A New Collaborative Filtering Approach for Increasing the Aggregate Diversity of Recommender Systems

Katja Niemann, Martin Wolpers
Fraunhofer Institute for Applied Information Technology FIT
Schloss Birlinghoven
53754 Sankt Augustin, Germany
{katja.niemann, martin.wolpers}@fit.fraunhofer.de

ABSTRACT

In order to satisfy and positively surprise its users, a recommender system needs to recommend items the users will like and most probably would not have found on their own. This requires the recommender system to recommend a broader range of items including niche items. Such an approach also supports online-stores that often offer more items than traditional stores and need recommender systems to enable users to find the not so popular items as well. However, popular items that hold a lot of usage data are more easy to recommend and, thus, niche items are often excluded from the recommendations. In this paper, we propose a new collaborative filtering approach that is based on the items' usage contexts. That is to say, an item is described by the items it is significantly often used with rather than by its users or content attributes. The approach increases the rating predictions for niche items with fewer usage data available and improves the aggregate diversity of the recommendations.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Retrieval and Search—*information filtering*

Keywords

Aggregate Diversity, Item-Item Similarity, Long Tail, Niche Items, Recommender Systems, Usage Context

1. INTRODUCTION

Recommender systems are steadily becoming more important in an expanding number of domains (movies, music, books, etc.) to filter relevant items for users. Recommending relevant items alone, though, is often not sufficient to satisfy user expectations, but other characteristics, such as diversity, novelty, serendipity and trust must be considered as well [4, 21, 27].

We focus on increasing the aggregate diversity, i.e. the number of distinct items recommended across all users [3], by improving the calculation of the expected ratings especially for niche items. It has been shown that popular items are recommended disproportionately often because they provide extensive usage data and, thus, can be recommended to more users [17, 32]. However, a system focussing on providing a wider range of items and not mainly popular items more likely recommends novel and diverse items to its users which often favour recommendations that are for items they would not have thought of by themselves [4, 27]. A large recent study analysing rating data shows that, for instance in the case of movies, users regularly give high ratings to niche items, suggesting that users value speciality items [19].

A broader range of recommended items is not only important for the users' satisfaction but plays an important role in online stores as well [8, 20]. Many markets have historically been dominated by a small number of bestselling products (Pareto Principle [8]). Internet markets, though, exhibit a significantly less concentrated sales distribution. Anderson describes the phenomenon that niche products can grow to become a large share of total sales as "The Long Tail" [5]. There are two explanations for this phenomenon. First, an online store can easily offer a larger amount of items than a traditional store. Second, by using tools such as recommender systems, users can be encouraged to buy items they would not have found by themselves [8]. For some online stores it might even be more beneficial to recommend niche items. For instance, Netflix[1] could encourage users to rent movies from the long tail, which are less costly to license than blockbusters [20]. However, a higher aggregate diversity often lowers the accuracy since it requires the recommendation of idiosyncratic items as well [3].

We address the task of recommending items from the long tail and increasing the aggregate diversity without lowering the accuracy by introducing a new collaborative filtering approach - called usage context-based collaborative filtering (UC-BCF) - where an item is described by the items it significantly often co-occurs with rather than by its users or content attributes. The idea is taken from linguistics where relations between words can be inferred from the words' contexts, e.g. the words they co-occur with in sentences [12].

The task of finding correlations between items in a dataset is well-known as association mining [10]. Association mining started with the analysis of shopping basket data to better understand and target the consumers' behaviour by discovering inferences and rules between items and item sets, e.g. a user who buys item A and item B also buys item C with a probability of 80%. Since then, association mining has

[1]http://netflix.com/

KDD'13, August 11–14, 2013, Chicago, Illinois, USA.

been applied to many different domains in which the relationships between objects can provide useful knowledge, e.g. risk analysis and clinical medicine.

However, our approach differs from association mining in that we do not assume two items to be related if they co-occur with each other, but if they significantly often co-occur with the same items. Thus, two items can be highly related even if they were never used together. For example, items A, B, and C are often used together as well as items A, B, and D. Thus, we assume the items C and D to be highly related because they are described through the same co-occurrences, here items A and B. Using this approach, we are able to create characteristic vectors also for rarely used items and to improve the calculation of the niche items' rating predictions. Thus, the aggregate diversity of recommendations can be improved without lowering the accuracy.

The rest of the paper is structured as follows: In chapter 2 we give an overview of recommendation approaches that try to increase the diversity of recommendations and compare them to the usage context-based collaborative filtering that is specified in chapter 3. In chapter 4 we describe the set-up of the experiments including the used data sets and evaluation metrics to discuss the results of the experiments in detail in chapter 5. Finally, in chapter 6, we summarize the presented work and give an outlook on future work.

2. RELATED WORK

Recently, several recommendation approaches have been invented that do not focus on accuracy alone but on new evaluation metrics such as diversity, novelty, unexpectedness, and serendipity of the recommended items to increase the user satisfaction [4, 21, 27]. So far, most work has been conducted for increasing the diversity which is divided in the individual and the aggregate diversity. The individual diversity describes the diversity of the items in a user's recommendation list, thus, increasing the individual diversity means avoiding overspecialisation. Strategies developed so far for increasing the individual diversity mostly calculate the quality of an item based on its similarity to the user and its dissimilarity to the items that are already selected for this user's recommendation list [7, 33, 34]. The aggregate diversity describes the total amount of items recommended to the users. Improving the individual diversity does not mean improving the aggregate diversity, e.g. if each user gets recommendations for the same 10 movies from 10 different genres, the individual diversity is high, but the aggregate diversity is still low. In the following, we will describe approaches that increase the aggregate diversity of recommender systems and compare these to our approach.

There are two lines of research that try to improve the aggregate diversity. The first line calculates the rating predictions using existing filtering approaches to then re-rank the items with the highest predicted ratings to push items from the long tail in the recommendation lists. The second line of research tries to improve the estimation process especially for rarely used items. We first present two approaches from the first and then two approaches from the second line.

Adomavicius and Kwon [3] propose to re-rank the list of candidate items for a user to improve the aggregate diversity. First, an ordered list of recommendations is calculated using any filtering technique. Second, for all items having a better expected rating than a given threshold, additional features are calculated, for instance the absolute and relative likeability of an item (how many users liked the item among all users or among all users who rated that item, respectively) and the item's rating variance. According to these features, the candidate items are re-ranked and only the top-N items are recommended. This way, niche items are pushed to the recommendation lists and very popular items are rejected. While this re-ranking technique can improve the aggregate diversity, it comes at the expense of accuracy.

In [2], Adomavicius and Kwon propose a graph-theoretic approach to select the top-N items for each user while yielding the maximum aggregate diversity. At first, the system calculates all candidate items for each user by applying any filtering technique. Then, a bipartite graph is created with the users and the selected items as vertices connected through edges that hold the predicted ratings as edge weights. By solving the "maximum bipartite matching problem" using a suitable algorithm, the maximum aggregate diversity is reached as each item is restricted to only one user. Users that receive less than N items in that process also receive recommendations for items that scored best in the first step. The accuracy and diversity of this graph-theoretic approach depends on the selection criteria for the items included in the graph. The more items are selected as candidate items for each user, the more diverse and less accurate are the recommendations and vice versa.

Park and Thuzhilin [28] group items from the long tail into clusters based on their attributes (e.g. name, description, price) to compensate the missing usage information. For each cluster, they build a model predicting ratings based on the known ratings and the items' attributes. Items from the head are not clustered, but for each of these items an individual predictive model is built. The splitting of the head and the tail is based on features like the items' average ratings or their popularity. This approach improves the estimation of rating predictions, but also relies on additional semantic metadata which is often not available.

Levy and Bosteels [25] explicitly push items from the long tail (here: songs from not well known artists) in the recommendation lists. After analysing the last.fm data set[2] they define each artist with less than 10.000 listeners as long tail artist. For each artist in the last.fm data set they calculate the k most similar long tail artists based on their listening history and tags applied to each artist. In order to create recommendations for a user, the similarities between the long tail artists and the artists in the user profile are calculated and the user gets recommendations for songs from those long tail artist that are most similar to her favourite artists. No evaluation has been published so far.

The UC-BCF approach we propose tries to improve rating predictions especially for niche items and thus falls in the latter research line mentioned above. In contrast to the other approaches of this line, however, it does not require any semantic metadata (which is often not available or imcomplete) but calculates the item vectors based on the items' usage contexts. Thus, it is not a hybrid but a new collaborative filtering approach. The first research line reranks candidate items based on their usage to push niche items from the long tail with the predicted ratings being calculated using any filtering technique. When using the UC-BCF approach to calculate the predicted ratings, these approaches can benefit from it as we show in subsection 5.3.

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

3. USAGE CONTEXT-BASED RECOMMENDATION

3.1 Introducing Usage Context

Similar to user- and item-based collaborative filtering, the UC-BCF takes the user-item matrix as input. However, before calculating the item pair similarities, the user-item matrix is transformed into an item-item matrix. In this matrix, each item is described by the k items it significantly often co-occurs with in a user profile. After the matrix transformation, the item pairs are compared using traditional information retrieval approaches, e.g. by calculating the cosine similarity of their vectors. This way, the UC-BCF combines features of the user- and the item-based approaches:

1) If two items share a significant amount of users that rated them similarly, they are described through similar item vectors because they often co-occur with the same items in a user profile.

2) If two items are rated similarly by a significant amount of similar users (not necessarily the same users), they are also described by similar item vectors because similar users co-rated the same items similarly.

Additionally, there is a third feature not contained in the user- or item-based approach:

3) If two items are rated dissimilarly by a significant amount of dissimilar users, they hold similar item vectors, because dissimilar users co-rated the same items dissimilarly.

Let's take the following examples to clarify the mechanisms of the UC-BCF. Table 1 shows the ratings of the two users u_1 and u_2 for the movies m_1 - m_5. For this example, the user-based collaborative filtering (UBCF) approach recommends movie m_2 to user u_1 and movie m_1 to user u_2. The item-based collaborative filtering (IBCF) approach fails because of the items' usage data sparsity.

Table 1: Exemplary Rating Data

	m_1	m_2	m_3	m_4	m_5
u_1	5	-	4	5	3
u_2	-	5	4	4	2

The UC-BCF approach first transforms the user-item matrix into an item-item matrix by calculating the most significant co-occurences for each item. For simplicity in this small example, we consider movies to be significant co-occurrences if at least one user rated them similarly (here: with a difference of 1 star or less). Additionally, we just use 0 and 1 for stating if movies are co-occurrences or not, later, we will exchange the binary attribute for the significance value of the co-occurrence. Table 2 shows the vectors describing the movies m_1 - m_5 calculated this way.

Table 2: Item Vectors

	m_1	m_2	m_3	m_4	m_5
m_1	0	0	1	1	0
m_2	0	0	1	1	0
m_3	1	1	0	1	1
m_4	1	1	1	0	0
m_5	0	0	1	0	0

The movies m_1 and m_2 are described by the same item vectors, therefore they receive a high similarity score and users who like m_1 are assumed to like m_2 as well and vice versa (of course also depending on the user's other ratings).

Table 3 shows exemplary rating data for the users u_3, u_4 and u_5 on the movies m_6 - m_9. Here, the UCBF suffers from the sparsity of user u_5's profile. The IBCF recommends movie m_6 to user u_5.

Table 3: Exemplary Rating Data

	m_6	m_7	m_8	m_9
u_3	4	5	5	3
u_4	2	2	4	2
u_5	-	4	-	-

The item vector calculation of the UC-BCF for the movies m_6 - m_9 results in the vectors given in table 4. Similar to the item-based approach, the movie m_6 gets recommended to user u_5.

Table 4: Item Vectors

	m_6	m_7	m_8	m_9
m_6	0	1	1	1
m_7	1	0	1	1
m_8	1	1	0	0
m_9	1	1	0	0

The last exemplary rating data is given in table 5. The users u_6 and u_7 are very dissimilar according to their ratings. Therefore, the ratings of u_6 will not be considered when calculating recommendations for user u_7 and vice versa when using a user-based approach. The item-based approach suffers from sparsity as in the first example.

Table 5: Exemplary Rating Data

	m_{10}	m_{11}	m_{12}	m_{13}
u_6	2	-	3	1
u_7	-	5	5	4

The transformation of this matrix into the usage context-based approach's item-item matrix results in the vectors given in table 6, which shows that the movies m_{10} and m_{11} are assumed to be similar. This is because the movies m_{12} and m_{13} were co-rated similarly (1 star difference or less) with the movies m_{10} and m_{11} by users u_6 and u_7. Here, it does not matter that u_6 rated the movies low and user u_7 rated them high. Thus, the approach would assume user u_7 to like movie m_{10} because he likes movie m_{11}.

Table 6: Item Vectors Excerpt

	m_{10}	m_{11}	m_{12}	m_{13}
m_{10}	0	0	1	1
m_{11}	0	0	1	1
m_{12}	1	1	0	1
m_{13}	1	1	1	0

These simple examples show, how the UC-BCF can combine the strength of the two standard UBCF and IBCF approaches and compensate their weaknesses. In the next sections, we will discuss how to choose the most significant co-occurrences for an item and how to calculate the similarity of two items based on their co-occurrences.

3.2 Significant Co-occurrences

We define two items to be co-occurrences if they are contained in at least one user profile in which they are rated similarly. We assume two items to be rated similarly if their rating difference is below a given threshold t or if both items are rated above or below the user's average rating. Not every co-occurrence is significant, rather most co-occurrences are coincidental, thus, we need to calculate a significance score for each co-occurring item pair.

Basic association measures calculate a significance score by comparing the observed frequency O of a co-occurrence with its expected frequency E, e.g. MI (mutual information) and z-score, see [26]. These simple association measures often give close approximation to the more sophisticated association measures (as described below) and are therefore sufficient for many applications. They also have some limitations as they, e.g. tend to fail when calculating the significance value for a frequent and an infrequent object [16].

In statistical theory, association measures are always based on a cross-classification of a set of items, e.g. using contingency tables. Table 7 shows the contingency table for the items i and j which are co-rated O_{11} times. Additionally, i was rated by O_{12} users who did not rate j, j was rated by O_{21} users who did not rate i and O_{22} users who did not rate any of these items at all. The expected values for these observed values are E_{11}, E_{12}, E_{21}, and E_{22}, respectively.

Table 7: Contingency table

	j	$\neg\, j$	
i	O_{11}	O_{12}	$=R_1$
$\neg\, i$	O_{21}	O_{22}	$=R_2$
	$=C_1$	$=C_2$	$=N$

Commonly used association measures that are based on contingency tables are the χ^2 test and log-likelihood, see [26]. The χ^2 test adds up the squared z-scores for each cell in the contingency table and puts it in relation to the expected frequencies. Since the normal approximation implicit in the z-scores becomes inaccurate if any of the expected frequencies are small [16], the Yates' continuity correction [13] shown in equation 1 offers a better approximation (corrected χ^2-test). Equation 2 shows the log-likelihood measure [14].

$$\chi^2\text{-corr} = \frac{(|O_{11}O_{22} - O_{12}O_{21}| - \frac{N}{2})^2}{R_1R_2C_1C_2} \tag{1}$$

$$\text{log-likelihood} = 2\sum_{ij} O_{ij} ln\frac{O_{ij}}{E_{ij}} \tag{2}$$

The association measure shown in equation 3 is based on the Poisson distribution. [22] gives a formal proof that justifies the assumption of a Poisson distribution for co-occurring objects in a data set if the frequency of most objects is much smaller than the size of the data set. Using the Poisson distribution requires the calculation of the faculty of the natural logarithm of O_{11}, which is numerically hard to handle for a large O_{11}, thus, using an approximation such as the Stirling's formula, it can be simplified [6].

$$\begin{aligned} &poisson = O_{11}(lnO_{11} - ln\lambda - 1) + \frac{1}{2}ln(2\pi O_{11}) + \lambda \\ &with\ \lambda = \frac{R_1C_1}{N} \end{aligned} \tag{3}$$

3.3 Detection of a Suitable Significance Threshold

After the calculation of the co-occurrences' significance values, the most significant ones must be selected for each item by choosing its k most significant co-occurrences or by using a threshold. Since there is no standard scale of measurement to draw a clear distinction between significant and non-significant co-occurrences [15], the calculation of a suitable k or a threshold is an exploratory investigation. In our experiments, we will vary the vector sizes but only consider items if they hold a significance value that is over-average for the described item to avoid noise.

3.4 Item Similarity Calculation

We calculate the similarity for each item pair using the cosine similarity which measures the angle between their co-occurrence vectors, see equation 4. Each item i is described by its vector V_i. The cosine similarity always takes a value between 0 and 1.

$$\text{cosine-sim} = \frac{V_i \cdot V_j}{||V_i||\ ||V_j||} \tag{4}$$

3.5 Expected Rating Calculation

We compute the expected rating $p(u,i)$ on an item i for a user u by averaging the ratings given by the user to the other items in her profile $P(u)$ while each rating is weighted by the corresponding similarity $sim(i,j)$, see equation 5.

$$p(u,i) = \frac{\sum_{j\in P(u), i\neq j}(sim(i,j) * r(u,j))}{\sum_{j\in P(u), i\neq j}|sim(i,j)|} \tag{5}$$

3.6 Computational Complexity

The computational complexity of the UC-BCF approach depends on the amount of time required to build the model (i.e. calculating the item-vectors and the item pair similarities) and the amount of time required to compute the recommendations using this model.

During the model building phase, we first need to compute the co-occurrence vectors by calculating the significance values of all co-occurrences and select the k most significant ones for each item. The upper bound on the complexity of this step is $O(n^2m)$ with n being the amount of items and m being the amount of usage contexts, i.e. user profiles, as we potentially need to calculate $\frac{n*(n-1)}{2}$ significance values, each requiring up to m operations. However, the actual complexity is significantly smaller because we only need to calculate significance values for those item pairs that were actually used together and the user profiles are generally sparse and the rated items are clustered [23]. In the second step of the model building phase, we need to compute the similarity of all item pairs using their co-occurrence vectors. The upper bound on the complexity is $O(n^2k)$ as we need to compute $\frac{n*(n-1)}{2}$ similarities, each potentially requiring k operations, with k typically being small (< 100, see section 5), thus, it can be simplified to $O(n^2)$.

Finally, the upper bound of complexity to compute the top-N recommendations for a given user profile $P(u)$ is described by $O(n|P(u)|)$ as for all items that have a similarity greater than 0 to at least one of the rated items (which is potentially n, but usually much smaller) a predicted rating is calculated by considering all rated items.

4. EXPERIMENTS

This section describes the data sets, the evaluation metrics and the experimental set-up used for the evaluation of the UC-BCF approach in comparison to baseline approaches, i.e. UBCF and IBCF as well as matrix factorization techniques. In the next section, we present the results and give a detailed discussion about the gathered insights.

4.1 Test Sets

4.1.1 MovieLens

The MovieLens[3] data set comprises 1 million ratings from 6,040 users for 3,952 movies (95,81% sparsity) on a rating scale from 1 to 5. We randomly split the data set in five subsets to perform a 5-fold cross validation. Following [3], all users that hold less than 20 high ratings in the test set are removed from it and the ratings are added to the training set. We do so because the recommendation lists we create have a length up to 20 and we want to be able to distinguish between relevant (liked) and irrelevant (not liked) items (rather than between liked and not yet reviewed items). As commonly done in recommender systems literature, we define an item as highly rated if the user rated it with at least 4 out of 5 stars [4]. This results in test sets containing on average 147,494 ratings (85,309 high and 62,185 low ratings) from 2,152 users on 3,423 movies, the remaining ratings form the training sets.

4.1.2 Netflix

The Netflix data set is the one used for the Netflix[4] prize. Because of the size of the data set and the number of experiments we conducted, we pre-processed the data set by randomly selecting 9,006 users who rated 17,208 movies. Overall, the data set comprises 1,863,197 ratings which results in a sparsity of 98,81%. In the same manner as for the MovieLens data set we split the data set in five subsets to perform a 5-fold cross validation and removed all users from the current test set that hold less than 20 high ratings. The test sets comprise on average 304,932 ratings (177,551 high and 127,581 low ratings) from 3,439 users for 12,515 movies.

4.2 Evaluation Metrics

4.2.1 Aggregate Diversity

Following [3], we present the aggregate diversity of a recommender system as the total number of distinct items recommended among all users, see equation 6, with $P_H(u)$ being the set of items highly rated by user u from the user set U and $Rec(u)$ being the set of items recommended to her.

$$\text{aggregate-diversity} = |\bigcup_{u \in U} (P_H(u) \cap Rec(u))| \quad (6)$$

Note that we only consider correctly recommended items, i.e. movies highly rated by the user in the test set. We do so, because we only allow the recommender to recommend items that were actually rated by the user in the test set (see subsection 4.1). Thus, we know if an item was highly rated by the user or not and we do not want to increase the aggregate diversity by recommending items not relevant for the users.

[3]http://grouplens.org

[4]http://www.netflixprize.com/

4.2.2 Novelty

Novelty is defined differently in several publications depending on the context and its purpose, e.g. the item's novelty in respect to the user (which is related to the individual diversity) or the item's novelty in respect to the total amount of recommended items (which is related to the aggregate diversity). We apply the population-based item novelty evaluation metric proposed in [31] which is called expected popularity complement (EPC) to measure the ability of a recommender system to recommend items from the long tail, see equation 7.

$$\text{EPC} = \frac{\sum_{u \in U} \sum_{r=1}^{N} \frac{rel(u,i_r)*(1-pop(i_r))}{log_2(r+1)}}{\sum_{u \in U} \sum_{r=1}^{N} \frac{rel(u,i_r)}{log_2(r+1)}} \quad (7)$$

Here, we calculate the average "non-popularity" $1-pop(i_r)$ of each item i_r (i.e. the item that is at ranking position r of the actual recommendation list with size N). The popularity $pop(i)$ is calculated based on the times the item has been rated so far, hence, the item's popularity is the ratio between the number of its ratings $Rat(i)$ and the number of ratings of the most rated item in the item set I, see equation 8.

$$pop(i) = \frac{|Rat(i)|}{max_{i \in I}|Rat(i)|} \quad (8)$$

Just as for the aggregate diversity calculation, only relevant items are considered, this means $rel(u, i_r)$ can take the values 0 or 1. Additionally, the items are weighted according to their position r in the recommendation list by using a logarithmic discount. Thus, a recommender gets a higher EPC value when it not only recommends items from the long tail but also ranks them highly in the recommendation lists.

4.2.3 Classification Accuracy

Classification metrics measure the amount of correct and incorrect decisions, i.e. whether an item is correctly or incorrectly recommended or not recommended, respectively. The most popular metrics in this category are precision and recall that state the percentage of correctly recommended items in respect to all recommended items (precision) and the percentage of correctly recommended items in respect to all items that are highly rated (recall), see [11].

However, in a considerable amount of recommender systems a fixed number of recommendations is offered to the user. Additionally, we need a uniform number of recommended items to evaluate our approach in respect to the aggregate diversity described below. Therefore, we use the accuracy in top-N approach, i.e. the average precision of the recommended items for all users, see equation 9.

$$\text{accuracy-in-top-N} = \frac{\sum_{u \in U} |P_H(u) \cap Rec(u)|}{|U| * N} \quad (9)$$

4.3 Baseline Algorithms

In order to create a baseline to evaluate our approach against, we use the standard collaborative filtering methods IBCF (with adjusted cosine similarity) and UBCF (with Pearson correlation based similarity). Although these methods do not explicitly support the notion of novelty, diversity or the long tail, they constitute fairly reasonable baselines for performance measures besides classical accuracy measures as pointed out in [9] and supported by [1].

Additionally, we tested the matrix factorization methods (MF) offered by the PREA (Personalized Recommendation Algorithms) toolkit [24] (i.e. Single Value Decomposition (SVD), Non-negative MF, Probabilistic MF, and Bayesian Probabilistic MF) as well as the MF methods offered by the Java port of the MyMediaLite Recommender System Library [18] (i.e. a standard MF as well as a Biased and a Factorized MF). Based on the performances of the different methods on our test sets and to not overload the diagrams, we choose to present the SVD [29] method from the PREA toolkit and the Biased Matrix Factorization (BMF) [30] from the MyMediaLite library.

4.4 Experimental Setup

We start our evaluation with predicting the top-N movies for each user in the MovieLens and the Netflix data set. Then, we calculate the aggregate diversity, the EPC, and the classification accuracy of the recommendation lists. We run the experiments with the result list size N = 5, 10, 15, and 20. The UC-BCF approach is tested with the association measures χ^2-corr (Chi), log-likelihood (Log), and Poisson-based (Poisson), see section 3.2. Altogether, we run 240 experiments combining the different features and varying the co-occurrence vector size from 10 to 100 (with the specific sizes 10, 20, 25, 30, 40, 50, 75 and 100). Due to size constraints and because the algorithms perform similar for the different values of N, we only present the results for N=10.

5. RESULTS

5.1 Aggregate Diversity, Novelty, and Classification Accuracy

Fig. 1 shows the aggregate diversity, i.e. the total amount of distinct recommended items for the different techniques. The baseline algorithms perform dissimilar for the two data sets, for instance the Biased Matrix Factorization method is the best performing baseline on the MovieLens data set while on the sparser Netflix data set, it is the worst performing one. Overall, the user-based collaborative filtering approach performs best in terms of aggregate diversity.

For both data sets, the UC-BCF approach is able to outperform the baseline algorithms whereupon the UC-BCF approach using the association measure χ^2-corr in combination with vector size 10 is the most promising one. For the Movielens data set it increases the amount of recommended items on average by 38.10% (from 26.59% compared to BMF to 46.10% compared to IBCF). For the Netflix data set it increases the amount of recommended items on average by 68.31% compared to the baseline algorithms (from 52.25% (UBCF) to 88.77% (BMF)).

It can be seen in Fig. 1 that an increasing vector size results in a decreasing amount of distinct recommended items. The less items are used to describe an item the more diverse are the item vectors and the more distinct items get recommended to the users. It is important to state that only correctly recommended items are used to calculate the amount of distinct recommended items as we want to avoid incorrectly recommended items no matter if they increase the aggregate diversity or not. For the sparser Netflix data set, even with vector size 100, the UC-BCF approach recommends more distinct items than the baseline algorithms. For the MovieLens data set, the vector size needs to be at maximum 10-30 depending on the used association measure, to perform better than the BMF approach.

Fig. 2 shows the comparison of the classification accuracy in the top-10 recommendation lists for the different approaches. Here, the baseline algorithms perform similarly on both data sets. As could be expected, the matrix factorization methods SVD and BMF outperform the standard collaborative filtering techniques IBCF and UBCF with SVD being the best and UBCF being the worst performing approach for both data sets.

The UC-BCF approach with association measure χ^2-corr reaches the best classification accuracy values with vector size 30 for MovieLens and vector size 50 for Netflix. For the MovieLens data set, the UC-BCF approach performs better than the BMF (0.46%), IBCF (1.10%), and UBCF (1.96%) approaches whereas the SVD performs better than the UC-BCF approach (0.82%). For the sparser Netflix data set, the UC-BCF approach is able to increase the classification accuracy compared to all baseline approaches (from 0.32% (SVD) to 2.72% (UBCF)).

As can be seen in Fig. 2, the accuracy of the UC-BCF approaches increases with an increasing vector size until a size of 30-50 depending on the underlying association measure; thereafter it slightly decreases again. This is due to the fact that the more co-occurring items are used to describe an item, the greater is the chance of including coincidental and thus non-significant items which can be described as noise.

Fig. 3 shows the expected popularity complement (EPC) which states how well a recommender system performs in recommending items from the long tail. In terms of EPC, SVD and UBCF are the best performing baseline algorithms.

All UC-BCF approaches are able to outperform the baseline algorithms in terms of EPC. Similar to the aggregate diversity, the UC-BCF approach using χ^2-corr as association measure with vector size 10 is the best performing one for the MovieLens and the Netflix dataset. This UC-BCF approach results in an improvement from 4.18% (UBCF) to 11.39 (IBCF) for the MovieLens data set and an improvement of 8.33% (UBCF) to 17.78% (BMF) for the Netflix data set.

5.2 Interpretative Summary

The goal of the UC-BCF approach is to increase the aggregate diversity by pushing items from the long tail into the users' top-N recommendation lists without decreasing the classification accuracy of the recommendations. Therefore, we need to choose a co-occurrence vector size that is as small as possible to increase the aggregate diversity and the expected popularity complement (EPC). Additionally, the size needs to be large enough to not decrease the classification accuracy. These considerations result in a vector size between 20 and 25 depending on the used association measure and the data set. Overall, χ^2-corr with vector size 25 is the most promising association measure to be used with UC-BCF.

For the MovieLens data set, the usage of the UC-BCF approach with χ^2-corr and vector size 25 raises the amount of recommended items up to 21.7% compared to the UBCF approach (from 919 to 1118) and at least by 5.45% compared to the BMF approach (from 1060 to 1118). In terms of accuracy, the SVD approach receives a 0.86% better value than the UC-BCF approach. Nonetheless, the UC-BCF approach outperforms all other baseline algorithms by 0.41%-1.91%.

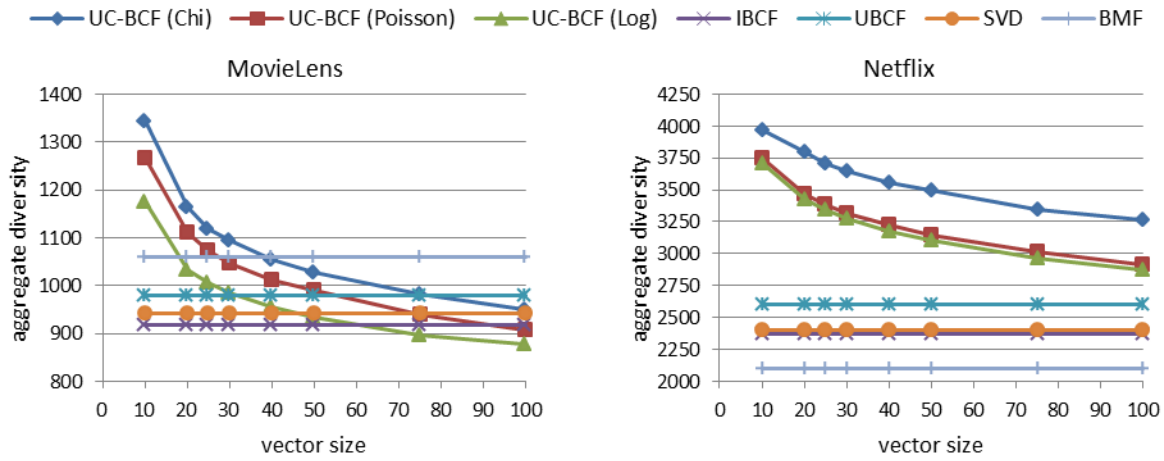

Figure 1: Aggregate Diversity in Top-10

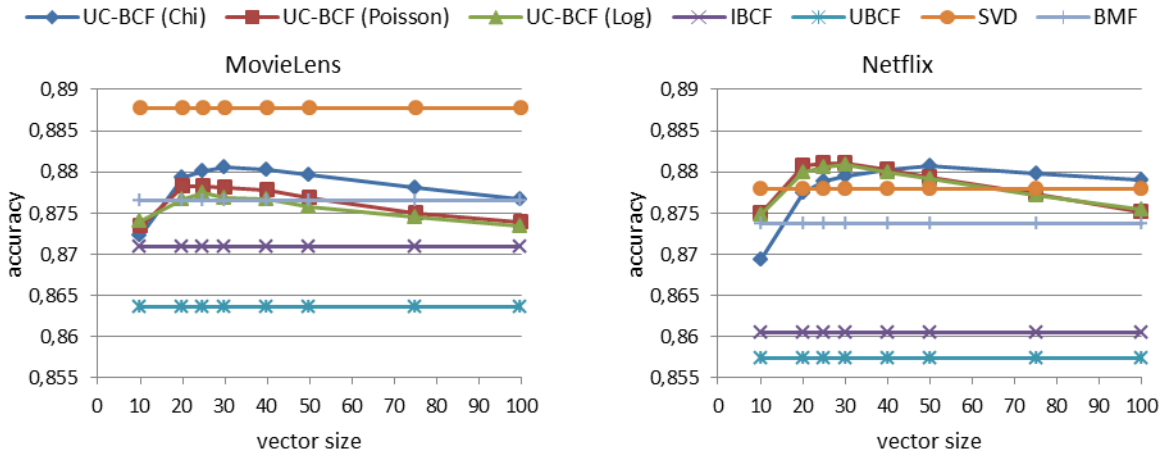

Figure 2: Accuracy in Top-10

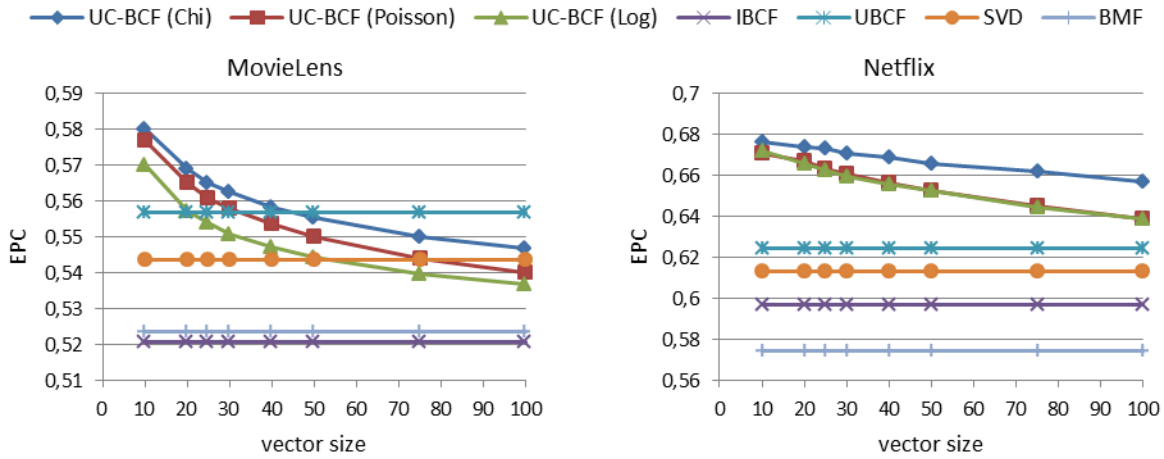

Figure 3: Expected Popularity Complement (EPC) in Top-10

Additionally, the EPC value shows that more items from the long tail are recommended.

For the Netflix data set, the amount of recommended items is improved up to 76.52% compared to the BMF approach (from 2101 to 3709 items) and at least by 42.37% compared to the UBCF approach (from 2605 to 3709 items) when using the UC-BCF approach with χ^2-corr and vector size 25. The accuracy is slightly increased as well (from 0.1% (SVD) to 2.5% (UBCF)) and the EPC value shows an improvement in recommending items from the long tail.

The improvement of the aggregate diversity and expected popularity complement is more significant for the Netflix data set than for the MovieLens data set, which can be accounted for by the fact that the Netflix data set holds more rarely rated objects (see section 4.1) that benefit from the new approach.

5.3 Combination with Existing Approaches

In chapter 2 we present several techniques that explicitly try to improve the aggregate diversity and that can be divided in two research lines. The first line tries to increase the aggregate diversity by re-ranking the most promising items. The second line tries to increase the aggregate diversity by improving the rating predictions for niche items usually using semantic metadata. The UC-BCF approach falls in the latter research line as it tries to improve the rating predictions as well but differs in that it does not require any additional semantic metadata which is an advantage as such data is often not available or incomplete.

However, the UC-BCF approach is not meant to be a mere concurrency for the presented approaches but can be used as an enhancement. Firstly, similar to other collaborative filtering approaches, it can be combined with approaches using semantic metadata if available, thus, the techniques from the second line can be applied to UC-BCF. Secondly, it can be used as an underlying algorithm for the re-ranking approaches from the first line.

In order to demonstrate such a combination, we select a simple popularity-based re-ranking approach as introduced in [3]. First, we calculate the missing ratings in the user-item matrix by applying the UC-BCF. Then, we choose all items having a higher rating prediction than a given threshold as candidates. These items are re-ranked according to their popularity (i.e. the number of ratings they hold) in reverse order to recommend the first ten items. Thus, a very popular item gets rejected, even if its predicted rating is high. We perform a 5-fold cross validation on the MovieLens and Netflix data sets as before to evaluate the recommendations. Additionally, in order to enable a better comparison, the threshold is varied from 3.8-4.2 stars to receive a common classification accuracy of 0.85 for each technique.

Fig. 4 shows the aggregate diversity reached using this re-ranking method for the baseline algorithms and the UC-BCF approach with χ^2-corr and vector size 25 for the MovieLens and the Netflix data set. Compared to the baseline algorithms, the UC-BCF approach is able to increase the aggregate diversity from 3.18% (BMF) to 24.42% (UBCF) for the MovieLens data set and from 18.37% (SVD) to 55.8% (UBCF) for the Netflix data set. Similar to the experiments conducted before, the Netflix dataset which holds more niche items than the MovieLens dataset benefits even more from the UC-BCF approach in terms of aggregate diversity.

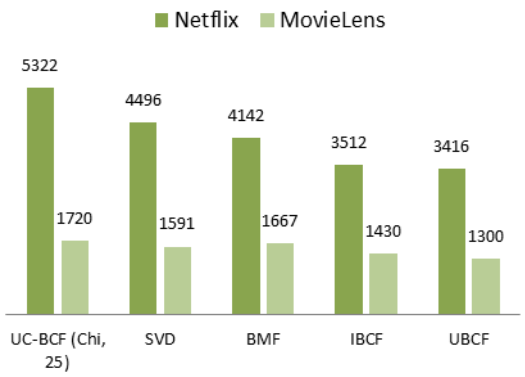

Figure 4: Aggregate Diversity after Re-ranking (for an accuracy of 0.85 in top-10)

6. CONCLUSION AND FUTURE WORK

In this paper we present a new collaborative filtering approach – named usage context-based collaborative filtering (UC-BCF) – in which items are described by items they significantly often co-occur with. In this way, items are similar if they often occur in similar usage contexts, i.e. in similar user profiles, but not necessarily together in the same user profiles. This approach combines and extends strengths from the well-known user-based and item-based collaborative filtering approaches and compensate their weaknesses. Using this new usage context-based approach, even niche items that do not hold an extensive usage history are represented by a characteristic co-occurrence vector and, thus, the calculation of their rating predictions is improved.

We show that the UC-BCF increases the aggregate diversity compared to the standard collaborative filtering and matrix factorization techniques with only one case of accuracy loss (in all other cases, the accuracy can even be improved). Additionally, the rating prediction of seldomly used items is improved and thus, more items from the long tail are recommended to the users. Therefore, the UC-BCF approach is better suited than the standard approaches to provide a user with idiosyncratic items and to increase the range of products sold by an online shop. Additionally, the UC-BCF approach is well suited to form the basis for approaches that re-rank possible recommendations to push even more items from the long tail.

In this paper we use user profiles as usage contexts, however, with more detailed usage data available usage contexts can be formed considering e.g. the time a movie was watched or the company it was watched with, which might result in even better results. Furthermore, when creating recommendations e.g. for songs or web pages, it might be more suitable to consider one session (e.g. every listening or click without a break of more than an hour) as one usage context. We will further continue on this track to answer these questions.

7. REFERENCES

[1] Adamopoulos, P., and Tuzhilin, A. On unexpectedness in recommender systems: Or how to expect the unexpected. In *DiveRS 2011, ACM RecSys Workshop on Novelty and Diversity in Recommender Systems* (October 2011), RecSys 2011, ACM.

[2] Adomavicius, G., and Kwon, Y. Maximizing aggregate recommendation diversity: A graph-theoretic approach. In *Workshop on Novelty and Diversity in Recommender Systems (DiveRS)* (2011).

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

[4] Adomavicius, G., and Tuzhilin, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transaction on Knowledge and Data Engineering 17*, 6 (2005), 734–749.

[5] Anderson, C. *The Long Tail: Why the Future of Business is Selling Less of More*. Hyperion, New York, 2006.

[6] Bordag, S. A comparison of co-occurrence and similarity measures as simulations of context. In *Computational Linguistics and Intelligent Text Processing* (2008), A. Gelbukh, Ed., vol. 4919 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 52–63. 10.1007/978-3-540-78135-65.

[7] Bradley, K., and Smyth, B. Improving Recommendation Diversity. In *Proc. of the 12th National Conference in Artificial Intelligence and Cognitive Science* (Maynooth, Ireland, 2001), D. O'Donoghue, Ed., pp. 75–84.

[8] Brynjolfsson, E., Hu, Y. J., and Simester, D. Goodbye pareto principle, hello long tail: The effect of search costs on the concentration of product sales. *Management Science 57*, 8 (August 2011), 1373–1386.

[9] Burke, R. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction 12*, 4 (Nov. 2002), 331–370.

[10] Ceglar, A., and Roddick, J. F. Association mining. *ACM Computing Surveys 38*, 2 (July 2006).

[11] Cleverdon, C., and Kean, M. Factors determining the performance of indexing systems. In *Aslib Cranfield Research Project,* (Cranfield, England, 1968).

[12] de Saussure, F. *Course in General Linguistics*. Open Court Publishing, Chicago, 1986.

[13] DeGroot M., H., and Schervish, M. J. *Probability and Statistics, 3rd edition*. Addison Wesley, Boston, 2002.

[14] Dunning, T. E. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics 19*, 1 (1993), 61–74.

[15] Evert, S. Corpora and collocations. In *Corpus Linguistics. An International Handbook*, vol. 2. De Gruyter, Berlin, pp. 223–233.

[16] Evert, S. *The Statistics of Word Cooccurrences: Word Pairs and Collocations (Dissertation)*. University Stuttgart, 2004.

[17] Fleder, D. M., and Hosanagar, K. Recommender systems and their impact on sales diversity. In *Proc. of the 8th ACM conference on Electronic commerce* (2007), EC '07, ACM, pp. 192–199.

[18] Gantner, Z., Rendle, S., Freudenthaler, C., and Schmidt-Thieme, L. MyMediaLite: A free recommender system library. In *Proc. of the 5th ACM Conference on Recommender Systems* (2011).

[19] Goel, S., Broder, A. Z., Gabrilovich, E., and Pang, B. Anatomy of the long tail: ordinary people with extraordinary tastes. In *WSDM* (2010), B. D. Davison, T. Suel, N. Craswell, and B. L. 0001, Eds., ACM, pp. 201–210.

[20] Goldstein, D. G., and Goldstein, D. C. Profiting from the long tail. *Harvard Business Review* (June 2006).

[21] Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst. 22*, 1 (Jan. 2004), 5–53.

[22] Holtsberg, A., W. C. Statistics for sentential co-occurrence. *Working Papers 48* (2011), 133–148.

[23] Karypis, G. Evaluation of item-based top-n recommendation algorithms. In *Proc. of the tenth international conference on Information and knowledge management* (2001), CIKM '01, ACM, pp. 247–254.

[24] Lee, J., Sun, M., and Lebanon, G. A Comparative Study of Collaborative Filtering Algorithms. *ArXiv e-prints* (May 2012).

[25] Levy, M., and Bosteels, K. Music recommendation and the long tail. In *Workshop on Music Recommendation and Discovery (WOMRAD)* (2010).

[26] Manning, C. D., Raghavan, P., and Schütze, H. *An Introduction to Information Retrieval*. Cambridge University Press, 2008.

[27] McNee, S. M., Riedl, J., and Konstan, J. A. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In *ACM SIGCHI Conference on Human Factors in Computing Systems* (Montréal, Québec, Canada, 22/04/2006 2006), ACM, ACM, pp. 1097–1101.

[28] Park, Y.-J., and Tuzhilin, A. The long tail of recommender systems and how to leverage it. In *RecSys* (2008), pp. 11–18.

[29] Paterek, A. Improving regularized singular value decomposition for collaborative filtering. In *Proc. of KDD Cup and Workshop* (2007).

[30] Rendle, S., and Schmidt-Thieme, L. Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In *Proc. of the 2008 ACM conference on Recommender systems* (2008), ACM, pp. 251–258.

[31] Vargas, S., and Castells, P. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proc. of the 5th ACM conference on Recommender systems* (2011), ACM, pp. 109–116.

[32] Zhang, M. Enhancing diversity in top-n recommendation. In *Proc. of the third ACM conference on Recommender systems* (2009), RecSys '09, ACM, pp. 397–400.

[33] Zhang, M., and Hurley, N. Avoiding monotony: improving the diversity of recommendation lists. In *Proc. of the 2008 ACM conference on Recommender systems* (2008), RecSys '08, ACM, pp. 123–130.

[34] Ziegler, C.-N., McNee, S. M., Konstan, J. A., and Lausen, G. Improving recommendation lists through topic diversification. In *Proc. of the 14th international conference on World Wide Web* (2005), ACM Press, pp. 22–32.