A Fuzzy Linguistic Summarization Technique for TV Recommender Systems

Antoine Pigeau, Guillaume Raschia, Marc Gelgon, Noureddine Mouaddib, Régis Saint-Paul

A Fuzzy Linguistic Summarization Technique for TV Recommender Systems

A. Pigeau, G. Raschia, M. Gelgon, N. Mouaddib, R. Saint-Paul

Institut de Recherche en Informatique de Nantes (IRIN/BaDRI)
2, rue de la Houssinière - BP 92208, 44322 Nantes cedex 03 - France
Tel: 02.51.12.58.00 Fax: 02.51.12.58.12 E-mail: {surname}@irin.univ-nantes.fr

***Abstract*—**

The increasing number of satellite and cable television channels is resulting in a soaring number of broadcast programs available to viewers. To alleviate this problem, Personal Video Recorders (multimedia platforms which record TV programs on a hard disk) should integrate a recommender system, which purpose is to filter programs according to their relevance. These systems are based on a user profile, acting as a representative for the user's interests. An important research issue resides in going beyond explicitly user-defined profiles.

This paper presents a TV recommender system using fuzzy linguistic summarization technique, which enables automatic learning of the user profile. The logical architecture of the recommender system based on the SAINTETIQ model, as well as the main ideas of the filtering task are introduced in this communication.

I. INTRODUCTION

The increasing number of satellite and cable television channels is resulting in a soaring number of broadcast programs available to viewers. Their potential can only be fully exploited, if there exist means for viewers to cope with this information overflow. Concurrently, we are observing the advent of a new generation of Personal Video Recorders (PVRs), which replace tapes with hard-disks and possess PC-like computation facilities.

Thus, there is a need to develop schemes dedicated to these home multimedia platforms, that assist users in the selection of programs of interest to them, i.e. that attempt to make *relevance to user*-based filtering of the program stream. In order to characterize the program content, one may either extract metadata from the audiovisual material itself, or rely on accompanying metadata. In the present work, we opt for the latter, given that such descriptors are progressively being made available in practice, either in the program stream, e.g. *Digital Video Broadcast Service Information* (DVB-SI), or as program guides on the internet. In this paper, we only consider descriptors that are global to a program.

Directly in the applicative field of TV recommenders, companies such as Tivo and ReplayTV have proposed commercial products based on the following principle. Upon initialization, the viewer manually defines a user profile through a graphical user interface. This profile is matched to the metadata so as to decide whether to record or discard each incoming program. Although this is very appropriate, selection remains limited to an explicitly formulated user profile. This assumes the user is able to express it through user interaction schemes that are nonetheless to be kept very simple and friendly, and she can be bothered to do so. The alternative, which is the focus of this paper, is to *learn* the user profile through the behaviour (viewing history) of the user. We in fact advocate a system that rely both on explicit and implicit profiling.

The present paper exposes a novel technique for tackling automatic learning of the user profile and for program filtering. This is carried out through the generation of a tree-based organization of fuzzy concepts [1]. The learning task is performed on-line, summarizing the metadata received by the system and characterizing the user interests in this tree. This offers a great flexibility in the design of profiles. A further feature of the proposed scheme is that it is well suited to handling, in the same manner, both numerical and symbolic metadata. Indeed, both forms are likely to appear in program descriptions, e.g. time and type (film, sports, ...). Finally, it supports joint explicit (user-specified)/implicit (learned) profiling.

The remained of this paper is organized as follows. In Section II, we survey existing work on the topic. Having recalled the main principles of the fuzzy set-based linguistic summarization SAINTETIQ framework, on which the present work relies (Section III), we then present the logical architecture of the proposed recommender system (Section IV). Next, we address on the user profiling task through the introduction of implicit profiles (Section V) and explicit ones (Section VI). The last section is dedicated to concluding remarks as well as some perspectives of this work.

II. RELEVANT WORK

Much work is being carried out in the broader area of recommender systems in general. Their purpose may be to suggest e.g. web pages[2], [3] or music bands [4]. Besides the goal of alleviating human-computer interaction, research in automated user profiling is also driven by the commercial product-push model. Focusing on TV recommenders, there exist several proposals that examine implicit profiling: electronic program guides [5], PTV [6] and a multi-agent TV recommender [7].

In most of the current commercial PVR, users have to specify their interests by themselves to get high quality recommendations. The drawback are that user intervention is required, and that the profile is henceforth static. In contrast, PTV [6], the

multi-agent TV recommender [7], and the electronic guide described in [5] enable automatic design of the profile.

PTV [6] is a client-server system operating over the web. It uses an explicit profile and a collaborative algorithm [8] to make recommendation. Thus the system have to face all the problems of collaborative algorithms [9], such as new user, sparse rating, recurring startup or scaling problem.

The multi-agent TV recommender [7] and the electronic program guide [5] implement both an explicit and an implicit profile using content-based filtering. The implicit profile is made automatically, not requiring any initialization from the user.

The implicit profile of the multi-agent recommender [7] is built from TV programs. Positive examples are created starting from wholly viewed programs, and for each of them, a negative example is created. Profile learning and program classification may be carried out using a Bayesian classifier or a decision tree, such as proposed in [7]. The authors conclude on the similarity of experimental results from these two techniques.

The electronic guide [5] also uses the metadata of TV programs to construct implicit profiles. Clustering of examples relies on an ontology which extends the DVB-SI standard with super and sub-categories: the profile is then represented by the class containing positive examples. A statistical analysis on words of the metadata is then carried out to refine the recommendations.

In the present paper, we propose an alternative profile learning schema. Thanks to a fuzzy set-based approach, it can provide a richer and more accurate classification than e.g. decision trees or crisp classifiers.

III. Overview of the SaintEtiQ model

The SaintEtiQ model enables summarization and classification of structured data stored into a database. It applies a conceptual clustering algorithm building a summary tree. Each node of the tree summarizes a part of the database. The most general summary is the root of the tree, whereas the most specific ones are the leaves. The fuzzy hierarchical model of summaries is defined from fuzzy labels (uncertain information) describing crisp tuples (regular database) by the way of a rewriting process. Moreover, linguistic variables and fuzzy partitions enable smooth classification thresholds as well as human-friendly description of the data, with different granularities.

Let us get across the general idea of the SaintEtiQ model by considering an example. For a more detailed presentation, the reader is invited to refer to [1].

SaintEtiQ considers records (or tuples) of a database R and produces summaries of those tuples. In the TV recommender context, SaintEtiQ deals with metadata of TV programs. Each tuple t of R is described over eight attributes: the *schedule*, the *channel*, the *title*, the *topic*, the *category*, the *actors*, the *director* and a brief *summary*. An example of metadata is shown in Table (I).

The first stage of the summarization task is the definition, by an expert, of a fuzzy knowledge base used to sample continuous attribute domains like *Schedule*, as well as to describe summaries of the metadata with customized *linguistic labels*. Furthermore, it supplies us with a unified framework for the representation of attribute values. Fuzzy partitions and linguistic variables are then manually built on the attributes involved in the summarization task.

All tuples t of R are rewritten using this knowledge base, such that a fuzzy linguistic label l, e.g. `early`, gathers several values $t.A$ (`20:30`), of different tuples t on an attribute A (*Schedule*). Label l is associated with a weight α corresponding to the highest membership grade of the $t.A$'s to l: $\alpha = \max_{t \in R}\{l(t.A)\}$. It defines the possibility of having l as a summary descriptor.

Table (II) gives an example of a knowledge base built on the attribute *Topic*.

Label	Value
`action`	$\{1.0/$`martial art` $+ 0.8/$`fear` $+$ $0.8/$`war` $+ 0.5/$`reality show`$\}$
`horror`	$\{1.0/$`fear` $+ 0.9/$`thriller`$\}$
`laughing`	$\{1.0/$`comedy` $+ 1.0/$`one man show`$\}$

TABLE II
KNOWLEDGE BASE ON THE ATTRIBUTE *Topic*

Example 1: The tuple t defined in Table (I) is rewritten on the attribute *Topic* by two fuzzy linguistic labels:

- $t[1].$*Topic* $= 0.8/$`action`
- $t[2].$*Topic* $= 1.0/$`horror`

Thus, it should be noticed that one tuple t could give birth to several nonexclusive candidate tuples, i.e. there exist one or more rewritten form(s) of a single database record according to fuzzy background knowledge.

The second stage of the summarization task in SaintEtiQ consists in clustering all the candidate tuples in a summary hierarchy. Each node z contains a short description of several candidate tuples.

Example 2: A summary z is defined as:

$$< \{0.5/\texttt{very early} + 1.0/\texttt{early}\}, \{1.0/\texttt{action} + 0.4/\texttt{fear}\} >$$

on the attributes *Schedule* and *Topic*, such that `very early` and `early` are fuzzy labels a priori defined in the knowledge base on the attribute *Schedule*.

Candidate tuples are introduced one at a time in the hierarchy with a top-down approach, and they are incorporated into best fitting nodes descending the tree. Thus, the root contains the summary of all the candidate tuples, whereas leaves represent only one combination of fuzzy linguistic labels over all the attributes.

Considering attributes *Schedule* and *Topic*, Figure (1) represents a summary hierarchy built from the three following candidate tuples:

$t[1] =< \{1.0/\texttt{early}\}, \{0.6/\texttt{action}\} >$,
$t[2] =< \{1.0/\texttt{early}\}, \{0.9/\texttt{fear}\} >$,
$t'[1] =< \{1.0/\texttt{early}\}, \{0.8/\texttt{action}\} >$.

The root contains a summary of all its children nodes, whereas the left node contains the summary of $\{t[1], t'[1]\}$ and the right node represents $t[2]$.

In the TV recommender system, all the metadata received by the PVR are classified and summarized in a hierarchy using

Schedule	*Channel*	*Title*	*Topic*	*Category*	*Actors*	*Director*	*Summary*
20:30	Tv film	Shining	fear	film	Nicholson J.	Kubrick S.	Once upon a time...

TABLE I
METADATA OF A TV PROGRAM

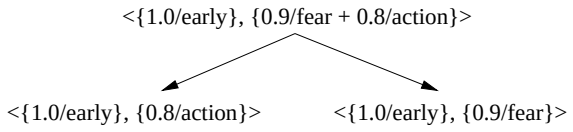

Fig. 1. A summary hierarchy in SAINTETIQ

the SAINTETIQ model outlined above. The main idea is to represent the user profile as several sub-trees of this summary hierarchy. This fuzzy set-based approach takes benefits from the SAINTETIQ features such that:

- the construction of the tree is flexible and dynamic: TV programs are continuously summarized and a new tuple does not require to rebuild the entire tree;
- each node of the tree contains a summary of its children nodes, so it can represent a user interest point;
- each hypothetical user interest point, i.e. a node of the tree, is natively described by linguistic labels taken from user-driven fuzzy knowledge base.

In the next part, we present the extension of this model to achieve the user profiling task in a TV recommender system.

IV. LOGICAL ARCHITECTURE OF THE RECOMMENDER

The overall architecture of the system is represented on Figure (2). It considers metadata of TV programs and, based on the summaries provided by the SAINTETIQ module, it incrementally builds a fuzzy user profile according to successive TV sessions of the user. An interactive mechanism offers the user a way to force the PVR to record a program, even if it is not included in the user's interest as stated by the fuzzy profile. Finally, merging the information of the fuzzy profile and an usual wish list, the system is able to automatically recommend (*filter in*) the program to be recorded by the PVR.

Focusing on the most sensitive module of this logical architecture, it is to be noticed that the fuzzy user profile is composed of two distinct parts:

- an *explicit* profile,
- an *implicit* profile.

The fuzzy profile is built using the SAINTETIQ model. The main idea is to construct a TV program hierarchy from the metadata and to represent the interest of the user by sub-trees.

As already mentioned in Section III, the very first step consists in defining a knowledge base to summarize and classify all the metadata. Thus, it remains an open issue to select automatically relevant attributes. Since these attributes will describe the user's interest, their selection is significant for the quality of the process. The knowledge base being built, one classifies and summarizes metadata from each TV program received by the PVR. Supposing that information about program are received just few minutes before its diffusion the tree contains thus the summaries of all the previously or currently programs and those to come.

The next two sections present the design and usage, respectively of the implicit profile and the explicit profile.

V. IMPLICIT PROFILE

The design of the user implicit profile is automatic, since it is based on user interaction (programs which are visualized, recorded ...). Each program gives birth to an *example*, which is incorporated into an extended representation of the SAINTETIQ tree. Furthermore, using the implicit profile requires computing a *degree of interest* for each program, according to the SAINTETIQ tree. The next two paragraphs detail both parts of the implicit profile.

A. Extension of the SAINTETIQ tree

Examples are created from interactions of the user on the remote control. These interactions are interpreted as like and dislike notifications on programs. Examples are used to update the fuzzy profile and consequently, node descriptions of the SAINTETIQ tree.

Definition 1—Example: An example e is defined as:

$$E =< p, L = \{n_1, \ldots, n_i\}, S \in [0,1], C = \{c_1, \ldots, c_2\} > \ , \quad (1)$$

where:

- p is the program associated with example e;
- L is a list of leaves of the SAINTETIQ tree, where p has been incorporated;
- $s \in [0,1]$ is the score of the example;
- C is a list of characteristics which determines s, such as the proportion of p which was visualized, the number of times the user has tuned to the program p, or the fact that the previously recorded program was deleted while having been visualized little or not, ...

Since valuation of the score s is closely related to cognitive sciences and behavioral studies, it is clearly beyond the scope of this communication. However, in the crisp case, example programs that are liked or disliked are respectively associated with scores equal to 1 and 0. For instance, for each program p which is entirely visualized, and hence liked, we decide to create β *negative examples*, i.e. examples with a score of 0. Such counter-examples are chosen randomly and differently among programs broadcast simultaneous to p, and hence enable proper handling of disliked programs in the system. Initialization of β is a trade-off between the two following factors:

- deleting some relevant interests (if β is too high),

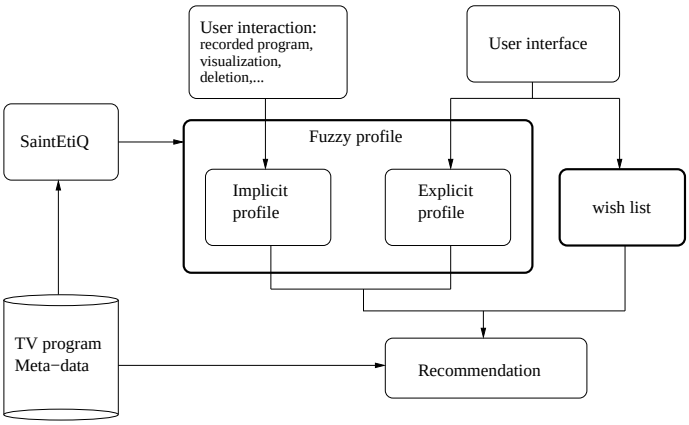

Fig. 2. Logical architecture of the recommender system

- increasing the number of irrelevant programs recorded (if β is too low)

Both positive and negative examples are then used to update the SAINTETIQ tree. The definition of a summary, i.e. a node of the tree, has been extended to take into account the example programs. Thus, each node z of the SAINTETIQ program tree is defined by the following features:

- a summary description, as presented in Section III,
- the number n_V of visualized programs,
- the number n_D of summarized programs in the node,
- the rate $f = n_V/n_D$,
- a score σ,
- a set E of k examples, $k \in \mathbb{N}$.

It should be noted that n_D is the number of rewritten, and not programs, contained in a node: a node represents an interest, so n_D is the weight of this interest in the tree.

A high score σ of a node z means that the user is interested by programs represented by this summary. The score of summary z can be computed in two different ways:

- if z is a leave, the score σ is an average of the example scores $e.s$ of $z.E$;
- otherwise, the score σ of z is an average of the scores of its children nodes.

Parameter k is related to the "memory" of the system (i.e. it affects inertia in temporal evolution of the profile). The higher its value, the more previous examples are considered in the computation of the leave scores. The rate f is used to compare two programs in competition to be recorded by the PVR. For instance, should the user watch nine programs out of ten in node z and twelve out of sixty in node z', the system concludes that the user prefers the summary associated with node z.

The update of the SAINTETIQ tree is carried out regularly from the examples : for each example e, finding all leaf nodes for which program $e.p$ is rewritten by means of the list $e.L$, and then successively updating the score of these leaf nodes and all of their parent nodes until the root. Figure (3) shows the updating process for an example e with $e.L = \{N.1.1.2, N.2.1.2\}$.

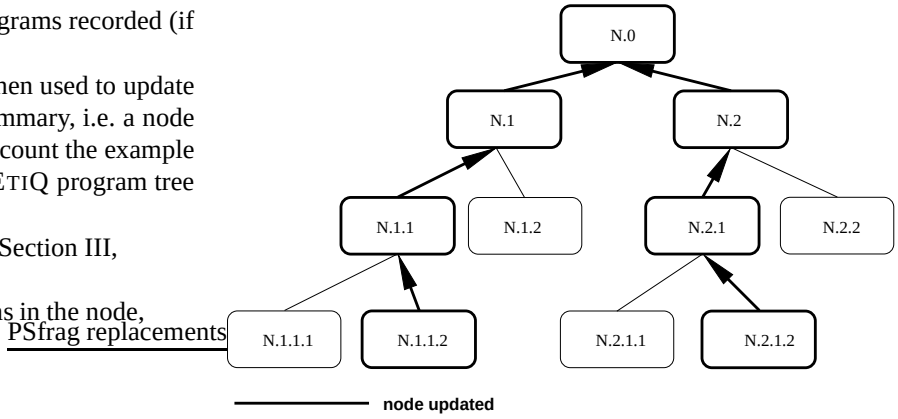

Fig. 3. Implicit user profile update: finding all leaf nodes for which program $e.p$ is rewritten by means of the list $e.L$, and then successively updating the score of these leaf nodes and all of their parent nodes.

B. Rating the interest of TV programs

To use the implicit profile in the program filtering phase, we compute a *degree of interest* for each new program received by the PVR. If the score of the program is higher than a threshold, denoted α, then it is of interest to the user. Let us point out that each new program has already been summarized and classified in the SAINTETIQ tree. For a program p, the degree of interest $d(p)$, based on the fact that programs are classified by similarity, is computed in the following way:

1) find all the leaf nodes z containing a candidate tuple $p[i]$ (rewriting form) of p;
2) for each candidate tuple $p[i]$ of p, find the biggest subtree T containing $p[i]$, and for which each node z' of the path from z to the root of T, the score $z'.\sigma$ of z' is greater than α. Assign the root score σ to $p[i]$, as a degree of interest for this candidate tuple;
3) the degree of interest d of p is then the maximum value of all the $p[i]$'s degrees.

The setting and possible temporal evolution of α determine the proportion of programs that will be filtered in by the system. In practice, this parameter could be driven by computer system considerations, such as the amount of remaining disk space.

Example 3: Figure (4) shows the calculation of a degree of interest for a program p having two associated candidate tuples $p[1]$ and $p[2]$:

$$\begin{cases} d(p[1]) = & [N.0.0].\sigma \\ d(p[2]) = & [N.1.0.1].\sigma \end{cases}$$
$$\Longrightarrow d(p) = \max([N.0.0].\sigma, [N.1.0.1].\sigma) \ .$$

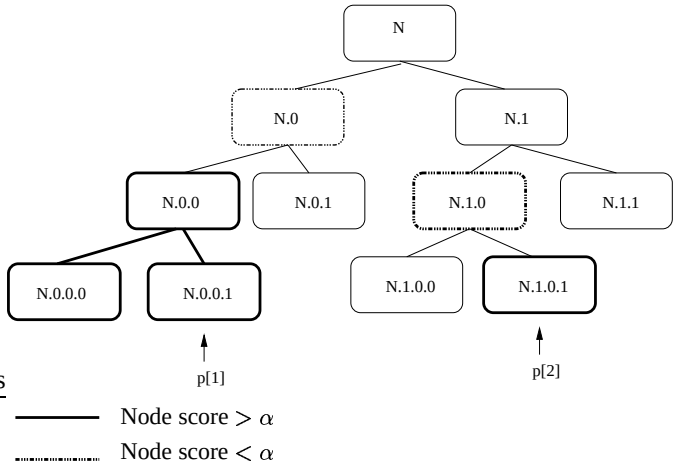

Fig. 4. Usage of the implicit profile: for each candidate tuple $p[i]$ of p, find the biggest subtree containing $p[i]$, and which root score is greater than α. The degree of interest of p is the highest score among all the root nodes.

VI. Integration of the explicit profile

Since the fuzzy profile is defined by both the implicit and the explicit profiles, this section presents the way explicit profiling is integrated into our TV recommender system.

A. Fuzzy tuples as an explicit profile

The design of an explicit profile is carried out by the user: by choosing weighted keywords among the set of linguistic labels over each attribute. The most general form of each attribute value is a fuzzy set of linguistic labels representing a filtering criterion, such as $\{1.0/\texttt{action} + 0.6/\texttt{horror}\}$ on the attribute *Topic*. Thus, an explicit profile is a set X of fuzzy tuples.

The explicit profile is illustrated in Table (III), assuming the attributes used for the summarization task are the *Topic*, the *Schedule*, the *Channel* and the *Category*.

Usage of the explicit profile consists in computing a matching degree between a new TV program and the explicit profile. Furthermore, since the attributes used for the summarization task could be of different importance in a user point of view, the recommender system consider weighted attributes, in a way that values of the most important attributes are more critical for the overall matching degree. Weights w_A of attributes $A \in \mathcal{A}$ verify the following additive constraint: $\sum_{\mathcal{A}} w_A = 1$.

Thus, the computation of the overall matching degree requires the computation of several partial matching degrees $m((X, p[i]))$ between the explicit profile X and each candidate tuple $p[i]$ of program p. The degree m is defined as the greatest weighted sum, over all the attributes, of similarity values between fuzzy sets $x.A$ and $p[i].A$:

$$m(X, p[i]) = \max_{x \in X} \sum_{A} w_A \cdot \max_{l \in \mathcal{L}(A)} \min\{p[i].A(l), x.A(l)\} \ . \quad (2)$$

Finally, the overall matching degree $\delta(X, p)$ between the explicit profile X and the TV program p is defined by an aggregation operator:

$$\delta(X, p) = agg.\ m(x, p[i]) \ . \quad (3)$$

Example 4—TV program vs explicit profile: consider agg defined by the function Max, a TV program p associated to two candidate tuples defined in Table (IV) and the weight vector $< 0.4, 0.4, 0.1, 0.1 >$ for the attributes *Topic*, *Category*, *Schedule* and *Channel*.

According to Equation (2), the matching degree $m(\{x_1, x_2\}, p[1])$ between the explicit profile of Table (III) and the first candidate tuple $p[1]$ of TV program p is computed as:

$$\begin{aligned} m(\{x_1, x_2\}, p[1]) = \max\{ & \\ & 0.1 \cdot 1.0 + 0.1 \cdot 0.0 + 0.4 \cdot 1.0 + 0.4 \cdot 1.0, \\ & 0.1 \cdot 0.0 + 0.1 \cdot 1.0 + 0.4 \cdot 1.0 + 0.4 \cdot 0.0\} \\ = \max\{0.9, 0.5\} = 0.9 \ . & \end{aligned}$$

In the same way, the matching degree between X and the second candidate tuple $p[2]$ of TV program p is computed as:

$$\begin{aligned} m(\{x_1, x_2\}, p[2]) = \max\{ & \\ & 0.1 \cdot 1.0 + 0.1 \cdot 0.6 + 0.4 \cdot 1.0 + 0.4 \cdot 1.0, \\ & 0.1 \cdot 0.0 + 0.1 \cdot 0.0 + 0.4 \cdot 1.0 + 0.4 \cdot 0.0\} \\ = \max\{0.906, 0.4\} = 0.906 \ . & \end{aligned}$$

Hence, according to Equation (3), the overall matching degree between the explicit profile X and the TV program p is equal to the maximum value of the m's, i.e. $\delta(X, p) = 0.906$.

The immediate interpretation to give to this value is that the TV program is fitting well the explicited user preferences.

B. Filtering

Integrating the explicit profile with the implicit one into the recommender system requires to mix the matching degree of explicit profile with the degree of interest of implicit profile for each new TV program. We propose to combine them as follows:

$$Q(p) = FOR(d(p) + \delta(X, p)) \quad (4)$$

with $FOR(a, b) = a + b - a \cdot b$.
Thus, programs which match either implicit or explicit profile only would have a high degree of interest.

Assuming that the PVR can not record more than one TV program, and that programs are simultaneously broadcasted, the recommender system has both to decide either to record a program, and to choose which one to record. Thus, for a set P of concurrent programs, the system first computes the $Q(p)$'s

Id.	*Schedule*	*Channel*	*Category*	*Topic*
x_1	1.0/evening	1.0/film	1.0/film	1.0/laughing
x_2	1.0/early	1.0/general	1.0/film	1.0/action + 0.6/horror

TABLE III

EXPLICIT PROFILE: A SET X OF TUPLES WHICH ARE A COMBINATION OF FUZZY LABELS OVER EACH ATTRIBUTE OF THE SUMMARIZATION TASK.

Id.	*Schedule*	*Channel*	*Category*	*Topic*
$p[1]$	1.0/evening	1.0/general	1.0/film	1.0/laughing
$p[2]$	1.0/evening	0.6/film	1.0/film	1.0/laughing

TABLE IV

CANDIDATE TUPLES OF PROGRAM p

and sorts programs according to these values. Then, either it considers only programs verifying $Q(p) > \alpha$, where α is a given threshold, or it ranks all the concurrent programs. Finally, the system recommends to record the best program, in the sense of the Q value. If there are more than one program having the same greatest value for Q, then the system compares the rate f of tree nodes containing p.

An experiment has been conducted over a toy sample of 100 000 metadata of TV programs. These metadata consist in descriptors among which some of the DVB-SI standard (*Digital Video Broadcasting - Service Information*). The summarization task has just been performed on attributes *Channel*, *Schedule*, *Topic* and *Category* (the issue related to multivalued attributs like *actors* are not solved yet). A small knowledge base has been built on each of these attributes. Even if the system is still too premature to be fully evaluated, we observed some good properties of the recommendation (filtering), as for instance the 'accurate' representation of user interests by SAINTETIQ subtrees, as well as the smooth evolution of the user profile during a few weeks of usage. Hence, this fuzzy set-based approach appears promising in the scope of recommender systems.

VII. CONCLUSION

A technic is proposed to automatically learn a user profile and to filter programs in the applicative framework of TV recommender system. The advantage of SAINTETIQ is that it performs a rich and robust hierarchical conceptual clustering, the centers of interests can thereby be emphasized precisely.

Node descriptions of the SAINTETIQ tree have been extended to take into account the implicit profile of a given user. A criterion has also been proposed to evaluate the relevance of a new TV program, according to the above implicit profile. Finally, a matching degree between an explicit profile and the program has been presented, providing a way of merging both the implicit and explicit profiles to filter programs through the recommender system.

Several problems still remain, closely related to this model:

- the profile depends on the quality and correctness of metadata, and also of the knowledge base used for the construction of the SAINTETIQ tree;
- it would be interesting to consider updating the knowledge base and the SAINTETIQ tree without loosing information of the implicit profile;
- to make operational the implicit profile, one first needs to summarize and classify a significant number of programs and next to acquire examples from the user TV sessions. It is the well-known problem of cold start;

ACKNOWLEDGEMENTS

This work is part of the DOMUS VIDEUM project on future home multimedia platforms, sponsored by contract n0 02.2.93.0108 RNTL from the RNTL - French Ministry of Industry. The authors wish to thank partners from Thomson Multimedia Research & Development, Rennes, France for discussions related to the present work.

REFERENCES

[1] G. Raschia and N. Mouaddib, "A fuzzy set-based approach to database summarization", *Int. Journal of Fuzzy Sets and Systems*, vol. 129, n. 2, pp. 137–162, July 2002.

[2] H. Lieberman, "Letizia: An Agent That Assists Web Browsing", in Chris S. Mellish, editor, *Proceedings of the Fourteenth International Joint Conference on Artifi cial Intelligence (IJCAI-95)*, pp. 924–929, Montreal,Canada, 1995, Morgan Kaufmann publishers Inc.: San Mateo, CA, USA.

[3] M. J. Pazzani, J. Muramatsu and D. Billsus, "Syskill Webert: Identifying Interesting Web Sites", in *AAAI/IAAI, Vol. 1*, pp. 54–61, 1996.

[4] U. Shardanand and P. Maes, "Social Information Filtering: Algorithms for Automating "Word of Mouth"", in *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*, volume 1, pp. 210–217, 1995.

[5] L. Ardissono, F. Portis, P. Torasso, F. Bellifemine, A. Chiarotto and A. Difi no, "Architecture of a system for the generation of personalized Electronic Program Guides", in *UserModeling 2001, Workshop on Personalization In Future TV*, Sonthofen, Germany, 2001.

[6] B. Smyth, P. Cotter and G. O'Hare, "Let's Get Personal - Personalised Television Listings on the Web", in *9th Irish Conference on Artifi cial Intelligence and Cognitive Science. Dublin, Ireland*, 1998.

[7] K. Kurapati, S. Gutta, D. Schaffer, J. Martino and J. Zimmerman, "A multi-agent TV recommender", in *UserModeling 2001, Workshop on Personalization In Future TV workshop*, Sonthofen, Germany, 2001.

[8] P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm and J. Riedl, "GroupLens: An Open Architecture for Collaborative Filtering of Netnews", in *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, pp. 175–186, Chapel Hill, North Carolina, 1994, ACM.

[9] W. S. Lee, "Collaborative Learning for Recommender Systems", in *Proc. 18th International Conf. on Machine Learning*, pp. 314–321. Morgan Kaufmann, San Francisco, CA, 2001.