Electronic Commerce Studies
Vol. 2, No.1, Spring 2004
Page 35-54

Applying Expert Knowledge and Social Information to Product Recommendations in E-Commerce

Wei-Po Lee
National Pingtung University of Science and Technology
Cheng-Che Lu
National Pingtung University of Science and Technology

Abstract

The advance of Internet and Web technologies has boosted the development of electronic commerce. More and more people have changed their traditional trading behaviors and conducted Internet shopping. However, the exponentially increasing product information provided by Internet enterprises causes the problem of information overload, and this inevitably reduces the customer's satisfaction and loyalty. To overcome this problem, in this paper we proposed an intelligent agent-based system that is capable of recommending optimal products based on the built-in knowledge and the customer's preferences obtained from the system-consumer interactions. In addition, the system also uses social information collected from previous consumers to predict what the current consumer may expect. Experiments have been conducted and the results show that our system can give sensible recommendations, and it is able to adapt to the most up-to-date preferences for the customers.

Keywords: Electronic Commerce, Intelligent Agents, Recommendations, Collaboration, Knowledge-Based Systems, Behavior Modeling

1. Introduction

In recent years, the advance of Internet and Web technologies has continuously boosted the prosperity of electronic commerce. Nowadays, companies have been digitalized to enhance their operating performance. Also the Internet enterprises have been developing new business portals and producing Internet advertisement to create more business opportunities. Yet, the advanced Internet hardware infrastructure and the entrancing Web sites are not the only decisive factors to guarantee a successful on-line business. In order to enhance the transactions, Internet companies must offer more value adding services. One popular way often employed by Internet enterprises to attract consumers is to provide prodigious amount of information about their products. However, the exponentially increasing information along with the rapid expansion of business sites causes the problem of information overload. In order to find the products that best fulfill ones personal needs, a consumer has to spend more and more time to know about the products and to survey the relevant product information for further comparison.

One way to overcome the above problem is to develop intelligent recommender systems to provide customized information services. The system can interact with the consumers to capture what they needs and can help them determine what to buy (Schaher *et al.*, 2001; Cheung *et al.*, 2001). Depending on the types of the products, different kinds of personalized recommender systems can be built to guide the consumers in a large product feature space. For the type of products that a consumer may purchase frequently, such as books or CDs, the recommender systems can be developed to reason about his personal preferences by analyzing his personal information, browsing history, and the products he purchased through the Internet in the past. Yet, for commodities such as computers or home theater systems that a general consumer does not buy so often compared to the above type of products, it is difficult and not necessary to reason a customer’s previous preferences because on the one hand there may not be enough information available about the customer’s past purchases and on the other hand, the customer may have his specific requirements in each single purchase. In addition, when the user would like to purchase products of this type, he normally has inadequate knowledge to evaluate the products. In this situation, advises from domain experts are especially in demand. Therefore, recommender systems of this kind are expected to have specific domain knowledge and play the roles of consultants to interact with the consumer. Consequently the systems can acquire and analyze a customer's current needs for the target products, and then evaluate the relevant products to help him recognize the optimal ones. By iteratively interacting with the recommender systems, consumers can save a lot of time spent on reading the electronic documents to make decisions and soon figure out the products that best fulfill his needs.

In this paper, we concentrate on the recommendations of 3C (computer, communication, and consumer electronics) products that a consumer normally does not buy often. An agent-based system is presented. This system 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. In this way the consumer can find the optimal products based on his personal preferences. We have built a prototype system for the recommendations of the fashionable 3C products, currently including notebook computers, home theater systems, digital cameras, and personal digital assistants. In our work, product knowledge is collected from the domain experts, and is embedded to the system to evaluate the quality of various kinds of products. Then the optimal products are determined and recommended to the consumer. Because the system has considered the general features of different kinds of products, the consumer does not have to specify the exact name of the product but only to describe what features and functions he emphasizes on. This is especially useful for the recommendations of today's 3C products—it is a trend to integrate multiple functions into one product. Based on the information derived from the consumer, the system can use expert knowledge to automatically find the most appropriate products for him. To save the efforts of user-system interaction, the system also uses social information to predict what the consumer may expect. The prediction is done by looking for the previous consumers with similar behavior patterns during the consultation, and recommending the products the similar consumers selected to the current user. In order to access the proposed approach, different experiments have been conducted and the results show that our system can give sensible recommendations, and it is able to adapt to the most up-to-date preferences for the customers.

2. Product Recommendations

Unlike the kind of systems mainly concerning about a consumer's previous preferences (e.g., (Basu *et al.*, 1998; Lee *et al.*, 2002), the recommender systems we investigate in this work do not recognize the individual consumers. Systems of this kind are designed to provide suggestions for the products that a consumer generally does not buy often, and he needs specific domain knowledge to evaluate the corresponding quality. For products considered here, the consumer has his specific needs in each single purchase that are normally independent from the previous ones. Therefore instead of modeling a customer's past preferences, the recommender systems look for the optimal products by using the ephemeral information that is provided by a consumer at the time he is consulting the system, and the built-in expert knowledge about the products. Recommender systems of this type aim to assist a customer to find out what he really wants, when he can simply describe the features or specific functions of the target product. Initially, the recommender system retrieves some products from the database, by measuring the similarity between the products in the database and the one described in terms of some features by the consumer. Then the consumer can increase or decrease his degrees of needs on certain features of the product recommended, and asks the system to suggest new items according to the modified needs. In this way the consumer can gradually find out the product that best meets his needs under the guidance of the 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, 1999), (Sen and Hernandez, 2000) and (Shearin and Lieberman, 2001) include typical examples of recommender systems that are based on the approach of this type.

Though the case-based reasoning method can find out the products most similar to the one a customer specifies, it ignores the optimality of the product. A problem may ensue when a consumer has not enough knowledge about the product and is complled to give some inappropriate specifications, the products recommended may not be suitable to the customer. One way to solve the above problem is to adopt multi-attribute decision making methods to simultaneously consider the customer's needs and the quality of the product. In the framework presented, we take such a design principle to evaluate and recommend products.

In the above kind of systems, the user-system interaction is an important factor in achieving optimal recommendations. During the interaction, the consumer can give more and more feedback to the system by explicitly expressing his personal opinions, and on the other hand the system can retrieve results accordingly from the databases or resources. The more concrete information a consumer provides, the higher probability of the optimal products can be found. Yet, the interactions inevitably take time. To speed up the consulting process, a *collaborative* approach is used in our work to predict what a consumer is targeting according to behavior patterns produced by the previous users. The collaboration-based approach does not directly analyze what a user likes, but taking the opinions of similar users. Generally the *k*-nearest neighbor method is performed to find other users with similar tastes for a specific user in which the similarity between different users is measured by certain correlation criteria. The preference prediction for this user is thus based on the evaluations of his nearest neighbors. (Shardanand and Maes, 1995), (Smyth and Cotter, 2000) and (Konstan *et al.*, 1997) describe systems by this method. However, unlike the traditional collaboration-based work in which the long term consumer information is recorded, for the type of products considered here only temporal consumer information is available. Therefore the similarity between two users must be measured by their behavior patterns obtained from the user-system interactions. Our system employs a dynamic programming approach to find previous users with similar behavior patterns for the current consumer, and then presents him the products selected by the similar users. More details are described in a later section.

3. A Hybrid System for 3C Product Recommendations

As is indicated, the 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 notebook computers. For products of this type, domain knowledge is expected and previous buying experiences may not be helpful. What a consumer needs here is some expertise to recognize the *ideal* product based on his current preferences and requirements. Under such circumstances, a more appropriate way to provide customized information services is to create an interactive environment in which a consumer can iteratively express his preferences or needs to the recommender system and the system can then use the ephemeral information from the consumer 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 physical shop, communicates with the human agent who can normally provide certain knowledge for products the consumer is interested in, and asks for his suggestions in making decision. This section describes how we design and implement a system that utilizes expert knowledge and social information for 3C product recommendations.

3.1 System Architecture

Developing intelligent agents to promote electronic commerce has been advocated in recent years (Liang and Huang, 2000; Maes *et al.*, 1999). Hence, in this work an agent-based methodology is adopted in which each agent is an expert in performing a specific task, and different expert agents work simultaneously to achieve the overall task. The goal of our recommender system is to analyze a consumer's current requirements and find 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, our system mainly includes four agents: an *interface agent* for interacting with the consumer and human expert, an *knowledge agent* for transferring external expert knowledge for internal use, a *decision-making agent* for calculating the optimality of each product, and a *behavior-matching* agent for looking for similar user patterns. The overall system architecture is illustrated in Figure 1. Currently the products considered in our system include notebook computers, home theater systems, digital cameras, and personal digital assistants. For simplicity, in the following we use the recommendations of notebook computers as an example to describe how the individual agents are developed.

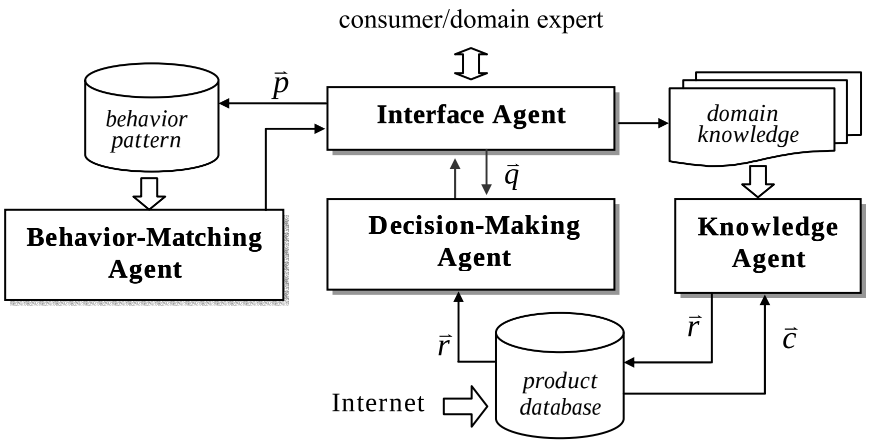

Figure 1: The system framework for the hybrid recommendations.

3.2 The Interface Agent

In order to capture and analyze a consumer's personal needs, the interface agent in Figure 1 presents him some specially designed questions about the products. It is presumed that the consumer does not have enough domain knowledge to answer quantitative questions regarding about the specifications of the product, therefore the system inquires 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 specifications obtained from the knowledge agent to find the ideal products.

Figure 2 shows the environment in which the interface agent interacts with the consumers. Here a consumer is asked to express his requirements (from 1 to 10) 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 to him some top product alternatives with their corresponding feature ranks. Figure 3 shows the typical recommending results in which the first two products presented are the results derived from expert knowledge and social information accordingly. The relative ranks of all feature dimensions for the product obtained from expert knowledge are also shown. In addition, the system gives another alternative (the third product in Figure 3) whose features partially match the ones specified by the user. This is especially useful to recommend products that have many functions, such as picturing, computing, communicating, etc.

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 will be used to find new candidates again. The column “need” in Figure 3 shows the user’s needs in different dimensions. In this iterative and interactive 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.

The interface agent also communicates with human experts in order to extract product knowledge from them. Figure 4 and Figure 5 show the environments that a domain expert can embed his personal knowledge to the system. As shown in Figure 4, both qualitative features and the hardware properties that could be relevant to each individual feature are pre-defined by the system and presented to the experts. The experts can then associate the hardware properties to their corresponding feature. Once an expert defines the associations (by pressing the “confirm” button), the system presents him another interface (i.e., Figure 5) so that he can further give a weight to each association link he has indicated. In this way, the system can extract product knowledge from the domain experts and integrate the opinions from different experts to give suggestions objectively. Section 3.3 will give more implementation details.

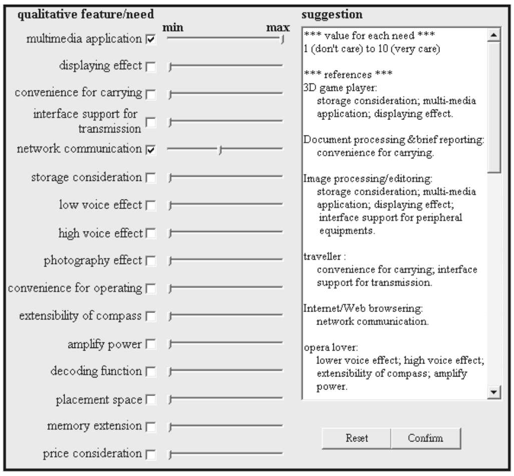

Figure 2: The questionnaire presents by the interface agent

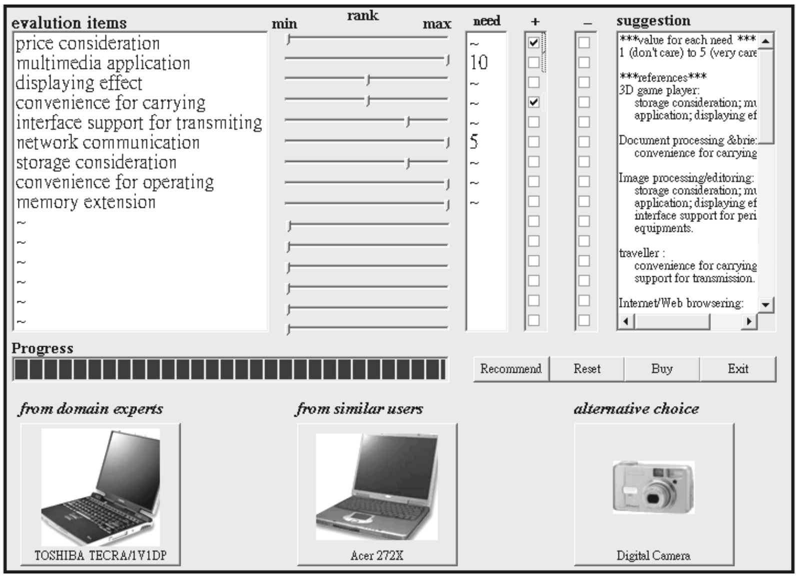

Figure 3: The typical recommendation result shown by the interface agent. If the consumer is not satisfied with the result, he can modify his needs by pressing the "+" (increase) or "－" (decrease) buttons shown.

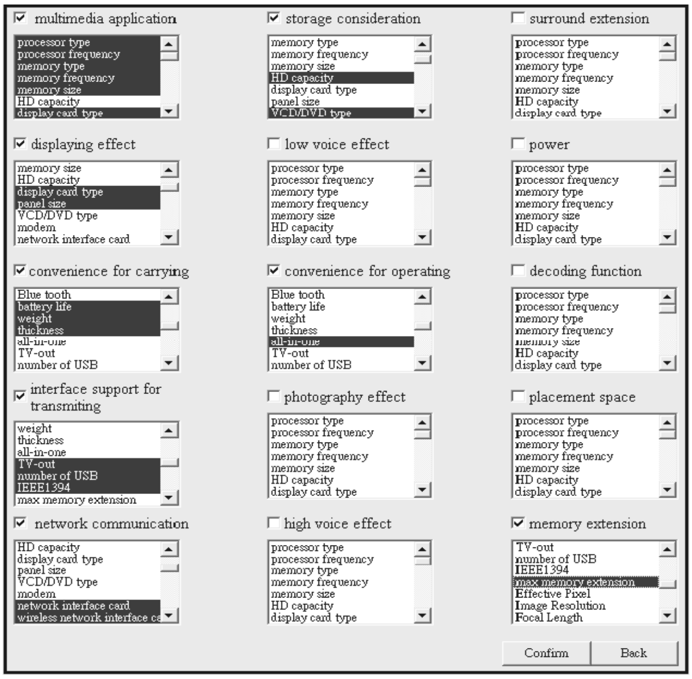

Figure 4: The interface for an expert to embed his knowledge by defining the relationships between qualitative features and hardware properties.

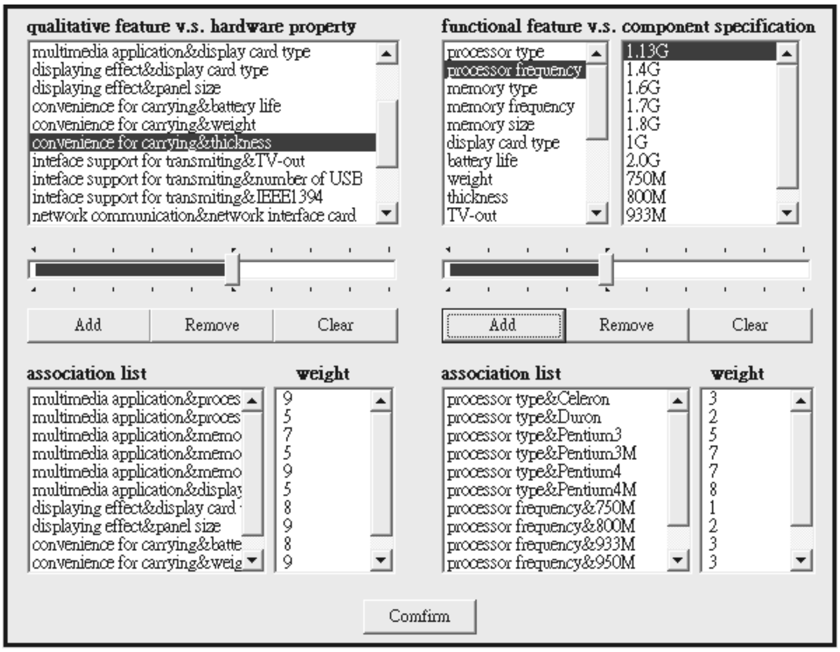

Figure 5: The interface for an expert to give a relative weight to each association link he has made.

3.3 The Knowledge Agent

In general a product is specified by a set of critical components $<c_1, c_2, \ldots, c_n>$ and different vendors have their own ways to categorize their products. For example, a computer can be described in terms of 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 $<q_1, q_2, \ldots, q_m>$. Therefore, in order to evaluate how a certain product P_i satisfies these needs, a knowledge agent is developed to transfer P_i from its vector form of original component specifications $<c^i_1, c^i_2, \ldots, c^i_n>$ to the corresponding vector of qualitative features $<q^i_1, q^i_2, \ldots, q^i_m>$ defined, so that further measurements can be performed.

In our work, the products are firstly collected from the Internet, and each product P_i is represented as a vector of component names $<c^i_1, c^i_2, \ldots, c^i_n>$ in the internal database. Then each product is converted to a vector of functional features $<f_1, f_2, \ldots, f_k>$ 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 processor type, the processor frequency, the memory frequency, etc. It should be noted that the dimensionalities (i.e., n and k) of the above vectors can be different; that is, the name of each component can be mapped into many different functional features. For example, a CPU named Pentium IV 1.6G is mapped into two performance values indicating its quality on the features of processor type and processor 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. The right hand side of the interface shown in Figure 5 is the environment for experts to define the performance values. For example, when an expert chooses a functional feature "processor frequency", the possibilities (e.g., 1.13G, 1.4G, etc.) for this feature are shown and the expert can associate one possibility to this feature. He can then move the bar in the middle to give a relative strength (performance value) to indicate the relative importance of the association, and press the "Add" buttons to insert this assignment to the system. Different experts may have their own settings; here the average is used to define each performance value.

As the decision-making agent in this work uses a multi-attribute decision making approach, derived from the TOPSIS (technique for order preference by similarity to ideal solution (Yoon and Hwang, 1995)), to estimate the optimality of each product for a consumer, the above functional features f_1, f_2, …, f_k must be further integrated to a pre-defined list of functional abilities $<a_1, a_2, \ldots, a_m>$ (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 many-to-many mapping in which the different functional features concerning about the same functional ability of the product are combined. As shown in Figure 4, the correspondence between the functional features and the functional abilities are also determined by the domain experts. An expert can then use the interface in Figure 5 (the left hand side) to indicate the strength of each association link he has defined in Figure 4. Normalization is also performed here before combining values from different dimensions.

Once a product P_i has been characterized as a vector of functional abilities $<a^i_1, a^i_2, \ldots, a^i_m>$, each value a^i_j 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 $<r^i_1, r^i_2, \ldots, r^i_m>$ where each r^i_j 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. As is analyzed, the knowledge agent is mainly a mediator by which both the consumer's needs and the names of the product components can be converted to a common form so that the optimality of the product can be measured. It should be noted that the final dimensionality m must correspond to the number of qualitative needs collected from the consumer.

3.4 The Decision-Making Agent

With the ranks transferred from the product names, whenever a consumer indicates his relative needs as described in the above section, the overall rank (or optimality) R of a product in the database is measured by:

$$R = \frac{R^-}{R^+ + R^-} \qquad \text{where}$$

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

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

In the above equations, n is the number of product features; r_i is the normalized rank of a product in the feature dimension i, and r_{i_best} and r_{i_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_best}) and the farthest distance from the negative-ideal one (i.e., the combination of all worst ranks r_{i_worst}). Once the currently available products have been ranked by the above criterion, the products with the top ranks are then recommended to the consumer (as shown in Figure 3). If the consumer 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 correspondingly are thus recommended to the consumer.

3.5 The Behavior-Matching Agent

As is mentioned in section 2, a consumer can iteratively interact with the system until he is satisfied with the results. Yet, in order to reduce the efforts of user-system interaction, a behavior- matching agent is built to compare the consulting patterns of the current user to the ones previously recorded. Then the system can recommend the products selected by the similar users to the current consumer. A behavior pattern here describes how a user modifies his needs during the consultation; it is defined as (p_1, p_2, …, p_l) in which p_t ($1 \leq t \leq l$) represents a set of operations the user has performed in different qualitative feature dimensions at a certain consulting step t (i.e., setting his need in a feature dimension to a value, as described in section 3.2), and l is the number of steps a user has interacted with the system. It should be noted that because different users may interact with the system for different numbers of steps, the length of their corresponding behavior patterns may thus be different. As can be observed, the problem of measuring the similarity of two behavior patterns (a_1, a_2, …, a_m) and (b_1, b_2, …, b_n) can be regarded as a sequence alignment problem—to adjust the two sequences (with some blanks "-") in some way and to score how good the resulting alignment is (Bafna, *et. al.*, 1997). For example, (a_1, a_2, a_3, -, a_4) and (b_1, -, -, b_2, b_3) is a possible alignment of the above two sequences, for the case of $m = 4$ and $n = 3$. The alignment with best score is then used to define the similarity of the two behavior patterns.

In order to evaluate an alignment, the following scoring rules are used: (i) if a_i ($1 \leq i \leq m$) is aligned with b_j ($1 \leq j \leq n$) for the above two sequences and $a_i = b_j$, then the score is increased by 2; (ii) if a_i is aligned with b_j and $a_i \neq b_j$, the score is decreased by 1; and (iii) if a_i or b_j is aligned with a blank (i.e., "-") inserted to the sequence, the score is also decreased by 1. Here, an element (e.g., a_i or b_j) in a behavior sequence represents a set of operations the user has performed in different feature dimensions at a certain step. Therefore, to determine whether a_i is equal to b_j is to compare what two users have done at a certain step. In this work, if more than (or equal to) half of the operations in the large set (with more operations performed) are included in the small set, the two sets are defined to be equal. With the above rules, the behavior-matching agent employs the dynamic programming approach (Aho, *et. al.*, 1974) to solve the alignment problem by the following formula:

$$A(m, n) = max\{A(m\text{-}1, n\text{-}1,)\text{-}1, A(m\text{-}1, n)\text{-}1, A(m, n\text{-}1)\text{-}1\} \quad \text{if } a_m \neq b_n$$
$$= A(m\text{-}1, n\text{-}1)+2 \quad \text{if } a_m = b_n$$

in which $A(m, n)$ denotes the score of the optimal alignment of two sequences (a_1, a_2, …, a_m) and (b_1, b_2, …, b_n).

Once the similarity of two behavior patterns can be measured by the above method, during the user-system interaction, the behavior-matching agent can thus find the most similar user patterns recorded in the database for the incomplete user pattern available so far from the current user. The system then predicts that what the current user is targeting may be one of the products the similar users selected previously. Hence, the system also recommends some products derived from the collaboration approach described above to the current user, in addition to the optimal products provided by the decision-making agent. In this way, the number of iterations of user-system interaction can be reduced, and the system can work even more efficiently.

4. Experiments and Evaluations

4.1 Recommendations by the Decision-Making Agent

This system is to recommend products that best satisfy the consumer's current needs and with the optimal quality. Therefore the experiments emphasize 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.

In the first set of experiments, we concentrate on examining the correctness of our system. Therefore we shall observe the correspondence between the consumer's modifications and the ranks of the products recommended in each single feature dimension. In the experiments, we simultaneously modified requirements in different feature dimensions at each step in order to examine the overall performance of the system. Figure 6 shows the typical experimental results for the recommendations of notebook computer, in which a consumer continuously modified his needs on four different product features at the same time. As can be seen, when the consumer increased/decreased his need in different feature dimensions by pressing the buttons of plus/minus, the weights (derived from the needs) and the ranks for these features changed accordingly. We have also examined the products recommended by the system for each step to confirm the correctness of our system.

In addition to the notebook computer, we also employ the same approach described in the previous section to conduct experiments for home theater system recommendations. A home theater system includes three independent equipments: DVD player, amplifier, and loudspeaker; it is specified by 37 different features. As with the notebook computer, the quantitative product features are transferred into the qualitative ones for the consumers. The typical results are presented in Figure 7. From these evaluation results, it can be seen that the system is able to successfully adapt to the consumer's changes.

It should be noted that possibly the ranks of the products recommended by the system are not able to truly response to the consumer's changes immediately. This is mainly because that each product here is a fixed combination of certain components, and the characteristics of exclusiveness or dependency between different features of the product may sometimes result in the situation that when the need on one feature increases, the need on the other may increase or decrease 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 a consumer has known what he needs and only the similarities between the product specified by the consumer and the ones available in the database are measured.

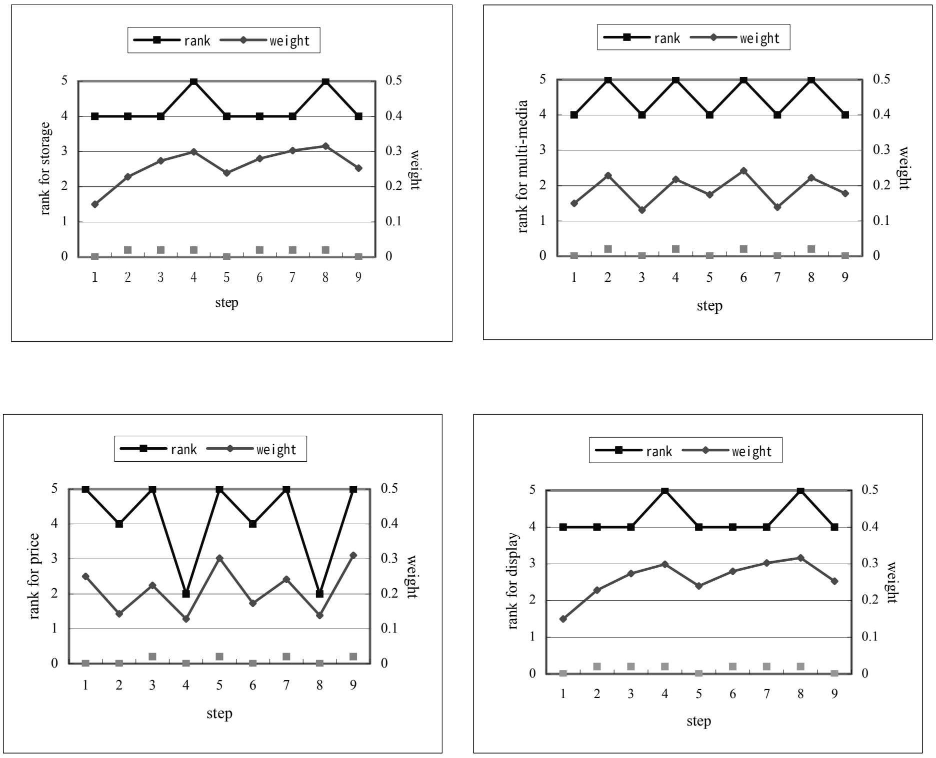

Figure 6: The correspondence among the feature ranks of the ideal product recommended by the system, the accumulated weights derived from the needs specified by the consumer, and each single modification (increase/decrease, the data points along the x-axis) in different feature dimensions.

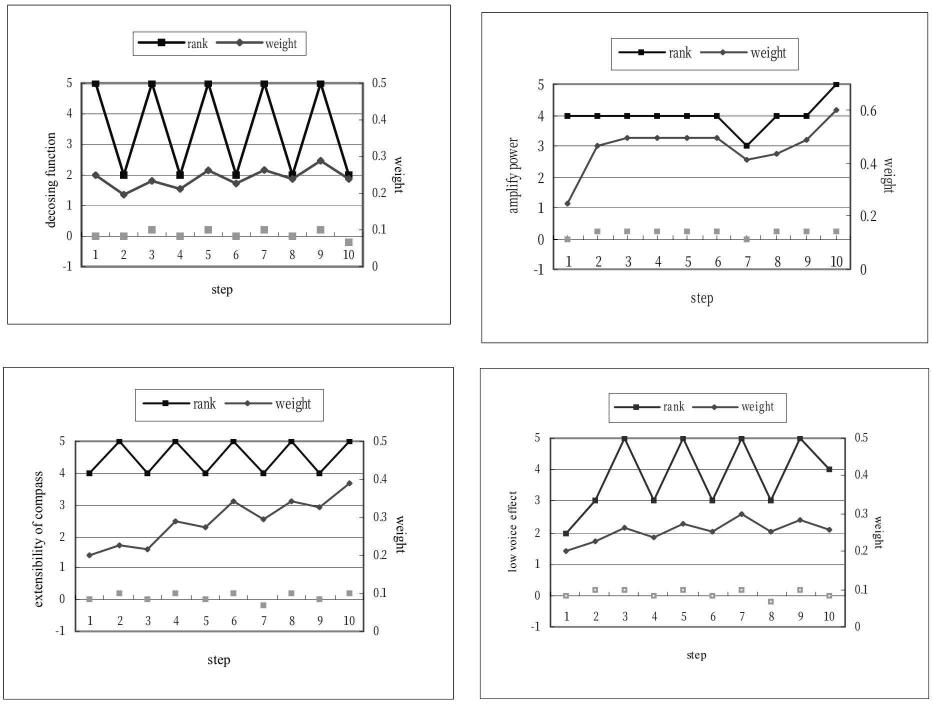

Figure 7: Some results for the recommendation of home theater system.

4.2 Collaboration-Based Recommendations

As is described, this work also uses a collaboration-based approach to enhance the performance of product recommendations. The dynamic programming method and the scoring rules used have been presented in section 3.4. In this section, we take a walk-through example to furthermore illustrate how the behavior-matching agent works. In Figure 8(a), the first sequence (a_1, a_2, a_3, a_4, a_5) shows the behavior pattern by a previous user *A*, in which a set of operations in a_t ($1 \leq t \leq 5$) represents the operations user *A* has performed at time step *t*, and each operation (d_i, *k*) means that user *A* has set his need in feature dimension d_i to be a value *k* at a certain time step. As shown in the right hand side of Figure 8(a), when a current user *B* was interacting with the system, he firstly performed a set of operations b_1. With the sequence (b_1), the behavior-matching agent aligned it to the sequence *A* and obtained a best alignment shown in Figure 8(b). In this alignment, two of the three operations (i.e., (d_3, 2) and (d_{16}, 2)) in a_1 are the same as the ones in b_1, therefore set a_1 and set b_1 are thus regarded as the same by definition and the alignment has a score $+2 \times 1 + (-1) \times 4 = -2$. After the current user *B* performed the second set of operations, the current sequence (b_1, b_2) and the sequence A were aligned again. Figure 8(c) shows the best alignment with a score $+2 \times 2 + (-1) \times 3 = +1$. Figure 8(d) and (e) are the consecutive results afterwards.

With the same manner, in the real situation the behavior-matching agent finds the behavior sequences with highest scores at different consulting steps, and retrieves the products selected by the similar consumers (i.e., the corresponding consumers of the behavior sequences found). These products are then recommended to the current user as alternatives to the optimal ones estimated by the decision-making agent.

previous user A

a_1	a_2	a_3	a_4	a_5
(d_3, 2)	(d_3, 2)	(d_3, 2)	(d_2, 1)	(d_2, 1)
(d_9, 4)	(d_9, 4)	(d_9, 3)	(d_3, 2)	(d_3, 1)
(d_{16}, 2)	(d_{16}, 1)	(d_8, 2)	(d_8, 2)	(d_8, 2)
		(d_{16}, 2)	(d_9, 3)	(d_9, 3)

current user B

b_1	b_2	b_3	b_4
(d_3, 2)	(d_3, 2)	(d_2, 1)	(d_2, 1)
(d_9, 2)	(d_9, 3)	(d_3, 2)	(d_3, 2)
(d_{16}, 2)	(d_{16}, 2)	(d_4, 2)	(d_4, 1)
		(d_8, 1)	(d_8, 2)
		(d_{16}, 2)	(d_{16}, 2)

(a)

a_1	a_2	a_3	a_4	a_5
b_1	—	—	—	—

(b)

a_1	a_2	a_3	a_4	a_5
b_1	—	b_2	—	—

(c)

a_1	a_2	a_3	a_4	a_5
b_1	—	b_2	b_3	—

(d)

a_1	a_2	a_3	—	a_4	a_5
b_1	—	b_2	b_3	b_4	—

(e)

Figure 9: An example illustrates how the behavior sequences are matched.

5. Conclusions and Future Work

In this paper, we have indicated the need for Internet enterprise to provide more advanced 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 intelligent recommender systems is a promising way to achieve this goal. Therefore in this work, we present an agent-based system to recommend 3C products a consumer does not buy frequently. To collect product knowledge, our work includes a specially designed interface to allow domain experts to easily embed their professional knowledge to the system. Instead of modeling a consumer's preferences, this system concentrates on calculating optimal products for a consumer by using the ephemeral information provided by him and the built-in expert knowledge. Here, a multi-attribute decision making method is used to recommend optimal products for a consumer, based on his needs and the quality of the product. In addition, a dynamical programming approach is used to exploit social information obtained from previous consumers for recommendations. To access our methodology, different kinds of experiments have been conducted. Experimental results have shown the promise of our systems.

Based on the work presented, we are extending this system to include more 3C products. Meanwhile, we are also investigating whether the proposed approach can be applied to recommend products of different natures, such as travel package. Apart from the recommendation, negotiation is the other activity most related to the decision-making process in electronic trading. Therefore, it will be worthwhile to explore the problem of automated negotiation in the electronic market as soon as the consumer has decided what to purchase.

References

Aho, A. V., J. E. Hopcroft and J. D. Ullman, The Design and Analysis of Computer Algorithms, 1974, Addison-Wesley, Reading, MA.

Bafna, V., E. Lawler and P. Pevzner, Approximation Algorithms for Multiple Sequence Alignment, Theoretical Computer Science, Vol.182, 1997, pp.233-244.

Basu, C., H. Hirsh and W. Cohen, Recommendation as Classification: Using Social and Content-Based Information in Recommendation, Proceedings of National Conference on Artificial Intelligence, 1998, pp.714-720.

Burke, R. D., The Wasabi Personal Shopper: A Case-Based Recommender System, Proceedings of National Conference on Artificial Intelligence, 1999, pp.844-849.

Cheung, K. W., C. H. Li, C. M. Lam and J. Liu, Customized Electronic Commerce with Intelligent Software Agents, S. M. Rahman (ed.), Internet Commerce and Software Agents, 2000, pp.150-176.

Konstan, J. A., B. N. Miller, A. Maltz, J. L. Herlocker, L. R. Gordan and J. Riedl, Grouplen: Applying Collaborative Filtering to Usenet News, Communications of the ACM, Vol.40, No.3, 1997, pp.77-87.

Lee, W.-P, C.-H. Liu and C.-C. Lu, Intelligent Agent-Based Systems for Personalized Recommendations in Internet Commerce, Expert Systems with Applications, Vol.22, No.4, 2002, pp.275-284.

Liang, T.-P. and J.-S. Huang, A Framework for Applying Intelligent Agents to Support Electronic Commerce, Decision Support Systems, Vol.28, 2000, pp.305-317.

Maes, P., R. H. Guttman, and A. G. Moukas, Agents that Buy and Sell: Transforming Commerce as We Know it, Communication of the ACM, Vol.42, No.3, 1999, pp.81-83.

Schaher, J., J. Konstan and J. Riedl, E-Commerce Recommendation Application, Journal of Data Mining and Knowledge Discovery, Vol.5, No.1-2, 2001, pp.115-152.

Sen, S. and K. Hernandez, A Buyer's Agent, Proceedings of International Conference on Autonomous Agents, 2000, pp.156-162.

Shardanand, U. and P. Maes, Social Information Filtering: Algorithms for Automating "Word of Mouth," Proceedings of the ACM Conference on Human Factors in Computing Systems, 1995, pp.210-217.

Shearin, S. and H. Lieberman, Intelligent Profiling by Examples, Proceedings of International Conference on Intelligent User Interface, 2001, pp.145-151.

Smyth, B. and P. Cotter, A Personalised TV Listings Service for the Digital TV Age, Knowledge-Based Systems, Vol.13, 2000, pp.53-59.

Yoon, K. and C.-L. Hwang, Multiple Attribute Decision Making: An Introduction, 1995, Thousand Oaks, CA.

Wilke, W., CBR and Electronic Commerce on the World Wide Web, Proceedings of International Conference on Case-Based Reasoning, 1997.