Justified Recommendations based on Content and Rating Data

Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos

Aristotle University, Department of Informatics, Thessaloniki 54124, Greece
{symeon, alex, manolopo}@csd.auth.gr

Abstract. Providing justification to a recommendation gives credibility to a recommender system. Some recommender systems (Amazon.com etc.) try to explain their recommendations, in an effort to regain customer acceptance and trust. But their explanations are poor and unjustified, because they are based solely on rating or navigational data, ignoring the content data. In this paper, we propose a novel approach that attains simultaneously accurate and justifiable recommendations. We construct a feature profile for the users, to reveal their favorite features. Moreover, we create biclusters (i.e. group of users which exhibit highly correlated ratings on groups of items) to exploit partial matching between the preferences of the test user and each community of users. We have evaluated the quality of our justifications with an objective metric in a real data set, showing the superiority of the proposed approach. We also conducted a user study to measure users' satisfaction against the existing and our proposed justification style. The user study shows that our justification style is users' favorite choice.

1 Introduction

Collaborative Filtering (CF) is a method that provides personalized recommendations, based on suggestions of users with similar preferences. Up to the present day, the development of CF algorithms have focused mainly to provide accurate recommendations [4]. Nevertheless, besides the accuracy of its recommendations, the acceptance of a recommender system is increased when users can understand the strengths and limitations of the recommendations [5]. This can be attained when users receive, along with a recommendation, the reasoning behind it. Such a combination is denoted as *justified recommendation*. Justified recommendations offer credibility to a recommender system [5].

As the need for justified recommendations has started to gain attention, several recommender systems, like that of Amazon, adopted the following style of justification: "Customers who bought item X also bought items $Y, Z, \ldots$". This is the so called "nearest neighbor" style [3] of justification. In contrast, with the so called "influence" style, justifications are of the form: "Item Y is recommended because you rated item X".[1] Thus, the system isolates the item,

[1] Amazon offers "influence style" justification through a link named "Why was I recommended this?" next to the recommended items.

X, that influenced most the recommendation of item Y. Bilgic et al. [3] claimed that the "influence" style is better than the "nearest neighbor" style, because it allows users to accurately predict their true opinion of an item. Nevertheless, both styles can not justify adequately their recommendations, because they are based solely on data about user preferences, i.e., ratings or navigational data (for simplicity, we henceforth refer to user-preference data as rating data), ignoring the content data.

Several CF systems have proposed the combination of *content* data with *rating* data. However, they aimed only to improve the accuracy of their recommendations [2, 10]. The combination of content with rating data helps to capture more effective correlations between users or items, which yields more accurate recommendations. Besides improving the accuracy of its recommendations, the combination can provide high quality justifications as well. Nevertheless, all the aforementioned systems do provide accurate recommendations, but they can not adequately justify them. The reason is three-fold: (i) They are hybrid (run CF on the results of CB [2] or vice versa [10]), thus they miss the dependency between users and features. (ii) They can not detect partial matching of user's preferences. (iii) They lack objective metrics to evaluate the quality of justifications. Up to the present day, the quality of justifications has been evaluated only with subjective criteria [3, 5].

We propose the following: (i) To capture the interaction between users and their favorite features we construct a feature profile for the users. (ii) To detect partial matching of user's preferences, we propose the generation of biclusters (i.e. group of users which exhibit highly correlated ratings on groups of items). (iii) To cover the lack of an objective evaluation method for the quality of justifications, we propose the *coverage*, which is the percentage between the sum of features that are favorite to a user and used to justify a recommendation to the total sum of favorite features of a user. If coverage is high, then the justification is more effective, as the features that are included in the justification can be easily recognized and accepted by the user.

The contributions of this paper are summarized as follows: (i) We construct a feature profile for the users, to reveal the real reasons of their rating behavior. We also apply a feature-weighting scheme, to find those features which better describe a user, and those which better distinguish him from the others. (ii) We create biclusters, to be able to detect partial matching of users' preferences. We also use a similarity measure based on items and features, which can provide accurate and simultaneously justifiable recommendations. (iii) Differently from prior work, we measure the quality of our justifications with an objective metric, coverage ratio, in a real data set (Movielens), illustrating the superiority of the proposed approach over existing CB, CF and hybrid approaches.

The rest of this paper is organized as follows. Section 2 summarizes the related work, whereas Section 3 contains the problem description. The proposed approach is described in Section 4. Experimental results are given in Section 5. Finally, Section 6 concludes this paper.

2 Related work

In 2000, Herlocker et al. [5] proposed 21 different interfaces of explaining CF recommendations. They demonstrated that the "nearest neighbor" style is effective in supporting explanations. To prove this, they conducted a survey with 210 users of the Movielens recommender system, demonstrating that explanations can improve the acceptance of CF systems.

In 2005, Bilgic et al. [3] claimed that "influence" style and "keyword-based" style are better than the "nearest neighbor" style explanations, proposed by [5], because they help users to accurately predict their true opinion of an item. In contrast, "neighbor style" explanation caused users to overestimate the quality of an item. To prove their claim, they conducted an online survey on their book recommender system LIBRA [8], which was initially developed as a purely content-based system containing a database of 40,000 books. The current version employs a hybrid approach called Content-Boosted Collaborative Filtering (CBCF) [6].

In recent years, in the same direction as CBCF, there have been several hybrid attempts to combine CB with CF. The Fab System [2], measures similarity between users after first computing a CB profile for each user. In contrast, the CinemaScreen System [10] runs CB on the results of CF. Melville et al. [6] used a content-based predictor to enhance existing user data, and then to provide personalized suggestions though collaborative filtering.

All these are hybrid systems that mainly aim in advancing the accuracy of recommendations. Our approach aims also to improve the quality of justifications, exploiting partial matching between a user and his nearest biclusters. Moreover, we propose an objective metric to measure the quality of justifications of our approach with representative algorithms of CB, CF and other hybrid approaches. We used in the comparison a pure content-based filtering (CB) algorithm. From the CF algorithms family, we used the user-based (UB) and item-based (IB) algorithms [11]. Finally, as representative of the hybrid algorithms, we have implemented a state-of-the-art algorithm, the Cinemascreen Recommender Agent [10], denoted as CFCB.

3 Problem Description

In this section, we define the basic notions that will be used through out the paper. Our goal is to make accurate and simultaneously justifiable recommendations, by providing a list of favorites features to a user, that cover a high percentage of his profile.

Rating and Content data: CF algorithms process the rating data of the users to provide accurate recommendations. An example of rating data is given in Figure 1a, where I_{1-7} are items and U_{1-8} are users. The null cells (no rating) are presented with dash.

Definition 1 *The rating profile $R(U_k)$ of user U_k is the k row of matrix R.*

For instance, $R(U_1)$ is the rating profile of user U_1, and consists of the rated items I_1,I_3 and I_5. Thus, the rating of a user U_k over an item i is given from the element $R(U_k, i)$ of matrix R.

	I_1	I_2	I_3	I_4	I_5	I_6	I_7
U_1	5	-	2	-	1	-	-
U_2	2	-	4	1	4	3	-
U_3	4	-	2	-	2	-	5
U_4	-	3	1	4	-	5	2
U_5	-	2	4	2	5	1	-
U_6	5	1	-	1	-	-	3
U_7	-	2	5	-	4	1	-
U_8	1	4	-	5	4	3	-

(a)

	f_1	f_2	f_3	f_4
I_1	1	0	0	0
I_2	0	1	1	0
I_3	0	0	1	1
I_4	0	1	0	1
I_5	0	1	1	1
I_6	0	0	1	1
I_7	1	0	0	0

(b)

	f_1	f_2	f_3	f_4
U_1	1	0	0	0
U_2	0	1	3	3
U_3	2	0	0	0
U_4	0	2	2	2
U_5	0	1	2	2
U_6	2	0	0	0
U_7	0	1	2	2
U_8	0	3	3	3

(c)

Fig. 1. Running example: (a) User-Item matrix R (b) Item-Feature matrix F (c) User-Feature matrix P.

As described, content data are provided in the form of features. In our running example illustrated in Figure 1b, for each item we have four features that describe its characteristics. We use matrix F, where element $F(i,\ f)$ is one, if item i contains feature f and zero otherwise.

Definition 2 *The item profile $F(I_k)$ of item I_k is the k row of matrix F.*

For instance, $F(I_2)$ is the profile of item I_2, and consists of features F_2 and F_3. Notice that this matrix is not always boolean. Thus, if we process documents, matrix F would count frequencies of terms.

Construction of the Feature profile: The combination of content with rating data discloses effective correlations between users and their favorite features. To capture the interaction between users and their favorite features, we construct a feature profile composed of the rating profile and the item profile.

For the construction of the feature profile of a user, we use a positive rating threshold, P_τ, to select items from his rating profile, whose rating is not less than this value. The reason is that the rating profile of a user consists of ratings that take values from a scale(in our running example, 1-5 scale). It is evident that ratings should be "positive", as it is not favorite an item that is rated with, e.g., 1 in 1-5 scale.

Definition 3 *The feature profile $P(U_k)$ of user U_k is the k row of matrix P whose elements $P(U_k,f)$ are given by Equation 1.*

$$P(U_k, f) = \sum_{\forall R(U_k,i) > P_\tau} F(i, f) \tag{1}$$

In Figure 1c, element $P(U_k,f)$ denotes the correlation between user U_k and feature f. In our running example (with $P_\tau = 2$), P(U_2) is the feature profile of user U_2, and consists of features f_2, f_3 and f_4. The correlation of a user U_k over a feature f is given from the element $P(U_k, f)$ of matrix P. As shown, features f_3 and f_4 describe him better, than feature f_2 does.

Based on the aforementioned problem data, we provide to user U_k two lists. A top-n list L of items $L(U_k)$=$\{I_1,...,I_n\}$, with which we provide our recommendation. Moreover, a list J of ordered pairs $J(U_k)$=$\{(f_1,c_1),...,(f_m,c_m)\}$, with which we provide our justification for the recommendation in L. Each ordered pair contains a feature f with its frequency c inside list L. In particular, the frequency c of feature f for a user U_k is given from Equation 2:

$$c_f(U_k) = \sum_{\forall j: I_j \in L(U_k)} F(j, f) \tag{2}$$

In our running example, assume we recommend a top-2 list of items, to user U_1 $L(U_1)$=$\{I_2, I_3\}$. The $J(U_1)$ list would be $J(U_1) = \{(f_3, 2)$, $(f_2, 1)$, $(f_4, 1)$, $(f_1, 0)\}$.

Evaluation Metrics: So far, for the evaluation of CF, CB and hybrid algorithms all metrics process only the rating profile of a user. For a test user that receives a top-N recommendation list L, let M denote the number of relevant recommended items(the items of the L list that are rated higher than P_τ by the test user). Precision is the ratio of M to N, while Recall is the ratio of M to the total number of relevant items for the test user (all items rated higher than P_τ by him).

However, precision and recall cannot distinguish between a relevant item from a more relevant item. For instance, different users might have a different interpretation of which item is more relevant and which is not. To cope with this problem, we introduce the coverage ratio, which is a user-oriented measure.

In order to be able to focus to the most relevant features in the feature profile of a user U_k, we use similarly to P_τ threshold, an F_τ threshold, which considers features whose value is not less than this value in his feature profile. Henceforth, features that have values higher than F_τ are denoted as relevant features.

Definition 4 *We define as coverage ratio of a user U_k the sum of the relevant features in the $J(U_k)$ list to the total sum of relevant features that exist in his feature profile.*

In particular, the coverage ratio of user U_k is given from Equation 3:

$$coverage(U_k) = \frac{\sum_{\forall f \in F} c_f(U_k)}{\sum_{\forall f \in F} P(U_k, f)} \tag{3}$$

If coverage is high, then the justification is more effective, as the features that are included in the justification can be easily recognized and accepted by the user. A relevant feature to a user should be determinative for distinguishing

him from the others. These relevant features will be extracted with a feature-weighting scheme, we will present later.

In our running example (with $F_\tau = 0$), assume we recommend a top-2 list of items to user U_8. If we recommend I_2 and I_4 we get 100% in precision and 50% recall. The same precision and recall we get, when we recommend I_5 and I_6. But the coverage ratio is not the same. The former top-2 list results 44% coverage (4/9 features), while the latter gives 55% (5/9 features). Thus, an algorithm which attains more coverage ratio can give more justifiable recommendations.

4 The proposed approach

4.1 The creation of biclusters

In this section, we propose the generation of groups of users *and* items at the same time. The simultaneous clustering of users and items discovers *biclusters*, which correspond to groups of users which exhibit highly correlated ratings on groups of items. Biclusters allow the computation of similarity between a test user and a bicluster *only* on the items or features that are included in the bicluster. Thus, partial matching of preferences is taken into account.

For this biclustering step, we have adopted another biclustering algorithm, xMotif, which looks for subsets of rows and subsets of columns with coherent values [9].

In Figure 2, we apply xMotif to matrix R of our running example, forming communities of users based on their rating profile. Otherwise, we could not preserve the information about items they have rated, to use it for our recommendations. The feature profile of users will be used for the justification of our recommendations.

	I_4	I_2	I_6	I_5	I_3	I_1	I_7
U_3	-	-	-	2	2	4	5
U_6	1	1	-	-	-	5	3
U_1	-	-	-	1	2	5	-
U_5	2	2	1	5	4	-	-
U_7	-	2	1	4	5	-	-
U_2	1	-	3	4	4	2	-
U_8	5	4	3	4	-	1	-
U_4	4	3	5	-	1	-	2

Fig. 2. Applying xMotif algorithm to matrix R.

We found four biclusters which consist, at least, of 2 users and 2 items. These bilcusters are summarized as follows:

Notice that there is overlapping between biclusters. We can allow this overlapping or we can forbid it. However, in order not to miss important biclusters, we allow a percentage of overlapping, modeling the possibility that the user may have different preferences or an item can belong in many genres.

$$
\begin{array}{llll}
b_1\colon U_{b_1} = & \{U_3, U_6, U_1\}, & I_{b_1} = & \{I_1, I_7\} \\
b_2\colon U_{b_2} = & \{U_5, U_7, U_2, U_8\}, & I_{b_2} = & \{I_5, I_3\} \\
b_3\colon U_{b_3} = & \{U_2, U_8\}, & I_{b_3} = & \{I_6, I_5, I_3\} \\
b_4\colon U_{b_4} = & \{U_8, U_4\}, & I_{b_4} = & \{I_4, I_2, I_6, I_5\}
\end{array}
$$

4.2 The application of a feature-weighting scheme

In this section, we weight the feature profiles, in order to find those features which better describe a user. Similarly, we apply a feature-weighting scheme to the features contained in each bicluster.

We will perform a TF/IDF weighting scheme [1] to the features of matrix P, in order to find (i) those features which better describe user u (describe the $\mathcal{F}_u$ set) and (ii) those features which better distinguish him from the others (distinguishing him from the remaining users in the $\mathcal{U}$ domain).

In our running example, the matrix P of Figure 1c is transformed into the matrix W of Figure 3. Notice that feature f_1 is dominant in the feature profile of users U_3 and U_6.

	f_1	f_2	f_3	f_4
U_1	0.43	0	0	0
U_2	0	0.20	0.61	0.61
U_3	0.85	0	0	0
U_4	0	0.41	0.41	0.41
U_5	0	0.20	0.41	0.41
U_6	0.85	0	0	0
U_7	0	0.20	0.41	0.41
U_8	0	0.61	0.61	0.61

Fig. 3. Weighted User-Feature matrix W

To be able to compare a user with a bicluster, similarly to the rating profile of users, we create a rating profile of biclusters. Thus, we define an R_b matrix, whose elements $R_b(b, i)$ are given from the frequency of item i in a bicluster b. Figure 4a, shows matrix R_b for our running example.

Similarly to the feature profile of users, we generate a feature profile of each bicluster. We define P_b matrix whose elements $P_b(b, f)$ are given from the frequency of feature f in a bicluster b. Finally, from P_b matrix we generate the weighted W_b matrix. The R_b, P_b and W_b matrices concern biclusters, while the R, P and W matrices concern users. Figures 4b and c, show the P_b and weighted W_b matrices for our running example.

4.3 The neighborhood formation of a user

In this section, we find biclusters containing features that have strong partial similarity with the test user. These features can be used for justifying our rec-

	I_1	I_2	I_3	I_4	I_5	I_6	I_7
b_1	3	0	0	0	0	0	2
b_2	0	0	3	0	4	0	0
b_3	0	0	2	0	2	2	0
b_4	0	2	0	2	1	2	0

(a)

	f_1	f_2	f_3	f_4
b_1	5	0	0	0
b_2	0	4	7	7
b_3	0	2	6	6
b_4	0	5	5	5

(b)

	f_1	f_2	f_3	f_4
b_1	3.01	0	0	0
b_2	0	0.50	0.87	0.87
b_3	0	0.25	0.74	0.74
b_4	0	0.62	0.62	0.62

(c)

Fig. 4. (a) Bicluster-Item matrix R_b. (b) Bicluster-Feature matrix P_b. (c) Weighted Bicluster-Feature matrix W_b.

ommendations. The neighborhood formation of a user, i.e., to find the k nearest biclusters, requires to measure the similarity of the test user and each of the biclusters. There are two different ways to measure similarity:

(i) To consider the similarity of test user and a bicluster *only* on the items that are included in the bicluster and not on all items that he has rated. As described, this allows for the detection of partial similarities. In Equation 4, we calculate the cosine similarity between a user u and bicluster b for the items he rated positively ($R(u,i) > P_\tau$) as follows:

$$\text{sim1}(u,b) = \frac{\sum_{\forall i \in X} R(u,i) R_b(b,i)}{\sqrt{\sum_{\forall i \in X} R(u,i)^2} \sqrt{\sum_{\forall i \in X} R_b(b,i)^2}}, X = \mathcal{I}_u \cap \mathcal{I}_b. \tag{4}$$

(ii) To consider the similarity of test user and a bicluster *only* on the features that are included in the bicluster. In our approach, as it is expressed by Equation 5, we calculate the cosine similarity between a test user and a bicluster.

$$\text{sim2}(u,b) = \frac{\sum_{\forall f \in X} W(u,f) W_b(b,f)}{\sqrt{\sum_{\forall f \in X} W(u,f)^2} \sqrt{\sum_{\forall f \in X} W_b(b,f)^2}}, X = \mathcal{F}_u \cap \mathcal{F}_b. \tag{5}$$

In our running example, the nearest bicluster ($k = 1$)of user U_1, considering only one of Equations 4 or Equation 5, respectively, gives the same bicluster b_1.

When the two similarity measures converge to close results, this means that the features confirm the user's rating behavior. But they do not always converge due to many reasons. For instance, the selected item features maybe a small fraction of a wider set of features, and can not describe well the user's rating behavior, giving less accurate recommendations. On the other hand, features can give more precise recommendations than items do, when they solve sparsity problems in the rating profile of a user, resulted from his unwillingness to rate items. For these reasons, we choose a mixed similarity measure that builds into the two aforementioned measures. In Equation 6, we calculate the cosine similarity between a user u and bicluster b as follows:

$$\text{sim}(u, b) = (1 - a) \cdot \text{sim1}(u, b) + a \cdot \text{sim2}(u, b) \quad (6)$$

Parameter a takes values between [0,1]. As we increase a, we demand from the system to give more justifiable recommendations, concentrating on the dominant features.

4.4 The extraction of the dominant features

In this section, we extract the dominant features inside the nearest biclusters of a test user u. Thus, items that contain those dominant features are favored and used in our recommendations. To extract the dominant features in the nearest biclusters of a user, we define matrix F_b, whose elements $F_b(i, f)$ are given from Equation 7:

$$F_b(i, f) = R'_b(b, i) \cdot W_b(b, f), \forall b \in B_u \quad (7)$$

Element $F_b(i, f)$ of matrix F_b denotes the influence of feature f of item i in the biclusters' neighborhood of a user u, while B_u is the set of nearest biclusters of user u.

In our running example, the two nearest biclusters ($k = 2$) of user U_1, considering only Equation 4 are first, b_1 and follows, one of b_2, b_3, b_4 with the same probability. To ease our discussion, let's assume that the system chooses, as second nearest bicluster, b_4. Then, the calculations for the extraction of the dominant items and features in the user's neighborhood are shown in Figures 5a, b, and c.

To find the total influence of an item in the user's neighborhood, we add $F_b(i, f)$ elements of matrix F_b for each individual item i, as it is shown in Equation 8 and Figure 5d.

$$Influence(i) = \sum_{\forall f \in F} F_b(i, f) \quad (8)$$

Thus, we reveal the items that contain the most dominant features. We keep the j (j>N) number of items with the highest aggregated values creating the B_b set of items. Then, we exclude those items from B_b that are rated already from the test user.

In our running example, we set j=5 and N=1. Thus, the items of B_b set are $\{I_1, I_7, I_2, I_4, I_6\}$. Then, we exclude item I_1 which is already rated by U_1 and the remaining in B_b items are $\{I_7, I_2, I_4, I_6\}$.

4.5 The generation of the recommendation and justification lists

As we described in section 3, our recommendations consist of a top-N list L of items, which is followed by a justification J list of features. The J list aims to contains features that maximize the coverage on the favorite features of a user. To be able to create this J list for a user u, we define a new matrix D whose elements $D(u, i)$ are given from Equation 9:

	b_1	b_2
I_1	3	0
I_2	0	2
I_3	0	0
I_4	0	2
I_5	0	1
I_6	0	2
I_7	2	0

(a)

	f_1	f_2	f_3	f_4
b_1	3.01	0	0	0
b_2	0	0.62	0.62	0.62

(b)

	f_1	f_2	f_3	f_4
I_1	9.03	0	0	0
I_2	0	1.24	1.24	1.24
I_3	0	0	0	0
I_4	0	1.24	1.24	1.24
I_5	0	0.62	0.62	0.62
I_6	0	1.24	1.24	1.24
I_7	6.02	0	0	0

(c)

	$Influence$
I_1	9.03
I_2	3.72
I_3	0
I_4	3.72
I_5	1.86
I_6	3.72
I_7	6.02

(d)

Fig. 5. Matrices (a) $R'_b(b,i)$ (b) $W_b(b,f)$ (c) $F_b(i,f)$ (d) Influence(i)

$$D(u,i) = W(u,f) \cdot F'(i,f), \forall i \in B_b \tag{9}$$

Element $D(u,i)$ of matrix D denotes the total influence of item i in the feature profile of a user u, while B_b is the set of items extracted from the user's neighborhood. We recommend first those items that are the most influential in the feature profile of the test user.

In our running example, we take the extracted items from the previous phase (i.e. I_2, I_4, I_6, I_7) and generate the D matrix as it is shown in Figure 6. It is obvious that only I_7 will be our recommendation. Item I_7 is recommended because it contains feature f_1, which covers 100% percentage the user's U_1 feature profile, due to item I_1 he has already rated.

	f_1	f_2	f_3	f_4
U_1	0.43	0	0	0

(a)

	I_2	I_4	I_6	I_7
f_1	0	0	0	1
f_2	1	1	0	0
f_3	1	0	1	0
f_4	0	1	1	0

(b)

	I_2	I_4	I_6	I_7
U_1	0	0	0	0.43

(c)

Fig. 6. Matrices $W(u,f)$, $F'(i,f)$ and $D(u,i)$

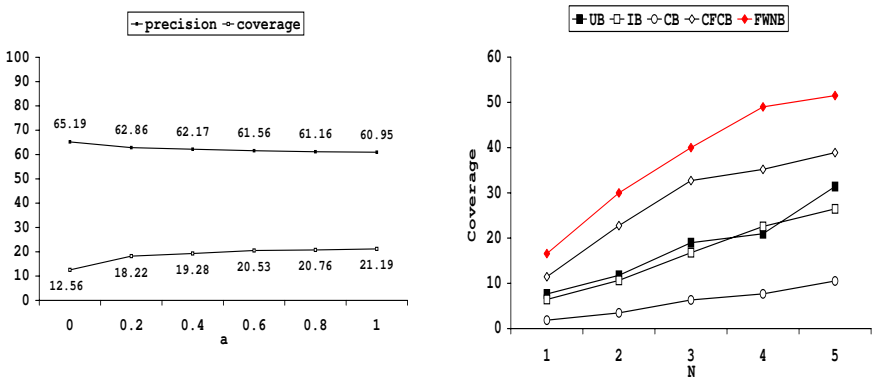
5 Experimental Configuration

In this section, we study the performance of our approach against well-known CF, CB and hybrid approaches, by means of a thorough experimental evaluation. Henceforth, our feature-weighted nearest biclusters algorithm is denoted as FWNB. We use in the comparison a pure content-based filtering (CB) algorithm, denoted as CB. From the CF algorithms family, we used the user-based and item-based algorithms [11] denoted as UB and IB, respectively. Finally, as

representative of the hybrid algorithms, we have implemented a state-of-the-art algorithm, the Cinemascreen Recommender Agent [10], denoted as CFCB. The metrics we use are recall, precision and coverage ratio. We perform experiments with a real data set that has been used as benchmarks in prior work. In particular, we examined the 100K MovieLens data set. The extraction of the content features has been done through the internet movie database (imdb). The join process lead to 23 different genres, 9847 keywords, 1050 directors and 2640 different actors and actresses. We performed 4-fold cross validation and the default size of the training set is set to 75%.

5.1 Tuning of a parameter

In Section 4.3, we introduced parameter a, which builds into, the item and feature similarity measures. As described, when both similarity measures converge to close results, this means that the feature profile of a user confirms his rating profile, revealing his rating behavior. Thus, we can attain high accuracy in our recommendations and simultaneously high coverage in our justifications.



(a) (b)

Fig. 7. (a) Precision and Coverage of FWNB vs. a (b) Comparison between UB, IB, CB, CFCB and FWNB in terms of coverage vs. N.

In Figure 7a, we present the precision and coverage ratio of FWNB vs. different values of a parameter. When a equals to 0, our measure is based exclusively on the item similarity measure. In contrast, when a equals to 1, it is based only on the feature similarity measure. As shown, as we increase a, precision ratio is almost stable(from 65,19% to 60,95%). This means that the feature similarity measure confirms the results of the item similarity measure. However, as we increase a the coverage ratio is raised drastically(from 12,56% to 21,19%). Having a high coverage ratio, we can get more justifiable recommendations. The best combination of precision and coverage is attained when a is set to 0.4 with

62,17% precision and 19,28% coverage ratio. In the following, we keep this as the default value.

5.2 Measuring the quality of justifications

The justification of our recommendations can be done easily, if we manage to reveal the dominant features in the feature profile of the users. Thus, users can receive, along with a recommendation, the reasoning behind it. In the following, we test the ability of the five algorithms to reveal the dominant features of the feature profile of the test user.

In Figure 7b, we present the coverage ratio of FWNB, with the UB, IB, CB and CFCB algorithms vs. different sizes of the recommendation list. As expected, FWNB outperforms all four algorithms(19,28% coverage ratio). In contrast, CFCB covers only a 16% of the user's features, when we recommend 20 items. The reason is that FWNB is able to extract collective features from the nearest biclusters. FWNB also uses a similarity measure based on both items and features, which can provide more justifiable recommendations.

5.3 Measuring the accuracy of recommendations

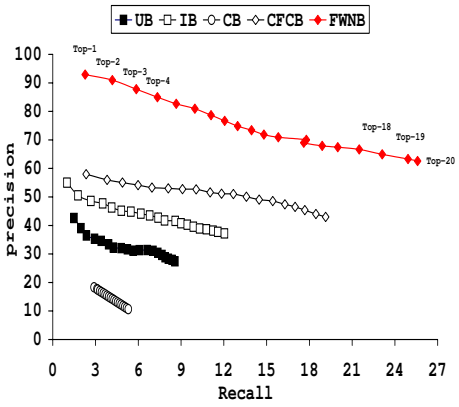
In this section, we proceed with the comparison of FWNB with UB, IB, CB and CFCB, in terms of precision and recall. This reveals the robustness of each algorithm in attaining high recall with minimal losses in terms of precision. We examine the top-N ranked list L, which is recommended to a test user, starting from the top item. In this situation, the recall and precision vary as we proceed with the examination of the top-N list.

In Figure 8a, we plot a precision versus recall curve for all five algorithms. As shown, all algorithms' precision falls as N increases. In contrast, as N increases, recall for all five algorithms is increases too. FWNB attains 70% precision, when we recommend a top-20 list of items. In contrast, CFCB gets a precision of 42%. Moreover, FWNB is more effective than CFCB getting a maximum recall of 27%, while the latter's is 20,5%. This experiment shows that FWNB is more robust in finding relevant items from the rating profile of the test user. The reason is that FWNB similarity measure is based on nearest-biclusters, and thus, being able to detect partial matching of users' preferences, can provide accurate recommendations.

5.4 User Study

As described, our justification style combines the "keyword-based" and the "influence" styles, having the following form: "Item X is recommended, because it contains features $a, b, \ldots$, which are included in items $Z, W, \ldots$ that you have already rated". In this section, we present some justifications that are obtained by our method for a real data set, the Movielens.

We selected a user at random (among the 943 users of the set), and recommended two movies. Table 1 depicts these recommended movies along with their



(a)

Expl. Styles	μ_p	σ_p
KSE	1.86	1.02
ISE	2.26	1.20
KISE	**3.71**	1.08

(b)

Fig. 8. Comparison between UB, IB, CB, CFCB and FWNB in terms of precision vs. recall. (b) Results of the user survey.

justifications. Notice that the second column of Table 1 concerns the "keyword-based" explanation style, whereas the third column of Table 1 concerns the "influence" explanation style. The combination of those two columns is our proposed justification style.

Recommended Movie title	**The reason is the participant**	**who appears in**
Indiana Jones and the Last Crusade (1989)	Ford, Harrison	5 movies you have rated
Die Hard 2 (1990)	Willis, Bruce	2 movies you have rated

Table 1. Justification example.

We also conducted a survey to measure user satisfaction against the three styles of explanations: "keyword-based" style (denoted as KSE), "influence" style (denoted as ISE), and our style of explanation (denoted as KISE), which combines the two aforementioned ones. We designed the user study with 42 pre- and post-graduate students of Aristotle University, who filled out an on-line survey. In particular, we asked each target user to provide us with ratings for at least five movies that exist in the Movielens data set. Then, we recommended to each target user a movie, justifying our recommendation by using the three justification styles (a different style each time). Finally, we asked target users to rate (in 1-5 rating scale) each explanation style separately to explicitly express their actual preference among the three styles.

In Figure 8b, we present the mean μ_p and standard deviation σ_p of ratings provided by the users to explicitly express their preference for each explanation

style. KISE attained a μ_p value equal to 3.71, which is the largest among all styles. We run paired t-test, and found out that the difference of KISE from KSE and ISE is statistically significant at the 0.01 level.

Our measurements show that KISE will be the users' favorite choice, because it can be more informative and combines the other two explanation styles. Thus, our proposed justification style can be suitable for real-world recommender systems.

6 Conclusions

We propose an approach to attain both accurate and justifiable recommendations. We perform experimental comparison of our method against well-known CF and a hybrid algorithm with two real data sets. Our approach builds a feature profile for the users, that reveals the real reasons of their rating behavior. Moreover, we group users into biclusters to exploit partial matching between the preferences of the target user and each group of users. Finally, the *coverage ratio* is an objective metric to measure the quality of justifications, which illustrates the superiority of our approach over the existing CF and hybrid approaches. We also conducted a survey with real users in order to verify that the proposed justification style is suitable for real-world recommender systems.

References

1. R. A. Baeza-Yates and B. A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.
2. M. Balabanovic and S. Y. Fab: Content-based, collaborative recommendation. *ACM Communications*, 40(3):66–72, 1997.
3. M. Bilgic and R. Mooney. Explaining recommendations: Satisfaction vs. promotion. In *Proc. Recommender Systems Workshop (IUI Conf.)*, 2005.
4. D. Goldberg, D. Nichols, M. Brian, and D. Terry. Using collaborative filtering to weave an information tapestry. *ACM Communications*, 35(12):61–70, 1992.
5. J. Herlocker, J. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Computer Supported Cooperative Work*, pages 241–250, 2000.
6. P. Melville, R. J. Mooney, and N. R. Proc. aaai conf. In *Content-Boosted Collaborative Filtering for improved Recommendations*, pages 187–192, 2002.
7. B. Mobasher, R. Burke, R. Bhaumic, and C. Williams. Towards trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Internet Technology*, 7(2):to appear, 2007.
8. R. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *Proc. ACM DL Conf.*, pages 195–204, 2000.
9. T. Murali and S. Kasif. Extracting conserved gene expression motifs from gene expression data. In *Proceedings of the Pacific Symposim on Biocompomputing Conference*, volume 8, pages 77–88, 2003.
10. J. Salter and N. Antonopoulos. Cinemascreen recommender agent: Combining collaborative and content-based filtering. *Intelligent Systems Magazine*, 21(1):35–41, 2006.
11. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proc. WWW Conf.*, pages 285–295, 2001.