Extending User Profiles in Collaborative Filtering Algorithms to Alleviate the Sparsity Problem

Toon De Pessemier, Kris Vanhecke, Simon Dooms, Tom Deryckere, and Luc Martens

Ghent University - IBBT, Department of Information Technology,
Gaston Crommenlaan 8, B-9050 Ghent, Belgium
{toon.depessemier,kris.vanhecke,simon.dooms,
tom.deryckere,luc.martens}@intec.ugent.be
http://www.wica.intec.ugent.be

Abstract. The overabundance of information and the related difficulty to discover interesting content has complicated the selection process for end-users. Recommender systems try to assist in this content-selection process by using intelligent personalisation techniques which filter the information. Most commonly-used recommendation algorithms are based on Collaborative Filtering (CF). However, present-day CF techniques are optimized for suggesting provider-generated content and partially lose their effectiveness when recommending user-generated content. Therefore, we propose an advanced CF algorithm which considers the specific characteristics of user-generated content (like the sparsity of the data matrix). To alleviate this sparsity problem, profiles are extended with probable future consumptions. These extended profiles increase the profile overlap probability, thereby increasing the number of neighbours used for calculating the recommendations. This way, the recommendations become more precise and diverse compared to traditional CF recommendations. This paper explains the proposed algorithm in detail and demonstrates the improvements on standard CF.

Keywords: Collaborative Filtering, Recommender System, Personalisation, Algorithm, Sparsity

1 Introduction

Various Web 2.0 sites (like YouTube, Digg, Flickr, Google Video...) have an overwhelming bulk of user-generated content available for online consumers. Although this exploding offer can be seen as a way to meet the specific demands and expectations of users, it has complicated the content selection process to the extent that users are overloaded with information and risk to 'get lost': though an abundance of content is available, obtaining useful and relevant content is often difficult. Traditional filtering tools, like keyword-based or filtered searches, are not capable to weed out irrelevant content or provide too much search results. An additional filtering based on the overall popularity (expressed by consumption patterns or user ratings) can assist, but requires a broad basis of user feedback before it can make reasonable suggestions. Furthermore, this technique does not consider personal preferences and individual consumption behaviour, since only the most popular content will be suggested. This situation reinforces the role of (collaborative) filtering tools and stimulates the development of recommender systems that assist users in finding the most relevant content.

The remainder of this paper is organized as follows: Section 2 provides an overview of related work regarding recommender systems in various application domains. Section 3 discusses collaborative filtering techniques and the problems related to sparse data sets. In Section 4, we present an extended version of the CF to overcome these problems. Our evaluation methodology and utilized datasets are described in Section 5. Section 6 elaborates on the obtained results and compares the proposed algorithm with the traditional CF. Optimisations and potential drawbacks of the proposed algorithm are discussed in Section 7. Finally, we offer a brief conclusion on our research results and point out interesting future work in Section 8.

2 Related Work

The overabundance of information and the related difficulty to discover interesting content have already been addressed in several contexts. Online shops, like Amazon [14], internet radios, like Last.FM [4], and video sharing website, like YouTube [6] apply recommendation techniques to personalize their website according to the needs of each user. Purchasing, clicking and rating behaviour are valuable information channels for online retailers and content providers to investigate consumers' interests and generate personalized recommendations [13].

Netflix is an online, mail-based, DVD rental service in the United States. Customers have the possibility to express their appreciation for a rented movie by a star-rating mechanism on the Netflix website. These user ratings, representing the user's preferences, are used as input for the Netflix recommender to generate personalized movie suggestions. Convinced by the potential of an accurate recommendation system, Netflix published a large dataset and started a competition to find the most suitable recommendation algorithm for their store in October 2006. In this context, many research groups have competed to find the best movie recommendation algorithm based on the Netflix dataset [1].

The introduction of digital television entails an increase in the number of available TV channels and the information overload linked thereto. Consequently, new standards to describe this content, e.g. TV-Anytime [7], and advanced electronic program guides, which simplify the navigation and selection of TV programs, become necessary . Several personalized TV guide systems which filter and recommend TV programs according to the user's preferences, have been developed for set-top boxes and personal digital recorders [21].

Besides these traditional premium content sources (i.e. provider-generated content), user-generated content (like personal photos, videos, or bookmarks) has received a more prominent role on the web in recent years. These Web 2.0 applications use more pragmatic approaches, like tagging, to annotate content than the traditional metadata standards. Moreover, user-generated content services are characterized by an immense content growth, which introduces sparsity problems for recommender systems.

3 Collaborative Filtering

3.1 Traditional Collaborative Filtering Algorithms

Most commonly-used recommendation algorithms are based on Collaborative Filtering (CF) techniques because they generally provide better results than Content-Based (CB) methods and require no metadata of the content [9]. To describe these recommendation algorithms, the 'item' concept is introduced as a general term for any kind of content or information (e.g., a book, video or picture) and accordingly, 'consumption behaviour' is a more general expression for user feedback (like ratings or purchases) on these items. Most literature reviews distinguish two important classes of CF: user-based and item-based, supplemented with several optional variations on them. To generate personal recommendations for a target user, user-based CF algorithms start by finding a set of neighbouring users whose purchased or rated items overlap this target user's purchased or rated items. To identify these neighbours, users are represented as an N-dimensional vector of potential consumptions, where N is the number of distinct catalogue items. Consumptions of items (like purchases or ratings) are recorded in the corresponding components of this vector. However, this profile vector may remain extremely sparse (i.e. containing a lot of missing values) for the majority of users who purchased or rated only a very small fraction of the available catalogue items. Subsequently, the composition of neighbourhoods of like-minded users is based on user similarity values.

The similarity of two users, j and k, symbolized by their consumption vectors, U_j and U_k, can be measured in different ways. A commonly-used method to determine the similarity is measuring the cosine of the angle between the two consumption vectors [17].

$$Similarity(\boldsymbol{U}_j, \boldsymbol{U}_k) = cosine(\boldsymbol{U}_j, \boldsymbol{U}_k) = \frac{\boldsymbol{U}_j \cdot \boldsymbol{U}_k}{||\boldsymbol{U}_j||\ ||\boldsymbol{U}_k||} \tag{1}$$

Next, the algorithm aggregates the items consumed by these neighbours while taking into account the similarity values. The items that the target user has already purchased or rated are eliminated, and the remaining items are candidate recommendations for this user [14]. An alternative for this user-based CF technique is item-based CF, a technique that matches each of the user's purchased or rated items to similar items and then combines those similar items into a recommendation list. For measuring the similarity of items, the same metrics can be used as with the user-based CF. Because of scalability reasons, this technique is often used to calculate recommendations for big online shops, like Amazon, where the number of users is much higher than the number of items [14].

3.2 Collaborative Filtering Based on Sparse Datasets

Despite the popularity of CF, its applicability is limited due to the sparsity problem, which refers to the situation that the consumption data in the profile vectors are lacking or insufficient to calculate reliable recommendations. This sparsity problem is a big concern for user-generated content systems, since the content offer is rapidly growing and most users only consume a small fraction of the available items.

As a direct consequence of this sparsity problem, the number of similar users, i.e. the neighbours of the target user, might be very limited with a user-based CF technique. Indeed, to determine the similarity, almost all metrics rely on the profile overlap, which might be very incomplete or even nonexistent. In addition, because of this sparsity, the majority of these neighbours might have a profile vector containing a small number of consumed items. Because the prospective personal recommendations are limited to the set of items consumed by neighbours, the variety, quality and quantity of the final recommendation list might be inadequate. A comparable reasoning is applicable to item-based CF techniques that work on sparse profile data. Users might have consumed a small number of items, which in turn also have a limited number of neighbouring items. Again, the CF algorithm is restricted to a narrow set of neighbouring items to generate the personal suggestions, which is pernicious for the efficiency of the recommender.

In an attempt to provide high-quality recommendations, even based on sparse consumption data, various solutions are proposed in literature [16]. Most of these techniques use trust inferences, transitive associations between users that are based on an underlying social network, to deal with the sparsity and cold-start problems of CF[20]. Nevertheless, these underlying social networks are often insufficiently developed in content-delivery systems or even nonexistent for (new) web-based applications that desire to offer personalized recommendations.

'Default voting' is an extension to the traditional CF algorithms which tries to solve this sparsity problem without exploiting a social network. A default value is assumed as 'vote' for items without an explicit purchase or rating[2] . Although this technique enlarges the profile overlap, it can not identify more contributory neighbours than the traditional CF approach. Other approaches to deal with sparse data profiles, such as link analysis techniques [11] or spreading activation algorithms [12] are too computationally intensive to be applied on large datasets.

3.3 Recommendations for User-Generated Content

Because of the inherent characteristics of user-generated content systems, the size of the content catalogue (i.e. the number of items) is often significantly bigger compared to provider-generated content systems. User-generated content requires less production efforts. Consequently, the content production rate and the number of distinct publishers are massive. For example, YouTube enjoys 65,000 daily new uploads [5]. Due to these varied content (production) characteristics, the sparsity problem is a major concern for user-generated content systems. As a result, the recommender accuracy might be inadequate if the traditional CF techniques are ported from provider-generated content systems to user-generated content systems without any adaptation.

Therefore, we developed an advance CF algorithm that extends the user profiles based on the probability that an item will be consumed in the future. The denser profile vectors increase the profile overlap probability, which increases the number of neighbours in a CF algorithm. These extended profiles and additionally identified neighbours, in response to the sparsity problem, lead to more precise and varied content recommendations.

4 Probability-Based Extended Profile Filtering

4.1 Algorithm Details

To increase the number of neighbours for a target user, and to strengthen the existing similarities, the profile overlap has to be augmented. This can be achieved by a larger amount of consumption behaviour. Because stimulating the users to consume more content is not an option in most cases, we opted for an artificial profile extension based on the future consumption probability. Our developed algorithm (called probability-based extended profile filtering) is an iterative process consisting of four phases.

In the first phase, a traditional CF algorithm is employed to generate a top-N recommendation list with a corresponding confidence value for each recommendation based on the existing, sparse profiles [18]. These recommendations, which might be inaccurate due to the sparsity, are not used as the final suggestions but only as an information source for subsequent calculations.

In the second phase, all the initial profiles that do not contain a minimum number of consumptions are extended. To make these sparse profiles denser, presumable future consumptions are inserted into the profile vectors. These additional consumptions are based on two information sources: the general probability and the profile-based probability that the item will be consumed in the near future.

In the user-based version of the proposed algorithm, existing user profiles are extended with the items that have the highest probability to be consumed by the user in the near future. The general probability that a specific item will be consumed by a specific user without a priori knowledge of that user is proportional to the current popularity of the item. So, this probability is estimated by a linear function of the popularity of the item: $f(x) = a \cdot x + b$, in which f(x) is the probability of consumption, x is the popularity of the item and a and b are parameters that can be optimized according to the content type. Especially for user-generated content systems, the popularity of the 'top items' can vary rapidly in time. Additionally, the probability that a specific item will be consumed by a user can also be calculated based on the user's profile as a priori knowledge. This probability is inversely proportional to the index of the item in the personal top-N recommendation list, and can be estimated by the confidence value which is calculated by the traditional CF system in phase 1. In fact, this top-N recommendation list is a prediction of the items which the user will like or consume in the near future.

In the item-based version of the extended CF, item profiles, which contain the users who consumed the item in the past, are supplemented with the most likely future consumers. The general probability that a specific user will consume a specific item, without any knowledge of the item, is proportional to the present intensity of the consumption behaviour of that user. Again, a linear function can be used to estimate this probability. With additional a priori knowledge of the item, the calculations can be repeated. Then, the probability is inversely proportional to the index of the specific user in the top-N list of users who are the most likely to consume the item in the future. This list and the associated confidence values can be generated based on the results of the traditional item-based CF algorithm [18].

Based on these general and profile-based probabilities, the user or item profiles are completed, if possible, until the minimum profile threshold is reached. Therefore, the predicted future consumptions resulting from the two methods (without and with a priori knowledge) are merged via an aggregator function. (In the bench-marks we used the maximum as an aggregator operator.) However, these predicted consumptions are marked as 'uncertain' in contrast to the initial consumptions which are linked to the user behaviour (i.e. the ratings or purchases). For example for a web shop, the real purchases are indicated by a value of 1, which refers to a 100% guaranteed consumption, whereas the potential future consumptions are represented by a decimal value between 0 and 1, according to the probability value, in the profile vector. This first and second phase may consist of several successive iterations until a minimum threshold for the profile size is reached.

Based on these extended profile vectors, the similarities are recalculated with the chosen similarity metric, e.g. the cosine similarity (equation 1), in a third phase. Because of the additional future consumptions, the profile overlap and accordingly the number of neighbours will be increased, compared to the first phase. In the item-based version, these similarities can be used as an extended 'related item' section (like on Amazon: customers who bought this item also bought...).

To produce personal suggestions, a recommendation vector is generated based on these extended profile vectors, in a fourth phase. For the user-based algorithm, the recommendation vector, R_j, for target user j can be calculated as:

$$\boldsymbol{R}_j = \frac{\sum_{k=1,k\neq j}^{M} \boldsymbol{U}_k \cdot Similarity(\boldsymbol{U}_j, \boldsymbol{U}_k)}{\sum_{k=1,k\neq j}^{M} Similarity(\boldsymbol{U}_j, \boldsymbol{U}_k)} \qquad (2)$$

where M stands for the number of users in the sysem. U_j and U_k represent the extended consumption vectors of users j and k respectively, which might contain real values. Subsequently, the top-N recommendations are obtained by selecting the indices of the components with the highest values from the recommendation vector, R_j, and eliminating the items which are already consumed by user j in the past.

4.2 Algorithm Example

The following concrete example of a small content delivery system (e.g. a video sharing website) can give more insights in the proposed recommendation algorithm. Table 1 illustrates the watching behaviour of 5 users and 8 videos in the system via the consumption matrix. The watched videos are indicated by a 'one' in the column of the user. E.g., user A has watched the videos 3, 6 and 8 in this example. Personal recommendations for the end-users can be calculated based on the data in this matrix. By way of illustration, the recommendations for user A are calculated in this example. Therefore, a traditional user-based CF algorithm is used in the first phase. To start, the neighbours of user A in the system are calculated based on the cosine similarity metric as illustrated in Table 2. Since all standard similarity metrics are based on the profile overlap between a couple of users, only 1 neighbour of user A can be identified, namely user B. Next, the personal recommendations are discovered by selecting the videos that neighbours watched in the past. Because the system does not suggest videos that the user has already watched, only 1 new video can be recommended to user A, namely video 5.

Table 1. The consumption matrix of the example. Videos watched by the end-user are indicated by a 'one'.

Watching Behaviour	User A	User B	User C	User D	User E
Video 1			1		
Video 2				1	1
Video 3	1	1			
Video 4			1		
Video 5		1	1	1	
Video 6	1	1			
Video 7					1
Video 8	1	1			

In contrast, our proposed algorithm tries to extend the profile vectors with potential future consumptions, before generating the recommendations. The general probability that a video will be watched, without a priori knowledge, is proportional to the current popularity of the video. In this small example, video 5 is the most popular video (3 views). So, this video might be added as a potential future consumption to the profile vector of user A and E (i.e. the users who have not yet watched the video). However for simplicity of the example, this general probability of watching a video is not considered during the selection of potential future consumptions.

Table 2. The similarity matrix that is calculated based on the original consumption matrix. Missing values indicate a zero similarity between two users.

Similarity	User A	User B	User C	User D	User E
User A	1	0.8660			
User B	0.8660	1	0.2887	0.3536	
User C		0.2887	1	0.4082	
User D		0.3536	0.4082	1	0.5000
User E				0.5000	1

Table 3. The consumption matrix after extending the profiles with potential future consumptions.

Ticketing	User A	User B	User C	User D	User E
Video 1		0.2887	1	0.4082	
Video 2		0.3536	0.4082	1	1
Video 3	1	1	0.2887	0.3536	
Video 4		0.2887	1	0.4082	
Video 5	0.8660	1	1	1	0.5000
Video 6	1	1	0.2887	0.3536	
Video 7				0.5000	1
Video 8	1	1	0.2887	0.3536	

Additionally, the probability of watching a video can be calculated based on the confidence value of the user's traditional CF recommendations. This confidence value is derived (e.g. by a linear combination) from the similarity values of the identified neighbours who consumed the recommended video. Table 3 illustrates the consumption matrix after the addition of the confidence values calculated by a simple CF algorithm. This matrix contains decimal values for the potential future consumptions making it less sparse than the former. This extended consumption matrix is the input for recalculating the user similarities, which are illustrated in Table 4. Subsequently, these updated user similarities are used in equation 2 for calculating the final recommendations. The recommendations for user A, together with the accompanying prediction values are listed in Table 5. The prediction values show that the most interesting suggestion for user C is still video 5 (just like the CF predicted). However, additional video recommendations can be provided based on Table 5. In descending order of prediction value: video 5, 2, 1, 4 and 7.

Alternatively, in addition to the traditional CF, the general popularity of the videos could be used to generate more recommendations (e.g., suggestions besides video 5 for user A). However, such a popular-item recommender can make no distinction between equally popular videos (like 1, 4 and 7 which are each watched by one user). In contrast, the proposed algorithm based on extended profiles can provide a (partial) ordering of these equally popular videos (video 1 = video 4 > video 7).

Table 4. The similarity matrix that is calculated based on the extended consumption matrix. Missing values indicate a zero similarity between two users.

Similarity	User A	User B	User C	User D	User E
User A	1	0.9637	0.4839	0.5785	0.1491
User B	0.9637	1	0.6758	0.7437	0.2747
User C	0.4839	0.6758	1	0.7961	0.3276
User D	0.5785	0.7437	0.7961	1	0.7752
User E	0.1491	0.2747	0.3276	0.7752	1

Table 5. The recommendations with the accompanying prediction values for user A, calculated with the probability-based extended profile filtering algorithm.

Prediction	Prediction Value
$R_{a,1}$	0.4589
$R_{a,2}$	0.5820
$R_{a,4}$	0.4589
$R_{a,5}$	0.9657
$R_{a,7}$	0.2015

5 Evaluation Methodology

To estimate the accuracy of personal recommendations, two different evaluation methods are possible. On the one hand, online evaluation methods measure the user interactivity (like clicks or buying behaviour) with the recommendations on a running service. On the other hand, offline evaluation methods use a test set with consumption behaviour which has to be predicted based on a training set with consumption history. Although online evaluation methods are closest to reality, we opted for an offline evaluation based on datasets because such an evaluation is fast, reproducible and commonly used in recommendation research.

Therefore, we compared the proposed extended CF algorithm with the traditional CF algorithm based on evaluation metrics which are generated by an offline analysis using a dataset with consumption behaviour. Because the datasets that are commonly used to bench-mark recommendation algorithms (like Netflix, Movielens or Jester) contain too little sparse profiles and handle only provider-generated content, we evaluated our algorithm on a dataset of PianoFiles[1]. Pianofiles is a user-generated content site that offers users the opportunity to manage their personal collection of sheet music they like to play. Currently, users can manage their personal collection of sheet music on PianoFiles but they do not yet receive personal recommendations. The main consumption behaviour, used to feed the recommendation algorithm, consists of the personal collections of the users. Each addition to a personal collection is used to populate the consumption matrix. The dataset contains 401,593 items (music sheets), 80,683 active users

[1] http://www.pianofiles.com/

and 1,553,586 distinct consumptions (individual additions of sheets to personal collections) in chronological order.

For evaluation purposes, we use 50% of the consumptions that are the most recent ones as the test set and use the remaining 50% of the consumption records as the input data. To analyse the performance of the algorithm under data of different sparsity levels, we compose 10 different training sets by selecting the first 10%, 20%, 30%, until 100% of the input data. The recommendation algorithm used these different training sets in successive iterations to generate personal suggestions. As commonly done for the evaluation of recommendations under sparse data [11], the test set was first filtered to include only consumptions that are possible to predict with the input data as a priori knowledge. A consumption of a sheet that is not contained in the input data or a consumption of a user without any consumption behaviour in the input data is not possible to predict with CF techniques. By eliminating the users and items without a priori knowledge, the CF algorithms can be compared more precisely based on the users (and items) which have specified preferences. Then, all users in this filtered test set were included into a set of target consumers. For each of these target consumers, the algorithm generated five ordered lists of 10, 20, 30, 40 and 50 recommendations respectively, which were compared with the test set. (Only the results for 10, 30 and 50 recommendations per user are shown in this paper, since the other results illustrate the same conclusions.) This offline evaluation methodology, in which a dataset is chronologically split into training set and test set, is commonly used for evaluating recommendation algorithms [8].

6 Results

6.1 Evaluation Metrics

One of the most used error metrics is the Root Mean Squared Error (RMSE) [19, 10], which is also adopted by the official Netflix contest. However, the Netflix contest is mainly focused on predicting accurate ratings for an entire set of items, while web-based content-delivery applications are most interested in providing the users with a short recommendation list of interesting items [3]. To evaluate this top-N recommendation task, i.e. a context where we are not interested in predicting user ratings with precision, but rather in giving an ordered list of N attractive suggestions to the users, error metrics are not meaningful. As a result, information-retrieval classification metrics, which evaluate the quality of a short list of recommendations, are the most suitable.

The most appropriate classification accuracy metrics for evaluating recommendations are precision and recall [3]. The precision is the ratio of the number of recommended items that match with future consumptions, and the total number of recommended items. In offline evaluations, the consumptions of the test set represent the future consumptions and the recommendations that match with these consumptions are called the *relevant recommendations.*

$$Precision = \frac{\#\ Relevant\ recommendations}{\#\ Recommendations} \tag{3}$$

The recall stands for the ratio of the number of relevant recommendations and the total number of future consumptions. Only the ‘future consumptions’ in the test set are considered as *relevant items* for the end-users in offline evaluations.

$$Recall = \frac{\#\ Relevant\ recommendations}{\#\ Relevant\ items} \tag{4}$$

It has been observed that precision and recall are inversely related and dependent on the length of the result list returned to the user [10]. If more recommended items are returned, precision decreases and recall increases. Therefore, to understand the global quality of a recommendation system, precision and recall is combined by means of the F1-measure.

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{5}$$

6.2 Bench-marked Algorithms

Since the item-based algorithms (the traditional item-based CF and the proposed item-based algorithm which extends profiles) generally achieved a very low performance on the PianoFiles dataset, we did not include the results of any item-based technique in the evaluation. This poor performance is mainly due to the nature of the dataset, which contains many more items than users. As a result, forming item neighbourhoods is actually much more difficult than forming user neighbourhoods [11]. Additionally, this ratio of users and items induces the big risk that item-based algorithms will trap users in a ‘similarity hole’, only giving exceptionally similar recommendations; e.g. once a user added a sheet of Michael Jackson to her collection, she would only receive recommendations for more Michael Jackson sheets [15]. Compared to this item-based CF, a user-based strategy achieved much better results based on the PianoFiles dataset.

6.3 Complete Training Set

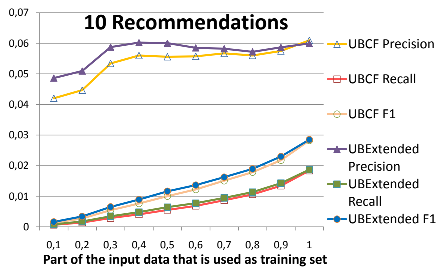
In a first evaluation, we bench-marked the standard user-based CF algorithm (UBCF), which operates on the initially existing profiles [18], against the user-based version of our probability-based extended profile filtering (UBExtended), which extends the sparse profiles before generating the actual recommendations. In this accuracy evaluation, the UBExtended algorithm was configured to extend sparse profiles to a target size of 6 consumptions and the cosine similarity (equation 1) was used as a measure to compare profile vectors in both algorithms. The graphs in Figure 1(a), 1(c) and 1(e) illustrate the evaluation metrics (precision, recall and F1) for these two algorithms (UBCF and UBExtended). Due to the large content offer (401,593 items) and the sparsity of the dataset, the recommendation algorithms have a hard job to suggest the most appropriate content items for each user. Because of this, the absolute values of the evaluation metrics seem rather low. However, precision and recall values between 1 and 10% are very common in bench-marks of recommendation algorithms [12], [11].

The graphs illustrate that the most accurate results are obtained for iterations which are based on a large quantity of training data. As the size of the training set increases, more data about user behaviour becomes available. This supplementary training data can refine the user preferences, which leads to a higher precision. However, after the profile size has reached a critical point, supplementary training data have no more additional information value, which leads to a stagnating precision value. Besides additional information of existing users, the extra data contain information about the behaviour of new users for which no information was available in the first part(s) of the training set. These additional data make it possible to generate recommendations for more users, which explains the increasing recall value. However, the recommendations for these new users, which generally have a lower precision value because of a limited early profile, strengthen the stagnation of the general precision. At last, the F1 metric follows the progress of this precision and recall graph closely because of its definition. Besides these general trends, the graphs prove that the UBExtended algorithm outperforms the standard UBCF in all three evaluation metrics (precision, recall and F1) and for different sizes of the recommendation list. This improvement is especially noticeable for small training sets, which mainly consist of sparse user profiles.

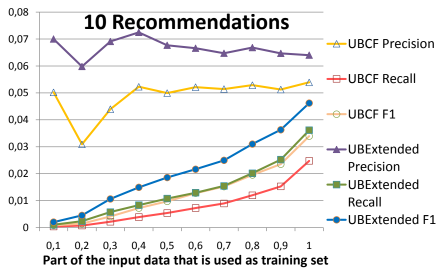
6.4 Filtered Training Set: Sparse Profiles Only

To illustrate this superiority of the UBExtended algorithm for sparse profiles, a second evaluation was performed. To study the recommender performance on the subset of users with a sparse profile, the training set was submitted to an extra filter. This filter removed all the users with more than x consumptions from the training set to simulate the situation of a very novel content delivery system without 'well-developed' user profiles. In accordance with the first evaluation in which we extended sparse profiles to a target size of 6 consumptions, we chose in this second analysis for a filter that removes all users with a profile that is larger than 6 consumptions. In this way, a standard UBCF that operates on a dataset with only sparse profiles (profile size ≤ 6 consumptions), was compared with the UBExtended algorithm which extends these profiles to the target size (profile size $= 6$ consumptions) before generating recommendations. Given the long-tail distribution of the profile size in content delivery systems, this subset of sparse-profile consumers constitutes a considerable segment of the system users. This is illustrated in Figure 1(d) which visualizes the histogram of the profile size at PianoFiles, showing that the big majority of users only have a limited number of sheets in their collection.

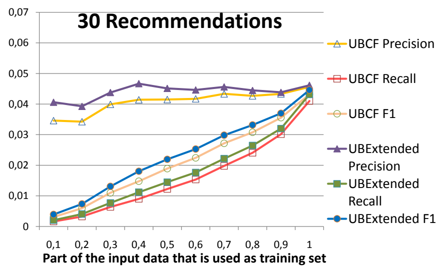
Since the extra filter modifies the training and test sets, the absolute values of this evaluation can not be compared with the absolute values of the first evaluation, which is based on the unfiltered datasets. However, the differences between the UBCF and the UBExtended algorithm in this second evaluation, illustrated in Figure 1(b), 1(d) and 1(f), compared with the differences between these algorithms in the first evaluation (Figure 1(a), 1(c) and 1(e)), confirm that the performance improvement of the UBExtended algorithm increases for



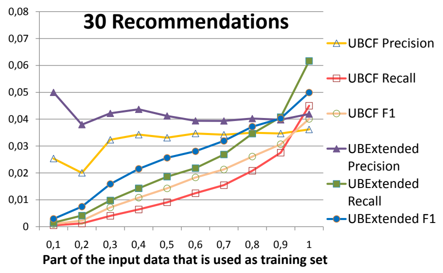
(a) 10 recommendations based on the initial training set



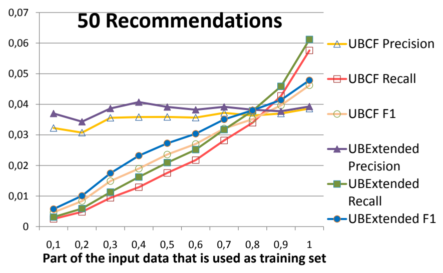
(b) 10 recommendations based on the training set which contains only sparse profiles



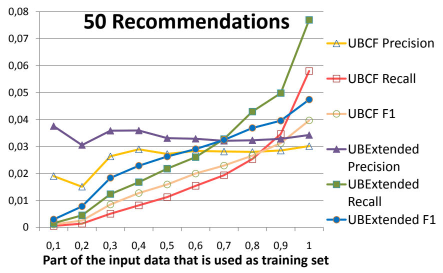
(c) 30 recommendations based on the initial training set



(d) 30 recommendations based on the training set which contains only sparse profiles



(e) 50 recommendations based on the initial training set



(f) 50 recommendations based on the training set which contains only sparse profiles

Fig. 1. The evaluation of the UBCF and UBExtended algorithm based on the precision, recall and F1 metric

sparser datasets. Finally, the graphs of this second evaluation show that for small training sets the precision might slightly fluctuate because of insufficient data and lots of new users.

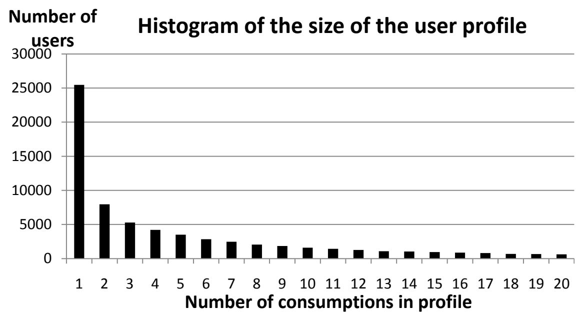


Fig. 2. Histogram of the number of sheets in the user profiles (i.e. the profile size) for the PianoFiles dataset. Profile sizes larger than 20 are not showed, because of the limited number of users that have such a profile.

7 Optimisations & Drawbacks

Since the parameters of the UBExtended algorithm are not yet optimized in this evaluation, the performance difference between the two bench-marked algorithms might even increase considerably. The UBExtended algorithm extends sparse profiles until each profile contains a predefined number of consumptions. This target profile size is an important parameter that has to be optimized as a function of the performance metrics. Although we have chosen a fixed size of 6 consumptions for the extended profiles in our evaluation, we believe this parameter might be a function of the general characteristics of the dataset, namely the overall sparsity of the data matrix, the number of items, and the number of users. Moreover, the procedure of extending the profiles, which is based on general and profile-based influences, can be fine-tuned. An optimal balance between this general and profile-based information to extend the profiles might result in more precise recommendations. Finally, some typical CF parameters have to be determined, such as the similarity metric and the number of neighbours used to calculate the recommendations.

Unfortunately, the accuracy improvement acquired with the UBExtended algorithm is associated with an extra calculation cost. Compared to the standard UBCF, the UBExtended algorithm consists of 2 extra phases: extending the profiles and recalculating the similarities after this extension. Especially the similarity calculation can be time-consuming due to its quadratic nature and therefore it may pose problems for systems that calculate the recommendations in real-time (i.e. generating recommendations when the web page is requested).

However, since most recommender systems schedule the calculations and update their recommendations periodically, the additional calculation time is no major problem.

8 Conclusions & Future Work

In this research, we proposed an advanced collaborative filtering algorithm which takes into account the specific characteristics of user-generated content systems. The algorithm extends sparse profiles with the most likely future consumptions based on general and personal consumption behaviour. Our experimental study, using a dataset from a user-generated content site (PianoFiles), proved that the user-based version of the proposed algorithm achieves more accurate results than the standard user-based collaborative filtering algorithm, especially on sparse datasets. These results confirm the need to adapt traditional collaborative filtering techniques to the specific characteristics, such as the sparsity, of user-generated content systems. In future research, we will try to optimise the algorithm parameters to further improve the performance results. Besides, we will investigate if the principle of profile extension is applicable in other types of collaborative filtering algorithms.

Acknowledgments. We would like to thank the Research Foundation - Flanders (FWO), for the research position of Toon De Pessemier (Aspirant FWO). We would also like to express our appreciation to Thomas Bonte, the founder of PianoFiles, for putting the dataset of his website at our disposal.

References

1. Bell, R.M., Koren, Y.: Lessons from the netflix prize challenge. SIGKDD Explor. Newsl. 9(2), 75–79 (2007)
2. Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. pp. 43–52. Morgan Kaufmann (1998)
3. Campochiaro, E., Casatta, R., Cremonesi, P., Turrin, R.: Do metrics make recommender algorithms? Advanced Information Networking and Applications Workshops, International Conference on 0, 648–653 (2009)
4. Celma, O., Cano, P.: From hits to niches?: or how popular artists can bias music recommendation and discovery. In: NETFLIX '08: Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition. pp. 1–8. ACM, New York, NY, USA (2008)
5. Cha, M., Kwak, H., Rodriguez, P., Ahn, Y.Y., Moon, S.: I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In: IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement. pp. 1–14. ACM, New York, NY, USA (2007)
6. Davidson, J., Liebald, B., Liu, J., Nandy, P., Van Vleet, T., Gargi, U., Gupta, S., He, Y., Lambert, M., Livingston, B., Sampath, D.: The youtube video recommendation system. In: RecSys '10: Proceedings of the fourth ACM conference on Recommender systems. pp. 293–296. ACM, New York, NY, USA (2010)

7. Evain, J.P., Martínez, J.M.: Tv-anytime phase 1 and mpeg-7. J. Am. Soc. Inf. Sci. Technol. 58(9), 1367–1373 (2007)
8. Hayes, C., Massa, P., Avesani, P., Cunningham, P.: An on-line evaluation framework for recommender systems. In: In Workshop on Personalization and Recommendation in E-Commerce (Malaga. Springer Verlag (2002)
9. Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. pp. 230–237. ACM, New York, NY, USA (1999)
10. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. 22(1), 5–53 (2004)
11. Huang, Z., Zeng, D., H.Chen: A link analysis approach to recommendation with sparse data. In: AMCIS 2004: Americas Conference on Information Systems. New York, NY, USA (2004)
12. Huang, Z., Chen, H., Zeng, D.: Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. ACM Trans. Inf. Syst. 22(1), 116–142 (2004)
13. Karypis, G.: Evaluation of item-based top-n recommendation algorithms. In: CIKM '01: Proceedings of the tenth international conference on Information and knowledge management. pp. 247–254. ACM, New York, NY, USA (2001)
14. Linden, G., Smith, B., York, J.: Amazon.com recommendations: item-to-item collaborative filtering. Internet Computing, IEEE 7(1), 76–80 (2003), `http://ieeexplore.ieee.org/xpls/abs\_all.jsp?arnumber=1167344`
15. McNee, S.M., Riedl, J., Konstan, J.A.: Being accurate is not enough: how accuracy metrics have hurt recommender systems. In: CHI '06: CHI '06 extended abstracts on Human factors in computing systems. pp. 1097–1101. ACM, New York, NY, USA (2006)
16. Papagelis, M., Plexousakis, D., Kutsuras, T.: Trust Management. Springer Berlin / Heidelberg (2005)
17. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Analysis of recommendation algorithms for e-commerce. In: EC '00: Proceedings of the 2nd ACM conference on Electronic commerce. pp. 158–167. ACM, New York, NY, USA (2000)
18. Segaran, T.: Programming collective intelligence. O'Reilly (2007)
19. Shani, G.: Tutorial on evaluating recommender systems. In: RecSys '10: Proceedings of the fourth ACM conference on Recommender systems. pp. 1–1. ACM, New York, NY, USA (2010)
20. Weng, J., Miao, C., Goh, A., Shen, Z., Gay, R.: Trust-based agent community for collaborative recommendation. In: AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems. pp. 1260–1262. ACM, New York, NY, USA (2006)
21. Yu, Z., Zhou, X.: Tv3p: an adaptive assistant for personalized tv. Consumer Electronics, IEEE Transactions on 50(1), 393–399 (Feb 2004)