PERGAMON

Expert Systems with Applications 22 (2002) 275–284

Expert Systems with Applications

www.elsevier.com/locate/eswa

Intelligent agent-based systems for personalized recommendations in Internet commerce

Wei-Po Lee*, Chih-Hung Liu, Cheng-Che Lu

Department of Management Information Systems, National Pingtung University of Science and Technology, Nei-Pu Hsiang, Pingtung, Taiwan, ROC

Abstract

The prosperity of electronic commerce has changed the traditional trading behaviors and more and more people are willing to conduct Internet shopping. However, the exponentially increasing information provided by the Internet enterprises causes the problem of overloaded information, and this inevitably reduces the customer's satisfaction and loyalty. One way to overcome such a problem is to build personalized recommender systems to retrieve product information that really interests the customers. For products that people may purchase relatively often, such as books and CDs, recommender systems can be built to reason about a customer's personal preferences from his purchasing history and then provide the most appropriate information services to meet his needs. On the other hand, for those commodities a general customer does not buy frequently, for example computers and home theater systems, more appropriate are the kinds of recommender systems able to retrieve optimal products based on the customer's current preferences obtained from the iterative system–customer interactions. This paper presents the above two kinds of recommender systems we have developed for supporting Internet commerce. Experimental results show the promise of our systems. © 2002 Elsevier Science Ltd. All rights reserved.

Keywords: Personalized information services; Agent-based systems; Web intelligence; Intelligent Internet applications

1. Introduction

In the recent years, the on-going advances of Internet and Web technologies have promoted the development of electronic commerce. Enterprises have been developing new business portals and providing large amount of product information in order to expand their markets and create more business opportunities. However, the exponentially increasing information along with the rapid expansion of the business Web sites causes the problem of information overload. Because of the prodigious amount of information provided by different sites, consumers have to spend more and more time to browse the net in order to find the information needed. Sometimes the contents of the Web pages are irrelevant to the consumer's expectation, but the consumer is helpless but has to read them in order to filter them out to get what he or she really wants.

One way to overcome the above problem is to develop intelligent recommender systems to provide personalized information services (Schafer, Konstan, & Riedl, 2001): retrieving the information a consumer desires and helping him determine which one to buy. Depending on the type of the product, different kinds of personalized recommender systems can be built to guide the consumers in a large product feature space. For some type of product that a consumer may purchase frequently, such as books, CDs, or DVD films, recommender systems can be developed to reason his personal preferences by analyzing his personal information, his browsing history, and the products he has purchased through the Internet in the past. Yet, for commodities such as computers or home theater systems that a general consumer does not buy very often as the other kinds just mentioned, it is difficult and not necessary to reason a customer's previous preferences because there may not be enough information about the customer's past purchases and the customer may have his specific requirements in each single purchase. In this situation, advises from domain experts are strongly demanded. Recommender systems are thus expected to have specific domain knowledge and capability to interact with the consumer. Consequently the systems can acquire and analyze a customer's current needs on the kind of product he has identified, and then evaluate the relevant products to help him recognize the optimal ones. With the personalized recommender systems, consumers can immediately access the information they are interested in, to save their time spent on reading the electronic documents. On the other hand, enterprises can get to know the customers' buying behaviors and then develop most appropriate marketing strategies to attract customers

* Corresponding author.
E-mail address: wplee@mail.npust.edu.tw (W.-P. Lee).

of specific types and efficiently deliver the information that is of interest to them. The customer's satisfaction and loyalty can thus be increased, and the increase in the visiting frequency of the customers can further create more transaction opportunities and benefit the Internet enterprises.

In this paper, we present two agent-based recommender systems for the two types of products described earlier. The first one focuses on learning a consumer's preferences for product recommendations and on the capacity to adapt to his change of interests. In our experiment we developed a system of this type for DVD films recommendations. The other aims to assist a consumer to navigate the product feature space in an interactive way in which the consumer has his own need in each feature dimension so that the customer can find the optimal products based on his personal preferences. We have also built a system of this kind for notebook computer recommendations. The experimental results show that both systems can give sensible recommendations, and are able to adapt to the most up-to-date preferences for the customers.

2. Intelligent personalized recommender systems

A recommender system can provide personalized information services in different ways; it depends on whether the system has been recording and analyzing a customer's previous preferences. In the first type of personalized recommender systems, a customer's personal information is first collected, then the system reasons about the customer's preferences by analyzing and modeling the personal information available. The ensuing user profile is recorded by the system for further product recommendations. To collect information the system can ask a consumer to manually express his interests by giving some positive/negative examples (or to rank some products in response to the inquiries of the system), or alternatively the system can implicitly observe and record the consumer's browsing behavior, and then analyze the content the consumer has read in order to build a model to distinguish between what he likes and what he does not.

Once the consumer's personal information is obtained, the recommender system can then construct a computational model for him to predicate his preferences for other items of the same application domain. Therefore, developing a high performance learning mechanism to precisely model the consumer's preferences is one of the main issues in constructing personalized recommender systems of this type. In fact, the work of recommendation can be regarded as classification: using the information already known to build a model to predict the events unknown. The model may be a set of rules, a decision tree, or a set of numerical numbers representing the corresponding weights for some specific terms.

Some learning approaches have been applied to construct customers' profiles. In Balabanovic and Shoham (1997), a statistic-based approach *tfidf* (term frequency and inverse document frequency (Salton, 1989)) was used to build a user's profile to recommend Web pages. In Joachims, Freitag, and Mitchell (1997) a reinforcement learning method was employed for Web pages recommendations, and book recommendations in Mooney and Roy (2000). In addition, a *k-nearest neighbor* (*k*-NN) method was used in Breese, Heckerman, and Kadie (1998) and Good et al. (1999) to find other customers who share the similar interests for a specific consumer. Consequently the system can recommend products to this consumer according to the evaluations from those who have similar tastes. In this work, we use an evolutionary approach to construct a learning agent to model a consumer's interests for DVD film recommendations, and show that our approach outperforms the traditional *k*-NN method.

Unlike the above systems concerned about a consumer's previous preferences, the other type of personalized recommender systems do not recognize a consumer. They are designed for the products that a consumer generally does not buy often and has his specific needs in each single purchase. In addition, a consumer needs specific domain knowledge to evaluate the corresponding quality for products of this kind. Therefore instead of modeling a customer's past preferences, the recommender systems use the ephemeral information provided by a consumer at the time he is consulting the system for suggestions, and the built-in expert knowledge about the products to look for the optimal ones. Recommender systems of this type aim to assist a customer to find out what he really wants, when he can simply identify the type of product he needs and describe the features or specific functions of the product. Initially, the recommender system retrieves some products from the database, by measuring the similarity between the products in the database and the target one described in terms of some features. Then the consumer can increase or decrease his degree of needs on certain features of the product recommended, and ask the system to suggest new items according to the modified needs. In this way the consumer can gradually find out the best product that meets his needs under the guidance of the recommender system.

As can be observed, the key issue in developing recommender systems of this kind lies in the estimation of product similarity. The case-based reasoning approach is usually taken to measure the similarity by calculating the weighted sum of different product features (Wilke, 1997). However, in most cases, a pre-defined overall weight vector of features is not feasible, as the consumer will attach different significance to product features depending on his preferences. A more flexible approach that allows the weight vector to dynamically change in response to the consumer's needs is required. Burke, Hammond, and Young (1997) and Sen and Hernandez (2000) are examples of recommender systems that are built based on such approach.

Though the case-based reasoning method can find out the products most similar to the one a customer specifies, it

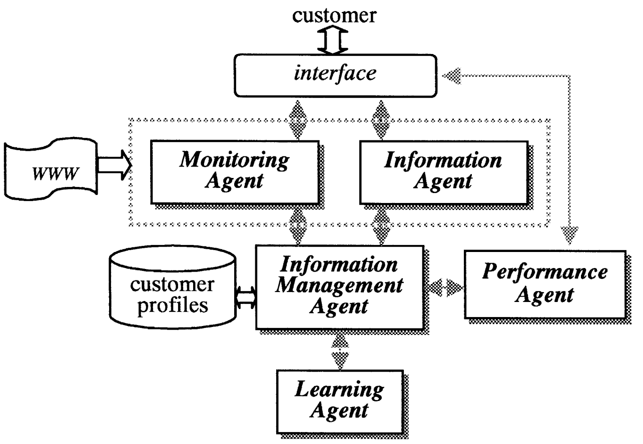


Fig. 1. The architecture of the first type of personalized recommender system.

ignores the optimality of the product. This may cause the problem that low quality products will be recommended to the customer, due to his lacking of specialized knowledge on the product and thus gives some inappropriate specifications. One way to remedy this problem is to adopt multi-attribute decision-making methods to simultaneously consider the customer's needs and the quality of the product. In the second system presented in this work, we take such a design principle to evaluate and recommend notebook computers.

3. An agent-based system for DVD films recommendations

3.1. System architecture

The major tasks of the first type of personalized recommender system include collecting a customer's personal interests, building a model to describe the information

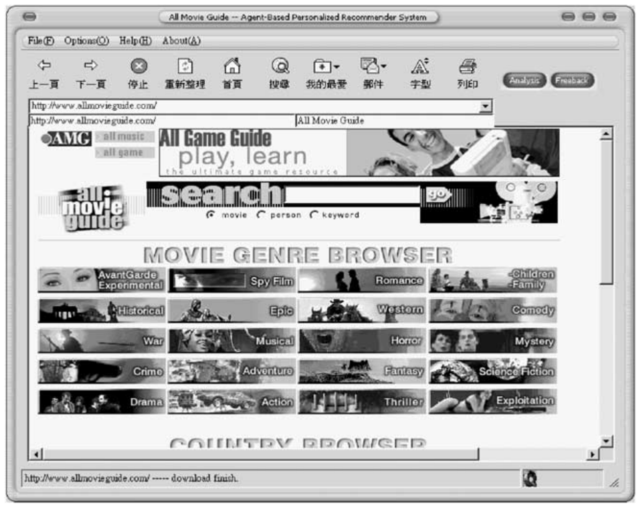
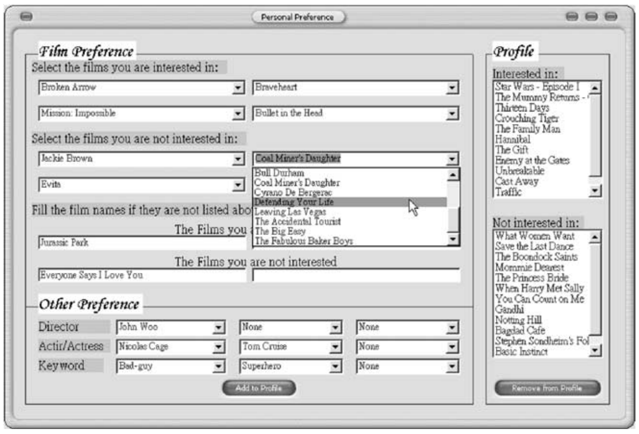


Fig. 2. Information being gathered by the monitoring agent.


Personal Preference

Film Preference

Select the films you are interested in:

Broken Arrow	Braveheart
Mission: Impossible	Bullet in the Head

Select the films you are not interested in:

Jackie Brown	Coal Miner's Daughter
Evita	

Bull Durham
Coal Miner's Daughter
Cyrano De Bergerac
Defending Your Life
Leaving Las Vegas
The Accidental Tourist
The Big Easy
The Fabulous Baker Boys

Fill the film names if they are not listed abo

The Films you

Jurassic Park

The Films you are not interested

Everyone Says I Love You

Other Preference

Director	John Woo	None	None
Actir/Actress	Nicolas Cage	Tom Cruise	None
Keyword	Bad-guy	Superhero	None

Add to Profile

Profile

Interested in:
Star Wars - Episode I
The Mummy Returns -
Thirteen Days
Crouching Tiger
The Family Man
Hannibal
The Gift
Enemy at the Gates
Unbreakable
Cast Away
Traffic

Not interested in:
What Women Want
Save the Last Dance
The Boondock Saints
Mommie Dearest
The Princess Bride
When Harry Met Sally
You Can Count on Me
Gandhi
Notting Hill
Bagdad Cafe
Stephen Sondheim's Fol
Basic Instinct

Remove from Profile

Fig. 3. Information being gathered by the information agent.

collected, managing a customer's personal information, and tracking the customer's feedback. In our work, an agent-based methodology is adopted in which each agent autonomously performs a specific task and they work simultaneously to achieve the overall task. Fig. 1 shows the system architecture.

In Fig. 1 the *monitoring agent* provides a browser-like environment in the front end for user's browsing; and it records the contents the customer has read, and categorizes them into different classes in the back end. Currently, our monitoring agent links the user to the on-line movie site *allmovie* (http://www.allmovieguide.com) and records his browsing behavior. Fig. 2 shows the environment. To recognize how a customer is interested in the product presented in a certain page, the monitoring agent measures the time he stays to view that page and a simple wrapper is designed to extract the information needed for constructing the personal profile. In addition to collecting information in the above implicit way, the *information agent* presents the customer a form that allows him to explicitly point out his interests. Fig. 3 shows the form. The above two agents both play the role of information collectors; a customer can activate them through the interface to gather his own preferences about the products.

The *information management agent* takes the responsibility of establishing and maintaining a personal profile for each customer in which a profile mainly contains two parts: a personalized prediction model and a set of product items with numerical values quantitatively indicating how much a customer likes them. This agent also analyzes and transfers each collected product into a feature (or attribute) vector in which the features are pre-defined by the system to best represent that type of product. A customer's personal profile will be updated when new information arrives or the personal model is re-built (due to the change of a consumer's preferences). Since the customers' profiles are maintained by the information management agent, thus the only way for other agents to access the information recorded in the consumers' profiles is through the intermediary of this agent.

The *learning agent* is the kernel of our recommender

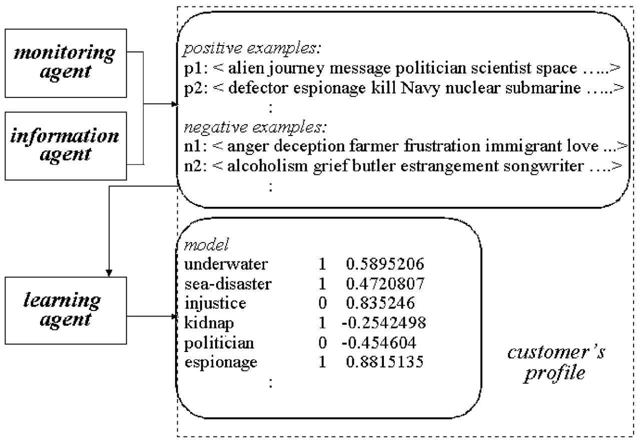


Fig. 4. The aspect of a typical customer's profile and its functional relationships with different agents. In the profile, the upper part includes the training examples and the lower part, the evolved model.

system; it builds the model of prediction for a customer, based on the information recorded in his profile. As is mentioned, this model can in fact be considered as a classifier that recognizes an unfamiliar item as one of the categories representing different degrees of preferences on the products. The most recent feature vectors of products collected and recorded in the profile are used as training examples to construct the classifiers. In our current implementation, the learning agent acts in a passive way. It is activated by the information management agent whenever necessary-at the time when a new customer registers to the system or when the performance of the recommender system declines due to the changes in a customer's preferences. An evolutionary approach is employed in our work to learn a customer's preferences and the details are described in a later section.

In contrast to the learning agent, the performance agent operates in a more active way: it keeps acting ever since a customer enters the system. With the learnt customer model, this agent can decide whether to recommend a specific product item to the corresponding customer or not. As a customer's interests may change occasionally, the system must continuously track his behavior and feedback in order to adapt to his most recent preferences. The performance agent is thus responsible for tracking the recommending performance and reporting the result to the information management agent. Based on the result, the information management agent can then determine if it is necessary to evoke the learning agent to establish a new model for this customer again.

3.2. The customer's profile

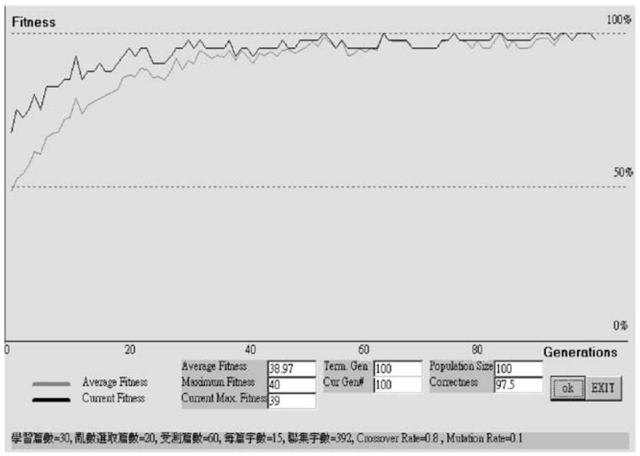
The above section has described the design principles and the general architecture of our system; in this section, we illustrate the personal profile used in this work. A DVD film can be described by features such as *genres*, directors, actors, and so on; here we choose the plot of a DVD film as the representative feature because it is often the most decisive factor a general audience used to determine whether he likes a specific film or not. In the preliminary experiments, we used the introductory note of a film as its corresponding plot, and employed the traditional text analysis approach (i.e. tfidf) to extract words to constitute a feature vector for a DVD film, as our previous work in document categorization (Liu et al., 2000). We have found that, however, the introductory note is generally too short to extract representative terms. Therefore we decide to directly use the keywords associated with a film (available from the on-line film database) to represent its plot and to be the product features. For example, the film *Jurassic Park* is represented as ⟨attack, clone, dinosaur, experiment, genetic,…⟩ in our system. The keywords appearing in the training examples are defined as product features and used to constitute a chromosome in our GA mechanism.

For simplicity, in our current implementation the system only recognizes whether a customer is interested in a specific DVD item or not, rather than the degree in which he favors the item. Both implicit and explicit ways for information collection are provided and the product examples collected are maintained in a customer's profile for later training. Fig. 4 shows the aspect of the customer's profile and its functional relationships with different agents.

3.3. Learning customer's preferences by GAs

An individual (chromosome) in our system is defined to include two strings of numerical values. The first string indicates which product features (in our case, keywords) should be taken into account in modeling a customer's preferences; it is represented as $\langle a_1, a_2, \ldots, a_n\rangle$ in which each gene a_i is an integer of 0 or 1. The second string means the relative importance of each feature listed in the first string; it is represented as $\langle b_1, b_2, \ldots, b_n\rangle$ in which each gene b_i is a floating point number between -1 and 1 as the associated weight for the corresponding feature term. As the learning agent will build a customer's model from the products collected, the string length n is thus the number of distinguishable feature appearing in the training examples. In the above structure the activation of each parameter/weight gene b_i is governed by a control gene a_i and the inactive genes always exit in the chromosome. In this way, the procedure of feature subset selection is also integrated into the overall evolutionary process. In our GA mechanism, the operations dealing with binary and floating point numbers (as described in Michalewicz (1994)) are applied independently to the two different strings.

With the above structure, to predict how a customer likes a certain product P, we first transfer the product into a n-dimension vector to match the feature description defined earlier, and then calculate the weighted sum of the product



Average Fitness 38.97
Maximum Fitness 40
Current Max. Fitness 39
Term. Gen 100
Cur Gen# 100
Population Size 100
Correctness 97.5

學習篇數=30, 亂數選取篇數=20, 受測篇數=60, 每篇字數=15, 聯集字數=392, Crossover Rate=0.8 , Mutation Rate=0.1

Fig. 5. A typical experiment run shows the converging behavior of the learning agent.

and the customer's model as:

$$\sum_{i=1}^{n} p_i \times a_i \times b_i$$

where p_i is the ith component in the binary vector form of a product P and it indicates whether the ith product feature appears in P; a_i and b_i the ith components of the string classifier defined earlier; and n is the length of the classifier. The possible range of the above weighted sum can be divided into multiple sub-intervals in which each sub-interval represents a certain degree of preference, and then the degree a customer likes a specific product can be measured by examining which sub-interval a weighted sum lies in. The reason why we use multiple discrete intervals here rather than a continuous one is to provide a simple way for a consumer to indicate his preference on a product. For instance, we can divide the overall range into five sub-intervals as increasing (or decreasing) scales of preference for the product (i.e. 1–5), and then request a customer to express how he feels about a product by choosing a scale value.

In the learning phase, the learning agent uses the products recorded in the customer's profile as training examples, and uses an evolutionary mechanism to select the appropriate features and to determine their corresponding weights as a consumer's model of preferences. During the evolutionary process, each individual (model) is used to predict how a customer likes the training examples by the weighted sum calculation already described above. For each training example the difference between the actual interval produced by the model and the desired interval pre-indicated by the customer is defined as the error of prediction, and its complement as the accuracy. Therefore the average accuracy over all training cases can be defined as the fitness of an individual. The individual evolved is then recorded in the consumer's profile and utilized for further product recommendations.

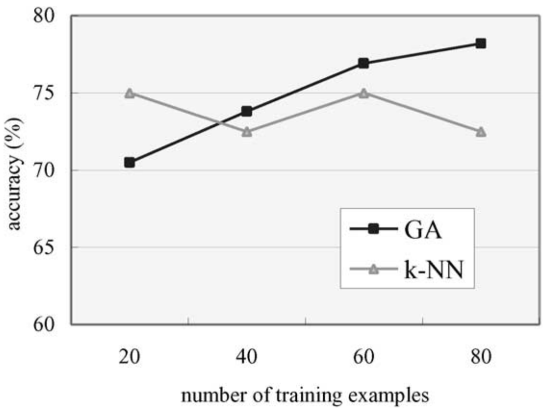


Fig. 6. The comparison of the GA and k-NN methods; values are averaged over 10 experiment runs.

3.4. Experiments and results

In order to assess the approach proposed and the system developed, this section describes the experiments conducted. In the experiments, we concentrate on evaluating the learning and performance agents. For the learning agent, we first investigate how a prediction model can be evolved by our GA mechanism, and then conduct a series of experiments to compare our GA approach with the well-known k-NN method. For the performance agent, we analyze how it monitors the system's performance that varies in response to the changes of a user's interests, and how it can adapt to a customer's most recent preferences.

3.4.1. Evaluation of the learning agent

To evaluate the performance of our learning agent, we implement a GA mechanism and use it to evolve models of preferences for consumers. In the training phase, 60 examples (the films recorded in the consumers' profiles) were used: a half of them were positive examples and the other half, negative. Here, we used a threshold value T to distinguish products: if the weighted sum calculated by the criteria described in Section 3.3 is higher than T, then the corresponding product is predicted as customer-like; if the weighted sum is lower than $-T$ it is customer-dislike. The proportion of the training examples predicted correctly (the cases with weighted sums between T and $-T$ are considered as wrongly predicted) is defined as the fitness of the specific model evaluated. Fig. 5 is a typical experiment (with $T = 0$) showing how the evolutionary system converged. The upper curve in Fig. 5 shows the accuracy of the best individual evolved in each generation; and the lower curve, the average accuracy of all population members. It indicates that an accurate solution can be evolved successfully.

After the learning, each evolved model was tested by another 40 cases not contained in the training set for further evaluations. Fig. 6 presents the results obtained from different numbers of training examples in which each data point is

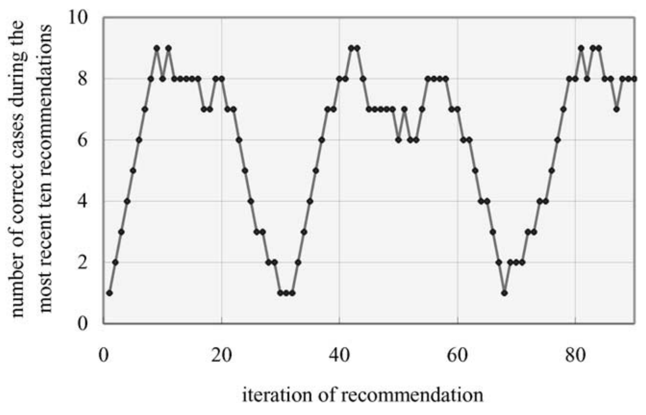


Fig. 7. An example illustrates how the system performance changes during the 90 consecutive recommendations.

averaged from 10 independent experiments. For the GA curve, the threshold value T was 3 in training, but 0 in testing. We have experienced different values and found that keeping two classes of items away by a certain distance can effectively reduce the effect of overfitting. Without the constraint, sensitive classifier could be produced through learning. The test results shown in Fig. 6 may be satisfactory as the numbers of training examples used were relatively small and thus not enough to cover all the customer-like items that distribute in different genres. Though using sparse samples to reason about a customer's preferences in a large product space could result in a relatively loose user model, it is nevertheless a more realistic situation.

In order to examine the performance of the evolved models further, we implemented the well-known k-NN method, which was shown to give the best performance in Pazzani's work on Web page recommendations (Pazzani & Billsus, 1997) for comparison. Three different values of k (1, 3, and 5) were used, and for each k different numbers of reference examples were provided as in the previous GA experiments. Fig. 6 shows the best results (with $k = 3$). As can be seen, in our recommendation cases, GA with separation distance gives better results than the k-NN method.

3.4.2. Analysis of the performance agent

As previously mentioned, in this system a customer's personal model will be updated if the corresponding recommendation performance declines. In this situation, when five wrong cases occur during the most recent 10 recommendations, or three consecutive wrong cases appear in our current implementation, the customer's preferences will be relearnt. The most recent 10 cases (five positive and five negative) collected will substitute 10 samples used in the previous training. Fig. 7 shows an example of how the system performance changes during the 90 consecutive recommendations for a certain customer, in which each data point indicates the number of correct cases accumulated from the most recent 10 recommendations. In this example, the customer changed his preferences into extremely different ones at the 20th (and the 60th) recommendation step (that is simulated by exchanging the positive and negative cases) and it caused the decline of system performance. Once the performance agent detected this situation, it informed the information management agent which then resumed the learning agent to build a new model. As can be observed, the new model did not improve the system performance immediately because the number of cases representing the new preferences was not enough to dominate the set of learning examples; thus the new model could not reflect the changed preferences truly. After most of the original training examples were replaced by the new ones, the model learnt could then represent the customer's current interests and the system performance was thus improved. At the 40th recommendation step the customer changed his interests toward slightly different ones. As it did not worsen the system performance so that the customer's model was not modified.

4. An agent-based system for NB recommendation

Apart from the ones described in Section 3, another type of recommender systems is demanded when a customer is going to buy products he or she generally does not often buy in a short period of time, for example computers. As analyzed in Section 2, for products of this type, previous personal buying experiences may not be helpful. What a consumer needed here is some expertise to recognize the ideal product based on his current preferences. Under such circumstances, a more appropriate way to provide personalized information services is to create an interactive environment in which a consumer can iteratively express his preferences or needs for the recommender system and the system can then use the ephemeral information provided by the customer with the built-in domain knowledge to find the ideal products as recommendations. This is similar to the scenario that a consumer really walks in a shop, communicates with the human agent who normally can provide certain knowledge for the kind of product the consumer is interested in, and asks for his suggestions in making decision. This section describes how we take an agent-based approach to design a recommender system of this kind.

4.1. System architecture

The second type of recommender system analyzes a consumer's current requirements and finds out the most ideal products for him. The ideal solutions here mean the ones best satisfying the consumer's requirements and with optimal quality at the same time. To achieve the above goal, the system we developed for notebook computer recommendations mainly includes three agents: an *interface agent* for interacting with the consumer, an *expert agent* for transferring external expert knowledge for internal use, and a *decision making agent* for calculating the optimality of each product. The overall system architecture is shown in

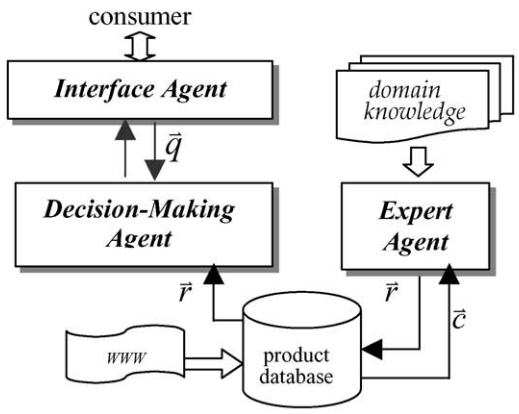


Fig. 8. The system architecture of the second type of recommender system. The explanations for vectors q, r, and c are in Section 4.2.

Fig. 8, and the same design principles can be applied to other products of this type.

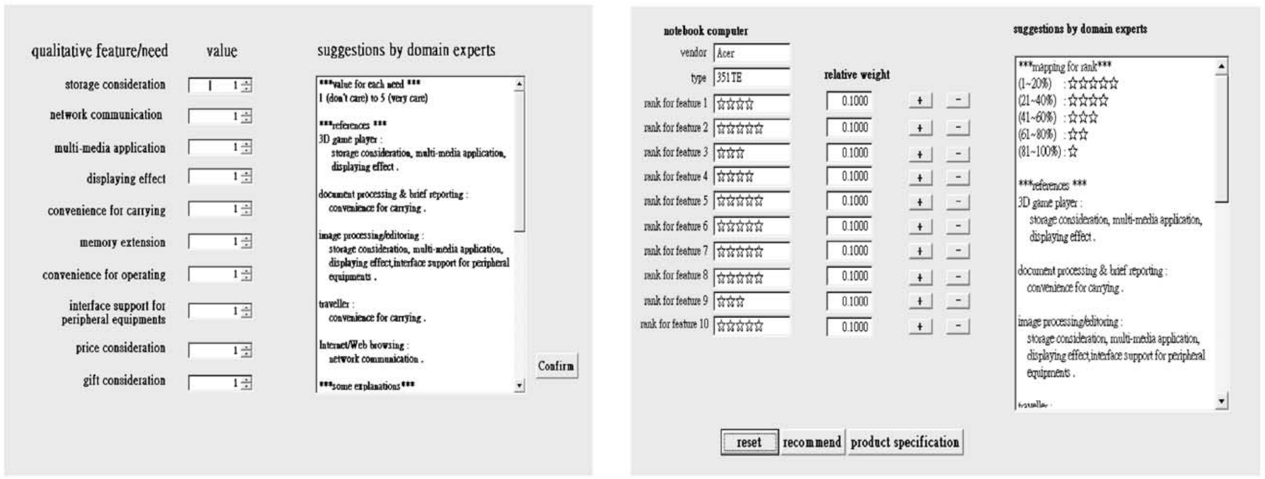
In order to collect and analyze a consumer's personal needs, the interface agent in Fig. 8 presents him some specially designed questions about the products. Presumably the consumer does not have enough domain knowledge to answer quantitative questions that concern about the specifications of the product, the system has to inquire some qualitative ones instead. For example, it is relatively difficult for an on-line game player to indicate the speed and the type of processor he prefers, but it is easy to express his need on the feature of multi-media. After gathering the consumer's qualitative needs, the interface agent can then deliver them to the decision-making agent that is capable of conducting certain mapping between the needs and the quantitative product features from the expert agent to find the ideal products. The quantitative features are derived by human experts from the specifications of the components that constitute the product. Section 4.2 will give more details on this issue.

Fig. 9 (left) shows the environment the interface agent interacts with the consumers in which a consumer is asked to express his needs (from 1 to 5) on some qualitative features about the product. As can be seen in this figure, there are also some descriptions provided to assist a consumer in indicating his needs. For instance, this agent suggests a 3D game player to give higher values on features concerning about 'multi-media', 'display', and 'storage'. Once the consumer completes this form, the system recommends some top alternatives with their corresponding feature ranks to him. Fig. 9 (right) shows the typical recommending results. If the consumer is not satisfied with the products recommended by the system, he can modify his requirements by increase/decrease his needs in different qualitative feature dimensions and the modified needs are used to find new candidates again. In this way, a consumer can gradually navigate the product space under the assistance and guidance of the recommender system to figure out what he really wants.

4.2. The expert agent and decision-making agent

In general products are specified by a set of critical components $\langle c_1, c_2,\ldots,c_n\rangle$ and different vendors have their own ways to categorize their products. For example, a computer can be described by processor, memory, monitor, etc., and the processors could be named as Pentium III or AMD K7 each with special meaning. On the other hand, some qualitative questions are designed to collect a consumer's needs represented as $\langle q_1, q_2,\ldots,q_m\rangle$. Therefore, in order to evaluate how a certain product P_i satisfies these needs, an expert agent is developed to transfer P_i from its vector form of original component specifications $\langle c_1^i, c_2^i, \ldots, c_n^i\rangle$ to the corresponding vector of qualitative features $\langle q_1^i, q_2^i, \ldots, q_m^i\rangle$ defined, so that further measurements can be performed.

In our work, the products (i.e. NB computers) are firstly



qualitative feature/need	value
storage consideration	1
network communication	1
multi-media application	1
displaying effect	1
convenience for carrying	1
memory extension	1
convenience for operating	1
interface support for peripheral equipments	1
price consideration	1
gift consideration	1

suggestions by domain experts

```
***value for each need ***
1 (don't care) to 5 (very care)

***references ***
3D game player :
   storage consideration, multi-media application,
   displaying effect .

document processing & brief reporting :
   convenience for carrying .

image processing/editoring :
   storage consideration, multi-media application,
   displaying effect,interface support for peripheral
   equipments .

traveller :
   convenience for carrying .

Internet/Web browsing :
   network communication .

***some explanations***
```

notebook computer

		relative weight		
vendor	Acer			
type	351TE			
rank for feature 1	☆☆☆☆	0.1000	+	-
rank for feature 2	☆☆☆☆☆	0.1000	+	-
rank for feature 3	☆☆☆	0.1000	+	-
rank for feature 4	☆☆☆☆	0.1000	+	-
rank for feature 5	☆☆☆☆☆	0.1000	+	-
rank for feature 6	☆☆☆☆☆	0.1000	+	-
rank for feature 7	☆☆☆☆☆	0.1000	+	-
rank for feature 8	☆☆☆☆☆	0.1000	+	-
rank for feature 9	☆☆☆	0.1000	+	-
rank for feature 10	☆☆☆☆☆	0.1000	+	-

suggestions by domain experts

```
***mapping for rank***
(1~20%)   :☆☆☆☆☆
(21~40%)  :☆☆☆☆
(41~60%)  :☆☆☆
(61~80%)  :☆☆
(81~100%) :☆

***references ***
3D game player :
   storage consideration, multi-media application,
   displaying effect .

document processing & brief reporting :
   convenience for carrying .

image processing/editoring :
   storage consideration, multi-media application,
   displaying effect,interface support for peripheral
   equipments .
```

Fig. 9. The questionnaire presented by the interface agent (left), and the result by the recommender system (right). In the right figure, the rank for each feature indicates the quality (1 for the lowest, 5 for the highest) for the product listed. The consumer can modify his needs by pressing the ' + ' (increase) or ' − ' (decrease) buttons shown.

collected from the Internet by a simple hand-coded wrapper, and a product P_i is represented as a vector of component names $\langle c_1^i, c_2^i, \ldots, c_n^i \rangle$ in the internal product database. Then each product is converted to a vector of functional features $\langle f_1, f_2, \ldots, f_k \rangle$ in which each f_i is the performance value for a certain functional feature i. The functional features here are selected by domain experts to consider the quality of the product from different views, such as the process type, the process frequency, the memory frequency, etc. It should be noted that the dimensionalities of the above vectors could be different; that is, the name of each component can be mapped into many different functional features. For example, a CPU named Pentium III 850 is mapped into two performance values indicating its quality on the features of processor type and process frequency respectively. In addition, to compare the products (or components) of different vendors, expert knowledge is required to define the common criteria. For example, we can set the performance value of Celeron type CPU to 1 and Pentium type CPU to 1.2., and so on. For the recommender system presented here, 24 different functional features are taken into account (i.e. $k = 24$) and the details can be found in Lu and Lee (2001).

As the decision-making agent in our work uses a multi-attribute decision making approach, derived from the TOPSIS (technique for order preference by similarity to ideal solution (Balabanovic & Shoham, 1997; Yoon & Hwang, 1995)), to estimate the optimality of each product for the consumer, the above functional features $f_1, f_2, \ldots, f_k$ must be further integrated to a pre-defined list of functional abilities $\langle a_1, a_2, \ldots, a_n \rangle$ (each functional ability a_i corresponds to a qualitative need defined previously and it is a quantity indicating the performance of a product in the dimension of qualitative feature i) by which the optimality of a product can thus easily be measured. That is, each product is converted from a list of quantitative features into a list of qualitative ones in this phase. The integration involves a two-stage many-to-many mapping in which the different functional features concerning about the performance of the same functionality of the product are first combined (e.g. the type and the frequency of a processor can be combined and the result represents the performance of CPU), and then the results related to the same functional ability are integrated (e.g. the functional ability of multi-media is the weighted sum of functionalities of processor and memory). In this work, the 24 functional features mentioned are integrated into 10 functional abilities in which the strategy for integration is determined by the domain knowledge and a series of empirical studies. Normalization is also performed here before combining the values from different dimensions.

Once a product P_i has been characterized as a vector of functional abilities $\langle a_1^i, a_2^i, \ldots, a_n^i \rangle$, each value a_j^i can be further transferred to a rank that represents the relative performance of the product P_i, among all the products collected, in the dimension of the functional ability j. As a result, P_i is finally represented as $\langle r_1^i, r_2^i, \ldots, r_n^i \rangle$ where each r_j^i is between 1 and 5. In this way, the multi-attribute decision making methodology can thus be employed to estimate the optimality for each product in the database.

Whenever a consumer indicates his relative needs as described in Section 4.1, the overall rank (or optimality) R of a product in the database is measured by:

$$R = \frac{R^-}{R^+ + R^-}$$

where

$$R^+ = \sqrt{\sum_{i=1}^{n} [w_i \times (r_i - r_{i(\text{best})})]^2} \qquad \text{and}$$

$$R^- = \sqrt{\sum_{i=1}^{n} [w_i \times (r_i - r_{i(\text{worst})})]^2}$$

In the above equations, n is the number of product features; r_i the normalized rank of a product in the feature dimension i, and $r_{i(\text{best})}$ and $r_{i(\text{worst})}$ are the best and worst ranks (normalized) in the same dimension, respectively; and w_i means the customer's relative need in this feature.

The above measurement is based on the principle that the selected solution should have the shortest distance to the *ideal* solution (i.e. the combination of all best ranks $r_{i(\text{best})}$) and the farthest distance from the negative-ideal one (i.e. the combination of all worst ranks $r_{i(\text{worst})}$). Once the currently available products have been ranked by the above criterion, the products with the top 10 ranks are then recommended to the customer. If the customer is not satisfied with the items recommended by the system, he can increase or decrease his requirements in different feature dimensions by pressing the plus or minus buttons associated with the features. The modified specifications are used to calculate the optimality for each product again, and those products with highest ranks are thus recommended to the customer.

4.3. Evaluations and discussions

The second type of recommender system described above is to recommend products that best satisfy the consumer's current needs and with the optimal quality. Therefore the experiments concentrate on evaluating the system behaviors; that is, we shall observe whether the overall system can response to the modifications made by the consumer in different feature dimensions. Fig. 10 shows the typical experimental results in which a consumer continuously modified his needs on four different qualitative product features at the same time. From this figure, it can be seen that the system is able to successfully adapt to the consumer's changes. But it should be noted that it is possibe that the ranks of the products recommended by the system are not able to truly reflect the consumer's changes immediately. This is mainly because of that each product here is a fixed combination of certain components. Therefore

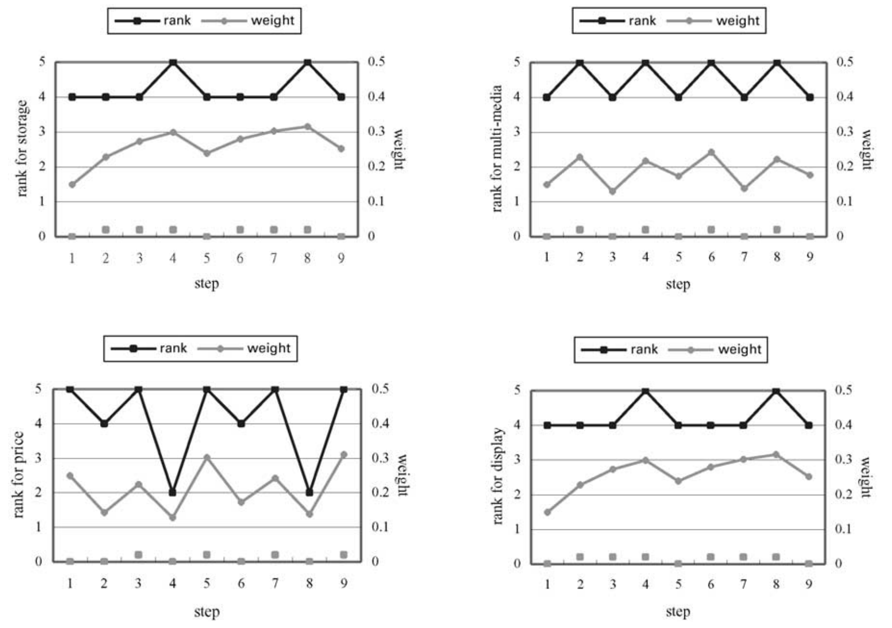


Fig. 10. The correspondence among the feature ranks of the ideal product recommended by the system, the accumulated weights derived from the consumer specified needs, and each single modification (increase/decrease, the data point along the *x*-axis) made by the consumer during the consecutive steps.

owing to the characteristics of exclusiveness or dependency between different features of the product, the increase of the need on one feature means the increase (or decrease) of the need on the other simultaneously. The multi-attribute decision making strategy employed in our work uses a global view to select the optimal products. This is different from the conventional case-based reasoning approach in which only the similarities between the product specified by the consumer and the ones available in the database are measured. Also in such a conventional approach, a consumer has to know what he needs. Our approach is more practical because most of the time while buying the second type of products, the consumer can not name what he exactly wants.

5. Conclusions and future work

In this paper, we explain the need for Internet enterprises to provide personalized information services in making a successful Internet business, in addition to developing or improving the software and hardware equipment directly related to the Internet infrastructure. We have also suggested that developing personalized recommender systems is a promising way to achieve this goal. Therefore in this work, we present two kinds of personalized recommender systems for the two kinds of products discussed as experimental examples. The first one focuses on learning a consumer's preferences and on the capacity to adapt to his changes of interests. In this system, an agent-based methodology is adopted in which each agent is responsible for a specific sub-task, including information gathering, customer modeling, information managing, and performance evaluation. An evolutionary method is employed to develop the learning agent to model a consumer's preferences. The second system concentrates on finding optimal products for a consumer by using the ephemeral information provided by him and the built-in expert knowledge. In this system, different agents are developed to interact with the consumer, transfer external domain knowledge for internal use, and calculate the optimality of each product. Here, a multi-attribute decision making method is used to recommend optimal notebook computer for a customer, based on his needs and the quality of the product. Experimental results have shown the promise of our systems.

In the experiments of learning a consumer's interests, we can observe that using sparse samples to model customer preferences could lead to results that are not robust and reliable enough. In addition to the insufficient training examples, there may be another reason: in our current implementation, a binary decision making strategy is employed and the products are therefore categorized into two extremely different classes of customer-like and customer-dislike. Under such circumstances, products that are slightly different could be dispatched to different groups and such a categorization makes no difference with that of two complete different products. A possible way to remedy the above situation is to evaluate products in terms of different levels of preferences; for example to ask the customer to rank products by a five-scale criterion. It will reduce the evaluation bias and more reliable and robust models can thus be obtained. We have been now conducting more experiments to address this issue.

Our work presented here directs to some prospects for future research. One is to explore the problem of scalability—using sparse samples to reason about customers' preferences in a large space of product items. In addition, it is worthwhile to investigate how the product knowledge from domain experts can be derived more easily. Another important issue is to examine the possibility of applying the personalization techniques to more Internet applications.

References

Balabanovic, M., & Shoham, Y. (1997). Fab: Content-based collaborative recommendation. *Communication of the ACM*, *40* (3), 66–72.

Breese, J., Heckerman, O., & Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. *Proceedings of International Conference on Uncertainty in Artificial Intelligence* (pp. 43–52).

Burke, R. D., Hammond, K. J., & Young, B. C. (1997). The FindMe approach to assisted browsing. *IEEE Expert*, *July/August*, 32–40.

Liu, J.-H., Lu, C.-C., & Lee, W.-P. (2000). Document Categorization by Genetic Algorithms. *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*. pp. 3869–3873.

Lu, C.-C., & Lee, W.-P. (2001). Developing Agents to Support Internet Commerce. *Proceedings of the Sixth Conference on Artificial Intelligence and Applications*.

Good, N., Schafer, J. B., Konstan, J. A., Borchers, A., Sarwar, B., Herlocker, J., & Riedl, A. (1999). Combining collaborative filtering with personal agents for better recommendations. *Proceedings of National Conference on Artificial Intelligence* (pp. 439–446).

Joachims, T., Freitag, D., & Mitchell, T. (1997). WebWatcher: A tour guide for the World Wide Web. *Proceedings of International Joint Conference on Artificial Intelligence*.

Michalewicz, Z. (1994). *Genetic algorithms + data structures = evolution programs*, Berlin: Springer.

Mooney, R. J., & Roy, L. (2000). Content-based book recommending using learning for text categorization. *Proceedings of the ACM International Conference on Digital Libraries* (pp. 195–204).

Pazzani, M., & Billsus, D. (1997). Learning and revising user profiles: The identification of interesting web sites. *Machine Learning*, *27*, 313–331.

Salton, G. (1989). *Automatic text processing*, Reading, MA: Addison-Wesley.

Schafer, J. B., Konstan, J., & Riedl, J. (2001). E-commerce recommendation applications. *Journal of Data Mining and Knowledge Discovery*, *January*.

Sen, S., & Hernandez, K. (2000). A buyer's agent. *Proceedings of International Conference on Autonomous Agents* (pp. 156–162).

Wilke, W. (1997). CBR and electronic commerce on the world wide web. *Proceedings of International Conference on Case-Based Reasoning*.

Yoon, K., & Hwang, C. -L. (1995). *Multiple attribute decision making: An introduction*. Thousand Oaks, CA.