Journal of Intelligent Information Systems, 18:2/3, 173–194, 2002

PVA: A Self-Adaptive Personal View Agent

CHIEN CHIN CHEN paton@iis.sinica.edu.tw
MENG CHANG CHEN mcc@iis.sinica.edu.tw
Institute of Information Science, Academia Sinica, Taiwan

YEALI SUN sunny@im.ntu.edu.tw
Department of Information Management, National Taiwan University, Taiwan

Abstract. In this paper, we present PVA, an adaptive personal view information agent system for tracking, learning and managing user interests in Internet documents. PVA consists of three parts: a *proxy*, *personal view constructor*, and *personal view maintainer*. The proxy logs the user's activities and extracts the user's interests without user intervention. The personal view constructor mines user interests and maps them to a class hierarchy (i.e., personal view). The personal view maintainer synchronizes user interests and the personal view periodically. When user interests change, in PVA, not only the contents, but also the structure of the user profile are modified to adapt to the changes. In addition, PVA considers the aging problem of user interests. The experimental results show that modulating the structure of the user profile increases the accuracy of a personalization system.

Keywords: machine learning, automatic classification, personalization, agent, and WWW

1. Introduction

The World Wide Web (WWW) has become a major information source for people from all walks of life. Via web browsers, people can receive various types of information from almost anywhere in the world. Although the WWW facilitates information distribution, the rapid growth of the number of Internet documents has made information discovery from the Internet a time consuming, and often fruitless, task. Two popular types of auxiliary WWW access tools, **web search engines** and **web directory systems**, have been developed to help people retrieve information from the Internet. Search engines, like Google, allow users to retrieve relevant documents by entering keywords. Although a search engine provides convenient web page search, its precision (Salton, 1989) is low. This means that the user has to check many returned web pages in order to locate the desired information. Web directory systems, like Yahoo!, organize their collected web pages in a hierarchical category structure that users can search to find relevant pages via top-down navigation. Although the precision of a web directory system is quite high, recall (Salton, 1989) is usually low because its collection of web pages is relatively small. This is because the task of categorization is performed manually by directory system maintainers and partially by information providers. Manual categorization makes it difficult for a web directory system to compete with search engines in terms of web page coverage.

To overcome the drawbacks of search engines (low precision) and web directory systems (low recall), a lot of research in this area has focused on personalization of WWW access. In general, personalization systems construct user profiles (Korfhage, 1997) by learning user interests. Then they utilize the information in user profiles to assist people when they are surfing the Web. For instance, a personalization system learns that the current interest of a user is the topic of finance. When the user queries a search engine by entering the keyword "bank," the personalization system can re-rank the returned web pages by moving finance related pages to the front of the list. Thus, precision can be raised.

In this paper, we present a system, the **Personal View Agent (PVA)**, that can automatically organize a personal view by learning user interests and adapt the personal view to changes in the user interest. There are three important features of PVA: (1) PVA learns a user profile without user intervention, such as relevance feedback (Salton, 1989). Goecks and Shavlik (2000) indicated that a central question in the area of learning user profiles concerns how systems obtain training examples. Although obtaining relevance feedback, where users are asked to manually rate pages, may make sense, it is troublesome for users and seldom done. In PVA, we use a proxy to acquire the training data implicitly. (2) Users usually have interests in multiple domains—music and basketball, for example. PVA, like previous works (Chen and Sycara, 1998; Widyantoro et al., 1999), models each domain as a separate vector in the vector space model (Salton, 1989). Furthermore, these vectors are organized into a hierarchical structure (called personal view). Each node in a personal view represents a topic of the real world. Like web directory systems, the hierarchical personal view can clearly describe the user's interests. (3) PVA can adapt to user interest changes over time. Since user interests often change, personalization systems need to have a mechanism by means of which they can adapt to these changes. While many previous works (Chen and Sycara, 1998; Hoashi et al., 2000; Widyantoro et al., 1999) only considered the contents of user profiles, PVA also modulates the structure of the personal view to adapt to user interest changes.

The rest of the paper is organized as follows: In Section 2, we describe existing approaches to web personalization. In Section 3, we show the structure of the user profile in PVA. In Section 4, we introduce the system components of PVA. We evaluate the system performance in Section 5. The PVA news filtering application is presented in Section 6. Finally, conclusions and future work are given in Section 7.

2. Related work

Most personalization systems need to use training data to build and update user profiles. According to the sources for their training data, recently proposed personalization systems can be divided into two categories. One is **Collaborative Filtering (CF)** (Mobasher et al., 2000b) systems that consider the behavior of a group of users, while the other category consists of systems which focus on a **single user**. CF analyzes a web log that contains the access patterns of a group of people in order to discover their general behaviors. CF is popular in e-commerce systems. For example, some e-tailers use CF to discover the shopping behaviors of their customers in various groups, and to provide users with shopping recommendations based on their purchase histories. Mobasher et al. (2000a, 2000b) proposed an interesting CF system that provides recommendation pages for individuals when they visit

Table 1. Recent works on single-user personalization systems.

System name	Use relevance feedback	Major feature
WebMate (Chen and Sycara, 1998)	Yes	Model domains of user interests serve as a set of vectors.
WebACE (Han et al., 1998)	No	Clusters user interests automatically.
Personal WebWatcher (Mladenic, 1999)	No	Suggests relevant hyperlinks.
News Dude (Billsus and Pazzani, 1999)	Yes	Predicts short-term interests.
Alipes (Widyantoro et al., 1999)	Yes	Exploits negative user profiles.
(Pretschner and Gauch, 1999b)	No	Structures ontology-based user profiles.
(Hoashi et al., 2000)	Yes	Exploits negative user profiles.

a site. It clusters the web pages of a web site based on its access log. Each cluster indicates the pages that are commonly accessed together. When a user browses the site, the system gives him a set of recommended next movements based on his navigation history. Although CF is popular in e-commerce systems, it has the following drawbacks. (1) The user profile of a CF system is derived from an access log; hence its function scope is limited to one site only. (2) It is hard to update the user profile of a CF system incrementally. If the structure of the site changes, the previously learned user profile will be obsolete. (3) Since CF learns the general behaviors of a group of people, it can not acquire the particular interests of each user unless it is given plentiful training examples.

In contrast to a CF system, the training data of a single-user personalization system comes from one user only. Therefore, its user profile is specific to a user's interests rather than to the general behaviors of a group of people. The Table 1 lists recent works on single-user systems.

Two tasks are critical when designing single-user personalization systems. One is modeling user interests in multiple domains, and the other is incrementally updating user profiles. In general, a user has interests in multiple domains; e.g., a user may be interested in basketball, movies, and computer games. In order to closely track user interests, personalization systems have to maintain multiple domains of user interests. WebACE uses a clustering algorithm to partition a user's access logs into clusters, each of which indicate a domain of user preferences. Alipes and WebMate exploit the vector space model in which each domain of user interests is represented by a keyword vector. Naive Bayes (Mitchell, 1997) is used in News Dude to calculate the probability of a document belonging to a domain of user interests. Although these systems can represent the multiple domains of user interests, the domains of a user profile are orthogonal. That is, the domains of a user profile are independent of each other. In the real world, concepts usually have a *general to specific* relationship. It would be more reasonable if the domains in a user profile were arranged in a hierarchical structure. Pretschner and Gauch (1999b) proposed a novel model of user profiles, which organizes user interests into a hierarchical structure, rather than a set of independent domains. Thus, there is a *general to specific* relationship in this user profile. Similar to personal bookmarks (Li et al., 1999), this hierarchical user profile can better describe the user preferences. Although the representation of this hierarchical user profile is innovative, Pretschner and Gauch did not make use of the characteristics of its hierarchical structure (e.g., by splitting or merging nodes in the user profile) to capture the dynamics of user interest changes.

To capture the dynamic changes of user interests, personalization systems need training data to update user profiles periodically. News Dude, WebMate, Hoashi et al. (2000), Klinkenberg and Renz (1998) and Alipes utilize relevance feedback to acquire training data explicitly. Usually, only positive information (i.e., documents that the user is interested in) is considered when the system updates the user profile. Alipes and Hoashi et al. (2000) found that negative information could help eliminate uncertainty in predicting user interests. Although relevance feedback is effective, users are often overloaded. Personal WebWatcher and WebACE utilize a proxy to obtain training data implicitly. Techniques, such as those which employ the page-browsing time or page-view frequency, are used to measure the popularity of a page, and pages of user interests are used to update user profiles incrementally. In general, changing interests are handled by time windows of fixed or adaptive size on the training data or by weighting data according to their age (Klinkenberg and Renz, 1998). Klinkenberg and Renz (1998) use user feedbacks to identify interest changes and adapt the size of the window to the current extent of interest changes. In this work, we utilize a proxy to acquire training data implicitly and use the method of aging training data to update user profiles.

In adapting to user interests, News Dude and Alipes reveal that there are two types of the user interests. One is short-term interests, and the other is the long-term interests. Short-term interests usually are related to hot news events and vanish quickly. On the other hand, long-term interests often reflect actual user interests.

Although each system has its own way to update user profiles, it is only concerned with the contents of user profiles. For example, WebMate tunes the weights of keyword vectors when a user interest changes. Experience with the short-term interests indicates that not only the contents of user profiles, but also the life cycles of each domain in a user profile should be considered when updating user profiles. In PVA, we organize user interests into a hierarchical structure and apply the theory of artificial life to adjust the life cycles of domains in the user profile. For more information about personalization systems, Pretschner and Gauch (1999a) gave a detailed survey of existing systems and approaches proposed in the recent years.

3. User profile representation

In PVA, a user profile is represented as a category hierarchy, called a **personal view**, where each category represents the knowledge about a domain of user interests. A personal view (PV) can be written as

$$PV = (V_1, E_1), (V_2, E_2), \ldots, (V_n, E_n) \tag{1}$$

Each category in a personal view consists of a keyword vector V_i and an energy value E_i. V_i describes the content of a domain of user interests, while E_i, which is a number, indicates the popularity of user interests. E_i, like the endogenous fitness of an artificial life agent (Menczer et al., 1995), controls the life span of a category in a personal view. The energy of a category increases when users show interest in web pages of that category, and it diminishes by a constant value for each period of time. Based on this energy value, categories that have high-energy will generate sub-categories to describe the user interests at a certain level of detail. Similarly, categories that receive little interest will gradually be abstracted and will finally die out. With the energy values of categories, the structure of a personal view can be modulated as the user's interests change. The hierarchical representation of a personal view has two advantages: one is efficiency in information search, and the other is the *general-to-specific* property of a hierarchy. Searching for relevant information can proceed efficiently in a category hierarchy via top-down navigation (Chen, 2000), and the *general-to-specific* relationship of the category hierarchy is intuitive.

For a personal view to be built, user interests have to be tracked and categorized. Two schools of learning algorithms, clustering and classification (i.e., non-supervised and supervised learning, respectively) (Mitchell, 1997), are frequently used to partition the domains of user interests. Many approaches, e.g., WebACE, employ methods based on clustering to structure user interests. However, automatic cluster labeling is not mature, as it is difficult to capture the true intension of a cluster of web pages. For this reason, we adopt the classification method based on a pre-defined category hierarchy, called a **world view**, as the superset of personal views. A world view represents the complete view of the world of which every possible personal view is a subset. PVA employs a world view as a reference and extracts the categories of user interests to build a personal view. In this study, our previous work, ACIRD (Lin et al., 1998), is employed as the world view. (Note that any well-defined world view can be used.) Figure 1 shows an example of a personal view and a world view.

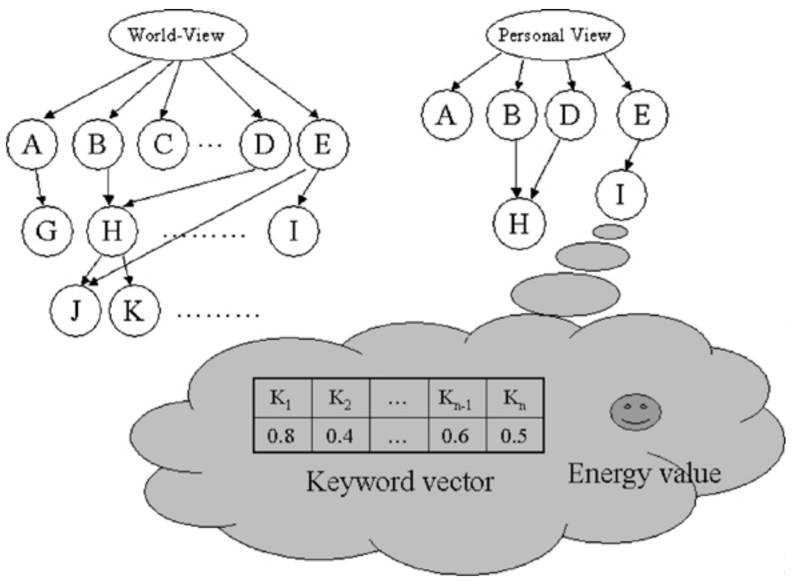


Figure 1. An example of a personal view and a world view.

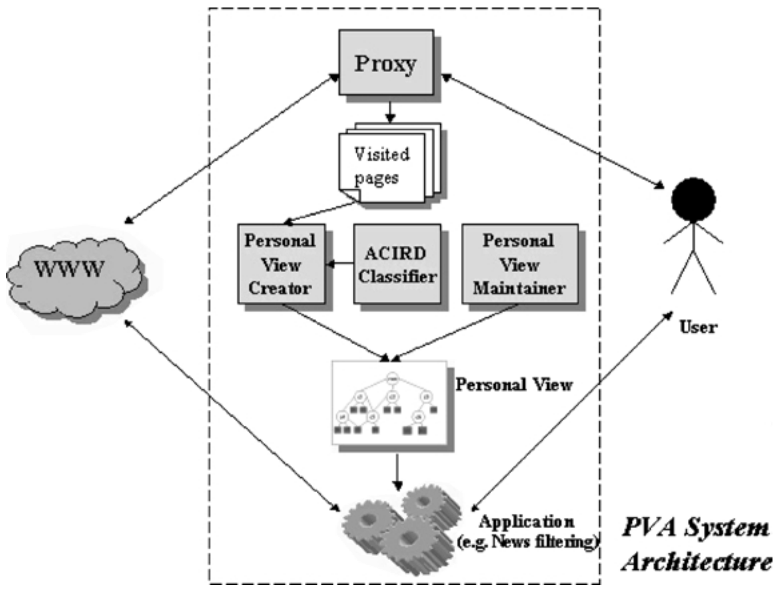


Figure 2. The PVA system architecture.

4. System architecture

PVA has three major components: a *proxy*, *personal view constructor* (**PVC**) and *personal view maintainer* (**PVM**). Figure 2 shows the system architecture of PVA. A user surfs the web through a local proxy. Then, the proxy analyzes the user's activities and sends pages of user interests to PVC, which constructs a personal view. To adapt to dynamic changes of user interests, PVM periodically adjusts the personal view to reflect the current user interests. Applications, such as query extension, information filtering and web page recommendation, can utilize the information in the personal view to improve their performance. The following sub-sections describe each component in detail.

4.1. Proxy

Besides relaying user's browsing requests, the major function of the proxy is to track user preferences implicitly. Periodically (e.g., every day), the proxy analyzes its log and sends pages of user interests to PVC. Techniques such as those employing the page view frequency, link visit percentage and page browsing time can be used to measure the degree of popularity of a page. Chan (1999) and Hijikata (1999) found that the browsing time of a page usually reflects the user's degree of interest. In Hijikata's experiment, 93% of the pages whose browsing times were longer than 60 seconds were relevant to user interests. Moreover, Pretschner and Gauch's experiment (1999b) indicated that 20% of web pages are visited for less than 5 seconds, while the mean browsing time of a page is 54.49 seconds.

Hence, in our approach, pages whose browsing times are longer than a preset threshold (e.g., 2 minutes) are sent to PVC, which generates and updates the user profile.

4.2. Personal view constructor

Periodically, PVC assimilates pages of user interests into a personal view. We organize the pages in a top-down manner. First, pages are placed in abstract categories. If a user continues to focus on these categories, they are split into detailed sub-categories. On the other hand, if categories do not receive interest, they are merged into the abstract category. We will introduce the process of constructing a personal view in this section and explain the split and merge operators used in Personal View Maintainer. As PVA employs a vector space model, a web page is parsed into a keyword vector. For each term i in a page p, we use the following formula to calculate its weight $W_{i,p}$:

$$W_{i,p} = \frac{freq_{i,p}}{\max_j[freq_{j,p}]} \tag{2}$$

where $freq_{i,p}$ is the term frequency of the term i in page p and $W_{i,p}$ is normalized to [0, 1].

Lin et al. (1998) reported that the pure term frequency keyword weighting function, also used in this study, performs better than the traditional TF*IDF weighting function in a hierarchical category learning system. Since the IDF (Inverted Document Frequency) is designed to enhance the discriminating capability of high frequency terms among categories, the TF*IDF scheme is not effective in the hierarchical learning model.

After a page is parsed, its keyword vector is classified by means of the world view to obtain the categories that the page belongs to. (Note that in this study, we adopt ACIRD (Lin et al., 1998) as the world view that classifies a web page into one or more categories.) Each obtained category is represented as a **classification path**, that is, a path from the root to the assigned category in the world view. For example, a web page of a professional basketball team is classified into the category "*NBA*," and its classification path is "*/Sport/Basketball/NBA/*". To insert the page into a personal view, we first check if the classified category exists in the personal view. If the category exists, the page is inserted into that category directly. If the category does not exist, we insert the page into the non-root closest ancestor in the personal view. If no non-root ancestor exists, we create the top category of the classification path in the personal view and insert the page into this newly generated category. Figure 3 shows an example of web page insertion into a personal view. In figure 3, Page 1 is assigned to "*/Sport/Basketball/NBA/*". As the category "*NBA*" exists, Page 1 is inserted to "*NBA*" directly. Page 2 is assigned to "*/Finance/Stock*". Since the category "*Stock*" does not exist, neither does its parent "*Finance*," so we create the category "*Finance*" in the personal view and then insert Page 2 into that category.

After the pages are inserted into the personal view, the keyword vectors and the energy values of each node in the personal view are updated. The keyword vector of a category c_i in the personal view is updated as follows:

$$V_i = \frac{\sum_{p \in P_i^{new}} V_p}{\left|P_i^{new}\right|} + \alpha * V_i \tag{3}$$

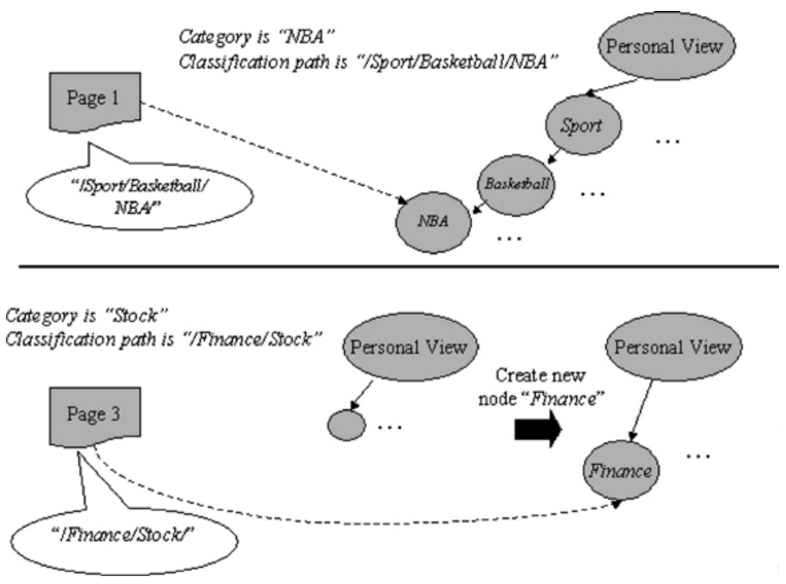


Figure 3. An example of web page insertion into a personal view.

where V_i is the keyword vector of category c_i, P_i^{new} is the set of newly inserted pages assigned to category c_i, $|P_i^{new}|$ is the number of documents in P_i^{new}, and V_p is the keyword vector of web page p. The parameter α, called the *aging factor*, is used to adjust the contribution of previously learned documents to the category. The value of α is set between 0 to 1; hence, the previously learned documents have less effect on the representation of the category. After the keyword vectors of the personal view are updated, we use the following method to calculate the new energy value of each category:

$$E_i = E_i + \sum_{p \in P_i^{new}} \cos(V_i, V_p) \tag{4}$$

where E_i is the energy value of category c_i, and $\cos(V_i, V_p)$ is the cosine similarity between the two vectors. In other words, the energy value of a category is the sum of the similarities of its documents. Calculating the energy value as the sum of the similarities between the newly inserted documents and the category is intuitive. A category is considered vital if the user is interested in documents of this category, and its energy value increases as more documents are inserted.

PVC uses the above methods to construct a personal view periodically. Like many previous works, the above function also updates the contents of user profiles incrementally. However, adjusting the knowledge contents of categories is not enough. The interests of users may change frequently. For instance, after the MLB finals, sports fans may shift their attention to the NBA. To adapt to such changes in user interests, we need a mechanism to adjust the life cycles of categories in the personal view periodically. In the following sub-section, we will introduce another component of PVA, the *Personal View Maintainer* (**PVM**), that adjusts the structure of the personal view to maintain the life cycles of the domains of user interests.

4.3. *Personal view maintainer*

After PVC inserts web pages into a personal view, PVM is used to adjust the life cycles of the categories in the personal view. Two maintenance operators, *split* and *merge*, are used to synchronize a personal view with user interests. The split operator specializes a category in a further step to reflect the user's current interests, while the merge operator generalizes a category that is now out of favor. The following sub-sections describe these two operators in detail.

4.3.1. Split. Since PVC constructs a personal view in a top-down manner, the categories in a personal view usually contain the concepts of their sub-categories. This becomes apparent for a category with a high energy value. For instance, the contents of the category "*Sport*" in the personal view of a sports fan might consist of the concepts of its sub-categories "*Basketball*," "*Baseball*," and "*Tennis*." When the user has strong interest in one concept, that concept will dominate the content of that category, and information about other concepts may fade. In order to correct this situation, PVM splits up a high energy category in order to generate a more specific category. Table 2 shows the algorithm of the split operation. First, each category's energy value is checked against a pre-defined threshold. If the category's energy value is greater than the threshold, one of its sub-categories with maximal gain after the split operation is performed is generated in the personal view. Afterwards, the keyword vectors and the energy values of both categories are updated.

The function *SplitGain* defined below measures the gain of splitting a sub-category from its parent:

$$SplitGain(C_{parent}, C_{child}) = Ent(C_{parent}) - \frac{|C_{parent-child}|}{|C_{parent}|} Ent(C_{parent-child}) \tag{5}$$

Table 2. The split algorithm of PVM.

```
1.  For each category $c$ in the personal view.
2.    While($E_c > threshold$)
3.      $c_{child} = ARGMAX_{c_{child}\ in\ c}\{SplitGain(c, c_{child})\}$;
4.      Create the category $c_{child}$ in the Personal View;
5.      Update the energy value of $c$;
6.      Calculate the energy value of $c_{child}$;
7.      Calculate the keyword vector of $c_{child}$;
8.      Update the keyword vector of $c$;
9.    End While;
10. End For;
```

where $C_{parent-child}$ is C_{parent} excluding C_{child}, the absolute value of a category returns the number of documents in the category, and the function $Ent(C)$ is the entropy value (Mitchell, 1997) of the category C and is defined as

$$Ent(C) = -\sum_{c \in C_{sub}} P(c) \ln P(c) \tag{6}$$

where C_{sub} is a set of C's sub-categories, and $P(c)$ is the ratio of the documents in category c to all the documents in C. When PVC inserts new pages into a personal view, their classification information, based on the world view, is also stored into two tables. One table keeps the number of documents per category, and the other records the document frequency (Salton, 1989) of each term. Hence, the value $P(c)$ is easily obtained by looking up the tables. Entropy is frequently used to measure the purity of a collection of data. The entropy is maximal when each sub-category in C has an equal number of documents. The entropy is minimal if all the documents in C belong to the same sub-category. The *SplitGain* function shows the entropy reduction achieved after a sub-category is split from its parent category. The sub-category with the maximal split gain is created and split from its parent in a personal view.

After a new child category is split from its parent, the keyword vectors and energy values of both categories have to be adjusted. The energy values are updated as follows:

$$\begin{aligned} E_{parent-child} &= E_{parent} * \frac{|C_{parent}|}{|C_{parent}| + |C_{child}|} \\ E_{child} &= E_{parent} * \frac{|C_{child}|}{|C_{parent}| + |C_{child}|} \end{aligned} \tag{7}$$

where E_{parent} is the energy value of the parent category before splitting, E_{child} is the energy value of the newly generated category, and $E_{parent-child}$ is the energy value of the parent category after splitting. The principle of updating energy values is based on the number of documents in each category.

Similarly, the weights of terms in keyword vectors of parent and child categories are adjusted according to document frequencies. For the child category, we use the following formula to create its keyword vector:

$$\begin{aligned} W^*_{i,child} &= W_{i,parent} * \frac{df_{i,child}}{df_{i,parent} + df_{i,child}} \\ W_{i,child} &= \frac{W^*_{i,child}}{\max_j W^*_{j,child}} \qquad (normalization) \end{aligned} \tag{8}$$

where $W_{i,child}$ and $W_{i,parent}$ are the weights of term i in the child and parent categories, respectively, and $df_{i,child}$ and $df_{i,parent}$ are the document frequencies of term i in the child and parent categories, respectively. Terms that mainly reside in the child category will have high weights in the child category.

Since information loss occurs in the parent category, we have to adjust the weights of the terms in the parent category. The following formula is used to revise the weights of the terms in the parent category:

$$\begin{aligned} W^*_{i,parent} &= W_{i,parent} * \frac{df_{i,parent}}{df_{i,parent} + df_{i,child}} \\ W_{i,parent} &= \frac{W^*_{i,parent}}{\max_j W^*_{j,parent}} \qquad (normalization) \end{aligned} \tag{9}$$

$df_{i,parent}/df_{i,parent} + df_{i.child}$ in the above formula measures the surplus value of a term in the parent category after the split operation is performed. Terms that mainly reside in the child category will have lower weights in their parent category.

4.3.2. Merge. In contrast to the split operator that highlights the categories in a personal view, the *merge* operator is the mechanism that removes the out-of-favor categories. Following the idea of the A-Life agent (Menczer et al., 1995), categories in a personal view are charged an energy cost for living. The cost is spent on grabbing food (i.e., documents inserted into the personal view) or generating offspring (i.e., splitting new categories). Categories with an unending supply of documents could continuously survive; otherwise, they will gradually become weak and finally die out. In PVA, every category's energy value will be reduced by a certain value during every period of time. When no or few documents are added to a category, its energy value will gradually decline. To keep a personal view in sync with the user's interests, categories with low energy values are removed. Before a low energy category is deleted, its knowledge is returned to its parent. Table 3 shows the algorithm of the merge operator.

This algorithm first reduces the energy value of every category periodically. Parameter β, called the *decay factor*, is used to control the recession rate. If a category's energy value is less than (or equal to) a pre-defined threshold (i.e., *th* in the algorithm), we remove the category from the personal view and merge its knowledge with its parent using the following formula:

$$\begin{aligned} W^*_{i,parent} &= W_{i,parent} * \left(1 + \frac{df_{i,child}}{df_{i,parent}}\right) \\ W_{i,parent} &= \frac{W^*_{i,parent}}{\max_j W^*_{j,parent}} \qquad (normalization) \end{aligned} \tag{10}$$

Table 3. Algorithm of decaying and merging categories in the personal view.

```
1. For each category c in the personal view
2.     $E_c = E_c - \beta$;
3.     If $E_c <= th$ then
4.         Update the keyword vector of c's parent;
5.         Remove category c from the personal view;
6.     End If;
7. End For;
```

It is interesting to note that the functions used to compute the weights in the parent's keyword vector in the split and merge operations are inverses of each other, i.e., $W_{i,parent} = merge(split(W_{i,parent}))$, as shown in the following equation:

$$\begin{aligned} merge(split(W_{i,parent})) &= W_{i,parent} * \frac{df_{i,parent}}{df_{i,parent} + df_{i,child}} * \left(1 + \frac{df_{i,child}}{df_{i,parent}}\right) \\ &= W_{i,parent} * \frac{df_{i,parent}}{df_{i,parent} + df_{i,child}} * \frac{df_{i,parent} + df_{i,child}}{df_{i,parent}} \\ &= W_{i,parent} \end{aligned} \tag{11}$$

5. Experiments

Three experiments were designed to verify the design concept and evaluate the performance of PVA. The world view and the automatic classifier of PVC were borrowed from an application (automatic news classification) of ACIRD. So far, the world view is a 3-level hierarchy that consists of 55 categories and is trained daily using news from 3 news agencies. In the first experiment, we showed the life cycle of a personal view over a three-week period, as well as auxiliary tools of PVA. Some interesting observations from this experiment are discussed below. Then, we evaluated the performance of PVA in terms of traditional IR precision and recall measurements. To show the benefit of PVA, PVA was compared with two personalization systems, WebMate and a system based on Pretschner and Gauch's paper (1999b), which only tune the weights of profile contents. The experimental results showed that modulating the life cycles of categories in the user profile could move the user profile closer to the user interests. Finally, we examined the influence of two control parameters, α and β, on the performance of PVA.

5.1. Experiment 1: An example of the life cycle of categories in a personal view

In this experiment, we observed a personal view over a 21-day period (from October 1, 2000 to October 21, 2000). We recorded the life cycles of categories in the personal view every day after a session of daily training, that is, insertion of pages of user interests into a personal view, splitting and merging a personal view. Three human experts were requested to use PVA to read news from several news agencies. Pages on which a user spent more than 2 minutes were regarded as interesting and added to the personal view. An average of 14 pages out of 60 pages per day were added into the personal view. Parameters α and β were set to be 0.95 and 0.5. The following figure shows the final structure of a personal view of a user. The degrees of brightness of categories indicate user preference, where darker means more interesting. Figure 4 shows that this user was interested in MLB, NBA and politics news.

Figure 5 shows the life cycle of a personal view, and figures 6 to 8 show each category in detail. Some interesting phenomena can be observed from these figures. Figure 6 shows that the category "*Olympics*" was split from the personal view at the beginning. This was because, due to the Olympic Games, most of the sport news was related to the Olympics. However, its

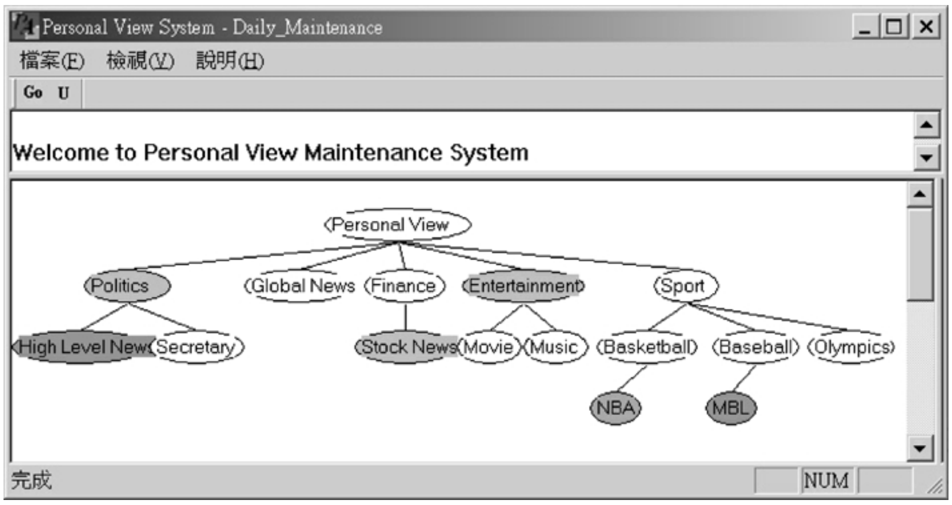


Figure 4. The personal view after 21 days.

energy value gradually decreased after Olympic Games ended on Oct. 2. Another interesting phenomenon is the growth of category "*MLB*" on Oct. 8. Its energy kept on increasing, and it finally became the hottest category in the personal view. October 8 was the first day of playoff in Major League Baseball, and this baseball enthusiast could not miss any baseball news. The growth of category "*NBA*" was similar to that of "*MLB*." Coincidentally, creation of category "*NBA*" in the personal view was tied to the pre-season period of the NBA. From this diagram, we can see that the personal view could reflect the increasing and decreasing interest in news events.

Figure 7 reveals another kind of user interest. The life cycle of these interests, unlike those of "*MLB*" and "*NBA*," are very steady. These categories can be considered as reflecting user's long-term interests, and interesting news in these categories was not restricted to a special time period.

Figure 8 shows some transient categories. One reason for the transient nature of these categories was misclassification of the world view. According to Lin et al. (1998), the **Top 1** precision of the world view was about 60%. Misclassification of pages would result in the creation of categories irrelevant to user interests. For example, news about traffic accidents was sometimes classified into the "*Travel*" category since it usually contained names of tourist spots and transportation terms. Fortunately, such misclassification happened infrequently. The energy decay mechanism and merge operator of PVA could quickly remedy this problem by eliminating these categories. Another reason was the occurrence of flash events (or short-term interests). One of the hottest events on Oct. 13 was *Kao XingJian* winning the Nobel Prize in Literature. The category "*Reading*" was generated since this news became interesting and was reported by several news agencies. However, the life cycle of categories in the personal view reveals that "*Reading*" was not an important user interest. Owing to the merge operator, "*Reading*" lasted for only a few days.

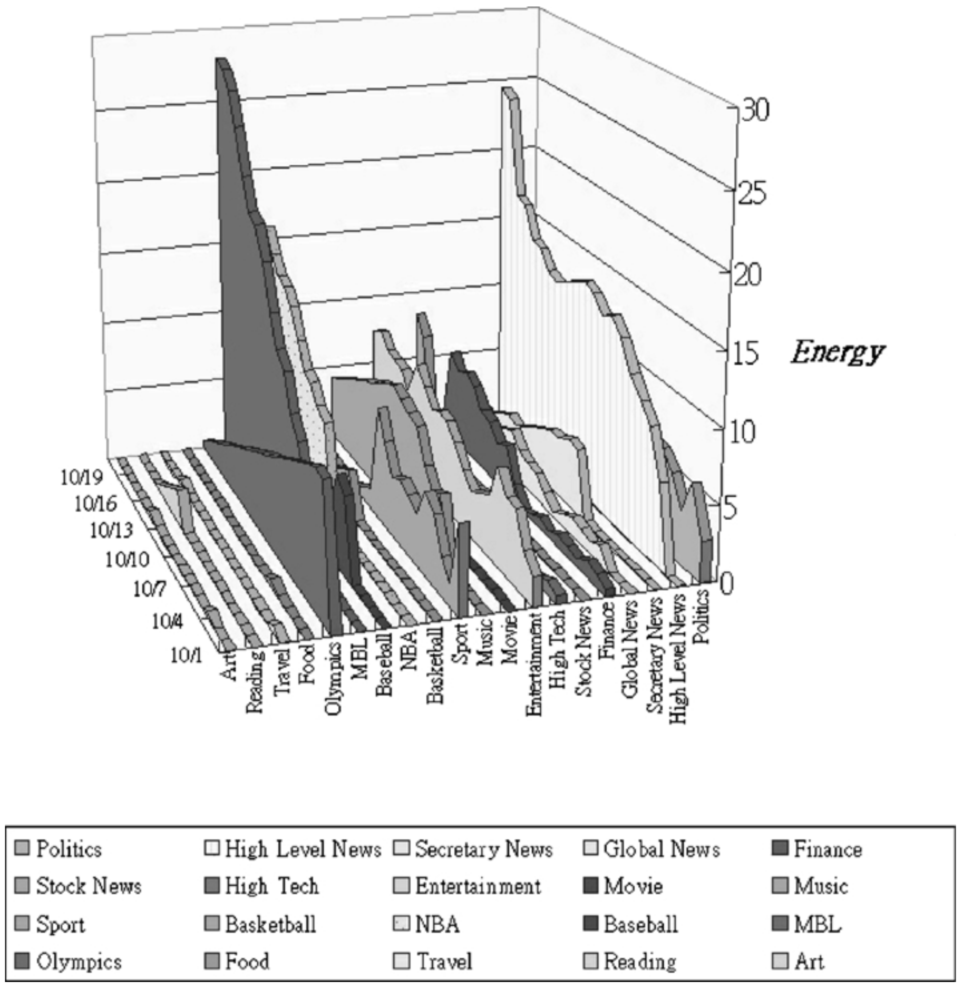


Figure 5. The life cycle of categories in a personal view.

This experiment shows the implicit and explicit effects of the split and merge operators. The split operator makes the personal view more detailed and moves it closer to current user interests, while the merge operator eliminates noise, which is unavoidable in machine learning, especially when training data is lacking.

5.2. Experiment 2: Precision and recall of a personal view

In this experiment, traditional IR measurement precision and recall were used to evaluate the performance of PVA. This experiment was based on the previous example. For each user,

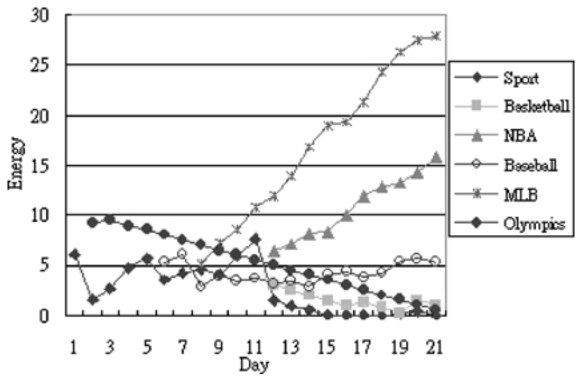


Figure 6. Life cycles of sports related categories.

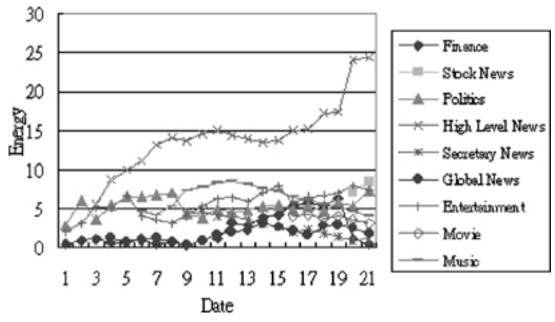


Figure 7. Life cycles of long-term interests.

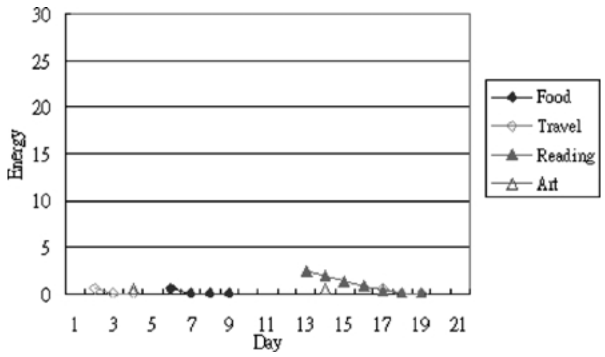


Figure 8. Growth of noise categories.

after a session of daily training, we randomly selected 100 documents of the same training day from a testing corpus to evaluate the performance of PVA. At this point, the testing corpus consisted of 10865 documents, and every document in the corpus was manually classified and marked as interesting or not interesting. The testing data were then filtered using the personal view based on the cosine similarity between documents and categories in the personal view. If the cosine similarity value was greater than a threshold, we classified the testing data into the category and called the data passed through the filter. The precision and recall functions are defined below:

$$\begin{aligned} Precision &= \frac{D_{filtered\_interesting}}{D_{filtered}} \\ Recall &= \frac{D_{filtered\_interesting}}{D_{interesting}} \end{aligned} \tag{12}$$

where $D_{filtered}$ is a subset of testing data that pass through the filter, $D_{filtered\_interesting}$ is a subset of $D_{filtered}$ that is of user interests, and $D_{interesting}$ is a subset of testing data that is of user intersts. We first examined the influence of the filtering threshold on the system performance and then compared PVA with other personalization systems. Figures 9 and 10 show the average precision and recall results of three users on different filtering thresholds.

Both precision and recall became stable after 10 days of training. From these figures, it can be observed that there was a direct ratio relationship between precision and the threshold. Precision was very high (around 70%) when the filtering threshold was 0.45. On the other hand, there was an inverse ratio relationship between recall and the threshold. Therefore, there was a tradeoff between precision and recall when a value was set for the filtering

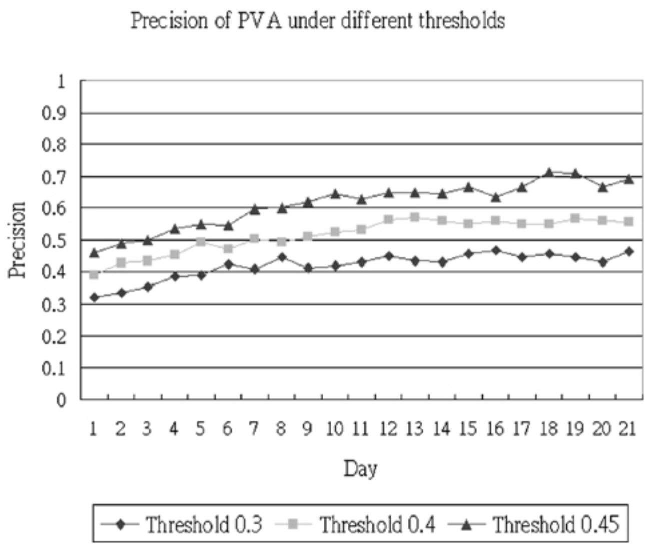


Figure 9. The precision of PVA under different thresholds.

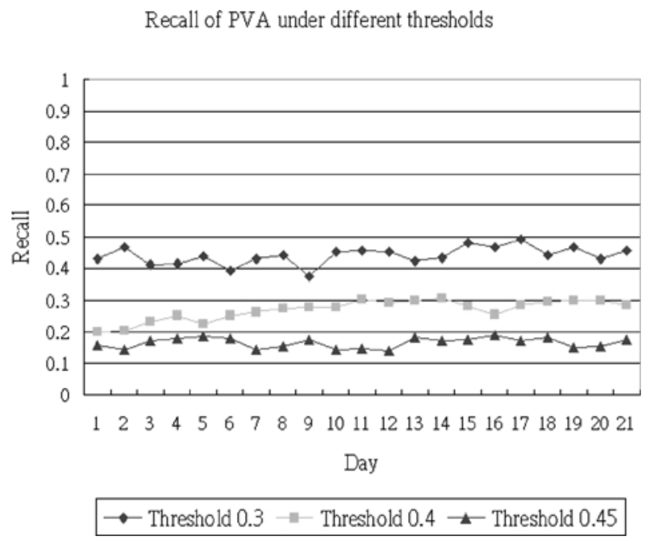


Figure 10. The recall of PVA under different thresholds.

threshold. In the following experiment, the threshold value was set to be 0.4, which resulted in precision of about 60% and recall of about 30%.

To examine the benefits of using the split and merge operators, PVA was compared with two personalization systems, WebMate and Pretschner and Gauch's system (1999b), which only adjust the contents of user profiles when user interests change. These two systems are very different since Pretschner and Gauch's system builds a very specific user profile, while WebMate produces a more general one. In contrast to these two systems, PVA constructs a user profile that is in between. All three systems were trained and tested in the same manner. The following figures show the results of comparison.

It can be observed from figures 11 and 12 that PVA performed better than the others. We present the reasons of our success in the following sub-sections.

5.2.1. Comparison with WebMate. From figure 11, we can see that the precision rate of WebMate was as expected, while its recall rate was quite low. The high precision rate of WebMate is due to the size of its user profile. As mentioned above, WebMate tries to build a more general user profile. The small number of categories in its user profile makes it more likely that training data and testing data will be correctly classified. Hence, it can achieve high precision. However, the lack of a split operator in WebMate results in the following drawbacks. (1) A topic in WebMate might comprise many concepts. For example, the "*Sports*" category might comprise the concepts MLB, NBA, NFL etc. This situation may result in a diversified keyword vector that reduces the classification correction rate. (2) The content of a category may be dominated by one topic. For example, NBA dominates the other sports in the "*Sports*" category during the basketball season. This situation may result in a low recall rate since other sports are excluded from the user profile. In PVA, topics of user interests can be specified by using split operators.

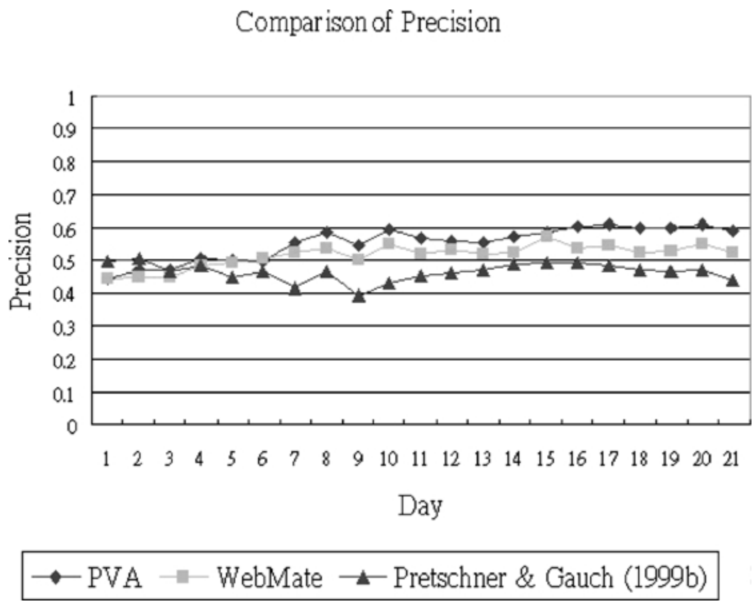


Figure 11. Precision results of the three systems.

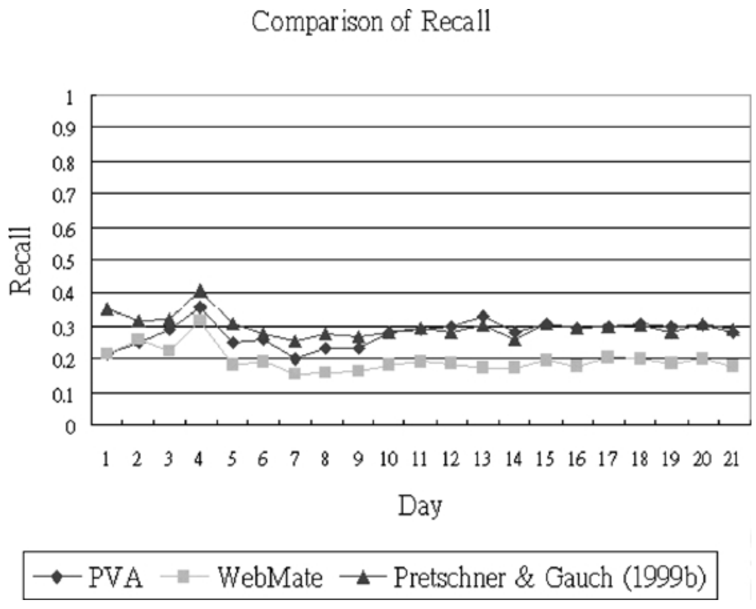


Figure 12. Recall results of the three systems.

5.2.2. Comparison with Pretschner and Gauch's system (1999b). Even though the recall rate of Pretschner and Gauch's system was quite high, its precision rate was low (and tended to worsen as time passed). The high recall rate of Pretschner and Gauch's system is due to the large size of its profile. Since Pretschner and Gauch's system tries to pinpoint a user's interests, it usually constructs a huge profile. The larger the user profile, the more topics of user interests can be covered. However, the relatively small amount of training data for each category made the knowledge of categories inaccurate, which caused the precision rate to go down as time passed. In PVA, this problem can be eliminated by using the merge operator.

By using the split and merge operators, we can enjoy the strengths of these two very different systems without inheriting their drawbacks.

5.3. Experiment 3: Effects of control parameters

In this section, we will discuss the effects of aging and decay factors (α and β) on PVA performance. The aging factor influences the knowledge content of categories, while the decay factor controls the life span of categories. We repeated Experiment 2 with 9 pairs of α and β. Parameters for these experiments were set to be $\alpha = \{0.95, 0.9, 0.85\}$, $\beta = \{0.5, 1, 1.5\}$. Each pair of parameters was evaluated five times with different testing data from the testing corpus. The following table shows the average precision and recall rates of each pair of α and β.

From Table 4, it can be seen that the precision rate increased with the increase of β. This is because a larger value of β consumes more energy, which makes it hard for the categories in the personal view to survive. This reduces the uncertainty of the personal view since all nonviable categories are removed. Therefore, documents that pass through the filter often match user interests, thus resulting in high precision. However, the recall rate goes down simultaneously. Since, with a larger value of β, a personal view is likely to be merged, the small number of categories in a personal view restricts the scope of user profiles and, therefore, lowers the recall rate. The average number of categories in a personal view was 15.2, 9.4, and 8.2 when β was 0.5, 1 and 1.5, respectively. There is a tradeoff between precision and recall for different values of β. However, the relationship between precision (and recall) and the aging factor α is not trivial since each user's interests may be long term or short term ones.

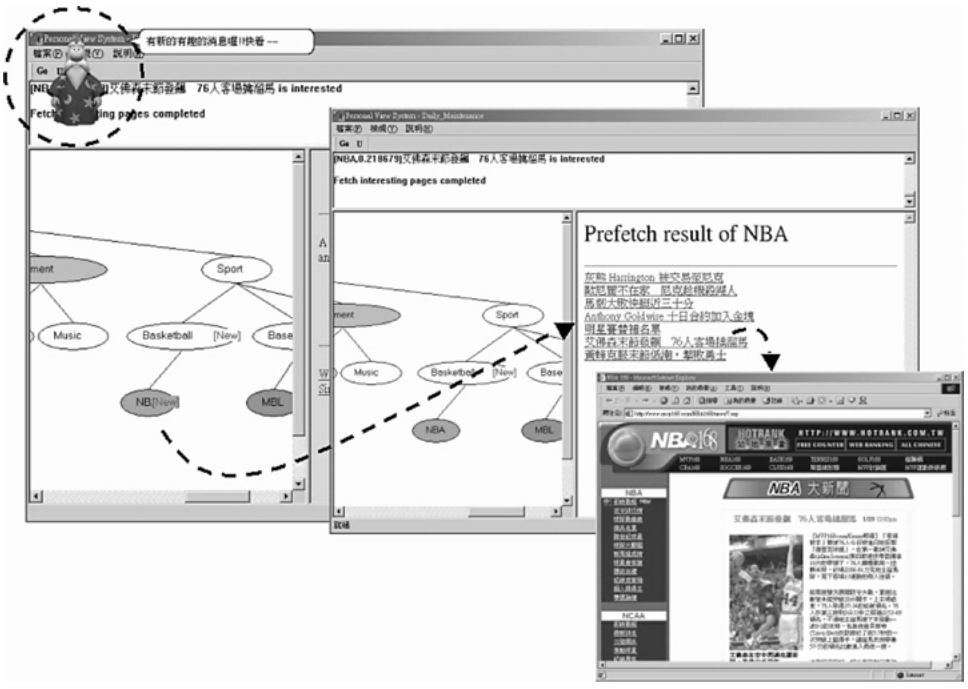
Table 4. Precision and recall rates for different pairs of α and β.

	α		
β	0.95	0.9	0.85
0.5	(0.55, 0.28)	(0.56, 0.28)	(0.54, 0.26)
1	(0.59, 0.23)	(0.58, 0.22)	(0.58, 0.22)
1.5	(0.63, 0.19)	(0.61, 0.21)	(0.61, 0.21)

6. PVA applications

The goal of web personalization is to facilitate users' activities on the Internet. Following are examples of applications that could benefit from PVA.

- *Information filtering*: Information filtering has been the major application in web personalization. Generally, an agent fetches news from several news agencies. These pages are then filtered based on a user profile. Only pages that are relevant to user preferences are selected. News Dude, WebMate, and Alipes all provide the function of information filtering.
- *Search re-ranking*: Search re-ranking systems modify the search results returned from search engines based on user profiles. Generally, the methods used to perform page ranking of search engines are based on global ranking algorithms. Re-ranking systems make personal perspectives feasible when they are employed in public search engines. Pretschner and Gauch's system is a re-ranking system.
- *Web page recommendation*: Personal WebWatcher is a recommendation system. When a user visits a web page, Personal WebWatcher analyzes the linked documents and highlights the links that the user might be interested in.


有新的有趣的消息喔!!快看～
Personal View System - Daily_Maintenance
Sport
Music
Basketball [New]
NB[New]
NBA
MBL
NBA168

[NBA,0.218679]艾佛森末節發飆 76人客場擒溜馬 is interested

Fetch interesting pages completed

Prefetch result of NBA

灰熊 Harrington 被交易至尼克
歐尼爾不在家 尼克趁機殺湖人
馬刺大敗差距近三十分
Anthony Goldwire 十日合約加入金塊
明星賽替補名單
艾佛森末節發飆 76人客場擒溜馬
賽特克談末節低潮，擊敗勇士

Figure 13. The news filtering application performed using PVA.

- *Search query expansion*: Usually, search queries typed by users are too short to describe the users' requirements. Query expansion systems refine queries based on user preferences. WebMate and WebACE utilize the information in user profiles to represent the specific information needs of users.

We implemented a prototype of an information filtering system based on PVA (see figure 13). In this system, a user can find out the status of his personal view by viewing auxiliary tools. Furthermore, a user can specify a set of URLs that are the locations of news agencies that the user likes to subscribe to. Periodically, the system checks the new links of these news agencies and calculates their relevance to the user's personal view. Pages that are close to user interests are classified into the personal view, and categories that have new information are given the suffix "[New]" to remind the user that they contain new documents. Then, the user can click on these marked categories to read the new stories. With PVA as a backbone, all the above applications and many others can be easily implemented.

7. Conclusions and future work

In this work, we have presented a system, PVA, which can learn user interests without user intervention. Problems related to modeling multiple domains of user interests and adapting to changes in user interests can be solved with PVA. Furthermore, we organize a user profile into a hierarchical category structure to provide rich semantics for applications like query extension, information filtering and web pages recommendation. The experimental results show that two personal view maintenance operators, split and merge, which adjust the life cycles of categories in user profiles, can improve the performance of personalization systems.

There are various types of life cycles of categories, as shown in the experiments. In the future, we will investigate these different types of life cycles and apply the results to identifying and defining long-term and short-term user interests, as well as the knowledge representation of a personal view, in order to better describe personal interests.

Acknowledgments

The authors would like to thank anonymous reviewers for their valueable comments. The work was partially supported by NSC under grant NSC 89-2213-E-001-008 and NSC 90-2213-E-001-021.

References

Billsus, D. and Pazzani, M.J. (1999). A Personal News Agent that Talks, Learns and Explains. In *Proceedings of the Third International Conference on Autonomous Agents*, Seattle, WA (pp. 268–275).

Chan, P.K. (1999). A Non-Invasive Learning Approach to Building Web User Profiles. In *Proceedings of KDD-99 Workshop on Web Usage Analysis and User Profiling*, San Diego, CA (pp. 7–12).

Chen, H. and Dumais, S. (2000). Bringing Order to the Web: Automatically Categorizing Search Results. In *Proceedings of the CHI 2000 Conference on Human Factors in Computing Systems*, Seattle, WA (pp. 145–152).

Chen, L. and Sycara, K. (1998). WebMate: A Personal Agent for Browsing and Searching. In *Proceedings of the Second International Conference on Autonomous Agents*, Minneapolis, MN (pp. 132–139).

Goecks, J. and Shavlik, J. (2000). Learning Users' Interests by Unobtrusively Observing Their Normal Behavior. In *Proceedings of the 2000 International Conference on Intelligent User Interfaces*, New Orleans, LA (pp. 129–132).

Han, E.H., Boley, D., Gini, M., Gross, R., Hastings, K., Karypis, G., Kumar, V., Mobasher, B., and Moore, J. (1998). WebACE: A Web Agent for Document Categorization and Exploration. In *Proceedings of the Second International Conference on Autonomous Agents*, Minneapolis, MN (pp. 408–415).

Hijikata, Y. (1999). Estimating a User's Degree of Interest in a Page during Web Browsing. In *Proceedings of IEEE Systems, Man, and Cybernetics*, Tokyo, Japan (pp. 105–110).

Hoashi, K., Matsumoto, K., Inoue, N., and Hashimoto, K. (2000). Document Filtering Method Using Non-Relevant Information Profile. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Athens, Greece (pp. 176–183).

Klinkenberg, R. and Renz, I. (1998). Adaptive Information Filtering: Learning Drifting Concepts. In *AAAI98/ICML-98 Workshop Learning for Text Categorization*.

Korfhage, R.R. (1997). *Information Storage and Retrieval*. New York, NY: Wiley Computer Publishing.

Li, W.S., Vu, Q., Chang, E., Agrawal, D., Hirata, K., Mukherjea, S., Wu, Y.L., Bufi, C., Chang, C.C.K., Hara, Y., Ito, R., Kimura, Y., Shimazu, K., and Saito, Y. (1999). PowerBookmarks: A System for Personalizable Web Information Organization, Sharing, and Management. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, Philadelphia, PA (pp. 565–567).

Lin, S.H., Shih, C.S., Chen, M.C., Ho, J.M., Ko, M.T., and Huang, Y.M. (1998). Extracting Classification Knowledge of Internet Documents with Mining Term Associations: A semantic Approach. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Melbourne, Australia (pp. 241–249).

Lin, S.H., Shih, C.S., Chen, M.C., Ho, J.M., Ko, M.T., and Huang, Y.M. (1999). ACIRD: Intelligent Internet Documents Organization and Retrieval. Technical Report, IIS, Academia Sinica. Also in *IEEE Transactions on Knowledge and Data Engineering*.

Menczer, F., Belew, R.K., and Willuhn, W. (1995). Artificial Life Applied to Adaptive Information Agents. In *Working Notes of the AAAI Symposium on Information Gathering from Distributed, Heterogeneous Databases*. Menlo Park, CA: AAAI Press.

Mitchell, T.M. (1997). *Machine Learning*. Boston, MA: WCB McGraw-Hill.

Mladenic, D. (1999). Machine Learning Used by Personal WebWatcher. In *Proceedings of ACAI-99 Workshop on Machine Learning and Intelligent Agents*, Chania, Greece.

Mobasher, B., Dai, H., Luo, T., Nakagawa, M., Sun, Y., and Wiltshire, S. (2000a). Discovery of Aggregate Usage Profiles for Web Personalization. In *Proceedings of the Web Mining for E-Commerce Workshop*, Boston, MA.

Mobasher, B., Cooley, R., and Srivastava, J. (2000b). Automatic Personalization Based on Web Usage Mining. *Communications of the ACM*, 43(8), 142–151.

Pretschner, A. and Gauch, S. (1999a). Personalization on the Web. Technical Report ITTC-FY2000-TR-13591-01, Information and Telecommunication Technology Center (ITTC), The University of Kansas, Lawrence, KS.

Pretschner, A. and Gauch, S. (1999b). Ontology Based Personalized Search. In *Proceedings of 11th IEEE International Conference On Tools with Artificial Intelligence*, Chicago, IL (pp. 391–398).

Salton, G. (1989). *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Reading, MA: Addison-Wesley.

Widyantoro, D.H., Ioerger, T.R., and Yen, J. (1999). An Adaptive Algorithm for Learning Changes in User Interests. In *Proceedings of the Eighth International Conference on Information Knowledge Management*, Kansas City, MO (pp. 405–412).