Providing Justifications in Recommender Systems

Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos

***Abstract*—Recommender systems are gaining widespread acceptance in e-commerce applications to confront the "information overload" problem. 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. However, their explanations are not sufficient, because they are based solely on rating or navigational data, ignoring the content data. Several systems have proposed the combination of content data with rating data to provide more accurate recommendations, but they cannot provide qualitative justifications. In this paper, we propose a novel approach that attains both accurate and justifiable recommendations. We construct a feature profile for the users to reveal their favorite features. Moreover, we group users into biclusters (i.e., groups of users which exhibit highly correlated ratings on groups of items) to exploit partial matching between the preferences of the target user and each group of users. We have evaluated the quality of our justifications with an objective metric in two real data sets (Reuters and MovieLens), showing the superiority of the proposed method over existing approaches.**

***Index Terms*—Collaborative filtering (CF), content-based filtering (CB), e-commerce, justification, recommender systems.**

I. Introduction

RECOMMENDER systems are gaining widespread acceptance in e-commerce applications to confront the "information overload" problem. Collaborative filtering (CF) is a method that provides personalized recommendations, based on suggestions of users with similar preferences. CF was initially developed to support information filtering systems, like the Information Tapestry project [8]. During the recent years, CF has attracted significant attention in the area of retail and particularly in e-commerce (e.g., recommender systems of Amazon and eBAY).

Up to the present day, the development of CF algorithms have focused mainly on how to provide accurate recommendations [8]. 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 [9]. 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 [9]. For instance, in e-commerce, justified recommendations help to improve customer attraction/retention and sales boosting, because customers can evaluate the provided recommendations more easily and accept them if satisfactory. The need for justification is nowadays even more crucial, due to *shilling attacks* by malicious web robots [14], which favor or disfavor a given item. In a recommender system that is under a shilling attack and does not provide justifications, users cannot understand why they receive unwanted recommendations (possibly with offensive material) resulting from such an attack.

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 [5] 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 X that influenced most the recommendation of item Y. Bilgic and Mooney [5] 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 cannot 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, which are extracted in the form of features that are derived from the items or documents. Content data comprise a valuable source of information in addition to rating data.

Pure content-based filtering (CB) systems [15] assume that each user operates independently. To overcome this limitation, several CF systems have proposed the combination of data about content with rating data [3], [11], [13], [18]. The combination of content with rating data helps to capture more effective correlations between users or items, which yields more accurate recommendations. However, besides the improvement of the accuracy of recommendations, the consideration of content can provide high quality justifications as well. Thus, based on CB, many systems [6], [15] were able to provide explanations for their recommendations. These works were pioneering for the problem of explanation and inspired subsequent research on combining CF and CB for explanation purposes. To further understand the merit of a "*content-based*" justification style, consider the following example.

Example 1: Assume a system that recommends news articles. Users may express interest about an article based on particular features in it, like its category (politics, athletics, etc.), author, or specific terms that it contains. A user has highly rated several articles about scientific news referring to global

Manuscript received April 19, 2007; revised December 20, 2007. Current version published October 20, 2008. This paper was recommended by Associate Editor Y. Narahari.

The authors are with the Department of Informatics, Aristotle University, 54124 Thessaloniki, Greece (e-mail: symeon@csd.auth.gr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCA.2008.2003969

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

warming. If the system decides to recommend another such article X, then a *content-based* style justification can be like: "Article X is recommended because its category is Scientific News and contains the terms {global, warming}, which are features contained in articles you rated." In contrast, an influence style justification will be: "Article X is recommended because you rated article Y." The latter justification burdens the user to make the connection between articles X and Y and understand, e.g., that both refer to global warming. In a system containing thousands of articles, such an effort can be rather discouraging for the users. Thus, the content-based style can be very convenient and effective.

Nevertheless, although existing recommender systems, which combine content with ratings, provide accurate recommendations, they cannot adequately justify them. The reason is threefold:

1) These systems are hybrid. They run CF on the results of CB [3] or vice versa [18]. CF considers the dependence between user and ratings, but misses the dependence between item and features [1]. CB considers the latter, but not the former [1]. Thus, they miss the interaction between the user and his favorite features that can be used for justifying a recommendation.
2) They cannot detect partial matching of the user's preferences. They measure similarity between two users on the entire set of items they have rated. For instance, assume two users that share similar preferences for science-fiction books and differentiate in all other kinds of literature. In this case, only their partial matching for science fiction, should be used for justifying their grouping.
3) Finally, they focus only on the accuracy of their recommendations. Thus, they have objective metrics for accuracy, but lack metrics to evaluate the quality of justifications. Up to the present day, the quality of justifications has been evaluated only with subjective criteria. For instance, Bilgic and Mooney [5] tested the users' satisfaction with the help of small surveys, whereas Herlocker *et al.* [9] followed an analogous approach. Subjective evaluation based on user surveys is important to understand user satisfiability. However, an objective evaluation procedure is required in order to systematically examine the quality of justification in terms of various parameters.

In this paper, we propose to capture the interaction between users and their favorite features by constructing a feature profile for the users. By considering the correlation between users and features, we reveal the actual reasons of their rating behavior. Moreover, we apply a feature-weighting scheme, to find those features which better describe a user and those which better distinguish him from the others.

To detect partial matching of the user's preferences, we propose to group users into biclusters (i.e., group of users which exhibit highly correlated ratings on groups of items). Each bicluster acts like a community for its corresponding items, e.g., in a system that recommends movies, such a group may be users that prefer comedies. Moreover, by using groups instead of individual users, we extract collective features that provide more robust justifications.

To cover the lack of an objective evaluation method for the quality of justifications, we propose the *explain coverage* measure, which is the percentage between the features that are favorite to a user (used to justify a recommendation) to the total 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. We also conducted a survey with real users in order to verify that the proposed justification style is suitable for real-world recommender systems.

The proposed justification style combines the "keyword-based" with the influence styles. It has 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." If these features are dominant inside the user's feature profile, this is a strong evidence for justifying the recommendations. Moreover, we identify that a tradeoff is possible to occur between the accuracy of a recommendation and the quality of its justification. That is, sometimes, a recommendation maybe accurate but cannot be well justified. The proposed algorithm considers this tradeoff and attains both accurate recommendations and strong justifications.

The rest of this paper is organized as follows. Section II summarizes the related work, whereas Section III contains the problem description. The proposed approach is described in Section IV. Experimental results are given in Section V. Finally, Section VI concludes this paper.

II. Related Work

Two families of CF algorithms have been studied extensively in the literature: 1) memory-based algorithms, which perform the computation on the entire database to identify the top k similar users to a target user, and 2) model-based algorithms, which recommend by first developing a model, i.e., by clustering different users into a small number of clusters, based on their rating patterns.

Regarding the memory-based CF algorithms, the GroupLens system [17] proposed an algorithm, known as user-based (UB) CF, because it employs users' similarities for the formation of the neighborhood of nearest users. In 2001, another algorithm was proposed by Sarwar *et al.* [19]. This algorithm is known as item-based (IB) CF algorithm, because it employs items' similarities for the formation of the neighborhood of nearest users. In 2006, Wang *et al.* [21] proposed the similarity fusion (SF) between the UB and IB methods, using also data from a third source (ratings of other similar users on other similar items). In particular, the final rating predictions are estimated by fusing predictions from three sources: 1) predictions based on UB; 2) predictions based on IB; and 3) predictions based on data from similar users rating other similar items.

Regarding the model-based CF algorithms, Xue *et al.* [22] proposed scalable CF using cluster-based smoothing. Their method allows a user to participate in several clusters. Once clusters are created, predictions for a target user can be made by averaging the opinions of the other users in the clusters that he participates, weighted by the degree of participation.

Another algorithm that also uses clustering is the decoupled model (DM) [10]. DM, in contrast to the other cluster-based CF methods, 1) performs separate clustering of users and items, 2) provides flexibility for a user/item to be in multiple clusters, and 3) decouples user preferences from its rating patterns. Finally, George and Merugu [7] used biclustering in CF to build a system that incrementally updates the biclusters as new users and new ratings are being continuously entered. As they claimed, their algorithm mainly aims in real-time efficiency, achieving almost the same accuracy as the other CF algorithms.

Many pure CB systems have tried to provide explanations to users. For instance, Billsus and Pazzani [6] proposed a personal news agent that could talk, learn, and explain. In particular, they used pure CB to recommend news articles to users, providing also explanations for reasoning their recommendations. Moreover, they exploited user's feedback to improve the recommendation process. In 2000, Mooney and Roy [15] proposed a method based also on pure CB for recommending books. Their content-based book recommender system utilized a machine-learning algorithm for text categorization. Based on CB, they were able to provide explanations for their recommendations.

In the area of CF, there is a little existing research on explaining. The GroupLens recommender system [17], which implements a CF algorithm based on common user preferences known as UB CF, has been used to provide CF explanations. In 2000, Herlocker *et al.* [9] 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 GroupLens recommender system, demonstrating that explanations can improve the acceptance of CF systems. Most users would like to see them added to CF systems.

In 2005, Bilgic and Mooney [5] claimed that the influence and keyword-based styles are better than the nearest neighbor style explanations, proposed by [9], 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 [15], 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) [13].

In recent years, in the same direction as CBCF, there have been several hybrid attempts to combine CB with CF. The Fab system [3] combines CB and CF in its recommendations by measuring similarity between users after first computing a profile for each user. Fab initially categorizes documents by a CB filter and then recommends them to the test user based on his relevance feedback. In contrast, CinemaScreen System [18] runs CB on the results of CF. In particular, CinemaScreen System computes predicted rating values for movies based on CF and then applies CB to generate the recommendation list. Melville *et al.* [13] used a content-based predictor to enhance existing user data and then to provide personalized suggestions through CF. Finally, Xin *et al.* [11] proposed a Web recommender system, in which collaborative and content features are integrated under the maximum entropy principle.

TABLE I
SYMBOLS AND DEFINITIONS

Symbol	Definition
$\mathcal{I}$	domain of all items
$\mathcal{U}$	domain of all users
$\mathcal{F}$	domain of all features
$\mathcal{B}$	domain of all biclusters
$\mathcal{I}_u$	set of items rated by user u
$\mathcal{F}_u$	set of features of user u
$N_B(u)$	set of nearest biclusters of user u
I_b	set of items of bicluster b
U_b	set of users of bicluster b
F_b	set of features of bicluster b
F	the item-feature matrix
R	the user-item ratings matrix
R_B	the bicluster-item ratings matrix
P	the user-feature matrix
P_B	the bicluster-feature matrix
W	the weighted user-feature matrix
W_B	the weighted bicluster-feature matrix

The novelty of our approach compared to existing approaches is as follows.

1) Existing CF algorithms perform separate clustering of users and items. Therefore, they ignore the clear duality that exists between users and items, whereas our method takes it into account. This is verified by experimental results in Section V, where our method outperforms SF [21], a state-of-the-art CF algorithm.
2) George and Merugu [7] proposed a way to update biclusters incrementally. However, their experimental results show that this improves only execution times but results to low recommendation accuracy, even when compared with simple CF algorithms. In contrast, our method aims in advancing the accuracy of the recommendations. Moreover, differently from [7], our approach exploits partial matching between a user and his nearest biclusters.
3) The novelty of our approach compared to existing hybrid approaches is that they run CB and CF separately; thus, they miss the dependence between user ratings and item features, whereas our method exploits this dependence. This is verified by experimental results in Section V, where our method outperforms CinemaScreen [18], a state-of-the-art hybrid algorithm.
4) The novelty of our justification style is that it combines the "keyword" with influence styles [5], because we underline both the importance of features and, simultaneously, the importance of the items that contain them. Thus, our justification style combines descriptiveness, which results from the keyword style, with informativeness, which results from the influence style. Moreover, we introduce an objective metric to measure the robustness of our justifications compared with those of SF and CinemaScreen approaches.

III. PROBLEM DESCRIPTION

In this section, we define the basic notions that will be used throughout this paper and describe the examined problem, i.e., how to provide both accurate and justifiable recommendations. Table I summarizes the symbols that are used henceforth.

	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	1	0	1	1
I_7	1	0	1	0

(b)

Fig. 1. Running example: (a) User–item matrix R. (b) Item–feature matrix F.

A. Rating, Item, and Feature Profiles

CF algorithms are based on rating data. We assume that the rating data are given as a matrix R, where the rating of a user u over an item i is given from the element $R(u, i)$. An example is shown in Fig. 1(a), where I_{1-7} are items and U_{1-8} are users. The null cells (no rating) are presented with dash.

Definition 1 (Rating Profile): As rating profile $R(U_k)$ of user U_k, we define the set of rated (i.e., nonnull) items in the kth row of matrix R.

For instance, $R(U_1) = \{I_1, I_3, I_5\}$.

As described, content data are provided in the form of features. We assume that a matrix F contains the feature data, i.e., element $F(i, f)$ equals to the number of appearances of feature f in item i. For instance, in a collection of documents, element $F(i, f)$ denotes the frequency of term f in document i. In some cases, appearance frequency cannot be quantified. For example, in a collection of movies, an actor may or may not appear in the cast of a movie. Thus, in such cases, which are a specialization of the ones with quantifiable appearance frequencies, $F(i, f)$ elements are treated as Boolean. In our running example, for each item we have four features that describe its characteristics. The corresponding F matrix is shown in Fig. 1(b)

Definition 2 (Item Profile): As item profile $F(I_k)$ of item I_k, we define the set of features f for which $F(I_k, f)$ has a positive (higher than zero) value.

For instance, $F(I_2) = \{f_2, f_3\}$.

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 the items whose rating express a "positive" preference by the user. We assume a matrix P whose element $P(u, f)$ denotes the correlation between user u and feature f and is given by

$$P(u, f) = \sum_{\forall R(u,i) > P_\tau} F(i, f). \quad (1)$$

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

Fig. 2. User–feature matrix P.

In our running example, assuming that $P_\tau = 2$, the resulting P matrix is shown in Fig. 2.

Definition 3 (Feature Profile): As feature profile $P(U_k)$ of user U_k, we define the set of $P(u, f)$ elements in the kth row of matrix P with values higher than zero.For instance, $P(U_5) = \{(f_2, 1), (f_3, 2), (f_4, 2)\}$.

We have to note that the aforementioned approach treats all features equally. However, some features may be more relevant to a user than other features and can better distinguish him from other users. In the previous example, features f_3 and f_4 describe U_5 better than feature f_2 does. To identify the relevant features, we will use a feature-weighting scheme, which is detailed in Section IV-C.

B. Style of Justification

As described in Section I, our proposed justification style combines the keyword-based with influence styles. The proposed style has 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." Thus, our justification style combines the descriptiveness/specificity of the keyword-based style (by including the features $a, b, \ldots$) with the informativeness/generality of the influence style (by including the rated items $Z, W, \ldots$). Our justification style allows a user u to get separate justification for each recommended item. However, as the number of features contained in the justification may be large, they can be presented gradually (M features at a time). Therefore, it becomes crucial to get 1) a compact justification list, i.e., without too much features, and 2) a high coverage of the user's profile from the first presented features, as users may not be willing to view the entire contents of the justification list. In our experimental results, we show that the proposed method is effective, because it satisfies these two properties.

C. Explain Coverage Measure

In existing research, CF algorithms provide to a user u a top-n list $L(u) = \{I_1, \ldots, I_n\}$, where $I_1, \ldots, I_n$ are the recommended items. To measure the accuracy of a recommendation list $L(u)$, we use the well-known measures of *precision* and *recall*. Let N denote the size of $L(u)$ list and M the number of relevant recommended items (the items in $L(u)$ that are rated higher than P_τ by u). Precision is the ratio of M to N, whereas recall is the ratio of M to the total number of relevant items for the test user. Both these measures concern only the rating profile and measure the accuracy of $L(u)$. However, different users might have a different interpretation of which

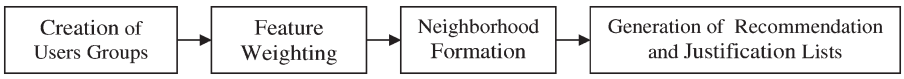

Fig. 3. Graphical representation of the approach.

item is more relevant and which is not. Recall and precision cannot distinguish between a relevant item from a more relevant item [2].

To address the problem of justifying the recommendations in L, we propose to additionally provide to user u a *justification* list $J(u)$. To measure the quality of the justification list $J(u)$, we define the measure of *explain coverage*. For a user u that receives a recommendation list L, we combine the justifications used for each recommended item in a *justification* list $J = \{(f_1, c_{f_1}), \ldots, (f_m, c_{f_m})\}$. J contains the union of features that are included in all separate justifications of items in L. Each pair (f_i, c_{f_i}) denotes that feature f_i has overall frequency c_{f_i} in L. In our running example, by assuming that $N = 2$ items are recommended, we can find that these are items I_7 and I_1. Since only feature f_1 exists in the profiles of I_7 and I_1 and U_1 (the target user), we have that $J = \{(f_1, 2)\}$.

Thus, for a user u that receives a recommendation list L, the *explain coverage* for the justification list J is defined as follows:

$$\text{Explain coverage}\,(u, J(u)) = \frac{\sum_{\forall (f_i, c_{f_i}) \in J(u)} \min\{c_{f_i}, P(u, f_i)\}}{\sum_{\forall f_i \in F} P(u, f_i)}. \quad (2)$$

The aforementioned ratio measures how much the features in $P(u)$ are covered by the features of the items included in $L(u)$. If explain coverage is high, then the justification is more effective, as the features that are included in the justification list $J(u)$ can be easily recognized and accepted by the user. Notice that, in order for explain coverage to be normalized into the [0, 1] range, in the numerator, we do not allow frequency c_{f_i} to exceed the corresponding value $P(u, f_i)$ in u's feature profile.

In our running example, assume that 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. We get the same precision and recall when we recommend I_5 and I_6. However, the explain coverage ratio is not the same. The former top-2 list results 40% coverage $((0 + 2 + 1 + 1)/(1 + 3 + 3 + 3) = 4/10)$, while the latter gives 60% $((1 + 1 + 2 + 2)/(1 + 3 + 3 + 3) = 6/10)$. Thus, an algorithm which attains more explain coverage ratio can give more justifiable recommendations.

IV. Proposed Approach

A. Outline of the Approach

Our approach is consisted of four stages, as shown in Fig. 3:

1) Creation of User Groups: To provide recommendations to a target user u, we have to find groups of users that have similar rating behavior as u. Therefore, users have to be organized into groups according to their ratings. The details of this procedure are given in Section IV-B, where we propose the simultaneous clustering (biclustering) of users and items, which discovers groups of users exhibiting highly correlated ratings on groups of items.

2) Feature Weighting: The next step is to find those features that better describe user u and distinguish him from other users. Thus, we have to weight u's feature profile. In Section IV-C, we describe a feature-weighting scheme that helps in balancing between the aforementioned two factors (intrauser similarity and interuser dissimilarity). Similarly, we weight the features contained in each found bicluster b.

3) Neighborhood Formation: In the following, we search the groups (biclusters) we have formed and find the k nearest ones to u. The neighborhood formation requires the measurement of the similarity of u to each of the biclusters. This similarity can be based on the items or on the features they have in common. In Section IV-D, we describe how to combine these two directions into a single similarity measure, in order to provide both accurate and justifiable recommendations.

4) Generation of the Recommendation and Justification Lists: As described in Section III-B, our justification style allows a user u to get explanation for each recommended item separately. Thus, a user u can receive, along with a recommendation, the reasoning behind it. In Section IV-E, we propose how to generate these two lists based on the found neighborhood.

In the remainder of this section, we elaborate on the aforementioned steps and describe the proposed method that is referred to as *Feature-Weighted Nearest Biclusters* (FWNB).

B. Creating Groups of Users

The creation of user groups is motivated by the fact that we have to find groups of users that have similar rating behavior as the target user u. Users can be organized into groups according to their ratings.[2] This kind of grouping can be performed with a clustering algorithm. Traditional clustering algorithms, however, will try to group users based on the ratings over the entire set of items they have rated. The shortcoming of this approach is that they cannot detect a partial matching of users' preferences, because two users may share similar preferences only for a subset of items. Moreover, other approaches apply separate clustering of users and items. However, these approaches are one-sided, in the sense that they separately examine similarities, either only between users or only between items. Thus, they miss the duality of data. In this paper, we apply simultaneous clustering of users and items.

The simultaneous clustering of users and items discovers *biclusters*, which correspond to groups of users exhibiting 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, a partial matching of preferences is taken into account.

[2]Notice that, besides similar rating behavior, one could group users according to similar feature profiles. However, we do not follow this approach, because it looses valuable information, as we cannot preserve the particular items that correspond to the features, in order to generate the recommendation list.

	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. 4. Applying the xMotif algorithm to matrix R.

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

Fig. 5. 4 Biclusters found in our running example.

Biclustering (a.k.a. *co-clustering, two-sided clustering*, etc.) has been used in diverse scientific fields, for instance, bioinformatics [16]. For purposes of CF, George and Merugu [7] used biclustering to build an efficient real-time system focusing in incremental updating of biclusters and scalability. Moreover, Symeonidis *et al.* [20] used a biclustering algorithm (Bimax) based on binary values to improve the accuracy of recommendations. In this paper, differently from previous works, we use a coherent biclustering algorithm [16], where biclusters are used to improve both the quality of recommendations (accuracy) and justifications (explain coverage).

For the biclustering step, we use the xMotif algorithm [16], which looks for subsets of rows and subsets of columns with coherent values, i.e., subsets of users that jointly present analogous rating behavior. Each bicluster is defined on a subset of rows and a subset of columns. Moreover, two biclusters may overlap, which means that several rows or columns of the matrix may participate in multiple biclusters. Another important characteristic is that each bicluster should be maximal, i.e., it should not be fully contained in another determined bicluster. Moreover, the xMotif algorithm permits the generation of xMotifs (biclusters) which can embody users or items in a flexible way. It allows a user to be included in a bicluster even if there exist a fraction of his rating values which cannot not be defined as interesting (ratings under the P_τ threshold or null values).

In Fig. 4, we have applied the xMotif algorithm in our running example. We found four biclusters which consist, at least, of two users and two items. Thus, the four found biclusters are shown in Fig. 5.

C. Feature Weighting

The motivation of a feature-weighting scheme is that, for a user u, we need to find those features that better describe and distinguish him from other users. Thus, the features in the user-feature matrix P have to be weighted to reveal those features that describe each user's preferences. For this reason, in the following, we describe a feature-weighing scheme that is applied over P. Similarly, for each bicluster b, based on the profiles of its items, we describe how to weight the corresponding features, so as to find those better describing the users in b.

	f_1	f_2	f_3	f_4
U_1	0.12	0	0	0
U_2	0.12	0.20	0.18	0.61
U_3	0.24	0	0.06	0
U_4	0.12	0.41	0.12	0.41
U_5	0	0.20	0.12	0.41
U_6	0.24	0	0.06	0
U_7	0	0.20	0.12	0.41
U_8	0.12	0.61	0.18	0.61

Fig. 6. Weighted user–feature matrix W.

Let $\mathcal{U}$ be the domain of all users and $\mathcal{F}_u$ the set of features that are correlated with user u, i.e., $\mathcal{F}_u = \{f \in \mathcal{F} | P(u,f) > 0\}$. Henceforth, user u and feature f are correlated when $P(u,f) > 0$. We will weight the features of matrix P, in order to find 1) those features which better describe user u (describe the $\mathcal{F}_u$ set) and 2) those features which better distinguish him from the others (distinguishing him from the remaining users in the $\mathcal{U}$ domain).

In our approach, motivated from the term frequency–inverse document frequency (TFIDF) scheme [2], we measure the frequency of each feature f for a user u. Henceforth, this frequency is referred as *feature frequency* (FF) factor. Furthermore, we measure the inverse of the frequency of a feature f among all users. Henceforth, this factor is referred as *inverse user frequency* (IUF) factor. Thus, *feature frequency* $\text{FF}(u,f)$ is the number of times feature f occurs in the profile of user u. In our model, it holds that $\text{FF}(u,f) = P(u,f)$. The *user frequency* $\text{UF}(f)$ is the number of users in which feature f occurs at least once. The *inverse user frequency* $\text{IUF}(f)$ can be calculated from $\text{UF}(f)$ as follows:

$$\text{IUF}(f) = \log \frac{|\mathcal{U}|}{\text{UF}(f)}. \tag{3}$$

In (3), $|\mathcal{U}|$ is the total number of users. The IUF of a feature is low, if it occurs in many users' profiles, whereas it is high if the feature occurs in few users' profiles. Finally, the new weighted value of feature f for user u is calculated as follows:

$$W(u,f) = \text{FF}(u,f) * \text{IUF}(f). \tag{4}$$

In our running example, the matrix P in Fig. 2 is transformed into the matrix W in Fig. 6.

Notice that feature f_1 is dominant in the feature profile of users U_3 and U_6, whereas feature f_4 is dominant in user U_2's feature profile.

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 average value of ratings for item i in a bicluster b.

Finally, similarly to the users' weighting scheme, we weight the features contained in each found bicluster b. In particular, similarly to the feature profile of users, we generate a feature profile of each bicluster from its rating profile in the R_B matrix. We define a P_B matrix whose elements $P_B(b,f)$ are given from the frequency of feature f in a bicluster b. Thus, from P_B matrix, we generate the weighted W_B matrix by applying feature-weighting in the same way used to obtain W from P.

D. Neighborhood Formation

For the formation of neighborhood, we find the k biclusters that are most similar to u. The motivation is as follows. As described in Section I, previous works [8], [9], [11], [13], [18] cannot detect the partial matching of users' preferences. This means that they measure similarity between two users, based on the entire set of items they have rated. In contrast, our approach matches target user u with the biclusters $\mathcal{B}$, considering *only* the similarity between the set of items/features that are included in b and the set of items/features rated by u. Moreover, since each bicluster acts like a community for its corresponding items, partial matching is performed with collective features that provide more robust justifications.

In particular, the set of the k most similar (nearest) biclusters forms the *neighborhood* of u, which is denoted as $N_B(u)$. The set of items contained in each bicluster $b \in N_B(u)$ are candidates for recommendation, i.e., for inclusion in the $L(u)$ list. Moreover, the features contained in b are candidates for providing justification, i.e., for inclusion in the $J(u)$ list. We have to resolve two issues: first, how to measure similarity between a user u and a bicluster b and, second, how to determine the exact items and features to be included in the $L(u)$ and $J(u)$ lists. The first issue is examined in the rest of this section, whereas the second issue is described in the next section.

To measure similarity between a user u and a bicluster b, we have two options:

1) To consider *only* the similarity between the set of items I_b that are included in b and the set of items $\mathcal{I}_u$ rated by u. In the following, we adapt the cosine similarity measure to consider the aforementioned option:

$$\text{sim}_I(u,b) = \frac{\sum_{\forall i \in \mathcal{I}_u \cap \mathcal{I}_b} R(u,i) R_B(b,i)}{\sqrt{\sum_{\forall i \in \mathcal{I}_u \cap \mathcal{I}_b} R(u,i)^2} \sqrt{\sum_{\forall i \in \mathcal{I}_u \cap \mathcal{I}_b} R_B(b,i)^2}}. \tag{5}$$

This way, u is partially matched with b by focusing only on the subset of items $I_u \cap I_b$ and ignoring the rest of the items that u has rated. Note that u can be matched against several biclusters; thus, each bicluster contributes, as candidates for recommendation, the items it specializes on.

2) To consider *only* the similarity of the set of features F_b that are included in b and the set of features $\mathcal{F}_u$ of u (these are the features of u given in the weighted matrix W). In the following, we adapt the cosine similarity to consider this option:

$$\text{sim}_F(u,b) = \frac{\sum_{\forall f \in \mathcal{F}_u \cap \mathcal{F}_b} W(u,f) W_B(b,f)}{\sqrt{\sum_{\forall f \in \mathcal{F}_u \cap \mathcal{F}_b} W(u,f)^2} \sqrt{\sum_{\forall f \in \mathcal{F}_u \cap \mathcal{F}_b} W_B(b,f)^2}}. \tag{6}$$

In our running example, considering only (5), the two nearest biclusters $(k = 2)$ of U_1 are b_1 and b_4.[3] Taking into account only (6), the two nearest biclusters of U_1 are b_3 and b_4.

[3]In this example, b_2 and b_3 have identical similarity with b_4, and we arbitrarily selected b_4.

The set of similar biclusters found with $\text{sim}_I(u,b)$ may differ from the set of those found with $\text{sim}_F(u,b)$, as also shown in the previous example. The reason is that the rating behavior of the user (which is considered by $\text{sim}_I(u,b)$) may not be fully explained by the given set of features (which are considered by $\text{sim}_F(u,b)$). The result is that, since $\text{sim}_I(u,b)$ captures the rating behavior, it is able to provide accurate recommendations, i.e., it can detect those items that will be rated positively by the target user. In contrast, $\text{sim}_F(u,b)$, since it captures the feature profile of the users, it is able to detect those items that cover the feature profile of the target user; thus, it increases the explain coverage of the justification. As our objective is to provide both accurate and justifiable recommendations, we combine the two similarity measures into a single one, which is given in

$$\text{sim}(u,b) = (1-a) \cdot \text{sim}_I(u,b) + a \cdot \text{sim}_F(u,b). \tag{7}$$

In (7), a takes values between [0, 1]. As we increase a, we demand from the system to give more justifiable recommendations, as it weights more $\text{sim}_F(u,b)$. Therefore, the final neighborhood $N_B(u)$ for a user u is formed using the $\text{sim}(u,b)$ measure. In our running example, for U_1 and $k = 2$, by setting $a = 0.5$, we have $N_B(U_1) = \langle b_1, b_4 \rangle$. The impact of a will be examined experimentally, and it will be shown that we can detect a single value for a that is suitable in most cases.

E. Generating the Recommendation and Justification Lists

To provide qualitative recommendations and justifications, it is important for an algorithm to identify those items that are highly preferred by the user and simultaneously contain features that are favorite to him. This motivates for the development of a method for the generation of recommendations and justifications with the aforementioned characteristics.

In our approach, for the target user u, the final step is to generate the recommended items and the justification for each of them. Thus, we identify the items in u's bicluster neighborhood, which 1) are both highly preferred by other users, according to their ratings in the R_B matrix, and 2) contain significant features, according to the weighted matrix W_B (we exclude items that have been already rated by u). By taking into account both factors, we sort and rerank the items and recommend the N top ones (N is user defined).

In our running example, assume that we want to recommend one item to user U_1 (thus, $N = 1$). By using only the first nearest bicluster of U_1 (i.e., $k = 1$), which is b_1, we get the features of the items in b_1. The items in b_1 are $I_{b_1} = \{I_1, I_7\}$. The item profile of I_1 is $F(I_1) = \{f_1\}$, and the item profile of I_7 is $F(I_7) = \{f_1, f_3\}$. Next, we find the frequency of the features inside the neighborhood of U_1. We get $\text{fr}(f_1) = 2$ and $\text{fr}(f_3) = 1$. Finally, for each item, we sum the frequency of its features and find its weight: $w(I_1) = 1$ and $w(I_7) = 3$. Thus, I_7 is recommended, because it has a higher weight than I_1.

The justification for each recommended item is generated by those features in the item's profile, which exist also in u's feature profile. The qualifying features are reranked according to their frequency in the u's feature profile and form the justification list for the recommended item.

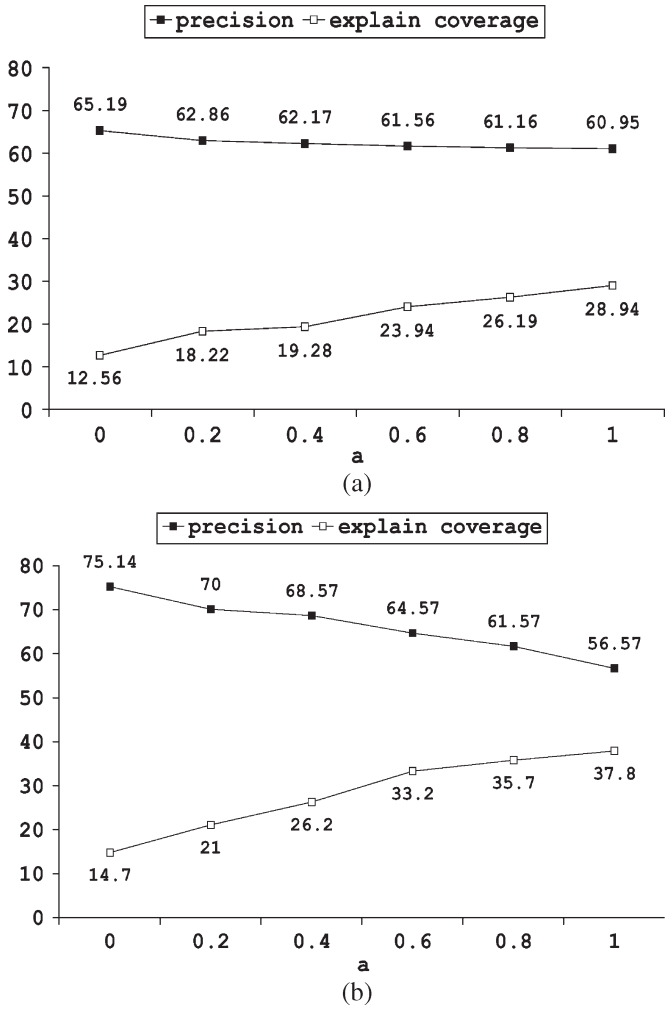

Fig. 7. Precision and explain coverage of FWNB versus a for (a) MovieLens and (b) Reuters data sets.

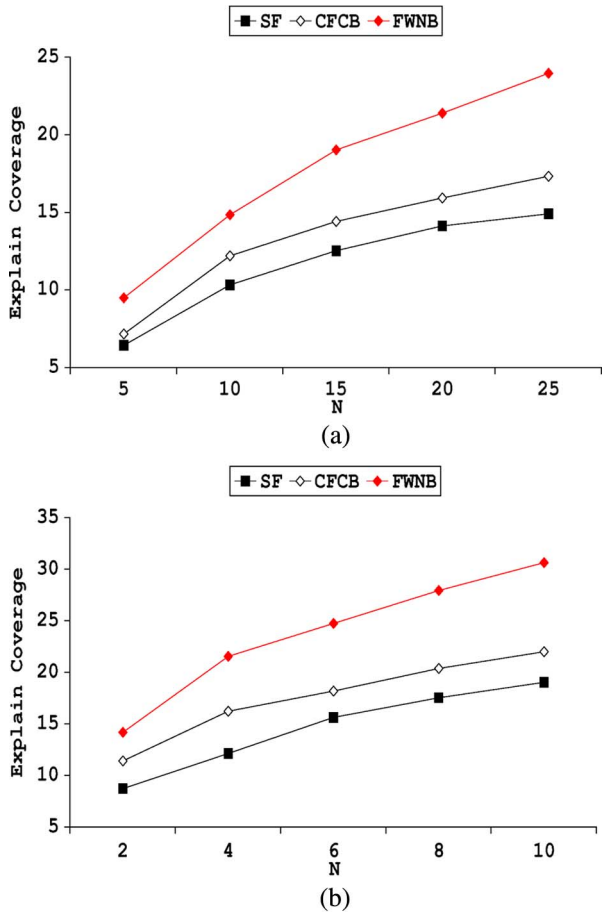

Fig. 8. Comparison between SF, CFCB, and FWNB in terms of explain coverage versus N for (a) MovieLens and (b) Reuters data sets.

In our running example, f_1 is the only feature existing both in the profile of the recommended item I_7 and the feature profile of the target user U_1. Therefore, the justification that U_1 receives for I_7 will be the following: "Item I_7 is recommended, because it contains feature f_1, which is included in item I_1 you have rated."

V. Experimental Results

In this section, we experimentally study the performance of the proposed FWNB method. For comparison purposes, in our experiments, we include a state-of-the-art CF algorithm called similarity fusion [21] and a state-of-the-art hybrid algorithm called CinemaScreen Recommender Agent [18] (henceforth denoted as CFCB). All algorithms were implemented in C++. In particular, for the generation of biclusters, we used the Biclustering Analysis Toolbox (BicAT) [4].

We perform experiments with two real data sets that have been used as benchmarks in prior work. In particular, we examined the following:

1) 100 000 MovieLens Data Set: It consists of 100 000 ratings assigned by 943 users on 1682 movies [17]. The range of ratings is between 1 (bad) and 5 (excellent). The extraction of the content features has been done by joining with the contents of the Internet movie database[4] and selecting four different classes of features (genres, actors, directors, and keywords). The join process yielded 23 different genres, 9847 keywords, 1050 directors, and 2640 different actors and actresses (we selected only the three most paid actors or actresses for each movie).

2) Reuters—21 578 Text Data Set: It is a collection of news articles[5] [12]. Each article has been assigned by humans to a set of categories, which are further analyzed to subcategories. We concentrated our study on a subset of the first 1000 articles by eliminating news categories with less than five articles. Preprocessing involved the removal of stop words, stemming, and TFIDF. The final result consists of 69 categories covering 1000 articles and a total number of 2894 terms. For the purposes of CF, we assume an analogy between news categories and users, and between articles and items. Moreover, the terms of each article comprises its features.

The metrics we use are recall, precision, and explain coverage. The default size of the recommendation list N is set to 20 for the MovieLens set and 10 for the Reuters data set. Respectively, P_τ is set to 3 for the MovieLens set and to 1 for

[4]Downloaded from ftp.fu-berlin.de, October 2006.
[5]Downloaded from http://www.daviddlewis.com.

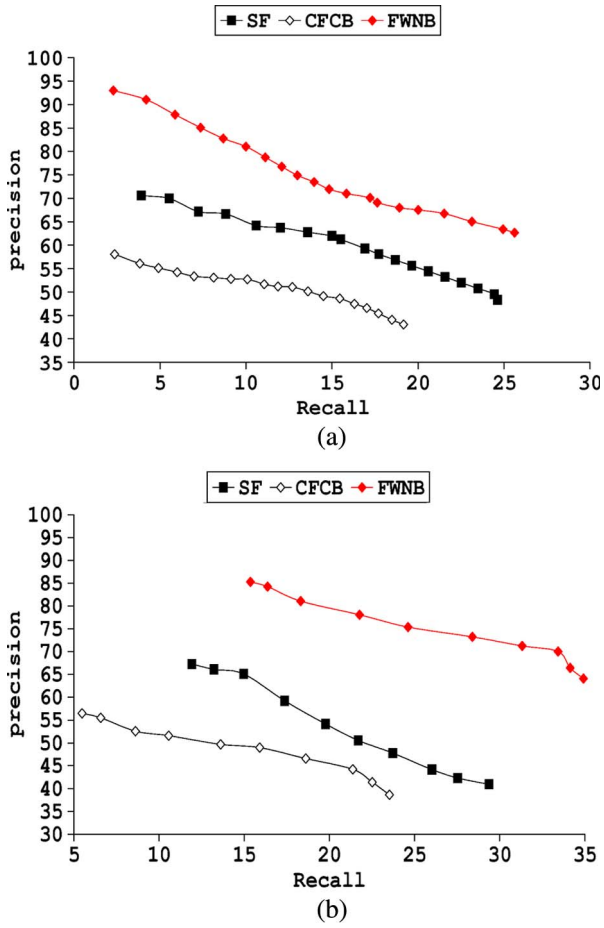

Fig. 9. Comparison between IB, CFCB, and FWNB in terms of precision versus recall for (a) MovieLens and (b) Reuters data sets.

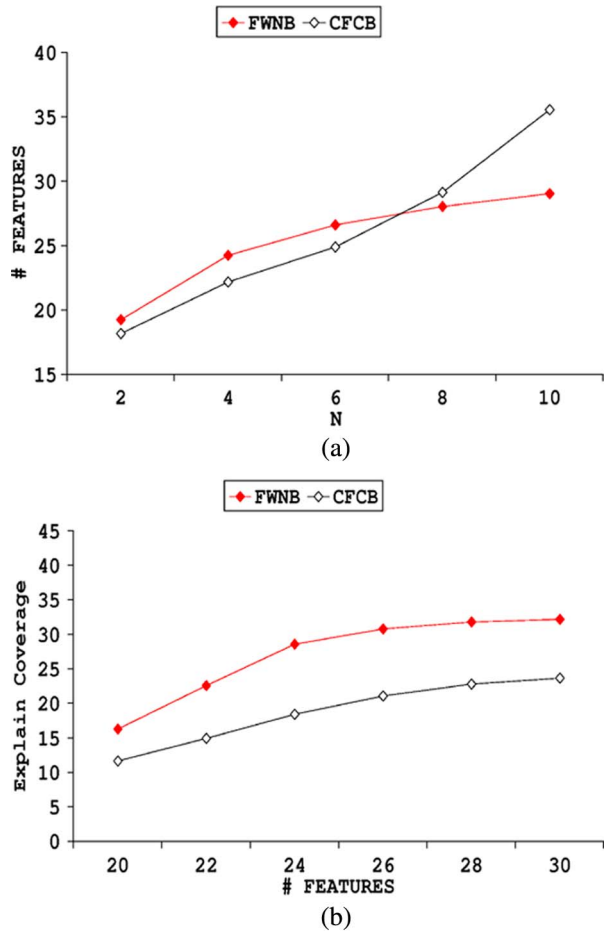

Fig. 10. Comparison between (a) CFCB and FWNB in terms of number of features in justification list versus N and (b) CFCB and FWNB in terms of explain coverage versus number of features in justification list.

the Reuters data set (1 means that an article belongs to a news category, while 0 means does not).

For the Reuters data set, we recommend articles for a given news category. We selected a random category, called "Coffee" and recommended two articles for this category. Table III presents the two recommended articles and their justifications.

A. Performance of the Similarity Measure

In Section IV-D, we examine the performance of the proposed similarity measure, and we verify its property to combine high accuracy and high explain coverage. As the balance between these two factors is attained through parameter a, we present results for precision and explain coverage versus a.

The results for the MovieLens and Reuters data sets are shown in Fig. 7(a) and (b), respectively. In both data sets, as expected, explain coverage increases with increasing a. The reason is that a is the weight we place on the sim_F factor [see (7)]; thus, with higher a values, we try to maximize explain coverage. Conversely, precision decreases when a increases. This decrease is less steep for the MovieLens data set, because in this data set, there is no deviation between the feature profiles of the users and their rating behavior. For the Reuters data set, we observe that, after a point, precision is significantly reduced. However, for a range of a values, both precision and explain coverage are sufficient. In the rest of the experiments, the default value for a is set to 0.5 to combine equally accuracy and explain coverage.

B. Measuring Explain Coverage and Precision–Recall

We examined all methods (FWNB, SF, and CFCB) against their explain coverage. For the MovieLens data set, the resulting explain coverage versus the size N of the recommendation list is shown in Fig. 8(a). For all methods, explain coverage increases with increasing N, because more recommended items (resulting from higher N values) are able to cover more the profile of the target users. However, FWNB outperforms CFCB and SF, because its similarity measure takes into account the partial matching between a user and biclusters. Notice that SF has the smaller explain coverage, because it is not hybrid, and thus, it does not exploit item features. Analogous results are obtained for the Reuters data set, as shown in Fig. 8(b).

Next, we compare FWNB, SF, and CFCB in terms of precision and recall. For the MovieLens data set, Fig. 9(a) shows the precision–recall diagram for all algorithms. FWNB attains the best precision in all cases. The reason is that the similarity measure of FWNB is based on nearest biclusters and, thus, being able to detect partial matching of users' preferences,

TABLE II
JUSTIFICATION EXAMPLE FOR THE MOVIELENS DATA SET

Recommended Movie title	The reason of recommendation 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 III
JUSTIFICATION EXAMPLE FOR THE REUTERS DATA SET (ICO STANDS FOR INTERNATIONAL COFFEE ORGANIZATION)

Recommended Article title	The reason of recommendation is the terms	because category "Coffee" contains
ICO Council ends in failure to agree quotas	Brazil, ICO, coffee, quotas	3 articles with these terms
Coffee traders expect sell off after ICO talks fail	ICO, coffee, dollars	2 articles with these terms

can provide accurate recommendations. Analogous results are obtained for the Reuters data set, as shown in Fig. 9(b).

C. Measuring Effectiveness

In Section III-B, we mentioned two properties that are required for the justification list $J(u)$: 1) to contain a small total number of features and 2) to get high partial coverage with a small number of features. We examined the effectiveness of FWNB in terms of these two properties and compare it against CFCB. In Fig. 10(b), we count the resulting number of features in $J(u)$ versus N (size of the recommendation list). For FWNB, the size of $J(u)$ starts to increase but it is early stabilized to a small number of features. In contrast, for CFCB, the size of $J(u)$ increases steadily. Thus, FWNB produces more compact justification lists and fulfills property 1). This ability of FWNB is because it extracts collective features from biclusters, whereas CFCB extracts features from individual users. In Fig. 10(c), we test the explain coverage of FWNB and CFCB versus the increasing number of features used in $J(u)$. FWNB is better than CFCB in all cases. FWNB can attain high partial coverage with a smaller number of features; thus, it fulfills property 2).

D. Measuring Justifications' Quality

As described in Section III-B, our justification style combines the keyword-based and 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 two real data sets, the MovieLens and Reuters.

For the MovieLens data set, we selected a user at random (among the 943 users of the set) and recommended two movies. Table II depicts these recommended movies along with their justifications. Notice that the second column of Table II concerns the keyword-based explanation style, whereas the third column of Table II concerns the influence explanation style. The combination of those two columns is our proposed justification style.

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 postgraduate students of Aristotle University, who filled out an online 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.

TABLE IV
RESULTS OF THE USER SURVEY

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

In Table IV, we present the mean μ_p and standard deviation σ_p of the 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 ran 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.

VI. CONCLUSION

We propose an approach to attain both accurate and justifiable recommendations. We perform an experimental comparison of our method against the well-known CF and a hybrid algorithm with two real data sets. Our approach builds a feature profile for the users, which 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 *explain 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.

Summarizing the aforementioned conclusions, our approach brings to surface the need for justification of recommendations. In this direction, content data can be very useful. In our future work, we will consider new ways of exploiting them. Moreover, we will consider other objective metrics to measure the quality of justifications. Finally, we aim to build an Internet-based application that will provide justifications to support movie recommendations based on the justification style developed in this work.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[2] R. A. Baeza-Yates and B. A. Ribeiro-Neto, *Modern Information Retrieval*. Reading, MA: Addison-Wesley, 1999.

[3] M. Balabanovic and S. Y. Fab, "Content-based, collaborative recommendation," *Commun. ACM*, vol. 40, no. 3, pp. 66–72, 1997.

[4] S. Barkow, S. Bleuler, A. Prelic, P. Zimmermann, and E. Zitzler, "BicAT: A biclustering analysis toolbox," *Bioinformatics*, vol. 22, no. 10, pp. 1282–1283, 2006.

[5] M. Bilgic and R. Mooney, "Explaining recommendations: Satisfaction vs. promotion," in *Proc. Beyond Personalization, Workshop Next Stage Recommender Syst. Res., IUI Conf.*, 2005, pp. 13–18.

[6] D. Billsus and M. Pazzani, "A personal news agent that talks, learns and explains," in *Proc. Auton. Agents Conf.*, 1999, pp. 268–275.

[7] T. George and S. Merugu, "A scalable collaborative filtering framework based on co-clustering," in *Proc. IEEE ICDM Conf.*, 2005, pp. 625–628.

[8] D. Goldberg, D. Nichols, M. Brian, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Commun. ACM*, vol. 35, no. 12, pp. 61–70, 1992.

[9] J. Herlocker, J. Konstan, and J. Riedl, "Explaining collaborative filtering recommendations," in *Proc. Comput. Supported Cooperative Work*, 2000, pp. 241–250.

[10] R. Jin, L. Si, and C. Zhai, "A study of mixture models for collaborative filtering," *Inf. Retr.*, vol. 9, no. 3, pp. 357–382, 2006.

[11] X. Jin, Y. Zhou, and B. Mobasher, "A maximum entropy web recommendation system: Combining collaborative and content features," in *Proc. ACM SIGKDD Conf.*, 2005, pp. 612–617.

[12] D. D. Lewis and P. J. Hayes, "Guest editorial," *ACM Trans. Inf. Syst.*, vol. 12, no. 3, p. 231, 1994.

[13] P. Melville, R. J. Mooney, and R. Nagarajan, "Content-boosted collaborative filtering for improved recommendations," in *Proc. AAAI Conf.*, 2002, pp. 187–192.

[14] B. Mobasher, R. Burke, R. Bhaumic, and C. Williams, "Towards trustworthy recommender systems: An analysis of attack models and algorithm robustness," *ACM Trans. Internet Technol.*, vol. 7, no. 2, pp. 23–60, 2007.

[15] R. Mooney and L. Roy, "Content-based book recommending using learning for text categorization," in *Proc. ACM DL Conf.*, 2000, pp. 195–204.

[16] T. Murali and S. Kasif, "Extracting conserved gene expression motifs from gene expression data," in *Proc. Pacific Symp. Biocomputing Conf.*, 2003, vol. 8, pp. 77–88.

[17] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: An open architecture for collaborative filtering on netnews," in *Proc. Comput. Supported Collaborative Work Conf.*, 1994, pp. 175–186.

[18] J. Salter and N. Antonopoulos, "CinemaScreen recommender agent: Combining collaborative and content-based filtering," *IEEE Intell. Syst.*, vol. 21, no. 1, pp. 35–41, Jan./Feb. 2006.

[19] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proc. WWW Conf.*, 2001, pp. 285–295.

[20] P. Symeonidis, A. Nanopoulos, A. Papadopoulos, and Y. Manolopoulos, "Nearest-biclusters collaborative filtering," in *Proc. 12th WebKDD*, Philadelphia, PA, 2006, pp. 21–29.

[21] J. Wang, A. Vries, and M. Reinders, "Unifying user-based and item-based collaborative filtering approaches by similarity fusion," in *Proc. SIGIR Conf.*, 2006, pp. 501–508.

[22] G. Xue, C. Lin, and Q. Yang, "Scalable collaborative filtering using cluster-based smoothing," in *Proc. ACM SIGIR Conf.*, 2005, pp. 114–121.

Panagiotis Symeonidis received the B.S. degree in applied informatics and the M.Sc. degree in information systems from Macedonia University, Thessaloniki, Greece, in 1996 and 2004, respectively, and the Ph.D. degree in web mining from the Aristotle University of Thessaloniki, Thessaloniki, in 2008.

Currently, he is a Postdoc Researcher with the Department of Informatics, Aristotle University of Thessaloniki. He also teaches informatics in a secondary education level school in Greece. His research interests include web mining, information retrieval, recommender systems, and social tagging systems.

Alexandros Nanopoulos was born in Craiova, Romania, in 1974. He received the B.S. and the Ph.D. degrees from the Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece, in 1996 and 2003, respectively.

Currently, he is a Lecturer with the Department of Informatics, Aristotle University of Thessaloniki. He is a coauthor of more than 60 articles in international journals and conferences. He has also a coauthor of the monographs *Advanced Signature Techniques for Multimedia and Web Applications* and *R-trees: Theory and Applications*. His research interests include data mining, web information retrieval, and spatial database indexing.

Yannis Manolopoulos was born in Thessaloniki, Greece, in 1957. He received the B.Eng. degree in electrical engineering and the Ph.D. degrees in computer engineering from the Aristotle University of Thessaloniki, Thessaloniki, in 1981 and 1986, respectively.

Currently, he is a Professor with the Department of Informatics, Aristotle University of Thessaloniki. He was with the Department of Computer Science, University of Toronto, Toronto, ON, Canada, the Department of Computer Science, University of Maryland, College Park, and the Department of Computer Science, University of Cyprus, Nicosia, Cyprus. He served as the General/PC Chair/Cochair of the 8th National Computer Conference (2001), the 6th Advances in Databases and Information Systems (ADBIS) Conference (2002), the 5th Workshop on Distributed Data and Structures (2003), the 8th Symposium on Spatial and Temporal Databases (2003), the 1st Balkan Conference in Informatics (2003), the 16th Scientific and Statistical Database Management Conference (2004), the 8th International Conference on Enterprise Information Systems (2006), and the 10th ADBIS Conference (2006). He is a coauthor of the following books: *Advanced Database Indexing* and *Advanced Signature Indexing for Multimedia and Web Applications* (Kluwer), as well as *Nearest Neighbor Search: a Database Perspective* and *R-trees: Theory and Applications* (Springer). He has published about 200 papers in refereed scientific journals and conference proceedings. His published work has received over 1700 citations from over 450 institutional groups. His research interests include databases, data mining, web and geographical information systems, bibliometrics/webometrics, and performance evaluation of storage subsystems.