Improving Case Representation and Case Base Maintenance in Recommender Agents

Miquel Montaner, Beatriz López, and Josep Lluís de la Rosa

Institut d'Informàtica i Aplicacions
Universitat de Girona
Campus Montilivi
17071 Girona, Spain
{mmontane, blopez, peplluis}@eia.udg.es

Abstract. Recommendations by salespeople are always based on knowledge about the products and expertise about your tastes, preferences, interests and behavior in the shop. In an attempt to model the behavior of salespeople, AI research has been focussed on the so called recommender agents. Such agents draw on previous results from machine learning and other advances in AI technology to develop user models and to anticipate and predict user preferences. In this paper we introduce a new approach to recommendation, based on Case-Based Reasoning (CBR). CBR is a paradigm for learning and reasoning through experience, as salesmen do. We present a user model based on cases in which we try to capture both explicit interests (the user is asked for information) and implicit interests (captured from user interaction) of a user on a given item. Retrieval is based on a similarity function that is constantly tuned according to the user model. Moreover, in order to cope with the utility problem that current CBR system suffer from, our approach includes a forgetting mechanism (the drift attribute) that can be extended to other applications beyond e-commerce.

1 Introduction

In the real world, if you always go to the same grocery store to buy your food, then the salespeople there will get to know a great deal about you. Based on the products that you normally buy, they might recommend new items that they think might interest you. For instance, if you always buy a certain brand of milk, a salesperson might recommend, with a good chance of success, another similar product that has a discount that particular week. They will also notice the changing interests of the customers over time. If you do not buy a given product for a long time, salespeople gradually stop recommending similar products. They guess that you are not interested in them anymore. The recommendations of salespeople are always based on knowledge about the products as well as about your tastes, preferences, interests and behavior in the grocery store. In short, your activity in the grocery guides salespeople to improved sales.

In an attempt to model the behavior of salespeople, AI research has been focussed on the so-called *recommender systems* [19]. The main task of a recommender system is to locate items, information sources and people related to the interest and preferences of a single person or a group of people [20]. This involves the construction of user models and the ability to anticipate and predict user preferences.

In addressing these types of tasks, recommender systems draw on previous results from machine learning and other AI technology advances. Among the various machine-learning technologies we concentrate on Case-Based Reasoning (CBR) as a paradigm for learning and reasoning through experience, as salespeople do. Previous recommender systems based on CBR include, for example, the Broadway approach [10, 24], WebSell [6], Adaptive Place Advisor [9] and Entree [2].

Recently, a quite exhaustive experimentation on similarity metrics, performed with Entree, showed that the quality of recommendations increases if they are based on knowing the reasons behind user preferences rather than on simply having large amounts of information [2]. So, learning that user A does not like an item that was recommended because it was successfully recommended to a similar user B, and knowing the reasons for such a difference between these similar users, increases accuracy. The experiments performed with Entree lend support to the case representation approach presented in this paper, based on two main parts: objective attributes of items (the problem) and subjective attributes (the solution), in which we capture the user's explicit interests (the user is asked) as well as his/her implicit interests (the user interaction is recorded) for a given item. Moreover, in order to cope with the utility problem, our CBR approach includes a forgetting mechanism that can be extended to other applications beyond e-commerce. Our view is in line with current trends in recommender systems development and follows agent-based approaches, and so we speak about personal recommender agents. The agent acts as the grocery salespeople: proactively recommending new products to the user and giving prompted answers to queries from the user.

The outline of this paper is as follows: the next section introduces a new approach to applying CBR to the recommendation domain. The following sections define the case base structure and each CBR phase of this new approach. Finally, section 8 presents related work and section 9 concludes this article.

2 Overview of the CBR Approach to Recommendation

The core of CBR is a case base which includes all the previous experiences that can give us information about new problems. Then, through the similarity concept, the most similar experiences are retrieved. However, similarity is not a simple or uniform concept. Similarity is a subjective term that depends on what one's goals are. For instance, two products with the same price would get maximum similarity if the user was interested in products with that same price, but would get very different similarity for other concepts, such as quality or trademark.

In our approach, the case base consists of a set of previous items explicitly and/or implicitly assessed by the user. Assuming that the user's interest in a new item is similar to the user's interest in similar items, in order to evaluate whether a new item could interest the user, the agent searches the case base for similar items. If the interest the user showed in them is high enough, the new item is recommended to the user.

With regard to the CBR cycle:

1. In the retrieval phase, as from a new item, the system searches similar items in the case base in order to find out whether the user might be interested in it. Local similarity measures are based on item attributes, the relevance of which is computed in each retrieval time. Thus, the similarity metric is automatically maintained.
2. In the reuse phase, as from the retrieved set of similar items, the system calculates a confidence value of interest, to recommend the new item to the user, based on explicit and implicit interests and the validity of the case according to the current user's interests.
3. In the revision phase, as from the relevance feedback of the user, the system evaluates the user's interest in the new item. The idea is to track user interaction with the system to get to know relevant information about the user's interest in the recommended item, as well as explicit and implicit information, in order to retain the new case.
4. In the retain phase, the new item is inserted in the case base with the interest attributes that were added in the revision phase. In order to control the case base size, it is also important to know whether the user never gives new feedback about items in the case base. In such a case, it is necessary to forget these interests with time. We propose the use of a new attribute that we call the *drift attribute*, that will be aware of such changes in the user preferences and contribute to case maintenance.

The CBR cycle is applied both when the agent gives prompted recommendations in the course of answering the user's queries, as well as when the agent proactively recommends something to the user looking for new products. For example, the latter case happens periodically when the recommender agent contacts the restaurants server looking for new restaurants similar to the one that best fits the user preferences.

In the following sections the structure of the case base and the different CBR phases of the new approach are described.

3 The Case Base

A case-based reasoner is heavily dependent on the structure / representation and content of its collection of cases. Moreover, the size of the case base is also an open problem, since it affects the performance of the system. Below, we introduce

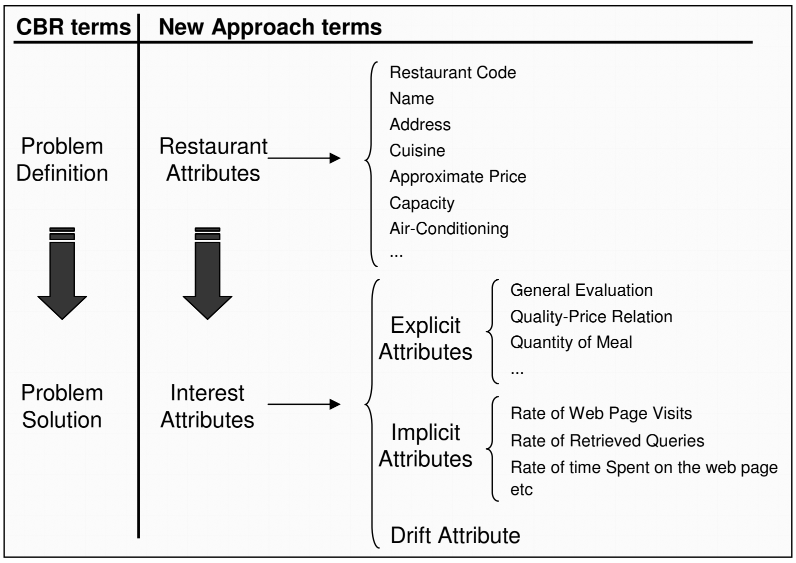

Fig. 1. An Example of Case Representation in the Restaurants Domain

the case base representation we use for recommendation and how we solve the utility problem with a control attribute called the drift attribute.

3.1 The Case Base Representation

In our approach a case is split into two parts: the first, a set of attributes describing the item (the definition of the problem in CBR terminology) and the second, a set of attributes describing the user's interest (the solution to the problem in CBR terminology). We decided to make the agent handle items from just one topic (e.g., restaurants). Extending the recommender system to other topics can be carried out in a cooperative environment (see for example [18]). In order to constrain the domain, we simplify the case base, since items from the same topic have the same attributes. As shown in Figure 1 from the restaurants recommendation domain, the case representation consists of a set of **item attributes** describing the restaurant (product description) and a set of **interest attributes** describing the opinion of the user about the restaurant (product assessment).

Firstly, in order to create a set of attributes for an item, we must use whatever available information there is about the item's qualities. Item descriptions in general do not tend to be very complex, consisting largely of descriptive adjectives, nouns or values. For example, when the CBR goal is to recommend restaurants, the system can deal with features of capacity (e.g., "100 places" or "150 places"), qualities of the cuisine (e.g., "traditional", "creative" or "bland") or approximate price (e.g., "from \$10 to \$15" or "from \$20 to \$30").

Secondly, interest attributes keep all the information the agent gathers from the user. These attributes depend on the technique used to obtain the relevant feedback: explicit or implicit [17]. Explicit feedback relies on the fact that any information can be asked of the user explicitly, such as quality-price relation or quantity of food. On the other hand, implicit information can be captured from the interaction between the user and the system, for example, the consulted items, the time spent consulting items, the number of visits to the web page or the number of queries made where the restaurant was retrieved as an answer.

Explicit attributes are the most relevant and accurate, since the user states his opinion. However, it can be a great nuisance to the user and, therefore, it is not always possible to obtain [5]. Implicit attributes are a lot less accurate, but there is no annoyance to the user, since the system just tracks his or her behavior and learns about it. Thus, we deal with both kinds of feedback and we represent the case interests through both explicit and implicit attributes.

Finally, if a case represents the user interest on a given item, the complete case base constitutes the user profile representation which models the user. So, each recommender agent keeps a case base that is the representation of the user on whose behalf the agent is acting.

3.2 The Case Base Maintenance

The main idea of the CBR is to solve new problems by adapting the solutions given for old ones. It should be remarked that, presumably, with a larger set of cases the system gives better results as long as cases cover a wide range of problems. However, several authors claim that when the case base reaches a certain number of cases, the performance of the system remains the same and sometimes decreases [12].

Thus, one of the main drawbacks of the CBR is the utility problem: the uncontrolled growth of case bases may degrade system performance as a direct consequence of the increased cost in accessing memory.

Therefore, a need arises for a technique that controls the case base size by forgetting irrelevant cases. Some approaches handle this problem by storing new cases selectively (for example, only when the existing cases in memory lead to a classification error) and deleting cases occasionally [11]. Other approaches incorporate a restricted expressiveness policy into the indexing scheme by planning an upper bound on the size of a case that can be matched [8]. More recently, McKenna and Smyth [15] proposed a competence model for CBR that has been followed by other researchers defining different performance-based metrics (e.g., [13]). Most of the measures proposed are based on case base coverage.

In the context of recommender agents, however, maintenance is even harder since it is necessary to handle the change of the human interests over time. Mitchell et al proposed learning the description of the user's interests only from the latest observations, with a time window [16]. Maloof and Michalski suggested giving examples an age and deleting instances from the partial memory that are older than a certain age [14]. Billsus and Pazzani implemented a dual user model consisting of both a short-term and a long-term model of user's interests [1]. Finally, Webb and Kuzmycz introduced forgetting old interests with the gradual forgetting function [26]. The main idea behind this, is that natural forgetting is a gradual process.

Based on this last approach, we propose the **drift attribute** in order to adapt to the user interests over time and to solve the utility problem of the CBR systems. The drift attribute is one of the interest attributes (see Figure 1) and its function is to age the cases in the case base. There is one drift attribute per case and it is updated according to the user-system interaction.

The drift attribute works as follows:

- The drift attribute value is confined to the [0-1] interval.
- New items are inserted in the case base with the maximum drift value when the user shows some interest about them.
- The value of the drift attribute is decreased over time, emulating the gradual process of people losing interest in something. The decreasing function is a simple one where the drift attribute δ_q of a case q is decreased by multiplying the last drift value by a factor β of between 0 and 1 (see Equation 1).

$$\delta_q = \delta_q * \beta \tag{1}$$

  The key issue then, is when to apply the decreasing function. We have to take into account that different users interact with the system more frequently than others. Therefore, the decreasing function should depend on the user interaction, rather than on a certain number of days or weeks. Our system decreases drift attributes each time a new item is incorporated into the case base.
- The value of the drift attribute is increased (rewarded) if the retrieved case results in a successful recommendation. The rewarding function is as simple as the decreasing one. The drift attribute δ_q of a case q is increased by dividing the last drift value by a factor λ of between 0 and 1 (see Equation 2).

$$\delta_q = \delta_q / \lambda \tag{2}$$

When a case reaches a drift value under a certain threshold (ξ), it is discarded. If the drift value is low enough, it does not make sense to retain the item in the case base. The confidence value for the interest that this item gives is insignificant and it is a useless case that only contributes to increasing the size of the case base and decreasing the performance of the system. Therefore, removing cases with a low drift value is the best solution for automatically controlling the size of the case base.

Initiating the case base adaptation needs a setup phase where the parameters are optimized to get the best performance out of the system. In other words, we will obtain different results by changing the rewarding function (λ), the decreasing function (β) and the threshold (ξ). Finding out the optimal values is an empirical task based on metrics to evaluate the system.

Finally, we want to point out that other authors have applied a gradual forgetting function to their systems in order to adapt the user profile to new interests [26, 21]. However, they have a weight/age for all the items of a given topic and when some event affects one of the items in the topic this weight/age is modified in such a way as to affect all the items in that topic. We think that this reduces system performance, especially when the same topic includes a large set of items. Because if in the same interest topic there is one single item that the user is interested in, the topic never drifts, even if the user is not interested in all the other items in the set. Alternatively, in our approach we assign a weight to each item, thus, every case is treated individually, hence solving this problem.

3.3 The Initial Case Base Generation

It is desirable to find out as much as possible from the user so that recommender agents provide satisfactory results from the very beginning. In our approach, the training set [17] seems to be the best technique to generate the initial profile (case base), since the training set list of items given by the user with the appropriate attributes of interest can be the initial case base.

Analyzing the initial profile generation techniques stated in [17], we found different advantages and drawbacks. In manual generation, the user tailors his or her own profile, thus it is a really transparent method. But it bothers the user to have to do this and it is difficult for users to define their preferences explicitly. The empty approach needs potentially a long time to get to know the user's preferences, that is, the initial recommendations are low quality. But in this case, the user is not bothered. The usual approach is to interview the user with a quick manual questionnaire that won't annoy him or her too much, but people are reluctant to give personal data. Typically, the user does not fill in the questionnaire or provides false data. The training set approach depends totally on the profile learning technique (case retain in CBR), since the user just gives a list of items that he likes and/or dislikes, and the learning technique generates the profile. There is nothing to annoy the users and the users easily define their preferences.

4 The Retrieval Phase

In CBR terminology, the retrieval task starts with a new problem description and ends when the best matching set of previous cases has been found. When we apply CBR to recommendation, this phase has the same purpose, but instead of retrieving similar problems, the system retrieves similar items. Thus, the retrieval task ends when the set of best matching previous items has been found (see Figure 2).

The most important step in the retrieval phase of CBR is to define the degree of similarity between cases. The success of CBR systems widely depends on the capacity of the system to exhibit how similar two cases are. With an efficient similarity measure, given a case, we can obtain an ordered list of similar cases. Taking advantage of this concept, when a user likes an item, his/her recommender agent can recommend to him/her a list of similar items that the user should like.

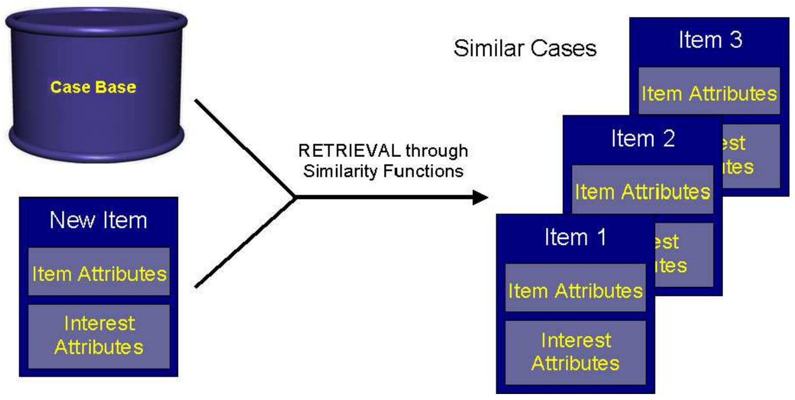

Fig. 2. Retrieve Phase

The degree of similarity between two items is computed by a global similarity function. Typically, the global similarity function is a weighted ponderation of the different attribute similarities based on the user model. For example, a particular user considers that two restaurants are similar if they have a similar price, but another user considers that they are similar if they have a similar cuisine. To get these attribute preferences you can ask the user explicitly when he /she asks the recommender agent for advice. However, such preferences are not available when a recommender agent proactively looks for new items. Therefore, we distinguish two roles of the agent that enforce two different similarity functions:

– Making prompted recommendations to the user when answering his/her queries. In our system the user can explicitly give his/her preferences on the attributes of the restaurant with a numerical value in [1-5] when he/she is looking for a new one. Then, we implement the global similarity function as a Weighted Average (WA) [23], since we can use a normalization in [0,1] of the preference values $pref_i$ given explicitly by the user as weights. In our model, we use the following function:

$$Sim(q,c) = \sum_{i=1}^{n} w_i * sim(q_i, c_i) \quad (3)$$

where:

- n is the number of item attributes of a case,
- q_i are the item attributes of case q,
- c_i are the item attributes of case c,
- $sim(q_i, c_i)$ are the different attribute similarities between q_i and c_i,
- and w_i are the weights of the ponderation, so that $w_i \in [0,1] \forall i$ and $\sum_i w_i = 1$.

In order to calculate the weights w_i based on the preference values $pref_i$, we use the following function:

$$w_i = \frac{pref_i}{\sum_{i=1}^{n} pref_i} \tag{4}$$

– Making proactive recommendations to the user who is looking for new products. When acting proactively, the agent has no information about the preferences of the user on item attributes. Thus, we implement the global similarity function as an Ordered Weighted Average (OWA) [28], since we are dealing with different preferences of the user and such preferences can be ordered according to their importance. Then, the similarity between cases q and c is calculated as:

$$Sim(q, c) = \sum_{i=1}^{n} w_i * sim_{\sigma(i)}(q_i, c_i) \tag{5}$$

where:
- n, q_i, c_i and w_i have been already explained in equation 3,
- $sim_{\sigma(i)}(q_i, c_i)$ are the different attribute similarities between q_i and c_i and
- $\sigma(i)$ is a permutation of the values $1, ..., n$ so that $sim_{\sigma(i-1)}(q_{i-1}, c_{i-1}) \geq sim_{\sigma(i)}(q_i, c_i)$ $\forall i = 2, ..., n$.

Thus, the key of the global similarity function is to define the weights w_i. In our model, we use the following function:

$$w_i = \begin{cases} \frac{1}{n-i} & if\ i < \frac{n}{2} \\ \frac{1}{n} & if\ i = \frac{n}{2} \\ \frac{1}{n+2*(i-\frac{n}{2})} & if\ i > \frac{n}{2} \end{cases} \tag{6}$$

Attribute similarities $sim(q_i, c_i)$ depend on the type of attribute. In our case base, we deal with both numerical and labelled attributes, thus, we consider the typical similarity metrics for both kind of attributes [27, 7].

Once the similarities between the new case and the cases in the case base are calculated, a set of best matches is chosen. In our implementation, we select the n best cases provided which exceed a minimum selection threshold.

5 The Reuse Phase

The reuse phase consists of adapting the old solutions of the retrieved cases to the new problem based on the differences among them. Once the system has retrieved a set of previous items (the most similar ones), the system knows the user's interest in similar items through the interest attributes (solution in CBR terminology). Assuming that the user's interest in a new item is similar to the user's interest in similar items, in the reuse phase, the recommender agent calculates an interest confidence value for the new item. This value is used to decide whether to recommend the new item to the user.

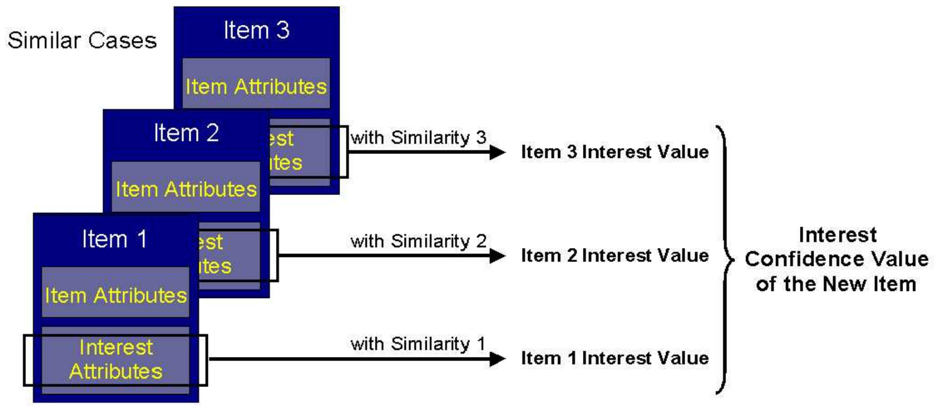

Fig. 3. Reuse Phase

The interest confidence value is a composite of the item interest value of the similar items selected in the retrieve phase (see Figure 3). We calculate it in a two-step process.

First, the item interest value V of each case is computed based on its interest attributes (see Equation 7). Given the following case representation:

Case/ Item	**Problem/ Item Attributes**	**Solution / Interest Attributes**		
		Explicit	Implicit	Drift Attribute
p	$p_1, p_2, ..., p_n$	$p^e_1, p^e_2, ..., p^e_{ne}$	$p^i_1, p^i_2, ..., p^i_{n_i}$	δ_p
q	$q_1, q_2, ..., q_n$	$q^e_1, q^e_2, ..., q^e_{qe}$	$q^i_1, q^i_2, ..., q^i_{n_i}$	δ_q

where p and q are items, $p_1, p_2, ..., p_n$ and $q_1, q_2, ..., q_n$ are the item attributes, $p^e_1, p^e_2, ..., p^e_{ne}$ and $q^e_1, q^e_2, ..., q^e_{ne}$ are explicit attributes, $p^i_1, p^i_2, ..., p^i_{n_i}$ and $q^i_1, q^i_2, ..., q^i_{ni}$ are implicit attributes and da_p and da_q are drift attributes; the item interest value of the item p is calculated as follows:

$$V_p = \delta_p * g(f^e(p^e_1, ..., p^e_{n_e}), f^i(p^i_1, ..., p^i_{n_i})) \tag{7}$$

where f^e is the function that combines the explicit interest, f^i is the function that combines the implicit attributes, g is the function that combines the results of f^e and f^i, and finally δ_p is the drift attribute (the temporal parameter related to the relevance of the product explained above). Aggregation techniques like [23] can be used for implementing f^e and f^i. We use the OWA operator for both f^e and f^i (see equation 5), applying the same procedure to obtain the weights that we showed in equation 6.

Finally, function g is a weighted arithmetic average (WA):

$$g(e, i) = \alpha_e * e + \alpha_i * i \tag{8}$$

The explicit attributes are the most relevant, since the user stated his/her opinion. Therefore, g gives more importance to explicit attributes (objective ones) than to the implicit ones (subjective). For instance, we use $\alpha_e = 0.7$ and $\alpha_i = 0.3$.

Second, the interest confidence value I of a new item r is a weighted ponderation function of the item interest value of each similar item:

$$I_r = \frac{\sum_{i=1}^{x}(Sim(r,i) * V_i)}{\sum_{i=1}^{x} Sim(r,i)} \tag{9}$$

where x is the number of similar items, Sim_i is the similarity between the item r and the item i (computed in the retrieve phase) and V_i is the item interest value of the item i. In this way, the most similar items are the most relevant in the final result.

Finally, if the interest confidence value of the new item is greater than a certain value (a confidence threshold), the item is recommended to the user. Otherwise, in the prompted role, the agent provides negative advice to the user regarding the queried item, while on the proactive role, the agent ignores it, the CBR cycle finalizes and there is no recommendation to the user. The item is not interesting enough to the user and the agent should not bother him/her with it.

6 The Revision Phase

Typically, the revision phase consists of evaluating the case solution generated by the reuse phase and learning about it. If the result is successful, then the system learns from the success (case retainment), otherwise it is necessary to repair the case solution using domain-specific knowledge. With regard to our approach, in the revision phase, as in the case of the relevance feedback of the user, the system is able to evaluate the user's interest in the recommended item. The idea is to track the user interaction by filling in the interest attributes of the item (case).

As shown in Figure 1, the interest attributes are distributed in two main groups: implicit and explicit attributes. Obviously, implicit attributes come from implicit feedback from the user, and explicit attributes come from explicit feedback. The idea is to find out the user's interest based on a hybrid relevance feedback system [17]. The user is explicitly asked about the new item, but, taking into account that users are very reluctant to give explicit feedback [5], the system tracks the user interaction with the system and tries to include additional information.

In CBR systems the solution is successful or wrong. When the solution is successful, the system retains the case inserting it into the case base. But when the solution fails, the system is interested in retaining the reason for the failure as well as the good solution, thus, there is an investigation task to find out additional information about the case. In the recommendation field, the user's interest can also be positive or negative, but, oppositely to the previous situation, the system is interested in retaining both positive and negative feedback. It is equally important to keep positive and negative information about the interests of the user, since it is useful to know what the user really "loves" and what the user really "hates". Thus, in this approach, in contrast to what usually happens in CBR systems, there is no investigation task to find out why the user is not interested in the new item, we simply retain the item as not interesting to the user. Therefore, in avoiding the investigation task, typically accomplished by a human expert, we get a completely automatic system.

7 The Retain Phase

In our approach, the new item is inserted into the case base with the interest attributes that were added in the revision phase. But, if the user did not give either explicit or implicit feedback, the item has no interest attributes and the case is not introduced into the case base. Only items with positive or negative interest are retained.

Moreover, when the user gives explicit or implicit feedback about an existing item in the case base, the case is updated. For example, if the user consults the web page of an item, the interest attribute representing the number of visits to the web page, and the attribute representing the time spent looking at the web page, are increased.

In order to control the case base size, it is also important to know whether the user ever gives new feedback about items in the case base. If not, it is necessary to forget these interests with time. This problem is solved with the drift attribute explained in section 3.2.

8 Related Work

A few research groups are investigating the application of CBR concepts and techniques to recommendation. Cunningham et al apply retrieval and adaptation techniques from CBR to intelligent product recommendation agents, particularly to their WebSell system [6]. As in our approach, the core of such application is a product database that describes the specific features of each available product. As a user profile, Websell keeps two selection lists of products which are used for collaborative recommendation service. These selection lists contain both interesting and not interesting products that, as in our approach, are used as a case base to pro-actively recommend new products to the user. Besides the lists, user profiles store more information (e.g., personal information, domain preferences), which can be compared to our approach (explicit information). The main difference is that we explicitly keep a set of interest attributes as a solution for each case, instead of general domain preferences deduced from the lists as WebSell does.

All "FindMe" systems [4] implement a similar CBR. They contain a database from which they retrieve items that meet certain constraints, and they rank the retrieved results by some criteria. For instance, the restaurant recommender Entree [3] makes its recommendations by finding restaurants in a new city similar to restaurants the user knows and likes. The main difference with our system is that Burke et al do not use an interest extensive representation of items in the case base as we do.

The Broadway approach [10, 24] also combines explicit and implicit information on cases by means of the use of a behavior summary of the user interaction. Similarly, we keep a detailed information of the user interests and, then, in the reuse phase, we aggregate such information. Moreover, they use interests in retrieval and adaptation while we are only using interests attributes in adaptation.

Adaptive Place Advisor [9] introduces an innovative approach to the retrieval phase based on diversity, which controls the repetition of the same recommendation to a user. Thus, they avoid recommending the same restaurant in a short period of time to the user. In our view, this approach, although innovative, could lead to somewhat suspicious behavior of a system which, with the same inputs, gives different outputs in successive moments, one of which must be untruthful or untrustworthy.

Another important difference between these systems and ours is that they do not handle the change of the user interests over time and, therefore, they do not take into account the maintenance of the case base . Other approaches, outside recommender systems, such as [13, 15] deal seriously with the problem of case maintenance since it is the principal drawback of CBR. However, we think that the metrics proposed are far from being useful in open domains as recommender agents. The alternative we propose is the drift attribute, which tackles the problem of maintenance in a way which is as natural as forgetting is for humans. Using the drift attribute is a new idea that we have introduced into the CBR, as applied to recommendation, thus, there is no previous work on this concept. However, this idea is based on the gradual forgetting function that some researchers have applied to other CBR systems. This concept was introduced by [26] and later applied in systems such as SiteIF [22] or LaboUr [21].

9 Conclusions

We have presented a new approach to recommendation based on CBR. The first contribution of this approach concerns case representation. Each case of the case base consists of a first set of attributes describing the item (as a problem definition) and a second set of attributes describing the user's interest (as a solution to a problem). Assuming that the user has a similar opinion about similar items, we can use the interest attributes of old cases to guess whether a new item will be of interest to the user.

The second contribution concerns the similarity metric. The relevance of attributes is computed every retrieval time and is based on information stored in cases. Such an approach reduces the need for maintenance of similarity metrics as well as the fact that no additional learning method is required to acquire the relevance.

Another contribution of this approach is the drift attribute. The drift attribute represents the age of the case in the case base and lets the personal agent distinguish between current and old interests. Consequently, the newest cases will be more confidently recommended than the oldest ones. However, the most important advantage of applying the drift attribute is that the utility problem is mitigated. Drift cases are deleted and the number of cases in the case base becomes stable while the performance of the system is maintained.

We have provided a detailed design of our system. Currently we have a prototype with some of the features implemented. Obviously we need to perform evaluations of the system. We plan to do this in the restaurants domain, based on our previous work [25].

Acknowledgments

This research has been developed within the DAF-DAF Project supported by the CICYT grant DPI2001-2094-C03-01.

References

1. D. Billsus and M. J. Pazzani. A hybrid user model for news classification. In *Kay J. (ed.), UM99 User Modeling - Proceedings of the Seventh International Conference*, pages 99–108. Wien, New York: Springer-Verlag, 1999.
2. R. Burke. A case-based reasoning approach to collaborative filtering. In *E. Blanzieri and L. Portinale (eds.), Advances in Case-Based Reasoning (5th European Workshop, EWCBR 2000)*, pages 370–379. Springer Verlag, New York, 2000.
3. R. Burke. Semantic ratings and heuristic similarity for collaborative filtering. In *AAAI Workshop on Knowledge-based Electronic Markets 2000 (KBEM'00)*. Austin, TX, 2000.
4. R. Burke, K. Hammond, and B. Young. The findme approach to assisted browsing. In *IEEE Expert*, volume 12(4), pages 32–40, 1997.
5. J. Carroll and M. B. Rosson. The paradox of the active user. In *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*, pages 26–28. J. M. Carroll (Ed.), Cambridge, MA: MIT Press, 1987.
6. P. Cunningham, R. Bergmann, S. Schmitt, R. Traphoner, S. Breen, and B. Smyth. Websell: Intelligent sales assistants for the world wide web. In *e-Business and e-Work 2001 (e-2001)*, 2001.
7. G. Falkman. Similarity measures for structured representations: A definitional approach. In *Blanzieri, E. and Portinale, L., eds., Advances in Case-Based Reasoning. Proceedings of the 5th European Workshop, EWCBR 2000, Trento, Italy*, volume 1898, pages 380–392. Springer-Verlag, Berlin, 2000.
8. A. G. Francis and A. Ram. The utility problem in case-based reasoning. In *CaseBased Reasoning: Papers from the 1993 Workshop. AAAI Press (WS-93-01). Washington*, 1993.
9. M. H. Goker and C. A. Thompson. The adaptive place advisor: A conversational recommendation system. In *Proceedings of the 8th German Workshop on Case Based Reasoning, Lammerbuckel, Germany.*, 2000.
10. M. Jaczynski and B. Trousse. Www assisted browsing by reusing past navigations of a group of users. In *Advanced in Case-based Reaosning, 4th European Workshop on Case-Based Reasoning*, volume 1488, pages 160–171. Lecture Notes in Artificial Intelligence, 1998.

11. D. Kibler and D. W. Aha. Case-based classification. In *Proceedings of the Case-Based Reasoning Workshop at AAAI'88*, pages 62–67, 1988.
12. D. B. Leake and D. C. Wilson. Categorizing case-base maintenance: Dimensions and directions. In *Advances in Case-Based Reasoning: 4th European Workshop, EWCBR-98, Dublin (Ireland)*, 1998.
13. D. B. Leake and D. C. Wilson. Guiding case-base maintenance: Competence and performance. In *Proceedings of the 14th European Conference on Artificial Intelligence Workshop on Flexible Strategies for Maintaining Knowledge Containers*, 2000.
14. M. A. Maloof and R. S. Michalski. Selecting examples for partial memory learning. In *Machine Learning*, volume 41, pages 27–52, 2000.
15. E. McKenna and B. Smyth. A competence model for case-based reasoning. In *9th Irish Conference on Artificial Intelligence and Cognitive Science. Dublin, Ireland*, 1998.
16. T. Mitchell, R. Caruana, D. Freitag, J. McDermott, and D. Zabowski. Experience with a learning personal assistant. In *Communications of the ACM*, volume 37:7, pages 81–91, 1994.
17. M. Montaner. A taxonomy of personalized agents on the internet. In *Technical Report, TR-2001-05-1, Departament d'Electrònica, Informàtica i Automàtica. Universitat de Girona*, 2001.
18. S. Ontañon and E. Plaza. Collaboration policies for case-based reasoning agents. In *Proc. Workshop on Learning Agents Autonomous (Agents'2001). Montreal*, 2001.
19. P. Resnick and H. Varian. Recommender systems. In *Communications of the ACM*, pages 56–58, 1997.
20. R. Sanguesa, U. Cortés, and B. Faltings. W9. workshop on recommender systems. In *Autonomous Agents, 2000. Barcelona, Spain*, 2000.
21. I. Schwab, A. Kobsa, and I. Koychev. Learning user's interests through positive examples using content analysis and collaborative filtering, 2001. Submitted.
22. A. Stefani and C. Strappavara. Personalizing access to web sites: The siteif project. In *Proc. 2nd Workshop on Adaptive Hypertext and Hypermedia HYPERTEXT'98*, 1998.
23. V. Torra. On the integration of numerical information: from the arithmetic mean to fuzzy integrals. In *Torra V. (Ed). Information fusion in data mining. Physiin-Verlag. (Forthcoming)*, 2001.
24. B. Trousse, M. Jaczynski, and R. Kanawati. Using user behavior similarity for recommendation computation: The broadway approach. In *Proceedings of the 8th international conference on human computer interaction (HCI'99), Munich*, 1999.
25. R. Vilà and M. Montaner. Implementació d'un sistema multiagent distribuit format per agents personals que recomanen restaurants aplicant raonament basat en casos i tècniques de trust, May, 2002. Projecte Fi de Carrera en Enginyeria Informàtica, Universitat de Girona.
26. G. Webb and M. Kuzmycz. Feature based modelling: A methodology for producing coherent, consistent, dynamically changing models of agents' competencies. In *User Modelling and User-Adapted Interaction*, volume 5, pages 117–150, 1996.
27. D. R. Wilson and T. R. Martinez. Improved heterogeneous distance functions. In *Journal of Articial Intelligence Research*, volume 6:1, pages 1–34, 1997.
28. R. R. Yager. On ordered weighted averaging aggregation operators in multi-criteria decision making. In *IEEE Transactions on SMC*, volume 18, pages 183–190, 1988.