Information access in Implicit Culture framework

Enrico Blanzieri
Department of Psychology
University of Turin - Italy
blanzier@psych.unito.it

Paolo Giorgini
Department of Mathematics
University of Trento - Italy
pgiorgio@science.unitn.it

Sabrina Recla and Paolo Massa
ITC/Irst - Italy
{recla,massa}@itc.it

ABSTRACT

The goal of a System for Implicit Culture Support (SICS) is to establish an implicit culture phenomenon, namely when the elements of a set behave according to the culture of a generally different group of agents. Earlier work claimed that Implicit Culture support can be seen as a generalization of Collaborative Filtering. In this paper, we recall the concept of Implicit Culture, show how it is useful for automatically exploit tacit knowledge and we present an implementation of a System for Implicit Culture Support.

1. INTRODUCTION

Systems for Implicit Culture Support [1] have the goal of estabilishing an Implicit Culture phenomenon. Supporting Implicit Culture is effective in solving the problem of improving the performances of agents acting in an environment where more-skilled agents are active. In fact, support of Implicit Culture can be useful for various applications: Computer Supported Collaborative Work, Group profiling, Cognitive modelling of social phenomena, e-books, Computer Mediated Communication Systems and, as we briefly present in the following, generalization of collaborative filtering, Knowledge Managment and requirements and interaction control of agent-based systems.

As suggested in [1] Collaborative filtering [5] (a popular technique exploited in recommendation systems) can be seen as a System for Implicit Culture Support (SICS).

In Knowledge Management, generally knowledge is categorized as being either codified (explicit) or tacit (implicit). Implicit Culture can be applied successfully in this context. In particular, the idea is to build systems able to capture implicit knowledge, but instead of sharing it among people, change the environment in order to make new people behave in accordance with this knowledge.

2. IMPLICIT CULTURE SUPPORT

The goal of a SICS is to establish an implicit culture phenomenon. In the following, we informally introduce the notions of implicit culture and implicit culture phenomenon (the formal definitions can be found in [2]). The second part of the section presents the general architecture of a SICS and shows how it relies on learning techniques.

We call Implicit Culture a relation between a set of agents G' and a group G such that the agents of G' perform actions that satisfy a cultural constraint for G. When a set and a group of agents are in Implicit Culture relation, we have an Implicit Culture phenomenon. The definitions do not require the empirical validation of the cultural constraint theory against the executed actions of G.

The general architecture of a SICS proposed in [1] allows to achieve the goal of estabilishing an implicit culture phenomenon following two steps. First, the elaboration of a cultural constraint theory $\mathbf{\Sigma}$ from a given domain and a set of situated executed actions of a group G. Second, the proposal to a group G' of a set of scenes such that the expected situated actions of the set of agents G' satisfies $\mathbf{\Sigma}$. Both the steps present learning problems.

A general SICS (see Figure 1-a) consists of three components: observer, inductive module and composer. The observer stores the situated executed actions of a group of agents G in oder to make them available for the other components. The inductive module uses these actions to produce a cultural constraint theory $\mathbf{\Sigma}$ for G. Finally, the composer, using the theory $\mathbf{\Sigma}$ and the actions, manipulates the scenes faced by a set of agents G' in such a way that their expected situated actions are cultural action w.r.t G. As a result, the agents of G' executes (on average) cultural actions w.r.t G (implicit culture phenomenon).

In general, our implemented architecture accepts cultural theories expressed by a set of rules of the form:

$$A_1 \wedge \cdots \wedge A_n \rightarrow C_1 \wedge \cdots \wedge C_m$$

in which $A_1 \wedge \cdots \wedge A_n$ is said antecedent and $C_1 \wedge \cdots \wedge C_m$ consequent. The idea is to express that "if in the past the antecedent has happened, then there exists in the future some scenes in which the consequent will happen".

The composer proposes to a set of agents G' a set of scenes such that their expected situated actions satisfy a cultural constraint theory $\mathbf{\Sigma}$ for a group G. The main idea is splitting the problem in two sub-problems: (i) find the cultural actions and (ii) find the scenes where such actions are the expected situated actions. Figure 1-b shows the composer in detail. Basically, the composer consists of two main submodules. The first is the *Cultural Actions Finder* (CAF), that takes as inputs the theory $\mathbf{\Sigma}$ and the executed situated actions of G', and produces as output the cultural actions

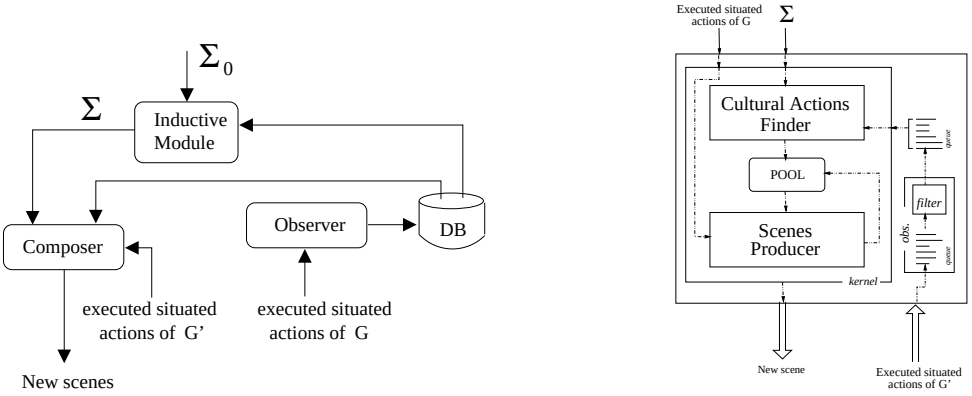

Figure 1: Architecture (a) The composer in detail (b)

w.r.t. G (namely, the actions that satisfy $\mathbf{\Sigma}$). The CAF matches the executed situated actions of G' with the antecedents of the rules of $\mathbf{\Sigma}$. If it finds an action that satisfies the antecedent of a rule, then it takes the consequent of the rule as a cultural action. The second is the *Scenes Producer* (SP), that takes one of the cultural action produced by the CAF and, using the executed situated actions of G, produces scenes such the expected situated action is the cultural action. The SICS architecture requires the solution of two learning problems:

Inductive Module Problem. Given a set of situated executed actions performed by the agents of G, find a cultural constraint theory.

Scene Producer Problem. Given a set of situated executed actions of the agents of G and G', and given a cultural action α for the agent x, find a scene s such that the expected situated action of x in the scene s is α.

The Inductive Module Problem is a rather standard learning problem: inducing the behavior patterns of a group and it is possible to solve using standard data mining techniques. As we previously noted, we do not require any specific validation of the theory. Obviously, very different effects will be reached depending on the fact that the theory is validated or not. In the Scene Producer Problem the request is on the effectiveness of the scene w.r.t the goal of producing the execution of a given action, namely its *persuasiveness*.

3. THE SCENE PRODUCER PROBLEM

The solution exploits the principles of instance-based learning (namely, memory-based or lazy). Given a cultural action α for the agent x that performed actions on the set of scenes $S(x)$, the algorithm used in the SP consists of three steps:

1. find a set of agents Q that performed actions similar to α;
2. select a set of agents $Q' \subseteq Q$ similar to x and the set of scenes S in which they performed actions;
3. select and propose to x a scene of S.

An agent y is added to the set Q if the similarity $sim(\beta_y, \alpha)$ between at least one of its situated executed actions β_y and α is greater than the minimum similarity threshold T_{min}. The scenes s in which the β_y actions have been executed are added to $S(y)$, that is the set of scenes in which y has performed actions similar to α.

Step 2 selects in Q the k nearest neighbors to x with respect to the agent similarity defined as follows:

$$w_{x,y} = \frac{1}{|S_{xy}|} \sum_{s \in S_{xy}} \frac{1}{N_x(s) N_y(s)} \sum_{\beta_x \in N_x(s)} \sum_{\beta_y \in N_y(s)} sim(\beta_x, \beta_y) \quad (1)$$

where $S_{xy} = S(x) \cap S(y)$ is the set of scenes in which both x and y have executed at least an action. $N_x(s)$ and $N_y(s)$ are the set of actions that x and y have respectively performed in the scene s. Eq. 1 can be replaced by a domain-dependent agent similarity function if needed (e.g., Pearson).

Step 3 selects the scenes in which the cultural action is the expected situated action. To do this, firstly we estimate for any scene $s \in S = \bigcup_{y \in Q} S(y)$ the similarity value between expected action and cultural action, and then we select the scene with the maximum value. The function to be maximized is the expected value $E(sim(\beta_x, \alpha)|s)$, where β_x is the action performed by the agent x, α is the cultural action, and $s \in S$ is the scene in which β_x is situated. The following estimate is used:

$$\hat{E}\left(sim(\beta_x, \alpha)|s\right) = \frac{\sum_{u \in Q'} \hat{E}_1\left(sim(\beta_u, \alpha)|s\right) * w_{x,u}}{\sum_{u \in Q'} w_{x,u}} \quad (2)$$

that is we calculate the weighted average of the similarity of the expected actions for the neighbor of the scene, $w_{x,u}$ is the similarity between the agent x and the agent u, whereas E_1 is estimate as follows:

$$\hat{E}_1\left(sim(\beta_u, \alpha)|s\right) = \frac{1}{|N_u(s)|} \sum_{\beta_u \in N_u(s)} sim(\beta_u, \alpha) \quad (3)$$

that is the average of $sim(\beta_u, \alpha)$ over the set of actions $N_u(s)$ performed by u in s.

The algorithms described above and the general architecture described in Section 2 are fully implemented in Java using XML for expressing the cultural constraint theory.

4. RECOMMENDATIONS EXPERIMENTS

Collaborative filtering can be seen as a particular SICS. In this section we present the results of some experiments aimed to compare a SICS parametrized for a recommendation problem with collaborative filtering. The goal of the experiments is validating the system against a well-estabilished method on a particular domain. For collaborative filtering we use a memory and neighborhood based algorithm presented by Herlocker et al. in [5].

action sim agent sim sim min neighbors	eq. 4 (domain-dependent similarity) Pearson Correlation coefficient T_{min}=0 k=10	k=30	k=50	k=80	k=150	k=300	T_{min}=0.8 k=80	eq. 1 T_{min}=0 k=80	*simple* Pearson T_{min}=0 k=80
ranks									
1	25.4	57.6	65.3	68.6	68.6	69.5	66.1	58.0	6.8
2	14.4	11.0	11.0	10.2	10.2	11.0	12.7	14.3	4.2
3	8.5	7.6	6.8	7.6	8.5	6.8	6.8	8.4	4.2
4	10.2	5.1	5.1	4.2	4.2	3.4	5.9	5.9	4.2
5	6.0	2.5	1.7	0.0	0.0	1.7	0.0	1.7	5.1
over 5	35.5	16.2	10.1	9.4	8.5	7.6	8.5	11.7	75.5

Table 1: Experimental results. Distribution in percentage of the items proposed by SICS in the rankings position proposed by the CF algorithm with $K_{CF} = 30$.

The SICS used for the experiments is characterized by: a set of agents (users) $\mathcal{P} = \{u_1, \cdots, u_n\}$; a set of objects $\mathcal{O} = M \cup V$ (M is the set of items and $V = \{0, 0.2, \cdots, 0.8, 1\}$ the set of the possible votes) and a set of possible actions $\mathcal{A} = \{vote, request\}$. For a recommender system the cultural theory is specified in advance (so no inductive module is needed) with the following rule:

$$request(x, t_1) \rightarrow vote(x, m, v_{max}, t_2) \wedge less(t_1, t_2)$$

that states that if x requests a suggestion, then x will assign the maximum vote to the proposed item. For a recommender system we require the satisfation of the user with the recommended items. For similarity between actions we use the following domain-dependent function:

$$sim(A_1, A_2) = \begin{cases} 0 & if(A_1 = vote \wedge A_2 = request) \vee \\ & (A_1 = request \wedge A_2 = vote) \vee \\ & (A_1 = vote(x, o, v_1) \wedge \\ & A_2 = vote(y, p, v_2) \wedge o \neq p) \\ 1 & if(A_1 = A_2 = request) \\ 1 - |v_1 - v_2| & if(A_1 = vote(x, o, v_1) \wedge \\ & A_2 = vote(y, p, v_2) \wedge o = p) \end{cases} \quad (4)$$

that means that the similarity is zero when the actions are different or when they are both *vote* but about distinct objects; it is maximum (namely 1) when the actions are both *request*; and finally, it is $1 - |v_1 - v_2|$ when the actions are both *vote* about the same object, but with different numerical votes.

For the experimentation we used the database EachMovie [6] that collects data of 72961 users who voted 1623 movies. We built a dataset considering the first 50 movies and the 119 users with at least one vote among the first 300 and we run a leave-on-out w.r.t the users. The CF algorithms returns a list of estimated ranking while in this case the SICS returns only a scene composed by a movie. We compared the movie proposed by the SICS and the best 10 movies ranked by the CF algorithm computing the distribution in percentage. We run our implementation of the CF algorithm obtaining a Mean Absolute Error comparable to the ones presented in the literature [3, 4]. We run the experiments on the SICS changing the number of neighbours, the threshold and the functions used to compute the similarity among agents and actions. Table 1 presents the results of the experiments.

A low k implies a low number of scenes (items) valutated and consequently low performance (compare columns 1-6) because it is likely to miss some valuable items. With $T_{min} = 0$ the number of valutated scenes is on average 38.4 against 25.2 with $T_{min} = 0.8$, hence the speed of presentation is higher with a slight decrease in the performance (compare columns 4 and 7). The domain-dependent agent similarity function (Pearson) is slightly better that the general one 1 (compare columns 4 and 8). Finally, a very simple (syntactic) action similarity performs poorly (compare columns 4 and 9). We can conclude that with domain-dependent similarities our SICS is comparable with CF. More interestingly, comparing the last cells of column 4 and 8 it is possible to conclude that also the version without the domain-dependent agent similarity but with a general one performs satisfactorily.

5. CONCLUSIONS AND FUTURE WORK

We have presented an implementation of a System for Implicit Culture Support that exploit an application of instance-based learning techniques. We have showed that, in a particular domain and with a simple a priori theory, the system is functionally equivalent to Collaborative Filtering. Moreover we have presented a real-world application. Our three-steps algorithm for proposing the scene can be considered a generalization of a Collaborative Filtering algorithm, where the similarity is performed on executed actions and not on ratings. This generalization is non–trivial for it puts the SICS in a wider framework than simple CF. In fact it is possible to vary the domain, the cultural contraint theory, and also to deal with artificial agents as we have done in [2].

Future work will be devoted to experimentation on domains with a wider range of actions, more complex scenes and more complex, possibly induced, cultural constraint theories.

6. REFERENCES

[1] E. Blanzieri and P. Giorgini. From collaborative filtering to implicit culture. In *Proceedings of the Workshop on Agents and Recommender Systems*, Barcellona, 2000.

[2] E. Blanzieri, P. Giorgini, P. Massa, and S. Recla. Implicit culture for multi-agent interaction support. In *Proc. of the Conf. Cooperative Information Systems COOPIS*, 2001.

[3] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. of the Fourteenth Conf. on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, 1998.

[4] A. Gokhale. Improvements to collaborative filters algorithms. Master's thesis, Worchester Ploytechnic Institute, May 1999.

[5] J. Herlocker, J. K. J., A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proc. of the 1999 Conf. on Research and Development in Information Retrieval*, 1999.

[6] P. McJones. Eachmovie collaborative filtering data set, dec systems research center, 1997. http://research.compaq.com/SRC/eachmovie/.