Collaborative Recommendations Using Bayesian Networks and Linguistic Modelling

Luis M. de Campos, Juan M. Fernández-Luna, and Juan F. Huete

Department of Computer Science and Artificial Intelligence
Technical School of Computer Science, University of Granada, 18071–Granada, Spain
{lci,jmfluna,jhg}@decsai.ugr.es

Abstract. This paper presents a model designed under the formalism of Bayesian Networks to deal with the problem of collaborative recommendation. It has been designed to perform efficient and effective recommendations. We also consider the fact that the user can usually use vague ratings for the products, which might be represented as fuzzy labels. The complete proposal is evaluated with MovieLens.

1 Introduction

Over the last ten years, there has been a massive increase in the amount of information available on Internet, and it is often hard for users to access this information, possibly because they are unaware it exists. This offers an attractive framework for research into new, accurate and efficient techniques for accessing this information. In this framework, *Recommender Systems (RS)* have emerged to help people deal with this overload of information. Broadly speaking, an RS provides specific suggestions about items (or actions) within a given domain, which may be considered of interest to the user [1]. There are several types of RS, classified according to the information used when recommending. This paper focuses on the variant called *collaborative filtering*, which attempts to identify groups of people with similar tastes to the user's and to recommend items that they have liked. In this case, the objective is usually to predict the utility of an unseen item for an active user based on items previously rated by other users.

In this paper, we propose to model an RS by using the *Bayesian network (BN)* [2] formalism. Given an unobserved item, we will therefore be able to obtain the most probable vote that a user would give to that item. It must be pointed out that fuzziness exists in the rating process. Rating an item usually implies the selection of a vote from a set of labels. However, there is often no meaningful way to set the boundary between two consecutive labels. For this reason, we shall also explore the advantages of considering the rating alternatives as vague concepts. Fuzzy sets formalize the idea of graded membership of an element to a given set. Although users might think in terms of vague concepts when rating, it is quite common for an RS to eventually store their ratings using crisp values with the consequent loss of information. What we offer in this paper, therefore, is a linguistic modelling in the system output so that the crisp votes generated by the BNs are converted into fuzzy labels which are closer to the users.

L. Rutkowski et al. (Eds.): ICAISC 2008, LNAI 5097, pp. 1185–1197, 2008.

Our approach for modelling RSs therefore involves processing two different types of uncertainty: probability arising from a lack of knowledge of how the different users are related, and fuzziness concerned with the ambiguity or vagueness in the description of the ratings. We shall show how the combination of both theories leads to an improvement when modelling a collaborative RS.

In order to describe our model, this paper is organised into the following parts: Section 2 will briefly review Bayesian networks and Recommender Systems; Section 3 will describe the model, explaining its topology and how it is learnt, the probability estimation, the inference mechanism and finally, how the linguistic modelling is carried out; Section 4 will show the experiments and results; and finally, Section 5 will discuss the main conclusions and further work.

2 Background

2.1 Bayesian Networks

In recent decades, Bayesian networks [2] have become one of the most consolidated methodologies for probabilistic inference. These graphical models are capable of efficiently representing and manipulating n-dimensional probability distributions by combining a qualitative and a quantitative representation of the problem by means of, on the one hand, a *directed acyclic graph* (DAG), $G = (V, E)$, where the nodes in V represent the random variables from the problem we want to solve, and the topology of the graph (the arcs in E) encodes dependence relationships between the variables, with the absence of an arc between any two nodes representing an independence relationship between the variables; and, on the other, a set of conditional probability distributions drawn from the graph structure: for each variable $X_i \in V$ we have a family of conditional probability distributions $Pr(X_i \mid pa(X_i))$, where $pa(X_i)$ represents any combination of the values in the parent set of X_i in G, $Pa(X_i)$.

Once the BN is completed, it specifies a complete joint probability distribution over all the variables, i.e. given a configuration $c = (x_1, x_2, \ldots, x_n)$ over the set of variables $X_1, \ldots, X_n$, with x_i being the value that variable X_i takes in c, then $Pr(c) = \prod_{i=1}^{n} Pr(x_i|pa(x_i))$, where $pa(x_i)$ are the values taken by the parent set of X_i in c. This decomposition of the joint distribution results in important savings in storage requirements.

In a probabilistic context, the user usually has some evidence of the state that a variable (or a set of variables) takes. The problem is to compute the posterior probability distribution for a variable given the evidence, $Pr(X_i|ev)$ [2].

2.2 Recommender Systems

The usual formulation of the recommendation problem is to predict the vote or rating that an active user would give to an unseen item. This estimation can be used to recommend those items with the highest estimated ratings to the user. RSs are usually classified into the three main categories based on how

the recommendations are made. The first type are *content-based RSs*, which store content information about each item to be recommended. This information will be used to recommend similar items to those favoured by the user in the past, based on how similar certain items are to each other or the similarity with respect to user preferences. The second kind are *collaborative filtering RSs*, which attempt to identify groups of people with similar tastes to those of the user and recommend items that they have liked (predicting the vote for a given user depends on the votes of people with similar tastes or preferences). In order to do so, they use some kind of aggregation measure considering the ratings of other (most similar) users for the same item. Alternatively, predictions may be made by building (offline) an explicit model of the relationships between items. This model is then used (on-line) to finally recommend the product to the users. In this approach, the predictions are not therefore based on any ad hoc heuristic but rather on a model learnt from the underlying data using statistical and machine learning techniques. Finally, *Hybrid RSs* combine both previous approaches.

Considering collaborative RSs, we can distinguish between two approaches. The first approach uses BN learning algorithms to learn a full joint probability distribution about items and then uses this distribution to make on-line predictions [3,4]. BN-based classifiers [5,6,7] have also been applied. The second approach builds several conditional models and predicts the likelihood of an individual item given a combination of the observed votes for other users [8].

3 The Collaborative Bayesian Network-Based Model

We shall consider a large number m of items $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$, a large set of n users, $\mathcal{U} = \{U_1, U_2, \ldots, U_n\}$. For each user, a set of ratings about the quality of certain observed items in $\mathcal{I}$. The user's ratings (preferences) are values in the set $\mathcal{S}$. The set of observed data can then be viewed as a very sparse $n \times m$ matrix, $\mathbf{R}$ (users only rate a very small proportion of items). In the matrix, $\mathbf{R}[a][j]$ represents the rate of user U_a for the item I_j and will also be denoted as $s_{a,j}$, assuming 0 when the item has not been rated by the user.

We are interested in representing the relation $\mathcal{I} \longrightarrow \mathcal{U}$ in a BN, modelling the database of user votes for the set of observed items, as well as the relation $\mathcal{U} \longrightarrow \mathcal{U}$, modelling the relationships between users. We shall therefore consider the set of items $\mathcal{I}$ and users $\mathcal{U}$ as variables in the BN (nodes in the graph).

It is clear that the voting pattern of each user (U_a) will depend directly on the vote given to each observed item. We shall include an arc from each item, I_j, voted by user U_a to the node representing that user. Each item $I_j \in \mathcal{I}$ will have an associated random binary variable, taking values from the sets $\{i_{j,0}, i_{j,1}\}$ (not relevant, relevant, respectively, to the user's interest). In the case of user variables, $U_a \in \{u_{a,1} \ldots, u_{a,r}\} \cup \{u_{a,0}\}$ (a new state $(u_{a,0})$ is added to represent the fact that the user has no interest in voting).

Our model might be able to represent relations between users, $\mathcal{U} \longrightarrow \mathcal{U}$. These should be modeled in the BN by the inclusion of arcs between any two similar users. As the similarities between two users tend to be symmetric, we

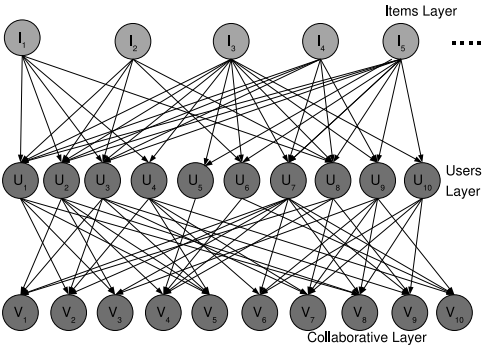

Fig. 1. Collaborative Recommending System Topology

would be including a cycle in the BN, something which is forbidden. To solve this problem, we propose that a new set of nodes $\mathcal{V}$ be considered to denote collaborative votes. There is therefore one collaborative node for each user in the system, i.e. $\mathcal{V} = \{V_1, V_2, \ldots, V_n\}$, which will also be used to predict the vote that the active user could give to an unseen item and they will therefore take their values in the set of valid rating labels, i.e. $\{v_{a,1}, v_{a,2}, \ldots, v_{a,r}\}$, omitting $v_{a,0}$ as an alternative state (see Figure 1).

3.1 Learning User Relationships

We shall now describe how the selection of similar users for a given one, V_a, is performed from the database of votes, in order to form its set of parents in the graph, $Pa(V_a)$ (those user variables, $U_b \in \mathcal{U}$, with U_a and U_b having the greatest similarity between their tastes). Thus, given a similarity measure, $Pa(V_a)$ is obtained using a threshold or p variables with the highest similarity.

A first measure is the *Cosine Measure* [9], based on the computation of the cosine of the angle formed by two vectors (any two rows of matrix $\mathbf{R}$). In the range $[0, 1]$, the greater the similarity between the vectors, the greater the cosine:

$$Cosine(U_a, U_b) = \frac{\sum_{I_j \in \mathcal{I}} r_{a,j} \cdot r_{b,j}}{\sqrt{\sum_{I_j \in \mathcal{I}} r_{a,j}^2} \sqrt{\sum_{I_j \in \mathcal{I}} r_{b,j}^2}} \tag{1}$$

A second alternative is *Pearson's Correlation Coefficient*, which determines whether there is a linear relationship between two variables (domain $[-1, 1]$). A 0 valoue means that there is absolutely no correlation; 1 means that there is an exact and positive correlation, and -1 that the correlation is exact but negative.

$$Pearson(U_a, U_b) = \frac{\sum_{I_j \in P(U_a) \cap P(U_b)} (r_{a,j} - \overline{r}_a)(r_{b,j} - \overline{r}_b)}{\sqrt{\sum_{I_j \in P(U_a) \cap P(U_b)} (r_{a,j} - \overline{r}_a)^2 \sum_{I_j \in P(U_a) \cap P(U_b)} (r_{b,j} - \overline{r}_b)^2}} \tag{2}$$

In this case, the summations over I_j are over those items for which both users U_a and U_b have recorded votes. In addition, $\overline{r}_a$ is the mean vote for user U_a.

A common problem due to the sparsity of the data set R arises: let us consider two users rating a common item. Then, $Pearson(U_a, U_b) = 1$, so U_b will be set as the parent of V_a and also U_a is a parent of V_b, resulting in low quality parent sets. In order to avoid this situation, we also propose taking into account the number of items that both U_a and U_b rated simultaneously, i.e. their overlap degree. The criterion can therefore be implemented with two alternatives [1]:

The *Jaccard Coefficient* [9] measures the overlap degree between two sets by dividing the numbers of items observed by both users (intersection) and the number of different items from both sets of rated products (union):

$$Jaccard(U_a, U_b) = \frac{\mid \{I_j \in \mathcal{I}/r_{a,j} \neq 0\} \cap \{I_j \in \mathcal{I}/r_{b,j} \neq 0\} \mid}{\mid \{I_j \in \mathcal{I}/r_{a,j} \neq 0\} \cup \{I_j \in \mathcal{I}/r_{b,j} \neq 0\} \mid} \tag{3}$$

A second choice is the *Overlap Coefficient*, which substitutes the denominator of Equation 3 for the number of products rated by one of the two users:

$$Overlap(U_a, U_b) = \frac{\mid \{I_j \in \mathcal{I}/r_{a,j} \neq 0\} \cap \{I_j \in \mathcal{I}/r_{b,j} \neq 0\} \mid}{\mid \{I_j \in \mathcal{I}/r_{a,j} \neq 0\}}. \tag{4}$$

The final similarity measures that we propose are combinations of both criteria: vote correlation between common items and the overlap degree, i.e.

$$\begin{aligned} PearsonJC(U_a, U_b) &= abs(Pearson(U_a, U_b)) \times Jaccard(U_a, U_b), \\ PearsonOC(U_a, U_b) &= abs(Pearson(U_a, U_b)) \times Overlap(U_a, U_b). \end{aligned} \tag{5}$$

where *abs* denotes the absolute value, as we take into account those users with a high positive correlation value (very similar tastes), and those with very low positive correlations (same items but totally opposite votes).

3.2 Probability Estimation

In order to complete the model's specification, the numerical values for the conditional probabilities must be estimated from the data sets, but prior to this, we shall introduce some notation. $x_{i,j}$ denotes the fact that variable X_i takes the j^{th}-value. We write $Pr(x_{i,j}|pa(X_i))$ for $Pr(X_i = x_{i,j}|pa(X_i))$, with $pa(X_i)$ denoting a configuration of the parent set of X_i, $Pa(X_i)$.

With respect to the set of items $\mathcal{I}$, since they are root nodes in the graph, they store marginal probability distributions which are linear in size to the number of states. Variables $\mathcal{U}$ and $\mathcal{V}$ must store a set of conditional probability distributions with an exponential size to the number of parents. Since a user can rate a large number of items and a collaborative node might be related to a large number of users, assessing and storing these probability values can be quite complex. We therefore propose a weighted-sum canonical model to represent these probabilities, enabling us to design a very efficient inference procedure.

[1] U_a is the user for which we are learning its set of parents.

Thus, for a given X_i, we define $Pr(x_{i,j}|pa(X_i)) = \sum_{Y_k \in Pa(X_i)} w(y_{k,l}, x_{i,j})$, where $w(y_{k,l}, x_{i,j})$ is a weight measuring how this l^{th} value of variable Y_k describes the j^{th} state of X_i. For every item, I_j, a priori probability distributions are estimated: $Pr(i_{j,1}) = \alpha$ and $Pr(i_{j,0}) = 1 - \alpha, \forall I_j \in \mathcal{I}$, α being a constant.

For every user node U_k, we need to assess a set of conditional probability distributions, one for each possible configuration of the set of items rated by U_k, representing its rating pattern. Considering the above restrictions, these will be computed using a canonical model: assuming that the user U_k rated an item I_j with the label l, their weights could be defined by means of: $w(i_{j,0}, u_{k,s}) = 0, \forall s \neq 0$; $w(i_{j,0}, u_{k,0}) = 1/|Pa(U_k)|$; $w(i_{j,1}, u_{k,s}) = \frac{\phi(s|l)}{|Pa(U_k)|}, \forall s \neq 0$, $w(i_{j,1}, u_{k,0}) = 0$.

Focusing on collaborative nodes $\mathcal{V}$, for each node V_a we must compute those weights $w(u_{b,\bullet}, v_{a,\bullet})$ given by users U_b with similar tastes, i.e. $U_b \in Pa(V_a)$. On one side, and in view of the fact that user ratings are related statistically, these weights might be considered to depend on the frequency that user U_a votes with value s given that user U_b has the state t, i.e. $freq(u_{a,s}|u_{b,t})$, and on the other, considering that the highest weights are assigned to the most similar users, it seems natural that these weights will also depend on the similarity degree between users. The way the weight associated to user U_b is distributed is therefore defined by means of the following equation:

$$w(u_{b,t}, v_{a,s}) = \frac{freq(u_{a,s}|u_{b,t}) \times sim(U_a, U_b)}{|Pa(V_a)| \sum_{U_b \in Pa(V_a)} sim(U_a, U_b)}, \text{ with } s \in \mathcal{S}, t \in \mathcal{S} \cup \{0\} \quad (6)$$

With respect to the estimation of $freq(u_{a,s}|u_{b,t})$: if user U_b has no interest in voting ($U_b = u_{b,0}$) then $Pr(u_{a,s}|u_{b,0}) = 1/r$, for all $s \in \mathcal{S}$ (the weight associated with the 'no interest in voting' situation will be distributed uniformly among the different candidate rates at collaborative nodes). If user U_b voted, then $freq(u_{a,s}|u_{b,t})$ with $t \neq 0$ can be estimated by means of:

$$freq(u_{a,s}|u_{b,t}) = \frac{N^*(u_{b,t}, v_{a,s}) + \beta q_s}{N^*(u_{b,t}) + \beta}, 1 \leq t, s \leq \ r.$$

$N^*(u_{b,t}, v_{a,s})$ is the number of items from $I(U_a) \cap I(U_b)$ that have been rated with value t by user U_b and with s by user U_a. $N^*(u_{b,t})$ is the number of items in $I(U_a) \cap I(U_b)$ voted with t by user U_b. Values β and q_s are the parameters of a Dirichlet prior over user ratings with $\sum_{i=1}^{r} q_i = 1$.

3.3 Recommending: Inference in the Bayesian Network

In order to predict the satisfaction degree that a user would give to a new item acting as evidence we shall compute the a posteriori probability distribution for the collaborative node V_a, $Pr(V_a = s|ev)$ for all $s \in \mathcal{S}$. Although general purpose algorithms do exist, they take exponential time with the number of parents when applied to a BN with the proposed topology [2]. Nevertheless, considering that the evidence only affects user nodes and the conditional independence statements represented in the network, the a posteriori probabilities for the collaborative

nodes can be computed efficiently by using the advantages of the canonical weighted-sum representation in Section 3.2.

Theorem 1: *Let l_{X_a} denote the number of states that X_a takes in the collaborative BN network and let Y_j be a node in $Pa(X_a)$. Let us assume that the set of conditional probability distributions over X_a are expressed using a canonical weighting scheme, i.e. $Pr(x_{a,s}|pa(X_a)) = \sum_{Y_j \in Pa(X_a)} w(y_{j,t}, x_{a,s})$, where $y_{j,t}$ is the value that variable Y_j takes in the configuration $pa(X_a)$ and $w(\cdot,\cdot)$ are a set of non-negative weights verifying that $\sum_{s=1}^{l_{X_a}} \sum_{Y_j \in Pa(X_a)} w(y_{j,t}, x_{a,s}) = 1, \forall\ pa(X_a)$. If the evidence, ev, is only on ancestors of X_a, the exact a posteriori probabilities can then be computed with the following formula:*

$$Pr(x_{a,s}|ev) = \sum_{Y_j \in Pa(X_a)} \sum_{t=1}^{l_{Y_j}} w(y_{j,t}, x_{a,s}) \cdot Pr(y_{j,t}|ev).$$

Propagation would therefore comprise two steps: computation of posterior probability in those user nodes, U_a, affected by the evidence, and with this information, the computation of the a posteriori distributions in the corresponding nodes, V_a. The next step would be vote selection, with two alternatives: the most probable vote, *MaxPostP* ($r_{V_a} = arg\max_l Pr(V_{a,l}|ev)$, and the vote with the largest difference between the a priori and a posteriori distributions, *MaxDifPostPPriP* ($r_{V_a} = arg\max_l(Pr(V_{a,l}|ev) - Pr(V_{a,l}))$. This would involve rewarding those states with the greatest increase in probability.

3.4 Vote Modelling

We assume that it is more difficult for a user to rate a product with an exact value from the set $\mathcal{S}$ of possible alternatives than to say 'this item is good'. Additionally, in this case he/she is not ruling out the fact that the item could be 'bad' or 'excellent', the previous and subsequent grades on the scale. Since the judgements are not usually strict and have a certain degree of flexibility, it seems appropriate to use the fuzzy-set formalism to describe the degree of satisfaction of a user evaluating an item. Following Zadeh's [10] definition, a fuzzy set A of a reference set Ω is identified by its membership function, $\mu_A : \Omega \longrightarrow [0, 1]$, where $\mu_A(x)$ is the membership degree of element $x \in A$, $\forall x \in \Omega$.

User ratings will then be considered as fuzzy observations from $\mathcal{S}$, i.e. each particular vote could be seen as a fuzzy set from $\mathcal{S}$, i.e. $\mathcal{S}_L = \{l_1, l_2, \ldots, l_r\}$[2].

As mentioned before, we are focusing on linguistic modelling at the output, i.e. instead of offering a value from $\mathcal{S}$ as the prediction, it would return a fuzzy label in $\mathcal{S}_L$ which is closer to the user's interest. Having computed $Pr(V_a|ev)$, we would then select the fuzzy label $l \in \mathcal{S}_L$ that best predicts the user's vote.

One alternative for carrying out this selection is to compute the a posteriori probability for each fuzzy event, $l \in \mathcal{S}_L$, and then return the most probable fuzzy

[2] For instance, with the MovieLens collection, $\mathcal{S}_L$ = { *1 = Awful, 2 = Fairly Bad, 3 = OK, 4 = Enjoyable, 5 = Must see* }, while $\mathcal{S} = \{1, 2, 3, 4, 5\}$.

set (*ProbFL*): vote $= \arg\max_l\{\sum_{s=1}^{r} \mu_l(s) Pr(V_a = s|ev)\}$. A second alternative is to use a similarity measure between the a posteriori probabilities and the fuzzy labels, giving as output the fuzzy label which is most similar to the a posteriori values. A direct similarity measure cannot be applied since we are not only talking about fuzzy labels and probabilities. It is therefore necessary to make transformations so as to allow both fuzzy labels and probability values to be compared with the same language: Possibility Theory [11].

We shall use $\Pi_{ev}(V_a)$ to denote the possibility distribution over the ratings obtained after transforming the a posteriori probability distribution $Pr(V_a|ev)$ and $\Pi_l(V_a)$ to denote the possibility distribution representing the fuzzy label l. We can then use a similarity measure between them in order to select the best rate, returning the label which best matches the (a posteriori) possibility values.

We propose to use a similarity measure based on a geometric distance model, the idea being that the smaller the distance between Π_A and Π_B, the greater the similarity between them. Given two possibility distributions, a one parameter class of distance functions can be defined as [12]:

$$d_z(\Pi_A, \Pi_B) = \left[\sum_{x=1}^{n} abs(\pi_A(x) - \pi_B(x))^z\right]^{\frac{1}{z}} \tag{7}$$

In this paper, we propose to use the parameter $z = 2$, and therefore the predicted vote will be the one with the lowest d_2, i.e. vote $= \arg\min_l\{d_2(\Pi_{ev}(V_a), \Pi_l(V_a))\}$. This approach will be called *DistPossM*.

In order to transform $Pr(V_a|ev)$ into $\Pi_{ev}(V_a)$, all that needs to be done is to normalize it by using the value of maximum probability [13], i.e. $\pi(x_i) = \frac{Pr(x_i)}{\max_{j=1}^{n} Pr(x_j)}$. For $l \in \mathcal{S}_L \longrightarrow \Pi_l(V_a)$, it is satisfied as directly as $\pi_l(s) = \mu_l(s)$. This technique will be noted as *MaxPoss*.

An alternative to *MaxPoss* is the so-called *AcumPoss*. The probabilities are sorted increasingly. The associated possibility would be the sum of the probabilities of the events which are below it in the ranking [13]: let σ be an increasingly sorted ranking of the events considering the associated posterior probabilities, and $\sigma(j)$ the event ranked j^{th}, then $\pi(x_i) = \sum_{j=1}^{i} Pr(x_{\sigma j} \mid ev), i = 1, \ldots, n$.

We must remember that with *MaxPostP*, we can apply both *ProbFL* and *DistPossM*. With this last one, it is possible to apply *MaxPoss* and *AcumPoss*, but with *MaxDifPostPPriP*, we are only able to apply *DistPossM* with *MaxPoss* (differences between probabilities are not probability distributions).

4 Experimentation

The most widely used experimental data set in the recommendation field, and therefore the one that we have selected for our experiments, is currently *MovieLens* [14], consisting of 100,000 ratings (1= Awful, 2= Fairly bad, 3= It's Ok, 4= Will enjoy, 5= Must see) for 1682 movies by 943 users. The data set is divided into 5 training and test sets (disjoint 80% – 20%) for 5-fold cross validation.

In order to test the performance of our model, we shall measure its capability to predict a user's true rating or preferences (system accuracy). Following [15],

Table 1. MAE for Cosine and Pearson variants per number of parents

	MaxPostP			MaxDifPostPPriP		
#P	Cosine	PearsonJC	PearsonOC	Cosine	PearsonJC	PearsonOC
5	0.8495 ± 0.0063	0.8513 ± 0.0085	0.8127 ± 0.0059	1.1268 ± 0.0042	1.3398 ± 0.0126	0.9888 ± 0.0049
10	0.8316 ± 0.0056	0.8263 ± 0.0069	0.7918 ± 0.0041	0.9427 ± 0.0041	1.0734 ± 0.0098	0.8550 ± 0.0013
20	0.8224 ± 0.0061	0.8117 ± 0.0068	**0.7861 ± 0.0037**	0.8592 ± 0.0046	0.9246 ± 0.0074	0.8073 ± 0.0032
30	**0.8191 ± 0.0061**	0.8075 ± 0.0059	0.7872 ± 0.0041	0.8350 ± 0.0045	0.8714 ± 0.0063	0.7939 ± 0.0036
50	0.8198 ± 0.0056	**0.8029 ± 0.0055**	0.7886 ± 0.0051	0.8193 ± 0.0044	0.8301 ± 0.0057	0.7848 ± 0.0042
75	0.8238 ± 0.0062	0.8047 ± 0.0045	0.7938 ± 0.0045	0.8123 ± 0.0047	0.8105 ± 0.0043	0.7837 ± 0.0038
100	0.8249 ± 0.0056	0.8056 ± 0.0049	0.7966 ± 0.0047	**0.8092 ± 0.0048**	**0.8020 ± 0.0038**	**0.7817 ± 0.0037**

we adopt the mean absolute error (MAE) which measures how close the system predictions are to the user's rating for each movie by considering the average absolute deviation between a predicted rating and the user's true rating: $MAE = \sum_{i=1}^{N} abs(p_i - r_i)/N$, with N being the number of cases in the test set, p_i the vote predicted for a movie, and r_i the true rating. In each result table presented, we shall show the average MAE obtained after repeating the experiment for each training and test sets and the standard deviation for the 5 experiments.

The main objective of the experimentation that we have designed is to measure the general performance of the model and compare it with other models. Specific objectives are: 1) to investigate the best vote selection method; 2) to determine the ideal size of the sets of parents of collaborative nodes; 3) to discover which similarity measure performs best; and 4) to observe whether linguistic modelling is a useful technique and how it should be carried out.

The first step is to design a battery of experiments without linguistic modelling in an attempt to find answers to objectives 1) to 3). From previous experimentation, the values of the parameters are: $\alpha = 0$, $\beta = 1$ and $q_i = 1/5$.

We need to conduct various experiments, considering a different number of parents ($\#P$) fixed for all the collaborative node (5, 10, 20, 30, 50, 75 and 100), selected using Cosine, PearsonJC and PearsonOJ similarities. In terms of the technique for selecting the favourite vote, we shall test *MaxPostP* and *MaxDifPostPPriP*.

The results of the experiments are shown in Table 1. The tendency is for the best performance to be reached systematically by the Pearson Coefficient corrected with the overlap over the other two, independently of the prediction method used. *MaxDifPostPPriP* is a slight improvement on *MaxPostP*. Regarding the suitable number of parents, it seems that an intermediate number (20, 30 or 50) is the most appropriate when using the *MaxPostP* method, and a large number of them with *MaxDifPostPPriP* (100). The behaviour of the first method seems to be as expected: with a low number there is not enough information, and with a large number, noise is introduced, leading to bad recommendations in both cases. We were, however, surprised by the behaviour of the second method.

One reason why this situation occurs is that when an item is instantiated to perform a recommendation for a given user, if the item has not been evaluated by either of its parents there is no change in its a posteriori probability distribution with respect to the a priori since the collaborative node does not receive any influence by any other node. When the number of parents is low, this situation is more probable, so there will be more collaborative users with identical

Table 2. MAE obtained using the a priori distribution of U_a

	MaxPostP			MaxDifPostPPriP		
#P	Cosine	PearsonJC	PearsonOC	Cosine	PearsonJC	PearsonOC
5	0.8286 ± 0.0052	0.8252 ± 0.0056	0.8043 ± 0.0047	0.8430 ± 0.0045	0.8487 ± 0.0051	0.8136 ± 0.0056
10	0.8201 ± 0.0050	0.8105 ± 0.0047	0.7871 ± 0.0038	0.8250 ± 0.0042	0.8285 ± 0.0049	0.7899 ± 0.0040
20	0.8154 ± 0.0057	0.8025 ± 0.0057	0.7832 ± 0.0037	0.8112 ± 0.0049	0.8109 ± 0.0058	0.7798 ± 0.0036
30	**0.8148 ± 0.0057**	0.8008 ± 0.0054	**0.7850 ± 0.0041**	**0.8062 ± 0.0048**	0.8012 ± 0.0052	0.7772 ± 0.0038
50	0.8167 ± 0.0054	**0.7984 ± 0.0051**	0.7870 ± 0.0047	0.8029 ± 0.0050	**0.7908 ± 0.0043**	**0.7745 ± 0.0043**
75	0.8215 ± 0.0060	0.8018 ± 0.0043	0.7926 ± 0.0043	0.8031 ± 0.0052	0.7872 ± 0.0039	0.7767 ± 0.0040
100	0.8232 ± 0.0055	0.8033 ± 0.0046	0.7956 ± 0.0045	0.8032 ± 0.0051	0.7860 ± 0.0037	0.7764 ± 0.0040

probability distributions, and therefore, the difference is 0, and the selection of the vote is basically random. This implies that lots of prediction mistakes are made. As the number of parent increases, it is more likely that someone has rated the instantiated items, so the distributions will be different, and fewer cases presented. In order to confirm this, we have counted the number of times that this situation arises, finding the following summarised results (for 5 and 100 parents): Cosine $(4464.4, 175.4)$, PearsonJC $(6839.8, 332.2)$ and PearsonOC $(2939.0, 144.2)$. When this number decreases, the number of parents increases.

Having detected the problem in a node V_a, we have adopted the solution of predicting the vote with the maximum a priori probability computed in its clone node U_a. Table 2 shows the results considering the 'parent help'.

In this case, the overlap variant performs better in both prediction methods. With respect to the number of parents, we can see how in both approaches, intermediate values show the best results. The values shown in this table are better than those presented in Table 2, so the use of this technique is suitable. In terms of which prediction method is preferable, *MaxDifPostPPriP* obtains the best results although the difference is not entirely significant.

Focusing on the application of the linguistic modelling to the output of the system, we have used the following set of linguistic labels in this second stage of the experimentation (with triangular membership functions):

$FL1$: $\mu_{FL1,l_1} = \{1/1, 0.5/2, 0/3, 0/4, 0/5\}$, $\mu_{FL1,l_2} = \{0.5/1, 1/2, 0.5/3, 0/4, 0/5\}$,
$\mu_{FL1,l_3} = \{0/1, 0.5/2, 1/3, 0.5/4, 0/5\}$, $\mu_{FL1,l_4} = \{0/1, 0/2, 0.5/3, 1/4, 0.5/5\}$,
$\mu_{FL1,l_5} = \{0/1, 0/2, 0/3, 0.5/4, 1/5\}$.
$FL2$: $\mu_{FL2,l_1} = \{1/1, 0.5/2, 0.25/3, 0/4, 0/5\}$, $\mu_{FL2,l_2} = \{0.5/1, 1/2, 0.5/3, 0.25/4, 0/5\}$,
$\mu_{FL2,l_3} = \{0.25/1, 0.5/2, 1/3, 0.5/4, 0.25/5\}$, $\mu_{FL2,l_4} = \{0/1, 0.25/2, 0.5/3, 1/4, 0.5/5\}$,
$\mu_{FL2,l_5} = \{0/1, 0/2, 0.25/3, 0.5/4, 1/5\}$.
$FL3$: $\mu_{FL3,l_1} = \{1/1, 0.25/2, 0/3, 0/4, 0/5\}$, $\mu_{FL3,l_2} = \{0.5/1, 1/2, 0.5/3, 0/4, 0/5\}$,
$\mu_{FL3,l_3} = \{0.25/1, 0.5/2, 1/3, 0.5/4, 0.25/5\}$, $\mu_{FL3,l_4} = \{0/1, 0/2, 0.5/3, 1/4, 0.5/5\}$
$\mu_{FL3,l_5} = \{0/1, 0/2, 0/3, 0.25/4, 1/5\}$.
$FL4$: $\mu_{FL4,l_1} = \{1/1, 0.5/2, 0.25/3, 0/4, 0/5\}$, $\mu_{FL4,l_2} = \{0.25/1, 1/2, 0.25/3, 0/4, 0/5\}$,
$\mu_{FL4,l_3} = \{0/1, 0.25/2, 1/3, 0.25/4, 0/5\}$, $\mu_{FL4,l_4} = \{0/1, 0/2, 0.25/3, 1/4, 0.25/5\}$,
$\mu_{FL4,l_5} = \{0/1, 0/2, 0.25/3, 0.5/4, 1/5\}$.

Our aim here is not to tune the model by experimenting with a wide set of fuzzy labels, but rather to determine whether there is an improvement in the recommending ability of our model when linguistic modelling is applied. We have therefore generated the fuzzy labels by changing the granularity degree between them, without their design being too exhaustive.

Table 3 shows the result of the experimentation when linguistic modelling is applied to *MaxPostP* and *MaxDiffPPostPPri* but only for $20, 30$ and 50 parents

Table 3. MAE values for various experiments related to linguistic modelling

#P	FL1	FL2	FL3	FL4
	PostP-MaxPoss-FL?-DistPossM			
20	**0.7384 ± 0.0053**	**0.7401 ± 0.0048**	**0.7735 ± 0.0040**	**0.9247 ± 0.0049**
30	0.7400 ± 0.0045	0.7427 ± 0.0047	0.7787 ± 0.0042	0.9347 ± 0.0059
50	0.7432 ± 0.0046	0.7470 ± 0.0040	0.7895 ± 0.0049	0.9558 ± 0.0068
	PostP-PriorP-MaxPoss-FL?-DistPossM			
20	**0.7357 ± 0.0047**	**0.7368 ± 0.0039**	**0.7690 ± 0.0030**	**0.9044 ± 0.0057**
30	0.7374 ± 0.0042	0.7395 ± 0.0039	0.7749 ± 0.0035	0.9211 ± 0.0066
50	0.7412 ± 0.0043	0.7442 ± 0.0035	0.7863 ± 0.0044	0.9473 ± 0.0073
	PostP-AcumPoss-FL?-DistPossM			
20	**0.7666 ± 0.0049**	**0.7795 ± 0.0047**	0.7407 ± 0.0054	0.7893 ± 0.0040
30	0.7678 ± 0.0048	0.7805 ± 0.0050	0.7396 ± 0.0054	**0.7905 ± 0.0044**
50	0.7672 ± 0.0055	0.7823 ± 0.0054	**0.7383 ± 0.0054**	0.7916 ± 0.0051
	PostP-PriorP-AcumPoss-FL?-DistPossM			
20	**0.8007 ± 0.0058**	0.8126 ± 0.0053	0.7757 ± 0.0061	**0.7977 ± 0.0047**
30	0.7924 ± 0.0053	0.8045 ± 0.0054	0.7649 ± 0.0055	0.7976 ± 0.0049
50	0.7834 ± 0.0059	**0.7982 ± 0.0057**	0.7549 ± 0.0056	0.7973 ± 0.0055
	PostP-FL?-ProbFL			
20	**0.7412 ± 0.0049**	**0.7512 ± 0.0042**	**0.8054 ± 0.0049**	**1, 03 ± 0.0048**
30	0.7434 ± 0.0047	0.7539 ± 0.0035	0.8131 ± 0.0048	1, 05 ± 0.0047
50	0.7477 ± 0.0049	0.7610 ± 0.0039	0.8277 ± 0.0054	1, 08 ± 0.0054
	PostP-PriorP-FL?-ProbFL			
20	**0.7381 ± 0.0045**	**0.7470 ± 0.0032**	**0.8001 ± 0.0038**	**1, 01 ± 0.0056**
30	0.7405 ± 0.0043	0.7502 ± 0.0027	0.8087 ± 0.0039	1, 03 ± 0.0054
50	0.7457 ± 0.0046	0.7579 ± 0.0033	0.8241 ± 0.0047	1, 07 ± 0.0059
	MaxDifPostPPriP-MaxPoss-FL?-DistPossM			
20	**0.8098 ± 0.0040**	0.8527 ± 0.0042	0.8198 ± 0.0038	0.7957 ± 0.0042
30	0.7960 ± 0.0052	0.8415 ± 0.0054	0.8073 ± 0.0051	0.7890 ± 0.0045
50	0.7891 ± 0.0054	**0.8355 ± 0.0054**	**0.8009 ± 0.0046**	**0.7862 ± 0.0048**

(the other sizes show worse results). From this, the main conclusion that we can draw is that the linguistic modelling usually helps improve the performance of the model when an appropriate selection of the parameters is carried out. More specifically, and focusing initially on *MaxPoss*, the best linguistic label is $FL1$, with the remaining labels obtaining worse values. The use of the information provided by the parents is a useful technique when the probability distributions in the nodes from $\mathcal{V}$ are equal. If we deal with the *AcumPoss* method, the tendency changes substantially as this method does not improve *MaxPostP*. In this case, the best values are obtained with $FL3$ and the technique of using the parent distribution does not improve the basic approach. Finally, with *ProbFL*, the behaviour is similar to *MaxPoss*. With respect to the linguistic modelling applied to *MaxDiffPPostPPri*, the *MaxPoss* method works best with *FL4*.

More specifically, we can draw the following general conclusions from this detailed experimentation: 1) the Jaccard coefficient modified with the overlap is the best measure for selecting user parents; 2) it is better to select an intermediate number of parents; 3) we think that it is better to make recommendations by using *MaxPostP* with and without linguistic modelling; 4) the application of linguistic modelling is interesting. The use of *MaxPoss* and *FL1* recommended.

We should compare our model's performance with other published RSs, experimenting with MovieLens and MAE, in order to discover its potential as a recommender system. The best results published so far are around $0.72 - 0.73$ of the MAE metric: Kim and Yum [16] with 0.70, Li and Kim [17] got 0.735, Sarwar et al. [18], 0.72; Chen and Yin [19], 0.732 and Mobasher et al. [20] got 0.73. With these data to hand, we can conclude that our model, obtaining a better MAE of 0.7357, competes well with the best published standards.

5 Conclusions and Further Research

In this paper, we have proposed a general BN-based model for collaborative recommendation, which is both effective and efficient. We have also studied the possibility of considering the set of ratings as vague concepts. Schematically, our system consists of two components: the first uses probabilistic reasoning to compute the probability distribution over the expected vote by means of a Bayesian network, and efficient methods for learning the topology, estimating the probability distributions and propagating; and the second computes the user's vote (the fuzzy set), thereby better representing this probability distribution.

By way of future work, we are planning to study mechanisms to incorporate better specifications of the products into the system and new methods for estimating the weights stored in the nodes of the BN. We do, however, wonder, like [15], whether *"users are sensitive to a change in the mean absolute error of 0.01.* This observation suggests that we might explore different directions instead of merely continuing to improve the MAE metric. In the future, we therefore plan to study problems such as how our system can communicate its reasoning to the users, the minimum amount of data (ratings) required for us to yield accurate recommendations, or how to include item information when recommending.

Acknowledgment. Work supported by the Spanish 'Ministerio de Educación y Ciencia' (TIN2005-02516), 'Consejería de Innovación, Ciencia y Empresa de la Junta de Andalucía' (TIC-276), and Spanish research programme Consolider Ingenio 2010: MIPRCV (CSD2007-00018).

References

1. Resnick, P., Varian, H.R.: Recommender systems. CACM 40(3), 56–58 (1997)
2. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann Publishers Inc., San Francisco (1988)
3. Schiaffino, S.N., Amandi, A.: User profiling with case-based reasoning and bayesian networks. In: IBERAMIA-SBIA, Open Discussion Track (2000) pp. 12–21 (2000)
4. Butz, C.: Exploiting contextual independencies in web search and user profiling. In: Proc. of World Congress on Computational Intelligence, pp. 1051–1056 (2002)
5. Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: 14th UAI Conference, pp. 43–52 (1998)
6. Miyahara, K., Pazzani, M.J.: Collaborative filtering with the simple bayesian classifier. In: Pacific Rim Int. Conf. on Artificial Intelligence, pp. 679–689 (2000)
7. Robles, V., Larrañaga, P., Peña, J., Marbán, O., Crespo, J., Pérez, M.: Collaborative filtering using interval estimation naive bayes. LNCS (LNAI), pp. 46–53. Springer (2003)
8. Heckerman, D., Chickering, D.M., Meek, C., Rounthwaite, R., Kadie, C.: Dependency networks for inference, collaborative filtering, and data visualization. J. Mach. Learn. Res. 1, 49–75 (2001)
9. van Rijsbergen, C.J.: Information retrieval. Butter Worths, London (1979)
10. Zadeh, L.A.: Probability measures from fuzzy events. Math. Anal. Applications. 23, 421–427 (1968)

11. Zadeh, L.A.: Fuzzy sets as a basis for a theory of possibility. Fuzzy Sets and Systems 1, 3–28 (1978)
12. Zwick, R., Carlstein, E., Budescu, D.: Measures of similarity among fuzzy concepts: a comparative analysis. lnternat. J. Approximate Reasoning 1, 221–242 (1987)
13. Klir, G., Parviz, B.: Probability-possibility transformations: A comparison. Int. Journal of General Systems 21, 291–310 (1992)
14. Miller, B., Albert, I., Lam, S., Konstan, J., Rield, J.: Movielens unplugged: Experiences with an occasionally connected recommender systems. In: Proc. of Int'l Conf. Intelligent User Interfaces, pp. 263–266 (2002)
15. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. 22(1), 5–53 (2004)
16. Kim, D., Yum, B.: Collaborative filtering based on iterative principal component analysis. Expert Systems with Applications 28(4), 823–830 (2005)
17. Li, Q., Kim, B.: Clustering approach for hybrid recommender system. In: IEEE/WIC Proc. of the Int. Conf. on Web Intelligence, pp. 33–38 (2003)
18. Sarwar, B., Karypis, G., Konstan, J., Riedl., J.: Application of dimensionality reduction in recommender system – a case study. In: Proc. ACM WebKDD (2000)
19. Chen, J., Yin., J.: Recommendation based on influence sets. In: Proc. of the Workshop on Web Mining and Web Usage Analysis (2006)
20. Mobasher, B., Jin, Y.Z.X.: Semantically enhanced collaborative filtering on the web. In: Berendt, B., Hotho, A., Mladenič, D., van Someren, M., Spiliopoulou, M., Stumme, G. (eds.) EWMF 2003. LNCS (LNAI), vol. 3209, pp. 57–76. Springer, Heidelberg (2004)