Efficient Optimization of Multiple Recommendation Quality Factors According to Individual User Tendencies

Michael Jugovac[a,*], Dietmar Jannach[a], Lukas Lerche[a]

[a]*TU Dortmund, Department of Computer Science, Otto-Hahn-Str. 12, 44227, Dortmund, Germany*

Abstract

Recommender systems are among the most visible applications of intelligent systems technology in practice and are used to help users find items of interest, for example on e-commerce sites, in a personalized way. While past research has focused mainly on accurately predicting the relevance of items that are unknown to the user, other quality criteria for recommendations have been investigated in recent years, including diversity, novelty, or serendipity. Considering these additional factors, however, often leads to the following two challenges. First, in many application domains, trade-offs like "diversity vs. accuracy" have to be balanced. Second, it is not always clear *how much* diversity or novelty is desirable in practice.

In this work, we propose a novel parameterizable optimization scheme that re-ranks accuracy-optimized recommendation lists in order to cope with these challenges. Our method is both capable of considering multiple optimization goals at the same time and designed to consider *individual user tendencies* regarding the different quality factors, like diversity. In contrast to previous work, the method is not restricted to a specific underlying item ranking algorithm and its generic design allows the algorithm to be parameterized according to the requirements of the application domain. Experimental evaluations with different datasets show that balancing the quality factors with our method can be done with a marginal or no loss in ranking accuracy. Given that our method can be applied in various domains and within the narrow time constraints of online recommendation, our work opens new opportunities to design novel finer-grained personalization approaches in practical applications.

Keywords: Recommender Systems, Quality Factors, User-Specific Optimization, Trade-Offs

*Corresponding author.

Preprint accepted for publication in *Expert Systems with Applications*.
Cite as: Michael Jugovac, Dietmar Jannach, and Lukas Lerche *Efficient optimization of multiple recommendation quality factors according to individual user tendencies*, Expert Systems with Applications, Vol. 81, 2017, pp. 321-331.
Final version available at: http://doi.org/10.1016/j.eswa.2017.03.055

1. Introduction

Automated and personalized recommendations have become an integral part of our online user experience. Nowadays, recommender systems (RS) are used to help users find relevant items in a variety of ways, e.g., by recommending items to purchase on e-commerce sites, music to listen to on streaming platforms, or other people to connect with on social networks. RS are among the most successful applications of intelligent systems in practical environments. As a consequence, RS are an active field of research and remarkable advances were made in recent years in terms of increasing the prediction and ranking accuracy of recommendation algorithms using, for example, novel methods for matrix factorization, ensemble learning, or learning-to-rank.

Predicting the relevance of an item for a user as accurately as possible is an important quality criterion of a recommender system. It is, however, not the only one (McNee et al., 2006; Jannach et al., 2016). Even when the recommended items match the user's taste very well, their value for this user might be limited, e.g., when the recommendations are too obvious and contain only the most popular items. In addition, the resulting recommendation lists can also be too monotonous and their limited diversity might prevent the user from discovering further relevant items or item categories. Several proposals have been made over the last years to deal with these problems, e.g., by increasing the diversity of the recommendations, by counteracting the popularity bias of algorithms and pushing novel, long-tail items, or by making serendipitous suggestions, see, e.g., (Ziegler et al., 2005; Iaquinta et al., 2008; Zhang and Hurley, 2008; Adomavicius and Kwon, 2012; Zhang et al., 2012; Said et al., 2013).

However, two main issues arise when additional quality factors are to be taken into account. First, in many application domains trade-offs have to be balanced. For example, when trying to help users discover new artists on a music streaming platform with a recommender system, one can recommend tracks of artists that the user has not yet listened to. However, recommending more novel or lesser-known items can be risky and even detrimental to the perceived quality of the service (Ekstrand et al., 2014; Chau et al., 2013). Recommending generally popular items, on the other hand, might be algorithmically easier (Steck, 2011) and less risky, but can also be of little value to users who are interested in discovering something new. One algorithmic problem is, therefore, to find a balance between different quality factors, e.g., to increase the diversity of the recommendations without including too many items that are of limited relevance to the user as done, e.g., by Adomavicius and Kwon (2012).

Second, most of the existing works assume that the same global level for each quality factor, such as diversity, is appropriate for the whole user base. For example, the approach of Adomavicius and Kwon

(2012) is designed to always *maximize* diversity for all users. In reality, however, the "right value" for each quality factor can depend on many aspects including *the domain* (homogeneity might be more desirable than diversity, e.g., in the music domain), *the individual user* (some users simply prefer a small subset of item categories), or *business goals*, e.g., the promotion of long-tail items.

In this paper, we present a novel algorithmic approach to address these two issues. At its core, our approach consists of a generic optimization scheme designed to balance accuracy with one or more quality factors. In contrast to previous approaches, e.g., by Oh et al. (2011), Vargas and Castells (2011), Adomavicius and Kwon (2012), or Kapoor et al. (2015), our method is capable of dealing with multiple optimization goals in parallel and is not limited to simple characteristics like the popularity of the individual items. Furthermore, our approach does not require the use of a specific algorithm for relevance-based item ranking, but is based on re-ranking the topmost items of an accuracy-optimized recommendation list that can be generated by any rating prediction or item ranking algorithm.

To deal with the problem of determining the right level for factors like popularity, we look at the *tendencies of the individual users* based on their past behavior and emulate these tendencies in the re-ranking process. If, for example, a user always had a strong preference for blockbuster movies from a wide range of genres, the re-ranked recommendations will reflect this by putting an emphasis on popular movies and high genre diversity. By using such an individualization, we can avoid the need for the specification of a "global" level of a certain quality feature and, at the same time, better match the user's preferences regarding quality factors that concern the set of recommendations as a whole.

Overall, the general nature and domain independence of our method allow it to be used in various application domains of recommender systems in which trade-offs have to be balanced. Technically, due to the greedy nature of the optimization scheme, it can be applied under the narrow time constraints of online recommendation and used as a post-processing step that further optimizes the outputs of today's high-performance machine learning algorithms. As a result, our method can help to build next-generation recommender systems that apply personalization not only when determining the relevance of individual items, but also on an aggregate level of an entire recommendation list.

The paper is organized as follows. In Section 2, we first discuss technical preliminaries regarding possible approaches of modeling past user tendencies and then present the proposed generic adaptation procedure. In Sections 3 and 4, we report detailed results of a comprehensive set of empirical evaluations in which we

test our approach with different data sets, compare it with previous approaches, and investigate system-wide effects of tuning the recommendations to individual user tendencies. Section 5 compares our work on a theoretical level with past approaches that have similar goals.

2. Technical Approach

2.1. Preliminaries

Generally, we assume that we are given a set of users with known past preferences, e.g., expressed through explicit item ratings, and a finite set of recommendable items. Our overall goal is to shape recommendations that reflect the user's preference tendencies in one or more quality dimensions while keeping accuracy high. With accuracy we refer to an algorithm's capability of estimating the relevance of an item for a user, which is typically quantified in the literature through computational metrics like precision, recall, or the RMSE.

To be able to quantify how well a recommendation list matches a user's past tendencies with respect to different quality characteristics, we introduce two functions: $\mathcal{P}(S_u)$ represents the user's preference tendencies, whose calculation is based on a sample set S_u, which is a subset of representative items of the user u. S_u can, for example, contain the set of top-rated items of the user, the set of recently listened music tracks, or any other set of explicit or implicit ratings considered to be relevant in the user's current context. $\mathcal{R}(T_u)$, in turn, represents the characteristics of the recommendations, whose values are based on T_u, the ranked top-n list of items recommended to u. This list can, as mentioned above, be generated by any existing recommendation algorithm.

2.1.1. Quantifying the Tendencies

Different approaches are possible to numerically quantify a user's past tendencies. In the following, we discuss three alternatives. However, our optimization method is not limited to these approaches and alternative and domain-specific measures can be used.

Mean and Standard Deviation. A straightforward way of quantifying the tendencies is to look at average values and standard deviations of a quality factor. For example, if a user likes blockbusters, the user's sample set will exhibit a high average popularity. In this case, we can try to reflect this tendency in the recommendation list by including items so that the average item popularity of the list is also high. To avoid that a re-ranking algorithm includes too many "extreme" values in the top-n list when trying to match the user's average value, we propose to use scores that combine the mean and standard deviation in a meaningful form, e.g., by using a weighted combination of their absolute values.

Aggregate Measures. Measures, such as the Intra-List-Similarity (ILS), also result in one single aggregated numerical score but cannot be obtained by averaging individual item features. Instead, the ILS is calculated as the mean of the pairwise item similarities. Another example of a measure of this type is the "coherence" of a music playlist, which can, e.g., be determined by looking at the tempo difference of two consecutive tracks. In our evaluations, we use the ILS score (Ziegler et al., 2005) to compare user tendencies and recommendation lists, i.e.,

$$\mathcal{P}_{sim}(S_u) = \sum_{i \in S_u} \sum_{j \in S_u} \frac{sim(i, j)}{|S_u|^2}$$

where $sim(i, j)$ is computed using the cosine similarity of the *Term Frequency-Inverse Document Frequency* (TF-IDF) representations of the items' content descriptions.

Earth Mover's Distance. The re-ranking approach proposed by Oh et al. (2011) shares goals with our work. The authors propose to use the *Earth Mover's Distance* (EMD) to compare two distributions, because the EMD has certain advantages over the Kullback-Leibler divergence in that context. Informally speaking, the EMD expresses the minimum cost of transforming one distribution into another. Determining the distribution distance corresponds to solving a transportation problem, which can be efficiently solved in quantitative value domains.

2.1.2. General Optimization Goal

Once the choice of how to compute $\mathcal{P}(S_u)$ and $\mathcal{R}(T_u)$ is made, the general optimization goal is to find a modified recommendation list T_u^* whose characteristics $\mathcal{R}(T_u^*)$ minimize the difference (error) $e = d(\mathcal{R}(T_u^*), \mathcal{P}(S_u))$. The error can, for example, simply be the absolute difference $d = |\mathcal{R}(T_u^*) - \mathcal{P}(S_u)|$ for single-valued tendency summaries or the Earth Mover's Distance for distributions.

2.2. Proposed Optimization Procedure

In the following, we will first present the general structure of our re-ranking strategy, which we call Personalized Ranking Adaptation (PRA), and then discuss possible algorithm configurations.

2.2.1. General Procedure and Design Rationale

The following main steps are performed *for each user* for whom we generate personalized recommendations.

1. Determine the sample set S_u, e.g., based on the top-rated items of a user, and compute the preference tendency $\mathcal{P}(S_u)$ w.r.t. the desired quality feature, for instance, diversity.

2. Retrieve a ranked item list from an accuracy-optimizing recommendation algorithm based on the training data.
3. Let T_u be the first n elements of the ranked recommendations, where n is the desired list length. Let the *Exchange List* X_u be the list of elements ranked immediately after T_u.
4. Systematically exchange elements i from T_u with elements j from X_u in such a way that $d(\mathcal{R}(T_u[i \rightarrow j]), \mathcal{P}(S_u))$ becomes smaller, leading to a modified list T_u^* of top items. The notation $T_u[i \rightarrow j]$ stands for a list in which element i is replaced by j. Since T_u and X_u are ordered by relevance, we iterate over the items in reverse order for T_u and in normal order for X_u to swap the least relevant items from T_u with the most relevant items from X_u when two or more swaps offer the same improvement. The exact details of the exchange procedure are given in Algorithm 1.
5. Return T_u^* as the list of recommended items.

Algorithm 1: PRA variant with a limit on the maximum number of steps. All possible swaps between T_u and X_u are evaluated. The swap with the strongest improvement on the desired quality property is executed afterwards.

input : int *maxSteps*; **Array** S_u, T_u, X_u
output: Array T_u as T_u^*

for 1 **to** *maxSteps* **do**
  $e_{base} \leftarrow d(\mathcal{R}(T_u), \mathcal{P}(S_u))$
  $\Delta_e \leftarrow 0$
  for $i \in T_u$ *(backwards),* $j \in X_u$ **do**
    $e_{new} \leftarrow d(\mathcal{R}(T_u[i \rightarrow j]), \mathcal{P}(S_u))$
    if $e_{base} - e_{new} > \Delta_e$ **then**
      $I_{best} \leftarrow i$
      $J_{best} \leftarrow j$
      $\Delta_e \leftarrow e_{base} - e_{new}$
  if $\Delta_e > 0$ **then**
    $T_u \leftarrow T_u[I_{best} \rightarrow J_{best}]$
    $X_u \leftarrow X_u[J_{best} \rightarrow I_{best}]$
  else
    `break`

Figure 1 illustrates how PRA can match the characteristics of quality factors of a recommendation list to the user's tendencies (using the example of diversity). In the example, the user's sample set (e.g., top rated items, in dotted lines) shows a tendency towards diversity, indicated by a large variety of colors. In contrast,

the first part of the recommendation list T_u generated by the accuracy-optimized baseline is comparably monotonous (Step 3). After swapping items with the exchange set X_u, the top-n list better resembles the user's diversity tendency (Step 5).

In summary, PRA iteratively and greedily re-ranks items to bring the top-n recommendations closer to the user's tendencies. The structure of the algorithm is independent of the optimization goal and multiple goals can be considered in parallel. Furthermore, the accuracy trade-off can be controlled by varying the length of the exchange list X_u.

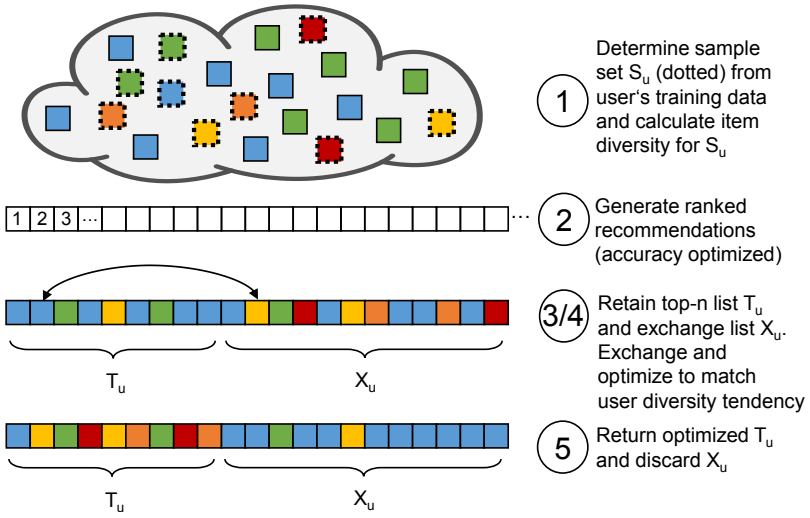

Figure 1: STEPS OF PRA using the example of diversity optimization (colors represent, e.g., an item's genre).

Next, we will discuss (a) possible variants and parameters of the item swapping process and the stopping criterion and (b) how to balance multiple optimization goals.

2.2.2. Algorithm Configurations

Depending on application-specific requirements, PRA can be parameterized and fine-tuned as follows.

- *Sample Set.* The size and the contents of the sample set S_u can be varied (see Section 2.1). In our experiments (Sections 3 and 4), we use the user's ten highest rated items in case of the movie dataset and the current playlist in case of the music data.
- *Exchange List Size.* When X_u is larger, more choices are available to PRA to optimize the desired quality factor(s). Picking items from the end of a larger set X_u might, however, have a stronger impact on accuracy.

- *Optimal vs. Greedy Swapping.* In every iteration, i.e., when two items are swapped, we can configure PRA to first systematically explore all possible swaps and then apply the exchange with the strongest effect each time. Alternatively, we can immediately apply every swap that leads at least to some error reduction. In our experimental evaluation, we report the results of the "optimal swap" strategy.

- *Stopping Criterion.* Different strategies are possible. First, the algorithm can be configured to stop when the *error reduction rate*, i.e., the quotient between the error reduction (Δ_e) of the current iteration and the error reduction of the previous iteration, falls below a certain threshold ϵ. Second, we can stop the execution of PRA if the *difference* between the sample set and the current top-n list falls below a certain level. Third, PRA can be configured to stop after a specified *number* of examined or successful swaps as done in Algorithm 1.

2.2.3. Balancing Multiple Optimization Goals

The proposed item re-ranking approach can be applied with multiple adaptation goals in mind. We could, for example, try to match the diversity and the popularity of the items in the top-n list at the same time.[1] Before applying a possible swap when dealing with multiple optimization goals, we thus have to look at the effects of the swap in all goal dimensions. The conservative strategy we used in our experiments is to apply a change only in case the *aggregated* error of all dimensions is reduced. One option to aggregate two or more errors is to calculate a weighted sum of the (normalized) errors, where the weights can be determined based on a cross-validation procedure or domain expertise. A less conservative approach could be to allow swaps that lead to a slight deterioration of one characteristic if there is a strong expected gain for another one.

2.2.4. Relation to Pareto Optimization

In our approach, we can consider multiple optimization goals at the same time in different ways, particularly by using an aggregated error (score) function as done in our experiments. Using such an aggregated score function in the optimization process allows us to use heuristics to find a good or close-to-optimal solution even in scenarios with narrow time constraints. This is in particular the case in our problem setting of online recommendation, in which we have to re-rank lists on demand.

In general, considering multiple (competing) optimization goals at the same time is the fundamental problem of multi-criteria optimization approaches (Marler and Arora (2004)). Pareto optimality is a central

[1] Compared to previous works that also consider past user-tendencies, this is a unique feature of our approach, as will be discussed later on in Section 4.2.1 and Section 5.

concept in such problem settings. A solution to a multi-criteria optimization problem is defined as Pareto optimal if none of the values of the objective functions for the individual goals can be improved without degrading the value of another objective function. The set of all Pareto optimal solutions is then called the Pareto frontier. In principle, our re-ranking problem setting can be framed as a traditional multi-objective optimization problem and existing algorithms can be used to find all points on the Pareto frontier. However, finding these points is computationally expensive in most cases, which is why many researchers rely on evolutionary approaches. But even when such heuristic search techniques are applied, it remains challenging to compute or approximate Pareto optimal solutions within the narrow time constraints of application scenarios like ours.

Ribeiro et al. (2014) proposed an approach that considered Pareto optimality in a recommendation scenario. Their problem setting and optimization goal are similar to ours, even though they optimized the recommendations only on a system-wide, and not a per-user level. However, as the authors state, the system can only be used in an online recommendation scenario because the system-wide Pareto frontier is precomputed offline. Re-ranking the recommendation lists online based on recent user interactions, as in our scenario, would therefore be computationally infeasible with such an approach.

In contrast to their approach, our method cannot guarantee in general that a returned solution is Pareto optimal because we use an aggregate objective function. Using such an aggregate function corresponds to what is called *weighted sum scalarization* in the multi-criteria optimization literature (Ehrgott, 2006). This strategy can only guarantee a Pareto optimal solution in case the solution space is convex. Otherwise, it returns what is called a *weakly optimal solution*. Due to the narrow time constraints of online recommendation, our re-ranking method uses a greedy heuristic strategy to approximate such (weakly) optimal solutions. The results of our empirical evaluations presented in Section 4.1 show, however, that our heuristics lead to results that are on average very close to these optimal solutions.

A general problem when using aggregate objective functions is that one has to determine the (importance) weights for the different optimization goals. Usually, this process of finding appropriate weights to balance the different criteria requires a certain degree of knowledge about the domain or a specific application. In a situation where all Pareto optimal points are known, this decision might be easier, because trade-offs between criteria can be visualized in terms of the Pareto frontier, and it might even be possible to find a *compromise solution* automatically (Ehrgott, 2006). However, in our application scenario we cannot generally assume that we know the elements on the Pareto frontier due to the potentially substantial computational costs to determine the frontier elements.

3. Empirical Evaluation – Algorithm Effectiveness

We evaluated the effectiveness of Personalized Ranking Adaptation, i.e., how good it is at balancing multiple optimization goals, in two application domains: movies and next-track music recommendation.

3.1. Data

To enable reproducibility, we first use the public MovieLens 1M dataset, which comprises 1 million explicit movie ratings. As a second dataset, we use a set of public playlists shared on Last.fm. The dataset comprises 22k different tracks, which are organized in 3.5k playlists shared by 511 users.[2] We chose this domain because quality criteria like artist diversity are particularly important in music playlist recommendation. Furthermore, the music dataset allows us to explore an alternative way of building the *Sample Set* S_u, which in this case is created from the user's recent listening history. Consequently, these measurements also help us to assess the effectiveness of PRA to adapt the recommendations to the user's short-term preferences.

3.2. Accuracy-Optimizing Baseline Algorithms

To be able to assess to what extent the effectiveness of PRA depends on the chosen baseline, we ran experiments using various state-of-the-art accuracy-optimizing algorithms of different families. Because the playlist recommendations are not personalized, but only depend on the recent history, we employed application-specific techniques from the literature. Specifically, the following baselines were used.

Movie Recommendation.

1. *Funk-SVD:* A matrix factorization (MF) method with gradient descent optimization (Funk, 2006). Parameters: Number of features = 50, steps = 100, $\lambda = 0.02$, $\gamma = 0.005$.
2. *Factorization Machines (FM):* A combination of feature engineering and factorization models with Alternating Least Squares optimization (Rendle, 2012). Parameters: Initial standard deviation = 0.3, steps = 50, $\lambda = 15$.
3. *Bayesian Personalized Ranking (BPR):* A ranking technique on implicit feedback with MF optimization (Rendle et al., 2009). Parameters: Number of features = 50, steps = 100, $\lambda = 0.0025$, $\gamma = 0.05$.

[2]The playlist dataset is available for download at: `http://bit.ly/1y3FJ0w`. To evaluate different aspects of the user preferences we retrieved additional information about the tracks (artist, year, tempo, loudness etc.) from public music services like `the.echonest.com`.

Next-Track Music Recommendation.

4. *KNN:* A playlist-based k-Nearest Neighbor technique ($k = 300$) using cosine similarity as a distance measure, which works particularly well in the music domain (Bonnin and Jannach, 2014).
5. *CAGH:* Recommendation of the greatest hits of artists that are similar to artists in the current playlist (Bonnin and Jannach, 2014). The method is very accurate for shorter list lengths, which is why it is particularly relevant in our setting.

The algorithm parameters were manually tuned to optimize the RMSE (Funk-SVD, FM) and the F1-measure (BPR) in the movie domains and for Recall (hit rate) as a typical accuracy measure for next-track recommendation.

3.3. Metrics and Protocol

PRA is designed to balance multiple optimization goals with ranking accuracy, which we measure in our experiments through F1 at list length 10. For the music domain, we applied the evaluation protocol used by Hariri et al. (2012). According to this protocol, we hide the very last track of each playlist in the test set and report the Recall values at list length 10.

As additional quality factors, we use *list diversity*, *item popularity*, and *item release years* in both domains. List diversity was calculated using ILS based on plot descriptions (movie domain) or artist information (music domain) as discussed in Section 2.1. To measure the item popularity of the recommendations, we determined the number of ratings/playlist occurrences in the dataset for the top-n recommended items (Jannach et al., 2015).

For the music domain, we additionally tested the effectiveness of PRA to optimize the recommendations in a way that they match other criteria, e.g., the *tempo* of music tracks in the sample set. Notice that optimizing the homogeneity of such numerical characteristics can be particularly helpful in the music domain, in which the goal can be to generate playlist recommendations that are similar to the most recently played tracks, for example, with respect to *tempo* or *loudness* (Sarroff and Casey, 2012).

To assess the impact of the re-rankings, we determine the measurement results for all quality factors before and after re-ranking the items with PRA. The numbers reported in the following sections are the result of a five-fold cross-validation procedure where the data was split user-wise, i.e., for each user there were items in the training set as well as hidden candidate items in the test set.

Table 1: PRA evaluation results for the movie domain. *Target* (= optimization target) can be Y = year, P = popularity, D = diversity, E_-_ = EMD variant. Values are shown in groups of three (Funk, BPR, FM) and represent the relative per-user tendency error improvement w.r.t. to the respective baseline algorithm. "*" marks a significant ($p < 0.05$) change in accuracy w.r.t. the baseline. Parameters of PRA: Stopping criterion: ϵ (error reduction rate) < 0.001, max. steps $= 500, |T_u| = 10, |X_u| = 20$.

	Relative improvement w.r.t. baseline [Funk-SVD | BPR | Factorization Machines]																							
Target	Avg. pop.			σ pop.			Avg. year			σ year			Diversity			EMD (pop.)			EMD (year)			F1@10		
P	.79	.95	.95	.72	.88	.85	-.16	.04	-.21	-.03	.04	-.16	.27	.10	-.03	.54	.55	.61	-.11	.03	-.19	.000	*-.009	-.002
Y	.11	.09	.16	.12	.01	.05	.93	.87	.89	.88	.66	.89	.19	.11	.03	.09	.06	.13	.62	.44	.65	.000	-.002	-.002
D	.05	.06	.13	.10	.02	.03	-.04	.02	-.05	-.05	.03	-.05	.99	.98	.97	.05	.04	.11	-.03	.01	-.05	.000	*-.004	.000
P Y	.78	.95	.94	.72	.88	.85	.25	.36	.17	.26	.30	.12	.27	.11	-.02	.54	.55	.62	.18	.22	.10	.000	*-.009	-.004
P D	.67	.83	.81	.58	.70	.64	-.12	.05	-.16	-.02	.05	-.13	.94	.89	.85	.48	.51	.56	-.07	.03	-.15	-.002	*-.009	-.002
Y D	.09	.07	.12	.13	.02	.03	.51	.34	.27	.34	.25	.20	.92	.87	.76	.07	.05	.10	.32	.19	.19	.000	-.002	-.002
P Y D	.67	.83	.81	.57	.69	.64	.10	.12	-.08	.11	.12	-.06	.94	.89	.85	.47	.51	.56	.07	.08	-.09	.000	*-.009	-.002
EP	.72	.85	.86	.57	.69	.62	-.16	.04	-.23	-.01	.05	-.15	.22	.09	-.03	.70	.85	.86	-.10	.03	-.20	.000	*-.007	-.002
EY	.17	.10	.18	.14	.01	.04	.87	.74	.85	.81	.59	.81	.23	.10	.02	.13	.07	.15	.79	.71	.80	-.002	*-.004	-.002
EP EY D	.32	.42	.36	.26	.23	.15	.55	.40	.42	.39	.30	.29	.92	.85	.78	.28	.34	.30	.46	.33	.36	.000	-.002	-.002

3.4. Results

3.4.1. Movie Domain

Table 1 shows the effect of applying PRA to the MovieLens dataset with different optimization targets. The entries in the table are interpreted as follows. Consider the value ".95" in the first data row circled in red. The row shows the results when the optimization target was "P", i.e., popularity as a single optimization goal. The value appears in the third column related to the quality factor "average popularity", which means that the baseline ranking was generated with Factorization Machines. The value 0.95 states that on average the popularity distance (error) between the user tendency and the re-ranked list could be reduced by 95 %, when comparing PRA with the baseline. Farther to the right in the same row, we see that at the same time the variance of the popularity values was better aligned with the user tendencies (85 %). Last, the accuracy numbers at the very right show that only tiny decreases in accuracy were the cost for this better alignment, i.e., a 0.2 % (-.002) lower F1@10 score. The background colors in Table 1 indicate whether improvements (shades of green) or deteriorations (shades of orange) were observed or if the numbers remained roughly unchanged w.r.t. the respective baseline.

Single-objective Optimization Results. In the following we focus on the results when quantifying the list characteristics with the previously mentioned *weighted combination of mean and standard deviation* (e.g. for popularity: $\alpha \cdot \overline{pop} + \beta \cdot \sigma_{pop}$). To demonstrate the ability of our algorithm to balance both the mean and the standard deviation, we set α and β to a fixed value of 1 in all experiments. We use this comparably

simple tendency quantification method in our first experiments to show that a basic method—which has the advantage of being interpretable—already leads to good results. The results obtained with the more complex EMD metric are discussed later on.

The first three rows of Table 1 show the results when optimizing the recommendations for a single goal. The improvements range from 66 % (variance of the release year for BPR) to 99 % (diversity for FUNK-SVD). Overall, PRA was successful in reducing the error in all cases. In some situations we see side effects, e.g., that optimizing for popularity or the release year also has a positive effect on diversity. In one case, when lists were optimized for popularity, we could observe a measurable negative side effect on the suitability of the release years.

Multi-objective Optimization Results. The next four rows of Table 1 contain the observations made when two or all three goals were considered during optimization. In all but one of the configurations, PRA was successful in achieving all optimization goals at the same time. For example, whenever diversity or popularity were part of the goals, a corresponding error reduction in both dimensions could be observed.

The only exception where we could not improve all specified goals at the same time again occurs when FM was used as a baseline and we optimized for popularity, release year, and diversity at once. While the popularity and diversity tendencies could be very well aligned, the match of the release years became slightly worse (-8/-6 %, circled in blue in the table). A possible trade-off between these goals was already observed in the single-objective measurements above. Furthermore, these side-effects seem to depend on the chosen baseline. Given that algorithms with similar accuracy often lead to quite different top-n lists (Jannach et al., 2015), this is not too surprising. A deeper analysis of why this phenomenon only appears in combination with the FM technique is part of our ongoing work.

Using Earth Mover's Distance. Finally, the last three rows of Table 1 show the results when the Earth Mover's Distance was minimized. The same general trend as for the mean-and-deviation-based goals can be observed. Using EMD-based tendencies during optimization "naturally" leads to stronger error reductions when the EMD is also used to assess the extent of the reduction.[3] The trade-off between "release year"-based and popularity-based optimization can also be observed when using the EMD as a distance measure. In the simultaneous optimization of all list characteristics ("EP EY D"), PRA was able to reduce the differences between user tendencies and list characteristics w.r.t. all distance measures for all baseline algorithms.

[3] We also tried to quantify the user tendencies via Gaussian Mixture Models and applied the Akaike Information Criterion as a distance measure. However, we omit the result as they are comparable to our observations for the EMD-based optimization.

Table 2: PRA evaluation results for the Last.fm dataset. *Target* (= optimization target) can be P = popularity, Y = year, T = tempo, L = loudness, D = diversity. Values are shown in groups of two (CAGH, KNN) and represent the relative per-user tendency error improvement w.r.t. the respective baseline algorithm. "*" marks a significant ($p < 0.05$) change in accuracy w.r.t. the baseline. Parameters of PRA: Stopping criterion: ϵ (error reduction rate) < 0.001, max. steps $= 500, |T_u| = 10, |X_u| = 20$.

					Relative improvement w.r.t. baseline [CAGH | KNN]															
Target					Avg. pop.		σ pop.		Avg. year		σ year		Avg.tmp.		σ tmp.		Avg. loud.		σ loud.		Diversity		Recall@10	
P					.52	.46	.09	.20	-.19	-.24	-.23	-.28	-.26	-.23	-.19	-.22	-.20	-.21	-.23	-.26	-.09	-.12	*.205	*.070
	Y				.11	-.05	-.33	-.44	.26	.32	.16	.27	-.22	-.24	-.21	-.23	-.19	-.20	-.29	-.28	-.11	-.10	*.299	*.204
		T			.15	-.05	-.31	-.49	-.19	-.27	-.22	-.29	.29	.29	.29	.25	-.22	-.24	-.30	-.27	-.16	-.14	*.188	.013
			L		.15	-.06	-.36	-.45	-.18	-.18	-.24	-.21	-.22	-.22	-.18	-.20	.30	.32	.20	.25	-.14	-.14	.120	.020
				D	.01	-.10	-.39	-.41	-.15	-.17	-.17	-.18	-.23	-.22	-.18	-.20	-.19	-.22	-.27	-.28	.19	.14	*.399	*.289
P	Y	T	L	D	.41	.28	-.02	.00	-.11	-.15	-.15	-.14	.01	.00	-.02	-.07	.21	.18	.09	.08	.16	.07	*.269	*.173

Overall, this last set of results indicates that EMD-based and mean-and-deviation-based optimization lead to comparable results. In fact, when optimizing w.r.t. one distance measure, the other often improves to some degree as well. Furthermore, the results suggest that EMD-based optimization can be used to effectively balance trade-offs between multiple quality criteria.

Impact on Accuracy. For the factorization methods FUNK-SVD and FM, no statistically significant differences could be observed between the accuracy of the baseline and the accuracy after post-processing (ANOVA $p > 0.99$). The strongest impacts can be observed in the F1 measure when BPR is used as a baseline, which is not too surprising because BPR optimizes a ranking criterion. But even in that case the largest deterioration is less than 1 %, and only some differences are statistically significant ($p < 0.05$, marked with * in Table 1).

3.4.2. Results – Music Domain

Table 2 shows the relative improvements obtained with PRA for the next-track recommendation problem using the Last.fm dataset. In addition to the aforementioned quality criteria, we included a number of other characteristics of musical tracks that can contribute to the user's quality perception in case they are aligned with the user's tendencies. For the sake of brevity, we only report a selection of the results and not all possible combinations of the various optimization goals. We also omit the results for the EMD-based optimization as they are mostly in line with the results for the mean-and-deviation measurements, which is similar to the observations we made for the movie domain.

Single-objective Optimization Results. Optimizing the lists returned by KNN and CAGH for a single optimization goal again consistently led to a reduction of the differences between user tendencies and list characteristics, indicated by the green diagonal in the table with improvements up to 52 %. In fact, making

the list of recommended music tracks more consistent with the recent listening history, i.e., the beginning of the playlists, helped to considerably *increase the accuracy* (marked with * if statistically significant at $p < 0.05$).[4] For example, when the artist diversity was the optimization target for KNN, Recall@10 improved significantly ($p < 0.05$) by 28.9 % (circled in red in the table). A closer look at the recommendations of the baseline method KNN shows that in some cases it has a tendency to create (inaccurate) recommendations that are dominated by tracks of a few artists. This happens in particular when the playlist to be continued contains a mix of popular and less popular artists. In these cases, re-increasing the artist diversity can, therefore, even be advantageous for accuracy. Another reason is that the Last.fm dataset contains a number of single-artist playlists. For such playlists, our PRA method—in an attempt to reproduce the characteristics of the playlist beginning—minimizes the number of artists and thereby increases the chance of finding suitable tracks.

Multi-objective Optimization Results. Multiple-target optimizations produce good results for both baseline algorithms and in most cases can also improve accuracy. For example, optimizing KNN simultaneously in all quality dimensions leads to improvements in the dimensions of popularity, loudness, and diversity. At the same time, Recall is increased by 26.9 % (CAGH) and 17.3 % (KNN), respectively.

However, we can again observe combinations that represent trade-offs, e.g., release years and tempo (circled in blue), that cannot be resolved given only a limited set of elements for re-ranking. Improving one measure—independently of the used optimization method—then inevitably means that we have to compromise on other aspects. The importance of the each of the quality characteristics for the user depends on the domain.

3.5. Summary of Effectiveness Analysis

Our results show that PRA is effective in different domains with respect to aligning recommendation lists with user tendencies. In the movie domain, the alignment leads to negligible accuracy degradations, and in the music domain, it even helps to significantly improve accuracy.

[4] Such improvements over an already optimized list are possible because here we optimize the top-n lists for the individual user and not for the whole user community in parallel.

4. Empirical Evaluation – Additional Analyses

4.1. Comparison with an Optimal Re-Ranking

Our PRA optimization method is *greedy* in the sense that it does not systematically explore all possible item exchanges. As a result, the algorithm can end up in a local minimum. However, exploring all possible exchanges is infeasible except for small exchange list sizes. In this section, our goal is thus to analyze how much of the existing optimization potential is tapped by our method.

Method. To quantify the harnessed optimization potential, we considered a number of single-objective optimization configurations from the movie domain. We again took the accuracy-optimized baseline methods as described above and determined the best possible value for the given optimization goal by exhaustively exploring all possible exchanges and the resulting recommendation lists.

Since the number of possible exchanges grows exponentially with the exchange set size $|X_u|$, we limited it to 10. We then also ran PRA with $|X_u| = 10$ and analyzed how close PRA comes to the optimal value. Since PRA is comparably fast due to its greedy nature, we additionally ran it with $|X_u| = 20$, which means that PRA had more flexibility in the optimization process. We finally report the accuracy values (in terms of F1) to see if measurable differences can be observed. Given the limited size of the exchange set, the effects are in general expected to be small.

Results. The obtained results using BPR as a baseline are shown in Table 3 (results for other baselines are comparable). The rows correspond to different optimization criteria and the columns different measurements. Columns are given in groups of three: The first cell of each group (e.g., the one circled in red) shows the result of the greedy strategy with $|X_u|$ set to 10. The cell right next to it displays the result for the optimal strategy and the final cell of the group shows the measurement for the greedy strategy with $|X_u| = 20$.

When both techniques use an exchange set size of 10, PRA is close to the optimum *in all cases* and in some cases even reaches it, i.e., no significant ($p < 0.05$) differences between the results produced by PRA and the optimum were measurable. Depending on the complexity of determining the distance measures, PRA is about 250 to 1000 times faster than the systematic procedure, and the re-ranking for a user can on average be determined in less than 1 ms. When PRA is configured with an exchange set of size 20, the post-processing time of the optimal procedure increases dramatically due to its exhaustive strategy, which makes it inapplicable in realistic settings. However, the running times of PRA with 20 exchange items only increase

Table 3: Comparison between the greedy PRA method and an optimal re-ranking with BPR as a baseline.

	[PRA: $\|X_u\|=10$ | Optimum: $\|X_u\|=10$ | PRA: $\|X_u\|=20$]								
Target	Rel. impr. of user error			Deterioration of F1@10			Post-proc. time (ms/user)		
Pop	0.81	0.82	0.93	*-.007	*-.009	*-.007	0.4	0.8	104.1
Year	0.61	0.63	0.77	-.000	-.002	.000	0.2	0.4	72.1
ILS	0.92	0.93	0.98	-.002	*-.004	*-.004	0.2	0.3	193.4
EMD (pop.)	0.71	0.71	0.85	*-.004	*-.007	*-.007	0.3	0.8	109.6
EMD (year)	0.56	0.56	0.71	-.002	*-.004	.000	0.3	0.7	89.0

marginally. Given that PRA now has much more flexibility, it can achieve a much better error minimization than in situations where only 10 exchange elements are available. The observed impacts on accuracy are in all cases very small (marked with * when significant at $p < 0.05$).

Overall, the results show that our greedy method not only harnesses most of the existing optimization potential, but is also computationally efficient, which allows us to examine larger exchange sets for optimization. The fast post-processing times of PRA also, in theory, enable its application in time-critical domains like e-commerce scenarios, where it is important to quickly adapt recommendation lists to users' short-term interests.

4.2. Comparison With Previous Methods

In this section, we compare our PRA technique with two other post-processing methods from the literature that aim to optimize the popularity of a recommendation list on a per-user and system-wide basis, respectively. Other re-ranking approaches will be discussed in Section 5.

4.2.1. Personal Popularity Tendency Matching

Personal Popularity Tendency Matching (PPTM) proposed by Oh et al. (2011) is a greedy re-ranking technique that adapts the popularity—as the only possible quality criterion—of a baseline recommendation list to the popularity preference of the individual user. To model the *Personal Popularity Tendency* for the top-rated items of a user, the discrete distribution of the binned popularity values of the items is used. The difference between the user preferences and the recommendation characteristics is then estimated with the help of the Earth Mover's Distance.

In fact, PPTM is designed to be used with the EMD as a distance measure and exploits some properties of the EMD to prune the recommendation candidate set and thus guarantees an optimal solution. Similar to

Table 4: Comparison between PRA and PPTM for the optimization target *EMD (popularity)*. Column "Param." shows the trade-off parameter configuration.

		[Funk-SVD | BPR | FM]								
		Rel. impr. of EMD pop. err.			Deterioration of F1@10			Post proc. time (ms/user)		
	Param.									
PRA	$\|X_U\| = 5$	.27	.53	.53	.000	-.002	.000	0.14	0.14	0.16
	$\|X_U\| = 10$	.48	.71	.72	.000	-.004	-.002	0.26	0.29	0.34
	$\|X_U\| = 20$	.70	.85	.86	.000	*-.007	-.002	0.46	0.75	0.46
	$\|X_U\| = 90$	.94	.97	.96	-.002	*-.009	*-.004	3.80	3.26	3.68
PPTM	$c = 0.5$	.56	.25	.67	.000	*-.002	.000	1.98	1.89	2.29
	$c = 1$	.76	.41	.84	.000	-.002	.000	2.00	1.32	2.92
	$c = 2$	.89	.61	.93	.002	.000	.000	2.04	2.07	2.48
	$c = 10$	.98	.91	.98	*-.002	.000	-.002	1.91	2.11	1.91

PRA, PPTM can, in theory, be applied to optimize other quality factors of a recommendation list, such as the release year. However, the approach is unable to optimize tendencies for which the EMD is not applicable as a distance measure, e.g., the ILS.

To compare the optimization capabilities of PPTM with PRA, we conducted another experiment with the movie dataset. For this analysis, we configured both PPTM and PRA to adapt the recommendation list popularity characteristics to the user tendencies using the EMD as a distance measure. Additionally, we varied the exchange set size $|X_u|$ for PRA and the parameter c for PPTM, which both determine the trade-off between accuracy preservation and the strength of the tendency match.

Table 4 shows the results of the comparison between PRA and PPTM. Overall, both PRA and PPTM can achieve an error reduction larger than 90 % with minimal decreases in accuracy. Additionally, changing the trade-off parameter to put the focus more on retaining accuracy leads to comparable outcomes for both algorithms. The choice of the baseline also seems to have an effect on the algorithms. For example, when the trade-off is set to favor the preservation of accuracy, PPTM deals better with recommendation lists produced by Funk-SVD or FM; PRA optimizes lists generated by BPR more effectively. Because of the efficient implementation of both algorithms, the post-processing times per recommendation list are all small enough for real-time processing.

In conclusion, both PRA and PPTM show comparable abilities to adjust the characteristics of accuracy-optimized recommendation lists to account for the user's popularity tendency. At the same time, both strategies offer the choice between accuracy preservation and stronger tendency matching via a parameter. However, the above mentioned usage limitations of PPTM make it less versatile when quality factors other than

popularity should be optimized. The generic and parameterizable approach of PRA, on the other hand, allows the simultaneous optimization of a variety of different quality criteria, which can easily be tailored to the specific application domain.

4.2.2. System-Wide Effects

So far, we focused our effectiveness analysis of PRA on user-wise metrics that quantify the difference between the characteristics of the user's sample set items and the recommendation list items. However, optimizing the popularity and diversity levels for each user can also have system-wide effects, e.g., when measuring the average item popularity across the top-10 lists of all users.

To analyze such global effects, we again compare PRA with PPTM and measure the impact both methods have on various system-wide metrics. In addition, we include the post-processing method of Adomavicius and Kwon (2012) in our comparison, which is explicitly designed to increase the *system-wide diversity*. Their post-processing scheme, called Item-Popularity-Based Ranking (IPBR), accomplishes this by re-ranking the top of each recommendation list according to the inverse popularity of the items. Similar to PRA, the overall accuracy of the recommendations is ensured by only considering items for re-ranking whose predicted rating is higher than a threshold T_R.

Figures 2 and 3 show the impact of PRA, PPTM, and Adomavicius and Kwon's IPBR on system-wide metrics, such as the average popularity across all top-10 lists.[5] Considering popularity aspects, IPBR produces the expected results, i.e., the post-processing method reduces the average popularity of the recommended items to promote long-tail items and thus increase aggregate diversity. PRA and PPTM also reduce the average popularity when Factorization Machines are used as a baseline but have the opposite effect when Funk-SVD's lists are post-processed. The explanation for this phenomenon is that the baseline algorithms—while sometimes being similar in terms of accuracy—often place quite different items in their top-n lists. Funk-SVD for example is more prone to recommend niche items, while FM and BPR have a stronger tendency to recommend popular items (Jannach et al., 2015). However, the average of the users' tendencies towards popularity seems to be somewhere in between what the algorithms recommend. Thus, PRA and PPTM try to adjust the recommendations towards this point in between, which can mean an increase or decrease of popularity. In contrast, the user-agnostic IPBR strategy always lowers the popularity to achieve a higher system-wide diversity.

[5] The results do not contain BPR as a baseline because IPBR only works for rating prediction algorithms.

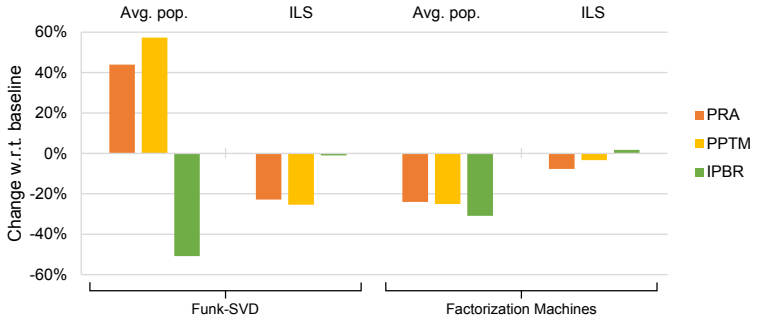

Figure 2: System-wide effects of PRA, PPTM, and Adomavicius and Kwon's IPBR on average popularity and intra-list similarity with FUNK-SVD and FM as baselines. Parameters: $|X_u| = 20$ (PRA); $c = 2$ (PPTM); $T_R = 4.5$ (IPBR).

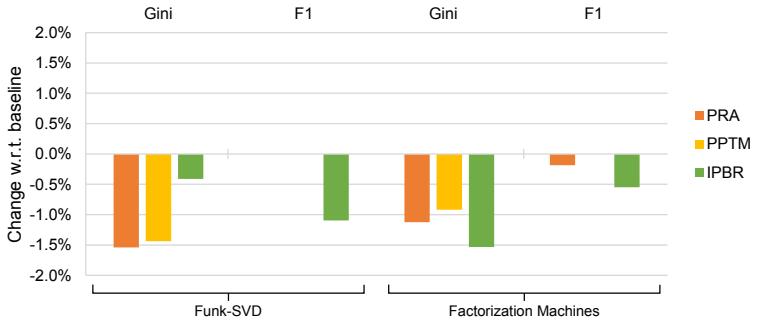

Figure 3: System-wide effects of PRA, PPTM, and Adomavicius and Kwon's IPBR on the Gini coefficient and the F1 measure with FUNK-SVD and FM as baselines. Parameters: $|X_u| = 20$ (PRA); $c = 2$ (PPTM); $T_R = 4.5$ (IPBR)

In addition to the average popularity, we also measured the effect of the different post-processing strategies on the ILS, as well as the Gini Index, and the F1 measure. PRA and PPTM again perform very similarly in that they, e.g., both increase diversity, reduce concentration, and have only negligible effects on accuracy. In contrast, IPBR does not have a noticeable effect on the ILS, and while it reduces concentration, the effect strength seems to depend on the baseline algorithm. Additionally, even with a conservative threshold setting of $T_R = 4.5$, we can observe a stronger accuracy deterioration when IPBR is used, which is expected since the user's tendencies are not taken into account. In contrast to PRA and PPTM, this can lead to strong accuracy decreases when the trade-off parameter is not well-chosen, e.g., a decrease of 8 % with $T_R = 4$ and FM as a baseline (not shown in Figure 3).

5. Previous Work

Our PRA method is a re-ranking strategy to adapt recommendations to user tendencies. A number of comparable post-processing and ranking techniques for a global or user-wise adaptation of different quality characteristics have been proposed in the past. The commonalities and differences to our approach can be summarized as follows.

Two-Factor Trade-Off Approaches. As mentioned before, the re-ranking approach of Adomavicius and Kwon (2012) aims to lower the system-wide popularity of the recommendations and thereby decrease the concentration bias. The focus of Bradley and Smyth (2001) and Ziegler et al. (2005), in contrast, is to increase the diversity of individual lists, and the items are correspondingly re-ranked based on their dissimilarity to each other. Lastly, Zhang and Hurley (2008) treat the trade-off between accuracy and diversity as a binary optimization problem. All these re-ranking approaches focus on the balance of accuracy and one additional quality factor, e.g., the (aggregate) diversity or the popularity of the recommendations. Our PRA method, in contrast, is able to optimize multiple goals besides maintaining high accuracy at the same time.

An alternative way of balancing potential trade-offs is to incorporate, e.g., diversity-enhancing mechanisms into the algorithms themselves. For example, in the work of Said et al. (2013), a "furthest-neighbor" method was designed and compared to a classic KNN method. Zhou et al. (2010), on the other hand, proposed a graph-based approach to avoid the recommendation of too similar items. Our method, in contrast, is independent from the inner workings of the underlying algorithms and allows us to fine-tune the desired level of, e.g., diversity. Our work is similar to these approaches w.r.t. the aim of balancing optimization goals, but is (a) not limited to one specific and single optimization goal and (b) furthermore capable of tailoring the recommendations on a per-user basis.

User-Wise Two-Factor Approaches. A user-wise re-ranking scheme was previously proposed by Jambor and Wang (2010). In their approach, the re-ranking is done with a per-user linear optimization scheme to change the relevance scores of an exchangeable baseline technique like SVD. The proposed approach promotes items from the long-tail and models the popularity preferences on a per-user basis similar to our PRA method. However, the approach re-ranks the entire recommendation list, while PRA focuses on optimizing only the head of the recommendation list to ensure accuracy. Additionally, their strategy is limited to rating prediction algorithms as baselines, while our re-ranking approach also works with recent learning-to-rank methods.

The PPTM re-ranking discussed in Section 4.2.1 also adapts popularity tendencies on a per-user basis but requires that the baseline algorithm provides recommendation scores. The framework proposed by Shi

et al. (2012), on the other hand, is specifically designed to adapt the *diversity* of the recommendations to the user profile of individual users. Similar to the PPTM method, their approach can balance accuracy with exactly one other optimization goal, by means of a weighted ranking technique in their case. Vargas et al. (2011) proposed a similar re-ranking strategy that also aims to increase recommendation list diversity. To assess the diversity of the list, they use the items' latent feature values extracted from a matrix factorization approach. They then balance the accuracy and list diversity, given the user's tendencies toward the latent feature values of the items in the list. In the work of Kapoor et al. (2015), finally, a personalized *novelty* preference is calculated for each user. The final item ranking is based on a trade-off between familiarity and novelty. In contrast to our approach, in which "content" information about the items is used to determine user preferences, these works use the variance of latent factors as a proxy for the diversity tendency and a time-based criterion for the novelty preferences, respectively.

Multi-Factor Optimization. Similar to PRA, some existing approaches incorporate more than one characteristic in the adaptation process. Zhang et al. (2012), for example, merge multiple algorithms through rank interpolation to optimize accuracy, novelty, diversity, and serendipity in the music domain. In contrast to our work, their optimization does not use exchangeable baseline methods but implements the optimizations inside the algorithms. Similarly, Ribeiro et al. (2014) combine multiple recommendation algorithms to optimize accuracy, diversity, and novelty and use a ranking and a hybridization method to calculate a Pareto frontier that represents the best possible combination of the algorithms in terms of the three optimization goals. However, both Ribeiro et al. (2014) and Zhang et al. (2012) differ from our work in that their recommendations are based on aggregated factors and not tailored to the individual user's preferences. Additionally, the approach of Ribeiro et al. (2014) is designed to be pre-calculated offline instead of on-the-fly based on recent user interactions, because of the high computational cost of determining the elements in the Pareto frontier.

A Structured Comparison with Previous Works. Table 5 presents the results of a structured comparison between our re-ranking method and other multi-factor optimization approaches from the literature. In this table, we provide a short description of each algorithm along with the following key characteristics: the level at which the optimization is performed (user-wise or global), the set of quality criteria that can be optimized, and the extent to which the optimization method is integrated into the underlying recommendation technique (fully integrated, hybridization, or re-ranking).

The results show that a number of approaches exist that are similar to PRA in that they either (a) tailor

Table 5: Past approaches from RS research that amplify or adapt alternative quality criteria in recommendation lists, e.g., diversity. The approaches are categorized as integrated algorithms (A), hybridization schemes (H) and re-ranking techniques (R).

Approach	Scope	Criteria	Method
"Bounded Greedy Selection" by Bradley and Smyth (2001)	Global	Diversity	R: Re-ranking of all recommendations with a greedy dissimilarity selection
"Sequential Topic Diversification" by Ziegler et al. (2005)	Global	Diversity	R: Re-ranking of all recommendations with a dissimilarity rank calculation
"SUGGEST" by Zhang and Hurley (2008)	Global	Diversity	R: Re-ranking of all recommendations with a binary optimization scheme
"Long Tail Constraint" by Jambor and Wang (2010)	Per-User	Popularity, (Potentially Others)	R: Re-ranking of all recommendations with an optimization for popularity adaption
"HeatS" by Zhou et al. (2010)	Global	Diversity	A: Integrated algorithm with a graph-based diversity modeling approach
"Personal Popularity Tendency Matching" by Oh et al. (2011)	Per-User	Quantitative Criteria (EMD applicable)	R: Re-ranking of first n items based on the Earth Mover's Distance to the user profile
"Intent-Oriented Diversity" by Vargas et al. (2011)	Per-User	Diversity	R: Re-ranking of first n items based on (latent) item feature similarity
"Item Popularity-Based Ranking" by Adomavicius and Kwon (2012)	Global	Aggregate Diversity	R: Re-ranking of first n items based on popularity and other indicators
"Latent Factor Portfolio" by Shi et al. (2012)	Per-User	Diversity	A: Integrated algorithm with latent factors as a proxy for diversity
"Auralist" by Zhang et al. (2012)	Global	Accuracy, Novelty, Diversity, Serendipity	H: Hybridization of multiple algorithms with a rank interpolation approach
"Furthest Neighbors" by Said et al. (2013)	Global	Diversity	A: Integrated algorithm with a kNN variant of dissimilar neighbors
"Pareto-Efficient Hybridization and Ranking" by Ribeiro et al. (2014)	Global	Diversity, Novelty	H: Hybridization of multiple algorithms with a Pareto optimization scheme
"Adaptive Novelty Recommendation" by Kapoor et al. (2015)	Per-User	Novelty	A: Integrated algorithm that takes the current user's novelty tendency into account
"**Personalized Ranking Adjustment**"	Per-User	Diversity, Popularity, Others	R: Re-ranking of first n items with a greedy item swapping scheme

their optimization goal to the user's tendencies or (b) allow the optimization of multiple goals. However, among the listed previous works, only the post-processing schemes of Jambor and Wang (2010) and Oh et al. (2011) feature both a *per-user* optimization scheme and the consideration of *multiple* optimization criteria. However, the optimization algorithms implemented in these two approaches are constrained in terms of the type of criteria that can potentially be optimized (e.g., PPTM can only optimize criteria for which the EMD can be calculated). In contrast, our proposed PRA approach is able to adapt the recommendations w.r.t. *a variety* of quality factors *simultaneously* and on a *per-user* basis.

6. Practical Implications

In recent years, it has become increasingly obvious that in many domains selecting items for recommendation only based on their estimated relevance for the user can be insufficient, e.g., when the elements of the resulting recommendation lists are too similar to each other. Such suggestions can be of limited value both for the consumer of the recommendation service, who finds the lists boring, and for the service provider, because the recommendations do not help the user to discover other parts of the catalog spectrum (Jannach and Adomavicius, 2016). Accordingly, different algorithmic proposals have been put forward in the last few years that aim to balance prediction accuracy with other potential quality factors like diversity or novelty. However, past algorithmic approaches—as indicated by our review of related work—often have different limitations and are, for instance, only applicable in combination with certain machine learning algorithms. Or, they can only consider accuracy and exactly one additional quality factor. Furthermore, most of the existing approaches balance different quality factors in the same way for all users, independent of each user's particular preference profile. The approach proposed in this paper addresses several of these practical challenges.

- Our method adopts a two-stage recommendation strategy, in which the initial ranking can be determined by any underlying recommendation algorithm. This allows the method to be applied both by recommendation service providers who want to rely on the latest and most sophisticated machine learning approaches as well as providers who prefer algorithms that are easier to develop and maintain. In fact, the winning algorithms developed in the Netflix Prize competition never made it into production, partly because of the required engineering efforts.[6]

[6] http://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html

- Our method can be configured to work in a greedy manner and our experiments show that the greedy optimization approach leads to solutions that are very close to the optimum. In particular, when this greedy strategy is used, the computational complexity is low and a close-to-optimal re-ranking can be computed under the narrow time constraints of online recommendation.

- Our method is not limited to one single quality factor or a certain set of quality factors. The only requirement when applying a quality measure in the optimization process is that this quality factor can be numerically quantified based on the items in a recommendation list. This allows service providers to implement application-specific or business-driven metrics into the optimization process as well. From a business perspective it could, for example, be desirable to include recommendations that both match the customers' preferences and, at the same time, lead to increased profit on the provider's side. To implement this strategy, the only requirement is the definition of a function that captures the business value of a given set of items.

Overall, our method goes beyond "one-size-fits-all" approaches that apply system-wide goals when balancing different quality factors. Our method takes the past preference patterns of individual users into account in the re-ranking process and thereby opens new opportunities to build next-generation recommendation services.

References

Adomavicius, G., Kwon, Y., 2012. Improving aggregate recommendation diversity using ranking-based techniques. IEEE Transactions on Knowledge and Data Engineering 24 (5), 896–911.

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

Bradley, K., Smyth, B., 2001. Improving Recommendation Diversity. In: Proceedings of the 12th Irish Conference on Artificial Intelligence and Cognitive Science (AICS '01). pp. 75–84.

Chau, P. Y. K., Ho, S. Y., Ho, K. K. W., Yao, Y., 2013. Examining the effects of malfunctioning personalized services on online users' distrust and behaviors. Decision Support Systems 56, 180–191.

Ehrgott, M., 2006. Multicriteria optimization. Springer Science & Business Media.

Ekstrand, M. D., Harper, F. M., Willemsen, M. C., Konstan, J. A., 2014. User perception of differences in recommender algorithms. In: Proceedings of the 8th ACM Conference on Recommender Systems (RecSys '14). pp. 161–168.

Funk, S., 2006. Try this at home. http://sifter.org/~simon/journal/20061211.html, last accessed: 06/2016.

Hariri, N., Mobasher, B., Burke, R., 2012. Context-Aware Music Recommendation Based on Latent Topic Sequential Patterns. In: Proceedings of the 6th ACM Conference on Recommender Systems (RecSys '12). pp. 131–138.

Iaquinta, L., de Gemmis, M., Lops, P., Semeraro, G., Filannino, M., Molino, P., 2008. Introducing serendipity in a content-based recommender system. In: Proceedings of the 8th International Conference on Hybrid Intelligent Systems (HIS '08). pp. 168–173.

Jambor, T., Wang, J., 2010. Optimizing multiple objectives in collaborative filtering. In: Proceedings of the 4th ACM Conference on Recommender Systems (RecSys '10). pp. 55–62.

Jannach, D., Adomavicius, G., 2016. Recommendations with a purpose. In: Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16). pp. 7–10.

Jannach, D., Lerche, L., Kamehkhosh, I., Jugovac, M., 2015. What recommenders recommend: an analysis of recommendation biases and possible countermeasures. User Modeling and User-Adapted Interaction 25 (5), 427–491.

Jannach, D., Resnick, P., Tuzhilin, A., Zanker, M., 2016. Recommender systems — beyond matrix completion. Commun. ACM 59 (11), 94–102.

Kapoor, K., Kumar, V., Terveen, L. G., Konstan, J. A., Schrater, P. R., 2015. "I like to explore sometimes": Adapting to dynamic user novelty preferences. In: Proceedings of the 9th ACM Conference on Recommender Systems (RecSys '15). pp. 19–26.

Marler, R., Arora, J., 2004. Survey of multi-objective optimization methods for engineering. Structural and Multidisciplinary Optimization 26 (6), 369–395.

McNee, S. M., Riedl, J., Konstan, J. A., 2006. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In: Proceedings of the 2006 Conference on Human Factors in Computing Systems (CHI '06). pp. 1097–1101.

Oh, J., Park, S., Yu, H., Song, M., Park, S., 2011. Novel recommendation based on personal popularity tendency. In: Proceedings of the 11th IEEE International Conference on Data Mining (ICDM '11). pp. 507–516.

Rendle, S., 2012. Factorization machines with libFM. ACM Transactions on Intelligent Systems and Technology 3 (3), 1–22.

Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L., 2009. BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI '09). pp. 452–461.

Ribeiro, M. T., Ziviani, N., Moura, E. S. D., Hata, I., Lacerda, A., Veloso, A., 2014. Multiobjective pareto-efficient approaches for recommender systems. ACM Transactions on Intelligent Systems and Technology 5 (4), 1–20.

Said, A., Fields, B., Jain, B. J., Albayrak, S., 2013. User-centric evaluation of a k-furthest neighbor collaborative filtering recommender algorithm. In: Proceedings of the 2013 Conference on Computer Supported Cooperative Work (CSCW '13). pp. 1399–1408.

Sarroff, A. M., Casey, M., 2012. Modeling and predicting song adjacencies in commercial albums. In: Proceedings of the 9th Sound and Music Computing Conference (SMC '12). pp. 364–371.

Shi, Y., Zhao, X., Wang, J., Larson, M., Hanjalic, A., 2012. Adaptive diversification of recommendation results via latent factor portfolio. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'12). pp. 175–184.

Steck, H., 2011. Item popularity and recommendation accuracy. In: Proceedings of the 5th ACM Conference on Recommender Systems (RecSys '11). pp. 125–132.

Vargas, S., Castells, P., 2011. Rank and relevance in novelty and diversity metrics for recommender systems. In: Proceedings of the 5th ACM Conference on Recommender Systems (RecSys '11). pp. 109–116.

Vargas, S., Castells, P., Vallet, D., 2011. Intent-oriented diversity in recommender systems. In: Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '11). pp. 1211–1212.

Zhang, M., Hurley, N., 2008. Avoiding monotony: Improving the diversity of recommendation lists. In: Proceedings of the 2008 ACM Conference on Recommender Systems (RecSys '08). pp. 123–130.

Zhang, Y. C., Séaghdha, D. O., Quercia, D., Jambor, T., 2012. Auralist: Introducing serendipity into music recommendation. In: Proceedings of the 5th International Conference on Web Search and Web Data Mining (WSDM '12). pp. 13–22.

Zhou, T., Kuscsik, Z., Liu, J.-G., Medo, M., Wakeling, J. R., Zhang, Y.-C., 2010. Solving the apparent diversity-accuracy dilemma of recommender systems. Proceedings of the National Academy of Sciences 107 (10), 4511–4515.

Ziegler, C.-N., McNee, S. M., Konstan, J. A., Lausen, G., 2005. Improving recommendation lists through topic diversification. In: Proceedings of the 14th International Conference on World Wide Web (WWW '05). pp. 22–32.