) as well as the mean absolute percentage error (MAPE) as stated in Equations 4 and 5, where $\hat{r}$ is the predicted rating and r the actual rating as contained in the test set. For the results stated Table 2, we compute the average error among all ratings r_i in the test set. Please note that for computing the error measures, we scaled the predicted rating $\hat{r}$ between 0 and 1 using min-max scaling to be able to directly compare the evaluated approaches.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(r_i - \hat{r_i})^2}{n}} \tag{4}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{r_i - \hat{r_i}}{r_i}\right| \tag{5}$$

For measuring the performance of the top-n recommendations, we rely on *recall*, *precision* and the F_1-measure. For computing the

recall-measure, we have to classify the tracks in the test set into relevant and non-relevant items. We consider an item as relevant, if the user has listened to this track in a certain cluster and thus $r = 1$. An item is considered as non-relevant for a given user if the user did not listen to it in a given cluster and thus, $r = -1$. For a certain user, a track can be relevant in certain clusters and simultaneously not relevant in other cluster. In case of the FM-based recommender, we have to transform the rating prediction task into a one-class classification task [26] on whether a given track relevant or not relevant for a given user in a given context to be able to compute the top-n measures. Therefore, we consider $\hat{r}$ as 1 if the computed probability that a user interacted with an item $P(r = 1)$ is higher than 50% as stated in Equation 6. As for the ranking, we rely on the predicted rating for ranking the recommendations to be able evaluate the top-n recommendations.

$$\hat{r} = \begin{cases} 1 & if\ P(r = 1) \geq 0.5 \\ -1 & otherwise \end{cases} \tag{6}$$

In Equations 7 and 8 we state how precision (P) and recall (R) are computed. Precision measures the number of true positives (TP) in relation to the number of recommendations n, which is the number of true positives plus the number of false positives (FP). We consider all items where $r = \hat{r} = 1$ as true positives. In contrast, Recall measures the ratio of true positives and the number of relevant items in the test set (RIT). These relevant items are the items a user has listened to in the given context and hence, have the rating $r = 1$. This recall computation implies that there is natural a cap of the recall determined by the number of recommendations n. The maximum recall is $\frac{n}{RIT}$. Hence, a low number of recommendations n naturally implies a low recall R.

$$P = \frac{TP}{TP + FP} \tag{7}$$

$$R = \frac{TP}{RIT} \tag{8}$$

For assessing the overall *precision*, *recall* and F_1-measure of the evaluated recommender systems, we compute the measures for each individual fold and compute the average among all users in a final step. We elaborate on the results of the presented evaluation in the following section.

5 RESULTS AND DISCUSSION

Based on the evaluation setup and measures described in the preceding section, we assess the performance of the following recommender systems: a pure CF-based recommender system (CF), context-aware CF with pre-filtering (PR-CF) as proposed by Pichl et al. [27], a SVD-based recommender system (SVD), a context-aware SVD-based recommender system with pre-filtering (PR-SVD), a context-aware random forest classifier-based recommender system (RF) as well as our proposed context-aware FM-based recommender system (FM). As outlined in Section 4.2, we consider the first five recommender systems as baseline approaches to our FM-based recommender. Additionally, we compare all recommender system against the random baseline (RB).

As described in Sections 4.3 and 4.4, we evaluate the rating prediction task and the top-n recommendations. Analogously to the previous section, we start with discussing the rating prediction before analyzing the top-n recommendations task.

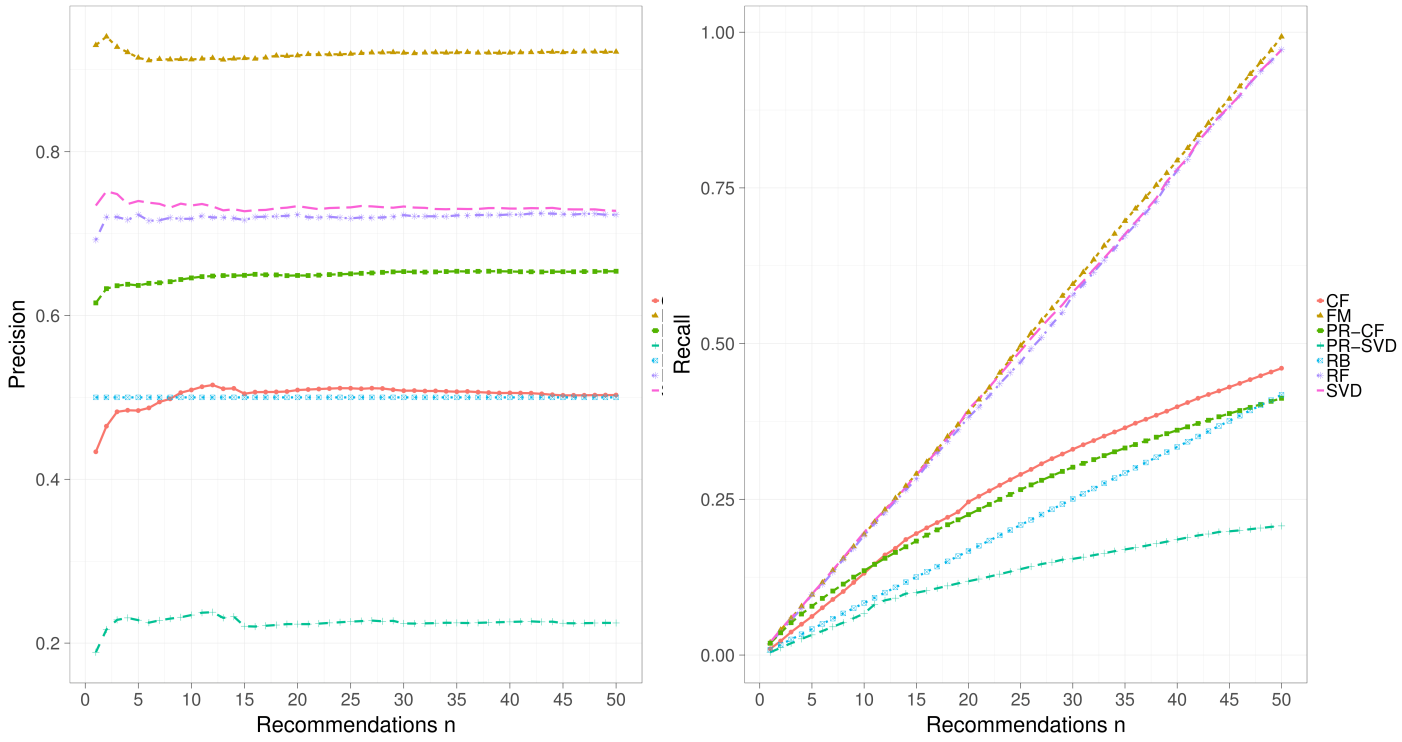
Recommender	RMSE	MAPE
CF	0.921	0.424
Pre-filtering CF	0.914	0.418
SVD	0.913	0.417
Pre-filtering SVD	0.914	0.418
RF	0.520	0.209
FM	0.560	0.282

Table 2: Evaluation of the Rating Prediction Task (all Tracks)

The results of the rating prediction task applied to all items in the test set are stated in Table 2. We find that with respect to the rating prediction task, the presented classifier-based context-aware approaches (RF and FM) clearly outperform all other approaches. RF and FM reach a RMSE of 0.520 and 0.560 and a MAPE of 0.209 and 0.282, respectively. The proposed baseline approaches reach RMSE values of > 0.9 and MAPE values of > 0.4. However, we also observe that none of the algorithms outperforms the random baseline of 0.5 w.r.t. RMSE (in contrast to MAPE). We lead this back to the fact that as RMSE naturally is more sensitive to high deviations between r and $\hat{r}$. Furthermore, the high error rate can also be explained by the fact that there are far more tracks a user did not listen to in a given cluster than tracks a user did actually listen to in a given cluster (i.e., the underlying matrix is highly sparse). Therefore, computing the error measures incorporating all tracks in the data leads to results biased towards imprecise rating predictions of low ranked (and hence, irrelevant) items. As the majority of tracks within our dataset are not relevant for a given user in a given context, evaluating RMSE and MAPE of all tracks within the dataset naturally includes tracks are not relevant for a user. Our recommender systems considers all tracks with a predicted rating $\hat{r} < 0.5$ as irrelevant to the user and these tracks are naturally not shown in the list of recommendations. We argue that the error for tracks with ratings $\hat{r} < 0.5$ are irrelevant for ranking the tracks on the recommendation list. To illustrate this bias we repeat the experiment for all tracks the recommendation algorithms consider as relevant for the user (i.e., tracks with a predicted rating of $\hat{r} \geq 0.5$ after the min-max scaling). The results of this evaluation are depicted in Table 3.

Recommender	RMSE	MAPE
CF	0.389	0.151
Pre-filtering CF	0.143	0.021
SVD	0.366	0.177
Pre-filtering SVD	0.939	0.927
RF	0.415	0.172
FM	0.380	0.221

Table 3: Evaluation of the Rating Prediction Task (relevant Tracks)



(a) Precision@n **(b) Recall@n**

Figure 2: Evaluation: Recall and Precision for $n = \{1 \dots 50\}$

When considering only tracks with a predicted rating of $\hat{r} \geq 0.5$, the results show that all algorithms except pre-filtering SVD outperform the baseline. The SVD-based recommender system even performs better than the RF-based one and slightly outperforms our proposed FM-based recommender. Furthermore, in this scenario, applying pre-filtering to CF improves results.

However, we argue that for the use cases we discuss later in this section, the top-n-recommendations evaluation is of higher importance as a user-centric evaluation that measures the utility of the top-n recommendations provided to the user is vital and of higher importance than actual error rates. Particularly, we argue that a top-n performance for low n is vital for users. Hence, we are particularly interested in the performance of the proposed recommendation approaches for lower n. Hence, not the precise rating prediction is crucial but ranking the track, such that the most relevant tracks for a user in a given context are listed within the top-n recommendations. This is, as the recommender system computes the list by sorting all potential track recommendations descending by the predicted rating $\hat{r}$ and returns the top-n tracks based in this list. Amongst others, in the remainder of the this section we empirically show the discrepancy between rating prediction accuracy and top-n prediction accuracy: although the RMSE and MAPE of CF is low, even lower than using RF, the performance evaluated measuring the accuracy of the top-n recommendations hardly outperforms the baseline.

For the presenting the results of the top-n performance evaluation of the proposed recommendations task, we depict the *precision*- and *recall*-curves in the Figures 2a and 2b for $n = \{1 \dots 50\}$. Aiming at making the performance of the recommender systems easily comparable, we integrated both, the *precision*- and the *recall* into the F_1 measure and plot the F_1 measure in Figure 3. Figure 2b shows that the FM, RF and SVD-based approaches perform substantially better in terms of recall than the other baselines across all number of recommendations n. Notably, the pre-filtering SVD approach performs worse than the random baseline across all n. As for precision (shown in Figure 2a) we detect a similar behavior. Again, pre-filtering SVD reaches substantially lower precision values than the other approaches. Interestingly, the SVD approach performs better than the pre-filtering SVD approach and reaches values similar to the random baseline. The FM-based approach performs substantially better than SVD and RF, followed by pre-filtering CF.

When examining the F_1 results in Figure 3, we consequently observe that all approaches outperform the baseline approach for $n < 25$. For $n >= 25$, only pre-filtering SVD reaches F_1 values lower than the random baseline. Considering the *precision* and *recall* plots of the algorithms in Figures 2a and 2b respectively, we observe that pre-filtering SVD performs poorly independent of the evaluation measure. However, we also note that a recommender system based on latent features computed via SVD provides accurate results and reaches high *recall* values. From this, we derive that the implicitly computed latent features represent track-context associations.

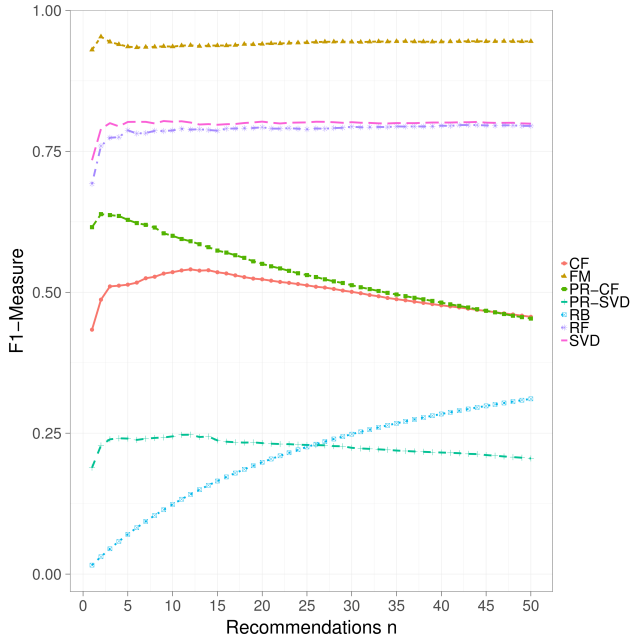


Figure 3: F1@n

Hence, pre-filtering limits the amount of input data available for computing latent features. Hence, we argue that pre-filtering SVD is not an effective approach for our recommendation task.

Moreover, we observe that all approaches besides pre-filtering SVD outperform classical CF. However, we have to note that CF hardly beats the random baseline, for which we assume that the chances to guess whether a track was listened by a user ($r = 1$) or not ($r = -1$) is 50%. We lead this back to a lack of non-boolean ratings as explicit ratings would allow a more precise computation of the user similarity and hence, more precise recommendations. We argue that this would improve the ordering of the tracks, which is especially crucial for the top-n recommendation task.

Additionally our experiments show that contextual pre-filtering is beneficial for CF. Pre-filtering CF beats the random baseline by a 46% higher F_1-score, which confirms the results of Pichl et al. [27]. However, as we observe in Figures 2a and 2b, pre-filtering is only highly beneficial for *precision*. The obtained *recall* value is slightly lower for the pre-filtering CF approach than for standard CF approach (-3,08%). We suspect two reasons for this: firstly, pre-filtering computes recommendations based on parts of the dataset. This is beneficial for the *precision*, as the number of recommendation candidates is limited. However, this configuration limits the *recall*. Secondly, as we compute user similarities on a restricted amount of data, not all similarities are captured which also possibly limits the set of possible recommendations.

Finally, we note that our proposed FM-based recommender system clearly outperforms all other approaches including SVD and RF in terms of *precision*, whereas the *recall* behaves similar for the three best approaches (FM, SVD and RF). We lead this behavior back to the way recall is computed. For each algorithm, the tracks are ordered by the predicted rating $\hat{r}$ and hence, by the likelihood of being relevant to a given user in a given context. Secondly, there is a natural upper bound of the recall dependent on the number of recommendations ($\frac{n}{RIT}$). As we sort recommendations by the predicted rating $\hat{r}$ evaluate the top-n tracks, the order of tracks is essential. The better an algorithm performs, the more relevant items with $\hat{r} = r = 1$ are contained in the top-n recommendations. This ultimately results in a higher number of RIT, as we compare the top-n recommendations to the actual rating value r. This is why the top-algorithms approach a recall of $n/50$.

Bollen et al. [6] addressed the problem of choice overload and state that user satisfaction is highest when presenting the user with top-5 to top-20 items—naturally assuming that the recommendation list contains a sufficient number of relevant items for the user. This is why we state the results for a small number of recommendations n in Table 4. Please note that we only list the top-3 algorithms here (FM, SVD and RF).

Recommender	F_1@1	F_1@5	F_1@10	F_1@20	F_1@50
FM	0.93	0.94	0.94	0.94	0.95
SVD	0.73	0.80	0.80	0.80	0.80
RF	0.69	0.79	0.79	0.79	0.80

Table 4: F_1-Measure for different n

The results in Table 4 show that for maximizing the user satisfaction according to Bollen et al. [6], our proposed FM-based approach clearly outperforms RF-based approaches (where we model the context explicitly) as well as SVD-based approaches (where we model the context-track associations implicitly via latent features). The FM model in Equation 2 depicts that the FM models the context explicitly as part of the variable's main effects: $\sum_{i=1}^{n} w_i x_i$ and additionally similar to the SVD approach implicitly in the pair-wise interactions: $\sum_{i=1}^{m} \sum_{j=i+1}^{m} \vec{v}_i, \vec{v}_j x_i x_j$. Underpinned by an empirical evaluation we argue that a hybrid approach combining regression with two-way interaction effects, where the weights of these effects are estimated via matrix factorization for classification (as provided by a factorization machine) is the best approach for context-aware music recommendation in a setting similar to the one presented in this work.

Summing up, in this work we show how contextual clusters can be leveraged for context-aware music recommendations. We find that contextual clusters can be leveraged for music recommendations without the drawbacks of the pre-filtering approach either by using a classifier approach or by incorporating latent features. Particularly, we find that by using Factorization Machines, the best results regarding the accuracy of recommendations can be obtained. Possible use cases for such recommender systems are (i) the generation of track suggestions during the playlist generation phase of a user and (ii) "contextual browsing" which helps users discovering music they like. For the first use case, the recommender system can recommend tracks that are likely to be interesting to the user that can be added to the currently curated playlist. Thus, the recommender system presents tracks to the user, which similar users added to contextually similar playlists. The second use case, the "contextual browsing", is based on the finding of Cunningham et al. [10] that people browse music collections to discover tracks they

like to listen to during different activities or situations. After a user selects a certain context (or the context is automatically inferred), our recommender system can provide lists of interesting tracks for this specified context. This use case is similar to the classical top-n recommendation task we evaluated in Section 4.

6 CONCLUSION

In this work, we propose a novel approach for incorporating contextual clusters extracted from the names of user playlists for the computation of context-aware track recommendations. Particularly, we present a recommendation approach based on Factorization Machines. We evaluate the prediction accuracy of different recommendation approaches based on a dataset of 15,000 users. Our k-fold cross-validations show that contextual clusters can indeed contribute substantially to recommendation accuracy by relying on either a classifier-based approach or approaches facilitating latent features. Particularly, the obtained results show that our proposed factorization machined-based recommender system is able to outperform the baseline approaches substantially. We consider these findings highly promising. Hence, in future work, we aim to evaluate different FM-models and configurations. Particularly, we are also interested in the use of higher order factorization machines [5].

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Trans. on Knowl. and Data Eng.* 17, 6 (June 2005), 734–749. DOI: http://dx.doi.org/10.1109/TKDE.2005.99

[2] Gediminas Adomavicius and Alexander Tuzhilin. 2010. Context-Aware Recommender Systems. In *Recommender Systems Handbook* (1st ed.), Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor (Eds.). Springer-Verlag New York, Inc., New York, NY, USA, Chapter 7, 217–253.

[3] Anupriya Ankolekar and Thomas Sandholm. 2011. Foxtrot: a soundtrack for where you are. In *Proceedings of Interacting with Sound Workshop: Exploring Context-Aware, Local and Social Audio Applications*. ACM, 26–31.

[4] Linas Baltrunas, Marius Kaminskas, Bernd Ludwig, Omar Moling, Francesco Ricci, Aykan Aydin, Karl-Heinz Lke, and Roland Schwaiger. 2011. InCarMusic: Context-Aware Music Recommendations in a Car. In *E-Commerce and Web Technologies*, Christian Huemer and Thomas Setzer (Eds.). Lecture Notes in Business Information Processing, Vol. 85. Springer, 89–100.

[5] Mathieu Blondel, Akinori Fujino, Naonori Ueda, and Masakazu Ishihata. 2016. Higher-Order Factorization Machines. In *Advances in Neural Information Processing Systems*. 3351–3359.

[6] Dirk Bollen, Bart P. Knijnenburg, Martijn C. Willemsen, and Mark Graus. 2010. Understanding Choice Overload in Recommender Systems. In *Proceedings of the Fourth ACM Conference on Recommender Systems (RecSys '10)*. ACM, New York, NY, USA, 63–70. DOI: http://dx.doi.org/10.1145/1864708.1864724

[7] Matthias Braunhofer, Marius Kaminskas, and Francesco Ricci. 2011. Recommending Music for Places of Interest in a Mobile Travel Guide. In *Proceedings of the Fifth ACM Conference on Recommender Systems (RecSys '11)*. ACM, New York, NY, USA, 253–256. DOI: http://dx.doi.org/10.1145/2043932.2043977

[8] Rui Cai, Chao Zhang, Chong Wang 0002, Lei Zhang 0001, and Wei-Ying Ma. 2007. MusicSense: contextual music recommendation using emotional allocation modeling.. In *Proceedings of the 15th ACM International Conference on Multimedia (2007)*.

[9] Zhiyong Cheng and Jialie Shen. 2014. Just-for-Me: An Adaptive Personalization System for Location-Aware Social Music Recommendation. In *Proceedings of the 2014 ACM International Conference on Multimedia Retrieval (ICMR)*. Glasgow, UK.

[10] Sally Jo Cunningham, David Bainbridge, and A. Falconer. 2006. More of an Art than a Science': Supporting the Creation of Playlists and Mixes. In *Proceedings of the 7th International Symposium on Music Information Retrieval (ISMIR 2006)*.

[11] Byeong-jun Han, Seungmin Rho, Sanghoon Jun, and Eenjun Hwang. 2010. Music emotion classification and context-based music recommendation. *Multimedia Tools and Applications* 47, 3 (2010), 433–460.

[12] David Hauger and Markus Schedl. 2012. Exploring Geospatial Music Listening Patterns in Microblog Data. In *Proceedings of the 10th International Workshop on Adaptive Multimedia Retrieval (AMR 2012)*.

[13] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM 2008)*. 263–272.

[14] Paul Jaccard. 1912. The Distribution of the Flora in the Alpine Zone. *New Phytologist* 11, 2 (1912), 37–50.

[15] Karen Sprck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation* 28 (1972), 11–21.

[16] Mohsen Kamalzadeh, Dominikus Baur, and Torsten Mller. 2012. A Survey on Music Listening and Management Behaviours. In *Proceedings of the 13th International Symposium on Music Information Retrieval (ISMIR 2012)*.

[17] Marius Kaminskas and Francesco Ricci. 2011. Location-Adapted Music Recommendation Using Tags. In *User Modeling, Adaption and Personalization*. Springer Berlin Heidelberg, 183–194.

[18] Marius Kaminskas and Francesco Ricci. 2012. Contextual music information retrieval and recommendation: State of the art and challenges. *Computer Science Review* 6, 2 (2012), 89–119.

[19] Marius Kaminskas, Francesco Ricci, and Markus Schedl. 2013. Location-aware Music Recommendation Using Auto-tagging and Hybrid Matching. In *Proceedings of the 7th ACM Conference on Recommender Systems (RecSys 2013)*. 17–24.

[20] Ja-Young Kim and Nicholas J Belkin. 2002. Categories of Music Description and Search Terms and Phrases Used by Non-Music Experts.. In *ISMIR*, Vol. 2. 209–214.

[21] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (Aug. 2009), 30–37. DOI: http://dx.doi.org/10.1109/MC.2009.263

[22] Jin Ha Lee and J Stephen Downie. 2004. Survey of music information needs, uses, and seeking behaviours: Preliminary findings.. In *ISMIR*, Vol. 2004. Citeseer, 5th.

[23] Jin Ha Lee, Yea-Seul Kim, and Chris Hubbles. 2016. A look at the cloud from both sides now: an analysis of cloud music service usage. In *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR 2016)*.

[24] Andy Liaw and Matthew Wiener. 2002. Classification and regression by randomForest. *R news* 2, 3 (2002), 18–22.

[25] Thais Mayumi Oshiro, Pedro Santoro Perez, and José Augusto Baranauskas. 2012. How Many Trees in a Random Forest?. In *In Proceedings of the 8th International Conference Machine Learning and Data Mining in Pattern Recognition (MLDM 2012)*. 154–168.

[26] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N. Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-Class Collaborative Filtering. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM '08)*. IEEE Computer Society, Washington, DC, USA, 502–511. DOI: http://dx.doi.org/10.1109/ICDM.2008.16

[27] Martin Pichl, Eva Zangerle, and Günther Specht. 2015. Towards a Context-Aware Music Recommendation Approach: What is Hidden in the Playlist Name?. In *15th IEEE International Conference on Data Mining Workshops (ICDM 2015)*. 1360–1365.

[28] Steffen Rendle. 2012. Factorization Machines with libFM. *ACM Trans. Intell. Syst. Technol.* 3, 3, Article 57 (May 2012), 22 pages.

[29] Steffen Rendle, Zeno Gantner, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2011. Fast Context-aware Recommendations with Factorization Machines. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 635–644.

[30] Steffen Rendle, Zeno Gantner, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2011. Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 635–644.

[31] Seungmin Rho, Byeong-jun Han, and Eenjun Hwang. 2009. SVR-based Music Mood Classification and Context-based Music Recommendation. In *Proceedings of the 17th ACM International Conference on Multimedia (2009)*. 713–716.

[32] Markus Schedl. 2016. The LFM-1b Dataset for Music Retrieval and Recommendation. In *Proceedings of the ACM International Conference on Multimedia Retrieval (ICMR)*. New York, USA.

[33] Markus Schedl, Arthur Flexer, and Julin Urbano. 2013. The neglected user in music information retrieval research. *Journal of Intelligent Information Systems* 41, 3 (2013), 523–539.

[34] Markus Schedl and Dominik Schnitzer. 2013. Hybrid Retrieval Approaches to Geospatial Music Recommendation. In *Proceedings of the 35th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2013)*.

[35] Markus Schedl, Andreu Vall, and Katayoun Farrahi. 2014. User Geospatial Context for Music Recommendation in Microblogs. In *Proceedings of the 37th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2014)*.

[36] Eva Zangerle, Wolfgang Gassler, and Günther Specht. 2012. Exploiting Twitter's Collective Knowledge for Music Recommendations. In *Proceedings of the 2nd Workshop on Making Sense of Microposts (#MSM2012)*.