Learning for Automatic Personalization in a Semantic Taxonomy-Based Meta-Search Agent

Abstract

Providing highly relevant page hits to the user is a major concern in Web search. To accomplish this goal, the user must be allowed to express his intent precisely. Secondly, page hit rating mechanisms should be used that take the user's intent into account. Finally, a learning mechanism is needed that captures a user's preferences in his Web search, even when those preferences are changing dynamically. To address the first two issues, we propose a semantic taxonomy-based meta-search agent approach that incorporates the user's taxonomic search intent. It also addresses relevancy improvement issues of the resulting page hits by using user's search intent and preference-based rating. To provide a learning mechanism, we first propose a connectionist model-based user profile representation approach, which can leverage all of the features of the semantic taxonomy-based information retrieval approach. A user profile learning algorithm is also devised for our proposed user profile representation framework by significantly modifying and extending a typical neural network learning algorithm. Finally, the entire methodology including this learning mechanism is implemented in an agent-based system, WebSifter II. Empirical results of learning performance are also discussed.

Keywords:

Personalization; Meta-Search Engine; Machine Learning; Taxonomy; Agent; Information Retrieval

1. Introduction

With the advent of the Internet and the Web, the amount of information available grows daily. However, having too much information at one's fingertips does not always mean high quality information, in fact, it may often prevent a decision maker from making sound decisions, by degrading the quality of the decision. Helping decision makers to locate relevant information in an efficient manner is very important both to the person and to an organization in terms of time, cost, data quality and risk management.

Although search engines assist users in finding information, many of the results are irrelevant to the decision problem. This is due in part, to the keyword search approach, which does not capture the user's intent, what we call meta-knowledge. Another reason for irrelevant results from search engines is a "semantic gap" between the meanings of terms used by the user and those recognized by the search engines. In addition, each search engine has its own proprietary and uncustomizable ranking system, where users cannot specify search and ranking preferences to a search engine. For example, a shopping agent may go for the lowest price, while the user might want the "most flexible return policy." Finally, most search engines lack learning capabilities to adapt and personalize user preferences. They cannot track large numbers of users. The personal agent approach can help to solve these problems.

To address the first three of these four problems, we proposed a semantic taxonomy-based personalizable meta-search agent approach in our previous research [1-3]. In this approach, we develop a tree-structured representation scheme with which users specify their search intent. We called this representation scheme the "Weighted Semantic Taxonomy Tree (WSTT)", in which each node denotes a concept that pertains to the user's problem-domain. To address the second weakness, we present an elaborate user preference representation scheme based on various components, each of which represents a specific decision-criterion. Users can easily and precisely express their preference for a search using this representation scheme.

In order to rate the relevance of a page hit, we use a rating mechanism combining the WSTT and the component-based preference representation. Since Web page rating can itself be viewed as a decision-making problem, where a decision maker (a user) must evaluate various alternatives (Web pages) for his/her problem (user's Web search intention), we use decision-analytic methods in the design of our rating mechanism.

The search performance of the WSTT and preference components based meta-search agent approach has been validated empirically against a leading meta-search engine and well-known search engines. Although our approach improves upon other personalization techniques, it suffers from the fact that the personalization features must be specified manually. In this paper, we propose a learning mechanism for the adaptive personalization of both the user's search intent as well as the user's ranking preferences.

Most of the typical search engines and research use a term-frequency vector as part of the user profile to learn the user's behavior (preferences and search intent), while our approach provides a richer search intent and preference representation scheme. Moreover, we feel that if we adopt our search intent and preference representation scheme as a basis of user profile representation and provide a user profile learning mechanism based on that representation model, we secure an automatic personalization method in our framework. We also improve information retrieval performance over previous information retrieval user profile learning methods. To achieve this goal, we proposed a connectionist model-based user profile representation scheme, which can leverage all features of the semantic taxonomy-based personalizable meta-search agent approach and its learning mechanism that is extended and modified from the well-known neural network learning rule, the generalized delta rule [4].

Finally, we have designed and implemented our learning scheme as a component system in a meta-search agent called WebSifter II [3]. For the empirical validation of our approach, we also present some real world examples of our system.

The remainder of the paper is organized as follows. Section 2 presents related research. Section 3 reviews the major aspects of our semantic taxonomy-based approach to represent user intention, and the multi-component-based rating of search hits. We present our connectionist model-based learning method for the personalization under the pre-proposed semantic taxonomy-based meta-search agent framework in Section 4. Section 5 presents a brief introduction to the WebSifter II system and the role of our personalization components in the entire system. Implementation issues and the results of empirical studies are presented in Section 6.

2. Related Work

We address the related work in terms of two different aspects, search enhancement itself and learning issues to achieve this goal.

2.1 Related Work for Search Enhancement

Most current Internet search engines such as Yahoo, Excite, AltaVista, WebCrawler, Lycos, Google, etc. suffer from *Recall* and *Precision* problems [5]. The relatively low coverage of individual search engines leads to using meta-search engines to improve the recall of a query. Examples are MetaCrawler [6], SavvySearch [7], NECI Metasearch Engine [8], and Copernic (http://www.copernic.com). This meta-search engine approach partly addresses the recall problem but still suffers from the precision problem.

We can categorize research regarding the precision problem into three major themes: content-based, collaborative, and domain-knowledge approaches.

The content-based approach first represents a user's explicit preferences and then evaluates Web page relevance in terms of its content and user preferences. Syskill & Webert [9], WebWatcher [10], WAWA [11], and WebSail [12] fall into this category. Further, some research takes into account not only Web page content but also its structure (e.g. hyperlinks) to evaluate relevance [13, 14].

The collaborative approach determines information relevancy based on similarity among users rather than similarity of the information itself. Example systems are Firefly and Ringo [15], Phoaks [16], and Siteseer [17]. In addition, some hybrid approaches incorporate both approaches for example Fab [18], Lifestyle Finder [19], WebCobra [20].

The domain knowledge approach uses user and organizational domain knowledge to improve the relevancy of search results. Yahoo! uses domain knowledge and provides a pre-defined taxonomy path. So, classifying Web pages automatically into a pre-defined, or a dynamically created taxonomy [21] is a related issue to this approach. NorthernLight (www.northernlight.com) is a search engine that supports this kind of dynamic taxonomy service.

Some research incorporates user domain knowledge in a more explicit way. For example, Aridor et al. [22] represent user domain knowledge as a small set of example Web pages provided by users. Chakrabarti et al. adopted both a pre-defined (but modifiable) taxonomy and a set of example user-provided Web pages as domain knowledge [23].

From this survey of related research, we have identified several aspects that merit further consideration. First, most approaches force users to use a search engine in a passive rather than active manner. Often, the user cannot understand why extraneous and irrelevant results are retrieved. There is a pressing need for users to be able to express their query intent in a more natural and structured manner. Second, current approaches lack sufficient expressive power to capture a users' search intent and preferences, because most of the representation schemes are based on a vector space model [24] or its variants. Third, most approaches do not take full advantage of domain-specific knowledge with which to scope the search, filter the hits, and classify the query result.

2.2 Personalization by Learning in Web Search

Several approaches mentioned in the previous section incorporate a learning component to enhance search precision by tracking and capturing user feedback or behavior. Generally, their learning features can be further classified in terms of three aspects: 1) the user profile representation scheme for learning, 2) the user feedback mechanism, and 3) the learning algorithm used.

We focus only on the first aspect, the representation of the user profile, because its impact on the personalization performance is much greater than the others. So far, the most popular user profile representation scheme is the word frequency vector, which originated from the vector space model. Skill

& Webert, WebWatcher, WebSail, Siteseer, Fab, Amalthaea [25], Alipes [26], and SIFT [27] are example systems which use the word frequency vector as their basis for user profile representation.

But as indicated in [28], learning a user profile based on the word frequency vector may result in many biased page hits when using it for searching and rating. To overcome this limitation in using the vector space model, many researchers have tried to extend the vector space model or incorporate other ways to represent the user profile. For example, ifWeb [29] and SiteIF [30] use semantic networks and PSUN [28] uses a kind of associative network of words. Also, SmartPush [31] and OBIWAN [32] use an ontology, in the form of a hierarchical concept tree.

Even though these approaches improve user profile representation, their user profile representation schemes are still based on word frequencies and word ordering. Therefore, similar limitations mentioned at the end of the previous section are observed as follows. First, there are many important aspects in user profile for Web search, which cannot be easily represented by using only the word frequency concept. By incorporating collaborative filtering through the learning of other users improvements are made in the value of the results [33]. User's preference for authority and popularity of page hit provide for this collaborative learning. Second, even when using domain knowledge such as an ontology, the domain knowledge is fixed and not user-generated. The ontology's perspective of the domain drives the user toward a specific result. When others construct the ontology it tends to achieve the ontology's preferred result. Whereas, if the ontology is user made it will arrive at, or closer to, the user's goal [34].

To maximize personalization, domain knowledge should be extracted from users and used to represent a part of the user profile. In fact, our previous research [1-3] partly addressed this user profile representation issue and proposed a sophisticated scheme for these two requirements. In this paper, we address how we can use this scheme as a user profile and how to learn from user feedback and adaptively reflect it in the profile. Before discussing our learning mechanism, we first briefly discuss our user's search preference representation scheme.

3. Semantic Taxonomy-Tree-Based Approach for Personalized Information Retrieval

3.1 Weighted Semantic Taxonomy Tree

Usually a keyword-based search representation is insufficient to express a user's search intent. By postulating a user's decision-making process as depicted in Figure 1, we can readily support query formulation and search.

Figure 1 here

This process starts with a problem identification phase and then a user seeks relevant information to solve the identified problem. Based on the collected information, listing alternatives, evaluating them, and selecting a solution are the subsequent steps. One implication of the decision-making process is that the more we understand a user's problem, the better we can support a user's information search. In our approach, we represent a user's search intent by a hierarchical concept tree with weights associated with each concept, thereby reflecting user-perceived relevance of concepts to the search.

Let's assume that a person has started a new business and is looking for office equipment. He wants to search for information about office equipment on the Web. Suppose he wants information about chairs, so he might build a query using a single term, "chair". If he were a more skilled user of Internet search engines, he might build a query using two terms, "office" and "chair" to obtain more precise results. He may also use the 'AND' or 'OR' operator between them. In this case, the term "office" provides added context for the search. However, this formulation is still very implicit and passive. One way to express this kind of context information is by using a taxonomy tree as shown in Figure 2. Figure 2(a) shows a simple taxonomy tree that represents a search intention to find a chair in the context of office, while a search for finding an office in the context of chair is expressed by Figure 2(b). The taxonomy tree provides more expressive semantics than simple keyword-based representations used by most current search engines.

Figure 2 here

The taxonomy tree approach is already used to classify pages in search engines such as Google and Yahoo! We have devised a tree-based search representation model that allows users to present their search intention by defining their own taxonomy topology. We call this the *Weighted Semantic Taxonomy Tree* (WSTT) model. The method to build such a tree is very similar to the approach in a conventional AHP approach, which has been shown to be a convenient way for a general user to describe his decision criteria in Decision Science. To build a WSTT model, a user defines a broad concept term that includes the term he seeks. Then he continues to refine that concept, specializing it by adding child nodes with more specific concept terms. At the same time he is assigning the relative importance levels to each concept term between sibling nodes. Through this procedure, users can build their own hierarchical taxonomy tree, and assign importance levels to each term within the context of their antecedent terms. In

the context of organizational ontologies, we note that organizations can specify focused ontologies that can be used by users to formulate search requests. They may also extend the organizational terms by specializing the terms as discussed above.

Figure 3 shows a realistic example of the businessman's search intention using our WSTT scheme. For example, we can translate the upper sub-tree as that a businessman wants to find information about chairs, desks, and phones within the context of office furniture and office equipment where the numbers that appear to the left to each term, 10, 9, and 6 denote his respective importance levels assigned for chairs, desks, and phones.

One drawback is that the terms may have multiple meanings, and this is one of the major reasons that search engines return irrelevant search results. To address this issue, we introduce the notion of "word senses" from WordNet [35] into our WSTT scheme to allow users to refine their search intention.

Figure 3 here

WordNet is a linguistic database that uses sets of terms that have similar semantics (*synsets*) to represent word senses. Each synset corresponds to terms with synonymous meaning in English and so each word may be associated with multiple synsets. In this paper, we rename this synset as *Concept* for our own use and the user can choose one of the concepts available from WordNet for the term of a specific node in WSTT. For example, the "chair" term has the following four possible concepts from WordNet.

(1) {chair, seat} // a seat for one person, with a support for the back,

(2) {professorship, chair} // the position of professor, or a chaired professorship,

(3) {president, chairman, chairwoman, chair, chairperson} // the officer who presides at the meetings of an organization, and

(4) {electric chair, chair, death chair, hot seat} // an instrument of death by electrocution that resembles a chair.

If the user wants to search for a chair to sit on, he would choose the first concept. If the user selects the first concept, then without loss of generality, we can assume that the remaining concepts are not of interest, thereby obtaining both positive and negative indicators of his intent. Now, let's distinguish the set of terms of the selected concept from the set of terms of the unselected concepts as *Positive Concept Terms* and *Negative Concept Terms*, and denote them as $pct(n)$ and $nct(n)$ for a node n, respectively. If a user selects the first concept from our example, according to the definitions, $pct(n)$ and $nct(n)$ are as

follows: *pct*(*n*) = {chair, seat} and *nct*(*n*) = {professorship, president, chairman, chairwoman, chairperson, electric chair, death chair, hot seat}.

For our office equipment example, Figure 4 shows an internal representation of the user's intention via the WSTT schema, after the concept selection process has finished; the user however sees the tree of Figure 3. Another advantage using the tree structure is that it is possible to represent many concepts at the same time. This allows the user to specify a broad range of interests simultaneously.

Figure 4 here

3.2 Multi-Attribute-Based Search Preference Representation

The ranking of Web search hits by users involves the evaluation of multiple attributes, which reflect user preferences and their conception of the decision problem. In our approach, we pose the ranking problem as a multi-attribute decision problem. Thus, we examine the search results provided by multiple search engines, and rank the pages according to multiple decision criteria. Multi-Attribute Utility Technology (MAUT) [36] and Repertory Grid [37] are two major approaches that address our information evaluation problem. Our ranking approach combines MAUT and the Repertory Grid. We define six search evaluation components as follows:

(1) *Semantic* component: represents a Web page's relevance with respect to its content.

(2) *Syntactic* component: represents the syntactic relevance with respect to its URL. This considers URL structure, the location of the document, the type of information provider, and the page type (e.g., home, directory, and content).

(3) *Categorical Match* component: represents the similarity measure between the structure of the user-created taxonomy and the category information provided by search engines for the retrieved Web pages.

(4) *Search Engine* component: represents the user's biases toward and confidence in search engine's results.

(5) *Authority/Hub* component: represents the level of user preference for *Authority* or *Hub* sites and pages. Authority sites usually have larger in-degree from Hub sites and Hub sites usually have larger out-degree to Authority sites [38]. This component is not yet implemented.

(6) *Popularity* component: represents the user's preference for popular sites. The number of visitors or the number of requests for the specific page or site can measure popularity.

Further, in this multi-component-based preference representation scheme, the user can assign a preference level to each of these components, and also to each available search engine within the search

engine component. Then, these components and the assigned preference level are eventually synthesized into a single unified value resulting in the relevance measure for a specific Web page. Figure 5 conceptually depicts our scheme. In this figure, each number assigned to an edge denotes the user's preference level for that component. This multi-component preference scheme allows users more control over their searches and the determination of a page's relevance.

Figure 5 here

Thus far, we have discussed how to capture and represent semantically the user's search intention and search preferences. Now, we turn our attention to deriving a good estimate of the relevancy of a Web page based on these semantics. In the following sections, we will discuss briefly how to obtain Web information using existing search engines and then address the derivation of relevance estimates.

3.3 Gathering Web Information based on Search Intention

Since we adopt a meta-search approach to Web information gathering to preserve the benefits of meta-search engines discussed in [6, 7, 22], we neither create nor maintain our own index database of Web information. At present, there is no search engine that accepts a search request based on the WSTT. We have developed a translation mechanism from our WSTT-based query, to Boolean queries that most of current search engines can process.

As already mentioned, we represent a user's search intention as a tree, as shown in Figure 4. The leaf nodes denote the terms of interest to the user, and the antecedent nodes for each node form a search context. We transform the entire tree into a set of separate queries where each is acceptable to existing search engines. To do this, first we decompose the tree into a set of paths from the root to each leaf node. Then for each path, we generate all possible combinations of terms, by selecting one term from the positive concept terms of each node in the path from a root node to a leaf node. Finally, we obtain the resulting page hits from the search engines by posing each query to them.

3.4 Unified Web Information Rating Mechanism

Each resulting page hit from the target search engines for the generated query statements is evaluated for each search evaluation component. Six relevance values of each Web page are computed first, and then a composite value of these six relevance values is computed based on a function of the multi-attribute-based search preference representation scheme. Through this rating mechanism, each Web page will have its own value representing the relevance level from the user's viewpoint. To perform these series of evaluation processes, we first define each evaluation component as a formal quantifiable measure and

also devise the methods to compute relevance value in terms of each component. In addition, we develop a synthesizing mechanism of relevance values from the components into a single unified relevance value, which becomes an ultimate criterion in providing the relevance information to user. In this paper, we have not addressed the detailed discussion about the rating mechanism. The reader may refer to [39] for details about this issue.

3.5 Experimental Evaluation of WSTT Retrieval Performance

The validation issue of the WSTT-based query representation with multi-attribute-based search preference representation and the performance of the rating mechanism is not the main focus of this paper. Nevertheless, overall retrieval performance of our approach is summarized in Table 1. In our previous work [39], our evaluation approach measured the hit ratio, which is the percentage of the relevant page-hits to the 20 highest ranked pages returned from the search engines. Three different experiments were performed for validation. Table 1 shows the average hit ratio from those experiments performed in [39] and compares our approach with the performance of a commercial meta-search engine Copernic as well as four major search engines.

Table 1 here

As shown in Table 1, our retrieval approach, WebSifter, outperforms other approaches with significant differences. One interesting fact shown in the table is that the leading meta-search engine Copernic shows relatively poorer performance than other search engines. This seems to be because most of the relevant page hits have low-rankings from the search engines, and meta-search engines such as Copernic tend to consider those highly-ranked page hits from other search engines in determining its overall page ranking.

Concerning the time complexity of our retrieval approach, the query processing time increases exponentially with the size of the WSTT, especially the number of nodes and the number of concepts in each node. If we assume the WSTT is a full binary tree and there are N nodes having fixed C number of concepts, time complexity increases proportional to $((N+1)/2)\cdot C^{\log_2(N+1)}$, where ($N$+1)/2 is the number of paths and $C^{\log_2(N+1)}$ is the number of combinations of the concepts per each path. Consequently, the time complexity in our case follows $O\left(N \cdot C^{\log_2 N}\right)$ and this fortunately means the processing time much smoothly increase than the pure exponential case such as O(C^N). Furthermore, it takes only 30 seconds to 1 minute on average to obtain the results for the WSTT shown in Figure 3. We feel that the average user will not create WSTTs that are more complex than that depicted in Figure 3. Therefore that time

complexity for query processing is balanced by the increased precision provided by the WebSifter meta-search agent.

The performance results of our approach indicate that our representation scheme for a user's search queries and preferences provides a better alternative for user profile representation for personalization than other methods such as the vector space model. But a user profile representation must be adaptable in a systematic way to reflect a user's behavior and to incorporate user feedback. The next section presents a formal user profile representation that can be adapted systematically by means of a learning mechanism.

4. Learning for Personalization

So far, we have discussed how to represent a user's search query and preferences, as well as how to rate the resulting page hits. However, these are not sufficient in several respects. First, even though a user may represent his preferences and search queries correctly, his search intent and preferences may change over time and this means the initial information about the user becomes stale. Second, as we provide users with more sophisticated tools, there is the chance that they will not be understood or they may be misused. Therefore, we propose a method that not only captures a user's search intent and preferences but also incorporates the acquired information into the profile's representation, on a continuous basis. To accomplish this, we define a user profile in a formal way and devise a feedback and learning mechanism.

4.1 User Profile

In this semantic taxonomy-based meta-search agent approach, we have a search intent representation, which consists of terms and associated weights organized in a hierarchical structure. We also have several types of search preferences such as component preference, search engine preference, and syntactic rule preference. This relatively more sophisticated representation scheme for a user's search intent and preferences allows users to present requirements more precisely than previous research, which considers only words or terms and their relationships in documents. Its retrieval performance has been empirically validated against a leading meta-search engine and several major search engines, although the experiments performed and initial results are preliminary.

These two observations imply that if such a representation approach could be used as a basis of user profile representation and learning, it would capture more accurately changes in a user's search intent and preferences thereby improving retrieval performance. Now, we summarize the available information in Section 3 for the construction of a user profile.

- Weights on the preference components, $cw^N(com)$: represent how important a user thinks each search preference component *com* is in his search.
- Weights on the term nodes in the WSTT, $tw^N(n)$: represent the importance a user places on each concept node *n* in his search.

- Weights on the search engines, *sw*(*s*): represent the importance a user attributes to each search engine *s* in his search.
- Weights on the syntactic Web page classification rules, *rsc*(*r*): represent how much a user prefers a certain syntactic matching rule *r* in his search.
- Weights on the parameters in the semantic and categorical match component relevancy computation, θ and α: θ denotes a rate a user wants to consider the irrelevancy measure in his semantic rating and α denotes how much a user prefers the co-occurrence level to the order consistency level in the categorical match rating. Readers can refer to the more detailed mathematical definitions of θ and α in [39].

However, this listing of the relevant information vectors and the parameters itself is not enough to represent the user profile in our approach, because that list of vectors and parameters is organized to eventually produce a relevancy measure for a document. The information about how those vectors and parameters are organized is also a part of the user profile. In this sense, our approach is distinguished from an approach such as the Rocchio algorithm [40], which uses a simple word (or word frequency) vector as a user profile. Rather, it is closer to the approaches that use a structural computation model such as the neural network [41] and the belief network [42]. However, those two latter approaches are still based on the vector space model and incorporate only terms and their relationships into the structural organization of the user profile model, while our approach incorporates not only terms but also additional factors such as the hierarchical structure of the terms, various preference components, etc. into a more comprehensible structural user profile model.

Therefore, we define a user profile as the list of profile vectors and parameters mentioned above together with their structural organization of the terms showing how they affect each other in deciding the relevancy of a document to the user profile. We adopt a connectionist model [43] and local representation method in this model to represent the user profile in our approach. Local representation here means that each node in the model designates a symbolic concept. This differs from simply adopting feed-forward neural network that assumes a black box, which implies we cannot know the meanings inside the network. One of the papers adopting such an approach is [41]. In our case, however, our model is a white box so that we can understand explicitly every node and every weight in the model. This representation allows us to learn each parameter used in WSTT-based information search framework and eventually to leverage all features from the semantic taxonomy-based meta-search engine approach.

Since our user profile representation is dependent on the WSTT, let's assume an example WSTT shown in Figure 6, where there are five terms t_is and their associated normalized weights $tw^N{}_i$s organized hierarchically. Then the resulting user profile representation can be established as shown in Figure 7.

Figure 6 here

Figure 7 here

Figure 7 depicts the components of the user profile that are used to compute the total relevancy of a document, within the connectionist model. In addition, the overall diagram shows how these user profile components affect each other. In this representation of a user profile, a dotted line stands for a fixed connection between nodes while the solid line stands for an adjustable connection. The arcs with no associated weight symbol means its weight is 1.

Now, let's briefly explain how each part of the diagram is organized. The first layer in Figure 7 starting from the left stands for our component relevance value composition mechanism discussed in Section 3.4 and each node in this layer denotes a corresponding preference component. To the right side of this first layer, each sub-network represents the detailed mechanism used to obtain the corresponding component relevancy from the top to the bottom.

Furthermore, N^r, N^s, and N^t denote the number of syntactic rules, the number of search engines, and the number of the term combinations, respectively. The symbols, RLPT_i and ILNT_i stand for "relevancy level of positive terms from the *i*-th path in the WSTT" and "irrelevancy level of negative terms from the *i*-th path in the WSTT", respectively. *Rule i* indicates the relevancy level measured by the *i*-th syntactic rule. $\mathrm{CL}_{i,j}$ and $\mathrm{OC}_{i,j}$ also mean the co-occurrence level and the order consistency level between the path *i* and the category information provided by the search engine *j*, respectively. Finally, $\mathrm{RK}^N_{i,j}$ stands for a normalized rank information for the term combination *i* from the search engine *j*. The detailed discussion about the measures represented by these symbols can be found in [39]. The symbols Σ, Σ_1, Σ_2, Σ_3, Σ_4, and f^* are related to the computational issues and will be discussed when we discuss the learning mechanism. We have omitted some of the sub-networks because of space limitations. The light-shaded nodes denoted with one of Σ_1, Σ_2, Σ_3, and Σ_4, have the same sub-network as the node denoting the same symbol; we have only depicted the sub-network fully for one of such nodes that have the same symbol.

When we apply this scheme of user profile representation to a web document to measure relevancy, the set of values in the leaf nodes obtained for the document forms a document representation. So, to evaluate a web document based on the user profile, those values are first computed for each document and then they are applied to the user profile to obtain total relevancy.

We have represented the user profile using a connectionist model together with a local representation method to adapt the user profile to changes in a user's search intent and preferences. We are now ready to

incorporate a feedback and learning mechanism into the user profile representation. The following sections will discuss the issues regarding how to get the feedback from user and learning based on the obtained feedbacks.

4.2 Feedback Mechanism

Generally, user feedback can be obtained in two ways, explicitly and implicitly. In the explicit manner, a user has to describe his perceived relevancy for the resulting page hits. In the implicit manner, a user doesn't need to provide any formal responses to the resulting page hits. Instead, some automatic monitoring of the user's navigation behavior needs to be performed. Feedback in the explicit case is usually more accurate than the implicit case.

In this paper, we adopt the explicit feedback approach, but our definition of the error can be easily extended to the implicit feedback approach. In our approach, a user is asked for his judgment on the relevancy of each resulting page hit. The user chooses from "relevant" and "irrelevant". The user may also use the default value of "don't know", to indicate no particular preference.

Let's denote the relevancy error occurring in a page pg by E_{pg} and define it formally as follows:

Relevancy Error Computing Rule

$$E_{pg} = \frac{1}{2}\left(rv^{U}(pg) - rv(pg)\right)^2 \tag{1}$$

where $rv^{U}(pg)$ is user's rated relevancy on the page pg and
$rv(pg)$ is relevancy value rated by our approach.

To quantify the user's answer, we assign 1 to $rv^{U}(pg)$ when the page is relevant, and 0, otherwise. For example, if our relevancy expectation on a page pg is 0.7 and the user's reply is 1, then E_{pg} becomes $1/2 \times 0.3^2 = 0.045$.

Based on (1), we can define total relevancy error, TE, for all pages, which has user's ratings, as follows:

$$TE = \sum_{pg} E_{pg} \tag{2}$$

The objective function for the learning process is to minimize the total relevancy error.

4.3 Learning Mechanism

Before deriving a learning mechanism for our user profile representation, let's note that the model shown in Figure 7 also addresses how the overall computation of the relevancy is performed. In each node, if it is not the leaf node, the relevancy values of the child nodes are aggregated and the symbols appeared in the

node represents the aggregation method applied to it. If it is a leaf node, the relevancy value of the node is the input to the model. Each sigma symbol, Σ, indicates that the relevancy in the node is computed by the following formula (3) regardless of whether it has a subscript or not in the figure.

General Propagation Rule

$$o_{pg,j} = \sum_i w_{ji} o_{pg,i} \tag{3}$$

where $o_{pg,j}$ and $o_{pg,i}$ are the output values on the node, j and i for the case of a Web page pg, respectively and w_{ji} is a weight from the node i to the node j.

In addition, the symbol f^* means the relevancy in the node is computed by the following formula (4).

Theta Propagation Rule

$$o_{pg,j} = \left(\mathrm{RLPT}_i\right) \cdot \left(1-\theta\right)^{\mathrm{ILNT}_i} \tag{4}$$

where θ is a given [0, 1] scale degradation rate and RLPT_i and ILNT_i are the relevancy level of the connected incoming nodes to the node j.

Therefore, the user profile representation can be regarded as a feed-forward neural network model from the computational model point of view and so, we can consider the neural network learning scheme as a learning mechanism in our profile learning task. But, there are several problems that prevent us from simply applying one of the well-known neural network learning algorithms to our case.

The first problem is that our user profile representation model depicted in Figure 7 differs from the typical feed-forward neural network model in that some of the weights are overlapped and those overlapped weights are required to always have the same value. As shown in Figure 7, the parameters θ, α, and sw_i are overlapped in multiple places and this means that when we adjust each of the weights in the model to learn the user's feedback, the weights that share the same parameter must be controlled to have the same value. A similar restriction also happens in the case of WSTT. The same WSTT depicted in Figure 6 is also used three times in different places of the user profile representation model as shown in Figure 7. Therefore, the weights from different positions but sharing the same parameter, tw^N_i must always have the same value because tw^N_i cannot have multiple values at any given time.

To define this problem more precisely, we provide some definitions. For a given parameter p, if there are multiple weights that share this parameter as their value, we call this set of weights "parameter sharing weight set" of the parameter p and denote it with $PSWS(p)$. In addition, we denote the parameter that a weight w_{ji} share by $PS(w_{ji})$. For example, there are three such weights in $PSWS(\theta)$ as shown in the right-upper side of Figure 7. For all such $w_{ji} \in PSWS(p)$ for a parameter p, those weights should be

always equal to each other. Since the typical neural network learning algorithms cannot address this kind of restriction, we have to devise a way to resolve such a problem.

The second problem is that several sets of weights in the user profile representation model must obey the rule that the sum of the weights in the set should be 1 for the purpose of normalization. At first, the weight sets, $\{w_{ji} \mid PS(w_{ji}) = cw^N_k$ and $k = 1, 2, \ldots, 5\}$, $\{w_{ji} \mid PS(w_{ji}) = sw_k$ and $k = 1, 2, \ldots, N^s\}$ for each node j denoted by Σ_4, and $\{w_{ji} \mid PS(w_{ji}) = \alpha$ or $1 - \alpha\}$ for each node j denoted by Σ_3 follow this rule. In addition, each set of weights on the nodes that have the same parent in the WSTT also follows this rule. We call all these weight sets that follow the rule "normalization weight sets" and denote the set of such sets by *NWS*. To address this problem, we also need to provide a normalization method while learning is performed.

The third problem is all of the weights are bounded in a value range [0, 1] and the fourth and trivial problem is f^* is not a typical additive function but a multiplicative and exponential function, which means we have to derive a new weight updating rule for this function which is different from the conventional updating rule.

To resolve the above four problems in applying the typical neural network learning algorithm to our user profile learning task, we propose a user profile learning algorithm which embeds the generalized delta rule [4] as a core heuristic. Before discussing the overall architecture of the user profile-learning algorithm, we provide some definitions and formulae.

At first, to apply the generalized delta rule [4], we define a delta in a node j for a feedback Web page pg, $\delta_{pg,j}$ as follows:

Generalized Delta Computing Rule

$$\delta_{pg,j} = \begin{cases} rv^U(pg) - rv(pg) & \text{if } j \text{ is an output node} \\ \sum_k \delta_{pg,k} w_{kj} & \text{otherwise} \end{cases} \quad (5)$$

where k is a node in the upper layer to the layer to which the node j belongs.

Using this delta, we derive a weight-updating rule as follows:

Generalized Weight Updating Rule

$$w_{ji}^{updated} = w_{ji}^{old} + \eta \cdot \delta_{pg,j} \cdot o_{pg,i} \quad (6)$$

where η is a learning rate for the weight.

However, as mentioned before, these two rules, (5) and (6) are not enough to address all our learning requirements. We also derive a delta computing formula to address the fourth problem in the above as follows:

Delta for Theta Computing Rule

$$\delta_{pg,j}^{\theta} = \left(\sum_{k} \delta_{pg,k} w_{kj} \right) \cdot \left(- RLPT_i \right) \cdot ILNT_i \cdot \left(1 - \theta \right)^{\left(ILNT_i - 1 \right)} \tag{7}$$

where the node j is a f^* type node and RLPT_i and ILNT_i are the relevancy level of the connected incoming nodes to the node j.

Then, we use (6) for updating the parameter θ in the f^* function but since the value of the incoming node of this θ connection is always fixed to 1, its updating rule can be simplified as follows:

Theta Updating Rule

$$\theta_j^{updated} = \theta_j^{old} + \eta \cdot \delta_{pg,j}^{\theta} \tag{8}$$

where the θ_j is a corresponding θ in a f^* type node j.

In addition to these definitions and formulas, to address the first problem mentioned above, we first define a set of weights sharing the same parameter as "parameter sharing weight set" and denote it by $PSWS(p)$ for a given parameter p. Then we devise a weight synchronization procedure, which assures the weights in a $PSWS$ always have the same value as follows:

Weight Synchronization Rule

$$w_{ji}^{synchronized} = \sum_{w_{st} \in PSWS} w_{st}^{unsynchonized} \Big/ n\left(PSWS\left(p_k \right) \right) \tag{9}$$

where the $w_{ji} \in PSWS(p_k)$, p_k is a given k-th parameter, and $n(PSWS(p_k))$ is the number of elements in $PSWS(p_k)$.

We also devise a weight normalization procedure to address the normalization constraints mentioned in the second problem. The resulting weight renormalization rule is formalized as follows:

Weight Normalization Rule

$$w_{ji}^{renormalized} = w_{ji}^{before-normalized} \Big/ \sum_{w_{jk} \in NWS_i} w_{jk}^{before-normalized} \tag{10}$$

where the $w_{ji} \in NWS_i$ and $NWS_i \in NWS$.

Now, we propose a user profile learning algorithm based on the user profile representation as shown in Figure 7 using the definitions and the formulas derived thus far, while resolving all four mentioned problems. The algorithm is described in Figure 8 using pseudo code and it is organized to show the procedure to perform the learning task. Further, when the algorithm was implemented, we took computational efficiency considerations into account.

Figure 8 here

In the following sections, we will first show how the issues related to not only learning for automatic personalization but also semantic taxonomy-based meta-search agent are organized into a system and implemented. Then we also discuss the details of implementation of the proposed user profile learning algorithm and its performance.

5. WebSifter II System Architecture

In this section we present the architecture of WebSifter II. Figure 9 shows the overall architecture of WebSifter II and its components. Major information flows are also depicted. WebSifter II consists of nine subsystems and four major information stores.

Now let's briefly introduce each of the components, their roles, and related architectural issues.

Figure 9 here

1) WSTT Elicitor

The WSTT elicitor supports the entire process (see section 3.1) of specifying a WSTT in a GUI environment. A user can express his search intent as a WSTT through interactions with the WSTT elicitor. This includes building a taxonomy tree, assigning weights to each node, and choosing a concept from an available list of WordNet concepts. To achieve this goal, the WSTT elicitor also cooperates with an Ontology agent, a Stemming agent, and a Spell Check agent. Once a user finishes building a WSTT, then the WSTT elicitor stores the WSTT information into the WSTT base in XML format.

2) Ontology Agent

The ontology agent is responsible for requesting available concepts of a given term via a Web version of WordNet (http://www.cogsci.princeton.edu/cgi-bin/webwn/) and also for interpreting the corresponding HTTP-based results. The agent receives requests for the concepts from WSTT elicitor and returns available concepts in an understandable form. Although WebSifter presently supports cooperation only with WordNet, its design can be easily extended to cooperate with other ontology servers such as CYC [44] and EDR [45].

3) Stemming Agent

Our stemming agent is based on Porter's algorithm [46]. It has two major roles: 1) to cooperate with the WSTT elicitor in transforming the terms in a concept to stemmed terms, and 2) to transform the content of Web pages into the stemmed terms internally through cooperation with a page request broker. As a result, the terms in concepts and the terms in Web pages can be compared to each other via their stemmed versions.

4) Spell Check Agent

The spell check agent monitors the user's text input to the WSTT elicitor and checks and suggests correct words to the user in real time.

5) Search Preference Elicitor

The search preference elicitor, via a GUI, supports the process (cf. section 3.2) of capturing the user's search preferences. A user can express his search preference by assigning their preference weights to each of the preference components and also to their favorite search engines. Moreover, it allows the user to modify the default values assigned to each syntactic URL class such as Direct Hit, Directory Hit and Page Hit. Whenever the user modifies them, it updates the related information stored in the Personalized Evaluation Rule Base, the Search Engine Preference Base, and the Component Preference Base.

6) Search Broker

The search broker performs the processes specified in section 3.3. It first interprets the XML-based WSTT and then generates all corresponding query statements. Using this set of queries, it requests information from a set of popular search engines simultaneously. Finally, it interprets the results returned from the search engines and then stores parsed information in a temporary data store. When it finishes its work, it activates the Web page rater to begin the rating process.

7) Page Request Broker

The page request broker is responsible for requesting the content of a specific URL and it cooperates with both the stemming agent and the Web page rater.

8) Web Page Rater

The Web page rater supports the entire Web page evaluation process specified in section 3.4 and also is responsible for displaying the results to users. This subsystem is the most complex and computationally intensive module of WebSifter II, and it uses all four major information stores and also communicates with the search broker and the page request broker.

9) User Profile Learning Agent

The user profile-learning agent first allows the user to provide feedback on the relevancy of the proposed Web page hits via an interactive user interface. Then, when the user invokes learning or when the user closes the system, the learning process starts and it updates various user preference parameters to reflect

the user's feedback information. The user can instantly refresh the search results based on the updated profile or can use it in another query later. During the update, the agent modifies all four information stores in WebSifter II.

6. Implementation and Performance Evaluation

6.1 Implementation

We have incorporated the framework of our semantic taxonomy-based meta-search agent approach and its user profile learning mechanism into a working prototype written in Java, except for one component, the spell check agent. Now, we plan to incorporate a commercial spell check agent into the system.

Figure 10 shows an illustrative screen where the user builds a WSTT using the WSTT elicitor. Figure 11 shows another screen of the WSTT elicitor supporting the selection of an intended concept from available concepts for a given term, obtained through cooperation with the ontology agent and WordNet.

Figure 12 shows a sample screen for a user to specify his search preference using our search preference elicitor. The four tab windows in Figure 12 are for adjusting the user's preference for the relevance components, search engines, various parameters in our mechanism, and classification rules for Web pages, respectively. However, only the tab window for preference components is shown in Figure 12.

Figure 10, 11 and 12 here

Finally Figure 13 shows a query result screen for WebSifter II. Note that the left-most column in the table for the resulting page hits, is reserved for obtaining user relevancy feedback. Whenever a user views a URL using the browser, which is invoked by clicking the URL on the screen, he can provide his rating feedback in the corresponding row on the feedback column by choosing one of the values, relevant or irrelevant using a dropdown list box. The user may select the default value "don't know" if he does not want to rate the page or feels unsure of his rating. Once the user finishes his rating, then he can invoke the learning process by selecting the learn menu from the menu bar. If he does not want to activate the learning mechanism explicitly, the system will invoke the learning process automatically when the user closes the system. In the case of activating the learning explicitly, the search results are instantly refreshed according to the new updated user profile and system parameters.

Figure 13 here

6.2 Example Computation for Learning and Performance Evaluation

To show how the learning actually works, let's assume the cases appearing in Figure 13 and demonstrate the required computations for the weight adjustments at the first layer in Figure 6. If a user rated only the first ranked page in Figure 13, 'www.officesuppliessuperstore.com', as a relevant page and he activated the learning mechanism, then the delta value of the output node in Figure 6 can be computed as 1 – 0.325 = 0.675 according to (4) since the suggested relevancy of that page by WebSifter, was 0.325. And because our five preference component relevancy values are 0.286, 1.0, 0.0, 0.034, and 0.0, respectively as shown in Figure 13, their weight adjustment levels become as follows, if we assume η is 0.5:

$$\Delta cw^{N}_{semantic} = 0.5 \times 0.675 \times 0.286 = 0.097$$

$$\Delta cw^{N}_{syntactic} = 0.5 \times 0.675 \times 1.0 = 0.338$$

$$\Delta cw^{N}_{categorical\ match} = 0.5 \times 0.675 \times 0.0 = 0.0$$

$$\Delta cw^{N}_{search\ engine} = 0.5 \times 0.675 \times 0.034 = 0.011$$

$$\Delta cw^{N}_{popularity} = 0.5 \times 0.675 \times 0.0 = 0.0$$

We can compute the updated weights for them by adding the above adjustment levels to the old weights, where they are 0.294, 0.235, 0.235, 0.176, and 0.059, respectively from the top to the bottom in the above case. Then, the resulting new updated weights on the above five components become 0.294 + 0.097 = 0.391, 0.235 + 0.338 = 0.573, 0.235 + 0.0 = 0.235, 0.176 + 0.011 = 0.187, and 0.059 + 0.0 = 0.059, respectively. But these new updated weights violate the constraint that requires their sum to be 1, so we additionally need to apply a re-normalization process to these weights according to (10). After finishing this process, the weights for five components become 0.271, 0.397, 0.163, 0.129, and 0.040, respectively. As a result, the weight only for the syntactic component increases but all other weights for the remaining components decrease in terms of their relative importance levels in the user preferences. This process will continue until at least one of the stopping conditions is satisfied, such as the maximum number of epochs.

To evaluate the performance of the user's search profile learning mechanism demonstrated here, we performed several empirical experiments under the respective scenarios. In the experiments, we measured the performance with the level of enhancement of hit ratio, that is, the ratio obtained by dividing the number of relevant page by the total number of page hits.

Now, let's see a scenario applied to the experiment. Using WebSifter II, a user requests a simple search query "chair" and gets the retrieved page hits shown in Table 2. Initial retrieval performance is not our main concern here and the initial retrieval performance of WebSifter II has been reported in [37]. The semantic taxonomy-based approach is competitive with other semantic approaches.

Table 2 here

Table 2 shows the top 20 page hits retrieved for the search query "chair" by WebSifter and also shows its evaluation of the relevancy of each page hit in the last column. "Y" means the corresponding page hit is relevant to the user while "N" means it is irrelevant. Here we assume the user wants to retrieve the page hits only related to a chair people sit on. In this scenario, the user gives his feedback on only top five page hits as shown in the column labelled "Feedback" in the table.

Based on this scenario, we performed the user's search profile learning and obtained the new retrieved page hit results shown in Table 3. These are based on the newly obtained weight and parameter results of learning.

Table 3 here

List of the page hits in Table 3 is much different from Table 2. There are many more relevant page hits compared to the initial page hit results. One interesting thing is that the page hits containing the term "seat" were evaluated as "irrelevant" by the user. We find the learning process effectively caught this point based on the fact that most of the page hits containing the term "seat" in Table 2 are dramatically downgraded or dropped from the top 20 lists in Table 3.

In terms of page-hit ratio, the learning achieves a 25% performance improvement. The hit ratio was 30% in Table 2 (before learning) and now it is 55% in Table 3 (after learning). In addition to this experiment with the search term "chair", four more search terms were used to evaluate the profile learning performance. These additional experiments also performed comparably well as in the "chair case", and Table 4 shows the page-hit ratios results from the experiment including the "chair" case. On the average, our profile learning approach achieves a 29% performance enhancement based on five user feedbacks on the top five initially retrieved page hits. This is a very impressive improvement with user feedback on only five hits. Other ongoing experiments performed also show a similar level of supportive results to our user's search profile approach. Even though the experiments in Table 4 are still very limited,

these initial results seem to be enough to indicate that the learning approach improves retrieval performance considerably.

Table 4 here

7. Conclusions

The semantic taxonomy-based meta-search agent approach [3, 39] has been proposed to achieve two important and complementary goals: 1) to allow users more expressive power in formulating their Web searches, and 2) to improve the relevancy of search results based on the user's real intent. These goals have been achieved in the WebSifter prototype system. However, one weakness to our approach is that it does not support user profile learning for personalization, even though it can represent a user's search intent and preference well. To overcome this shortcoming and to achieve better personalization, we have proposed adding a connectionist model-based user profile representation and learning mechanism to the semantic taxonomy-based meta-search agent approach.

The learning mechanism proposed in this paper enhances the functionality of WebSifter by allowing the automatic and dynamic update of the user's profile to adjust to the user preferences based on user feedback regarding the relevance of pages returned from a Web search. The connectionist model-based approach proposed has been shown to be effective in recognizing changes in user preferences and adapting those preferences to improve search results.

Now, let's briefly summarize our contributions as follows.

We propose a user's query intent and search preference profile representation scheme in conjunction with the search-intention representation scheme, the Weighted Semantic-Taxonomy Tree and the search preference representation scheme based on the various preference components. It allows representing a user's profile of search intent and preferences in a more sophisticated manner than previous approaches based on the vector space model.

Second, we present a connectionist model-based user profile representation and learning mechanism to learn user search intent and preferences in the Web search based on the proposed user profile representation scheme. To achieve this goal, we first represent the entire rating mechanism [39] as a connectionist model adopting a local representation method and then, we devise a profile-learning algorithm based on the generalized delta rule.

Third, we have designed and implemented a user profile-learning agent as a component of the meta-search agent system called WebSifter II, which cooperates with WordNet for concept retrieval, and most well known search engines for Web page retrieval. For the empirical validation of our user profile learning approach, we performed real world experiments of our system and empirically proved validity of our approach.

References

[1] Scime, A. and Kerschberg, L., "WebSifter: An Ontology-Based Personalizable Search Agent for the Web," *International Conference on Digital Libraries: Research and Practice*, Kyoto Japan, 2000, pp. 493-446.

[2] Scime, A. and Kerschberg, L., "WebSifter: An Ontological Web-Mining Agent for E-Business," *Proceedings of the 9th IFIP 2.6 Working Conference on Database Semantics (DS-9): Semantic Issues in E-Commerce Systems*, Hong Kong, 2001, pp. 200-214.

[3] Kerschberg, L., Kim, W., and Scime, A., "WebSifter II: A Personalizable Meta-Search Agent Based on Weighted Semantic Taxonomy Tree," *Proceedings of the International Conference on Internet Computing 2001(IC'2001)*, 2001, pp. 14-20.

[4] Rumelhart, D. E., et al., "Learning Representations by Back-Propagating Errors," *Nature*, vol. 323, 1986, pp. 533-536.

[5] Lawrence, S. and Giles, C. L., "Accessibility of Information on the Web," *Nature*, vol. 400, 1999, pp. 107-109.

[6] Selberg, E. and Etzioni, O., "The MetaCrawler Architecture for Resource Aggregation on the Web," *IEEE Expert*, vol. 12, no. 1, 1997, pp. 11-14.

[7] Howe, A. E. and Dreilinger, D., "Savvy Search: A Metasearch Engine that Learns which Search Engines to Query," *AI Magazine*, vol. 18, no. 2, 1997, pp. 19-25.

[8] Lawrence, S. and Giles, C. L., "Context and Page Analysis for Improved Web Search," *IEEE Internet Computing*, vol. 2, no. 4, 1998, pp. 38-46.

[9] Ackerman, M., et al., "Learning Probabilistic User Profiles - Applications for Finding Interesting Web Sites, Notifying Users of Relevant Changes to Web Pages, and Locating Grant Opportunities," *AI Magazine*, vol. 18, no. 2, 1997, pp. 47-56.

[10] Armstrong, R., et al., "WebWatcher: A Learning Apprentice for the World Wide Web," *Proceedings of the 1995 AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments*, 1995.

[11] Shavlik, J. and Eliassi-Rad, T., "Building Intelligent Agents for Web-based Tasks: A Theory-Refinement Approach," *Proceedings of the Conference on Automated Learning and Discovery: Workshop on Learning from Text and the Web*, Pittsburgh, PA, 1998.

[12] Chen, Z., et al., "WebSail: from On-line Learning to Web Search," *Proceedings of the First International Conference on Web Information Systems Engineering*, vol. 1, 2000, pp. 206-213.

[13] Chakrabarti, S., et al., "Enhanced Hypertext Categorization using Hyperlinks," *Proceedings of ACM SIGMOD International Conference on Management of Data*, Seattle, Washington, 1998, pp. 307-318.

[14] Li, Y., "Toward a Qualitative Search Engine," *IEEE Internet Computing*, vol. 2, no. 4, 1998, pp. 24-29.

[15] Maes, P., "Agents that reduce work and information overload," *Communications of the ACM*, vol. 37, no. 7, 1994, pp. 30-40.

[16] Terveen, L., et al., "PHOAKS: a System for Sharing Recommendations," *Communications of the ACM*, vol. 40, no. 3, 1997, pp. 59-62.

[17] Bollacker, K. D., et al., "Discovering Relevant Scientific Literature on the Web," *IEEE Intelligent Systems*, vol. 15, no. 2, 2000, pp. 42-47.

[18] Balabanovic, M. and Shoham, Y., "Content-Based, Collaborative Recommendation," *Communications of the ACM*, vol. 40, no. 3, 1997, pp. 66-72.

[19] Krulwich, B., "Lifestyle Finder," *AI Magazine*, vol. 18, no. 2, 1997, pp. 37-46.

[20] de Vel, O. and Nesbitt, S., "A Collaborative Filtering Agent System for Dynamic Virtual Communities on the Web," *Working notes of Learning from Text and the Web, Conference on Automated Learning and Discovery CONALD-98*, Carnegie Mellon University, Pittsburgh, 1998.

[21] Chen, H. and Dumais, S., "Bringing Order to the Web: Automatically Categorizing Search Results," *Proceedings of the CHI 2000 conference on Human factors in computing systems*, The Hague Netherlands, 2000, pp. 145-152.

[22] Aridor, Y., et al., "Knowledge Agent on the Web," *Proceedings of the 4th International Workshop on Cooperative Information Agents IV*, 2000, pp. 15-26.

[23] Chakrabarti, S., et al., "Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery," *Proceedings of the Eighth International WWW Conference*, 1999, pp. 545-562.

[24] Salton, G., et al., "A Vector Space Model for Automatic Indexing," *Communications of the ACM*, vol. 18, no. 11, 1975, pp. 613-620.

[25] Moukas, A., "Amalthaea: Information Discovery and Filtering using A Multiagent Evolving Ecosystem," *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi Agent Technology*, London, 1996.

[26] Widyantoro, D. H., et al., "Alipes: A Swift Messenger in Cyberspace," *Proceedings of the Spring Symposium on Intelligent Agents in Cyberspace*, Palo Alto, CA, March 1999.

[27] Yan, T. W. and Garcia-Molina, H., "SIFT - A Tool for Wide-Area Information Dissemination,"

Proceedings of the 1995 USENIX Technical Conference, 1995, pp. 177-186.

[28] Sorensen, H. and Elligott, M. M., "PSUN: A Profiling System for Usenet News," *Proceedings of the CIKM'95 Workshop on Intelligent Information Agents Workshop*, Baltimore, USA, December 1995.

[29] Asnicar, F. A. and Tasso, C., "ifWeb: a Prototype of User Model-Based Intelligent Agent for Document Filtering and Navigation in the World Wide Web," *Proceedings of the workshop on Adaptive Systems and User Modeling on the World Wide Web (Sixth International Conference on User Modeling)*, Chia Laguna, Sardinia, June 1997.

[30] Stefani, A. and Strapparava, C., "Personalizing Access to Web Sites: The SiteIF Project," *Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia (HYPERTEXT '98)*, Pittsburgh, USA, 1998.

[31] Kurki, T., et al., "Agents in Delivering Personalized Content Based on Semantic Metadata," *Proceedings of the 1999 AAAI Spring Symposium Workshop on Intelligent Agents in Cyberspace*, Stanford, USA, 1999, pp. 84-93.

[32] Pretschner, A. and Gauch, S., "Ontology Based Personalized Search," *Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence*, 1998.

[33] Good, N., Schafer, J.B., Konstan, J.A., Borchers, A., Sarwar, B., Herlocker, J., and Riedl, J., "Combining Collaborative Filtering with Personal Agents for Better Recommendations," *Proceedings of the 1999 National Conference of the American Association of Artificial Intelligence*, pp. 439-436, 1999.

[34] Andre, E. and Rist, T., "From Adaptive Hypertext to Personalized Web Companions," *Communications of the ACM*, vol. 45, no. 5, 2002, pp. 43-46.

[35] Miller, G. A., "WordNet a Lexical Database for English," *Communications of the ACM*, vol. 38, no. 11, 1995, pp. 39-41.

[36] Klein, D. A., *Decision-Analytic Intelligent Systems: Automated Explanation and Knowledge Acquisition*, Lawrence Erlbaum Associates, 1994.

[37] Boose, J. H. and Bradshaw, J. M., "Expertise Transfer and Complex Problems: Using AQUINAS as a Knowledge-acquisition Workbench for Knowledge-Based Systems," *Int. J. Man-Machine Studies*, vol. 26, 1987, pp. 3-28.

[38] Kleinberg, J. M., "Authoritative Sources in a Hyperlinked Environment," *Journal of the ACM*, vol. 46, no. 5, 1999, pp. 604-632.

[39] Kerschberg, L., Kim, W., and Scime, A., "A Semantic Taxonomy-Based Personalizable Meta-Search Agent," *Proceedings of the Second International Conference on Web Information System*

Engineering, Kyoto, Japan, 3-6 December 2001, pp. 53-62.

[40] Salton, G. and McGill, M. J., An Introduction to Modern Information Retrieval, McGraw-Hill, 1983.

[41] Schűtze, H., Hull, D. A., and Pedersen, J. O., "A Comparison of Classifiers and Document Representations for the Routing Problem, " *Proceedings of the SIFIR-95*, 1995, pp. 229-237.

[42] Koller, D. and Sahami, M., "Hierarchically Classifying Documents Using Very Few Words," *Proceedings of the 14th International Conference on Machine Learning*, 1997.

[43] Feldman, J. A. and Ballard, D. H., "Connectionist models and their properties," *Cognitive Science*, vol. 6, 1982, pp. 205-224.

[44] Lenat, D. B., "Cyc: A Large-Scale Investment in Knowledge Infrastructure," *Communications of the ACM*, vol. 38, no. 11, 1995, pp. 33-38.

[45] Yokoi, T., "The EDR Electronic Dictionary," *Communications of the ACM*, vol. 38, no. 11, 1995, pp. 45-48.

[46] Porter, M. F., "An Algorithm for Suffix Stripping," Program, vol. 14, 1980, pp. 130-137.

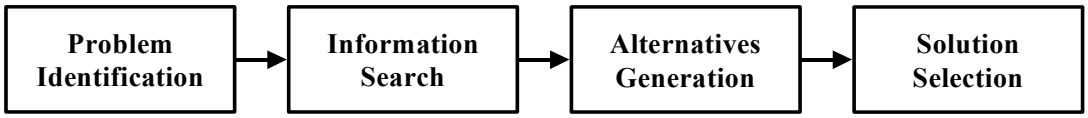


Figure 1 - Four Phases of Decision Making Process

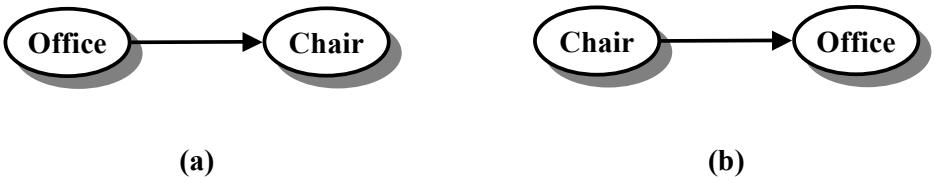


Figure 2 - A Simple Example of Taxonomy Tree

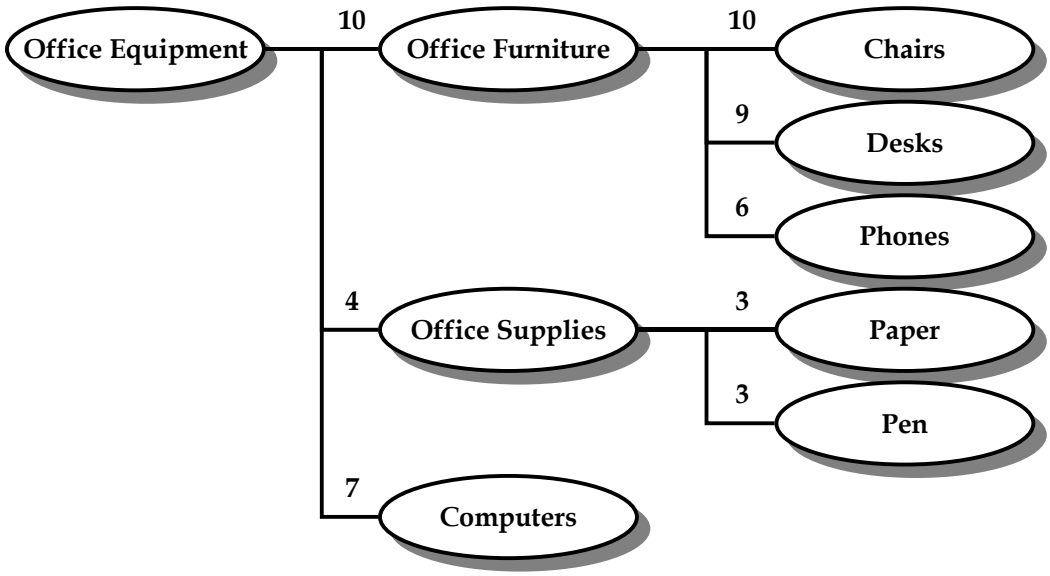


Figure 3 - An example of WSTT representing a businessman's search intention

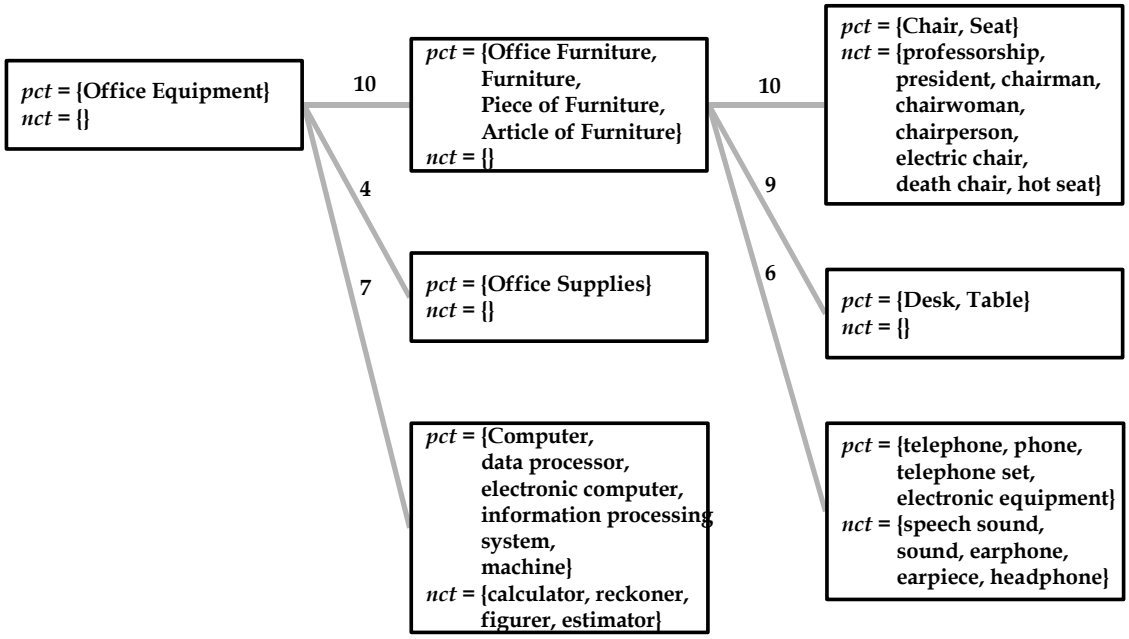


Figure 4 - An Example of Internal Representation of User's Search Intention

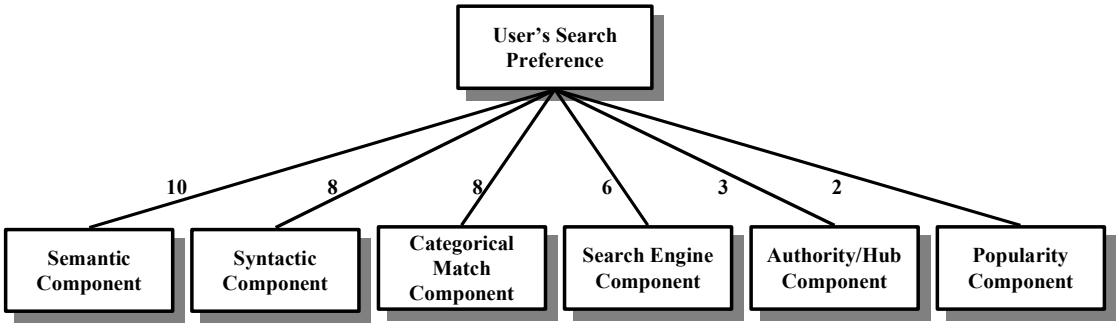


Figure 5 - A Conceptual Model of User's Preference Representation Scheme

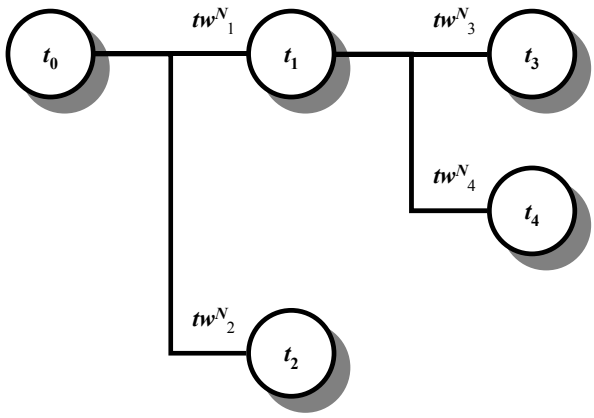


Figure 6 - An Example Structure of WSTT

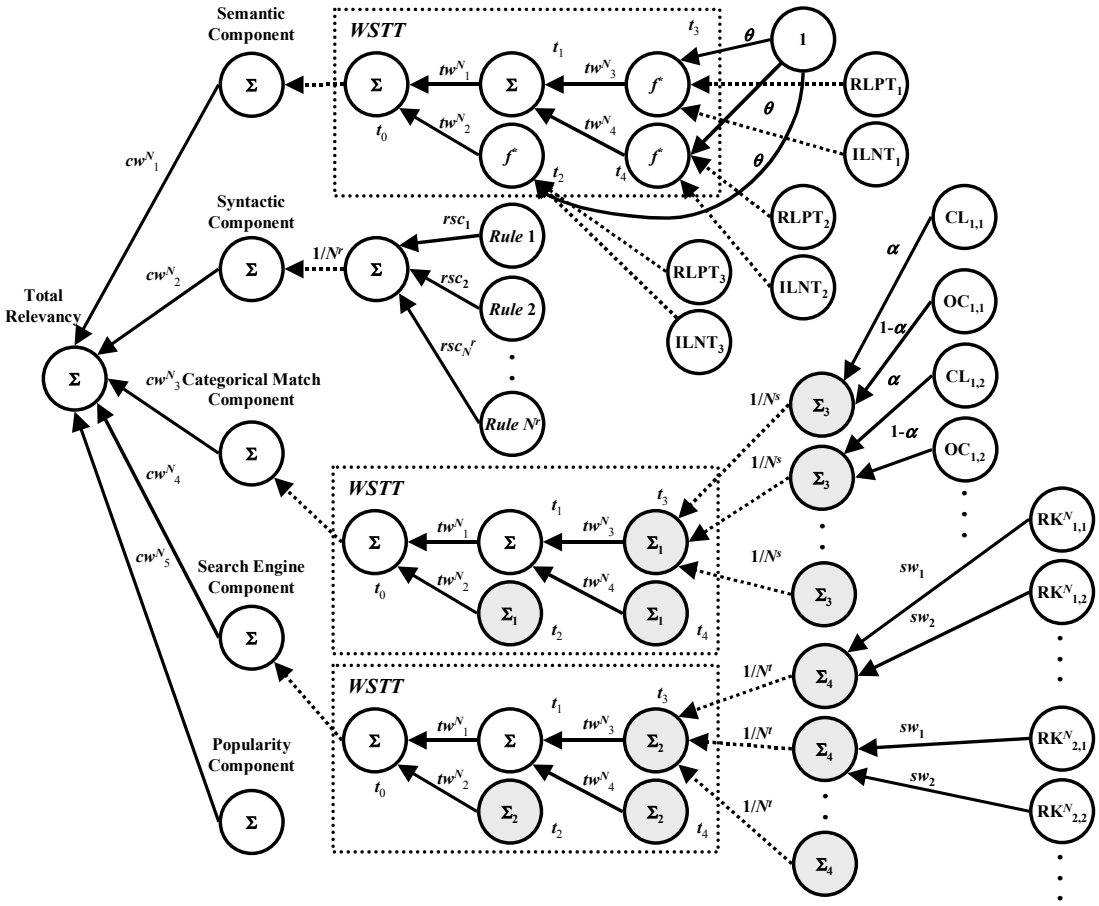


Figure 7 - Connectionist Model-based Representation of User Profile

User Profile Learning Algorithm

inputs: *MAX_EPOCHS*, *MIN_ERROR*, *UserProfileModel*, *DocumentSet*, *UserFeedBack*, *NWS*
local variables: *epochs*, *currentError*, *documentError*, *node*, *weight*, *document*, *nwse*, *parameter*, *documentVector*, *relevancyLevel*

loop do
  if *currentError* < *MIN_ERROR* **then return** success
  else if *epochs* > *MAX_EPOCHS* **then return** success
  currentError ← 0.0
  for each *document* **in** *DocumentSet* **do**
    compute *documentVector*
    for each *node* **in** the leaf node set of *UserProfileModel* **do**
      bind *node* with the corresponding value of *documentVector*
    end
    propagate *UserProfileMode*
      using **General Propagation Rule** and **Theta Propagation Rule**
    obtain *relevancyLevel* at the root node of *UserProfileModel*
    compute *documentError* using **Relevancy Error Computing Rule**
    for each *node* **in** the node set of *UserProfileModel* **do**
      if *node* is not related to θ **then**
        apply **Generalized Delta Computing Rule** to *node*
      else
        apply **Delta for Theta Computing Rule** to *node*
    end
    for each *weight* **in** the weight set of *UserProfileModel* **do**
      if *PS*(*weight*) is not equal to θ **then**
        apply **Generalized Weight Updating Rule** to *weight*
      else
        apply **Theta Updating Rule** to *weight*
    end
    for each set of weights *nwse* **in** *NWS* **do**
      for each *weight* **in** *nwse* **do**
        apply **Weight Normalization Rule** to *weight*
      end
    end
    for each *parameter* **in** the parameter set of *UserProfileModel* **do**
      for each *weight* **in** *PSWS*(*parameter*) **do**
        apply **Weight Synchronization Rule** to *weight*
        if *weight* < 0 **then** *weight* ← 0
        else if *weight* > 1 **then** *weight* ← 1
      end
    end
    currentError ← *currentError* + *documentError*
  end
  epochs ← *epochs* + 1
end

Figure 8 –User Profile Learning Algorithm

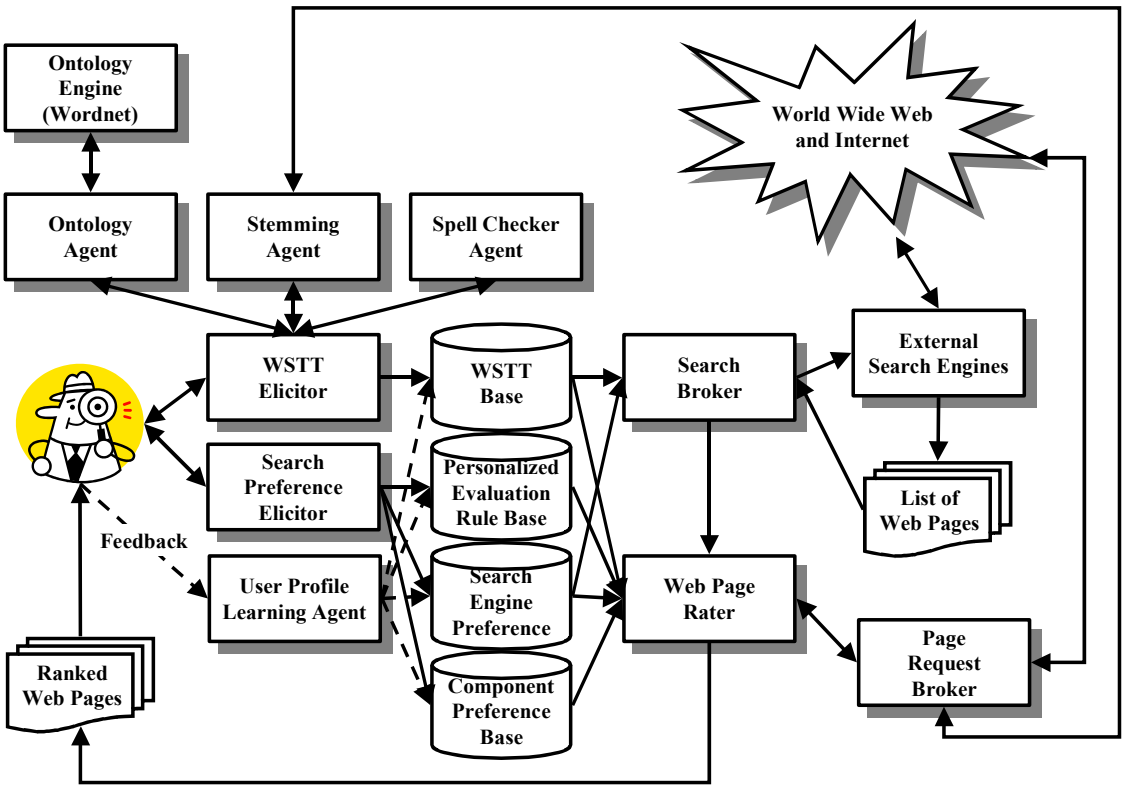


Figure 9 - System Architecture of WebSifter II

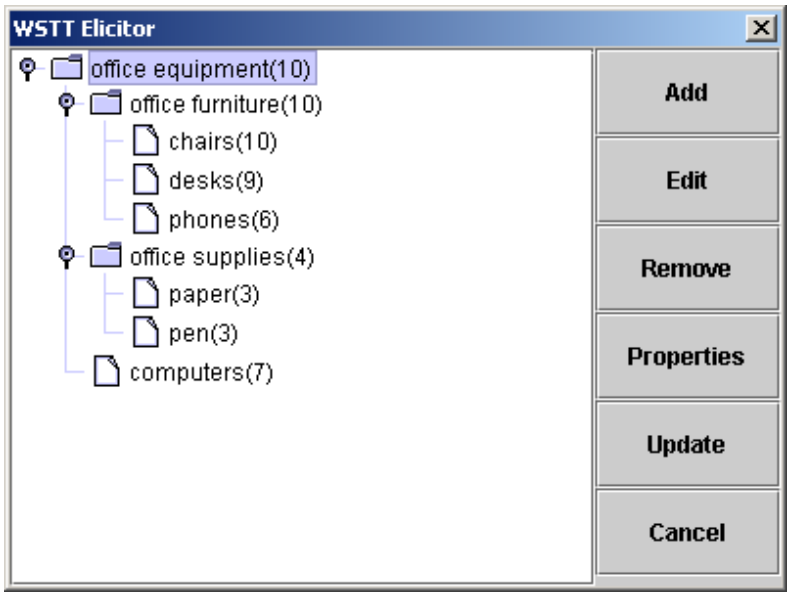


Figure 10 - An Illustrative Screen of WSTT Elicitor

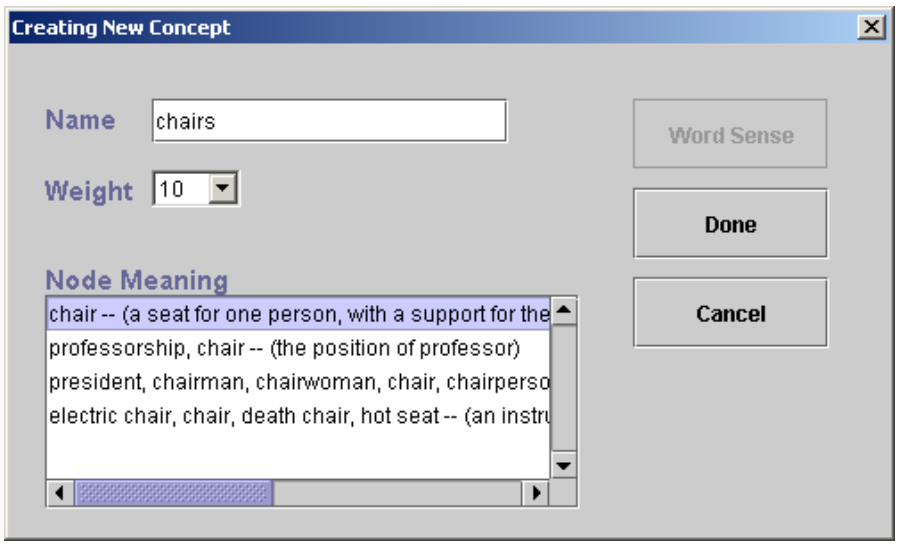


Figure 11 - An Illustrative Screen for Concept Selection

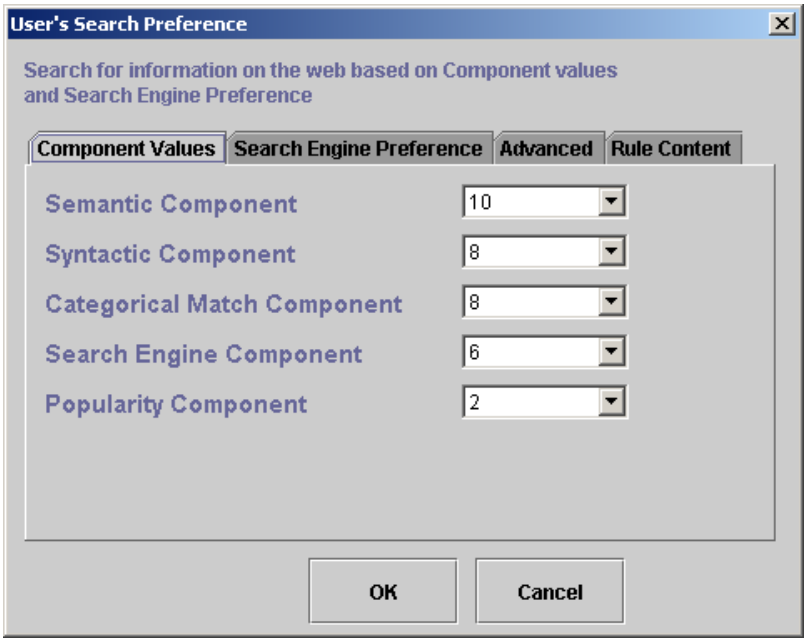
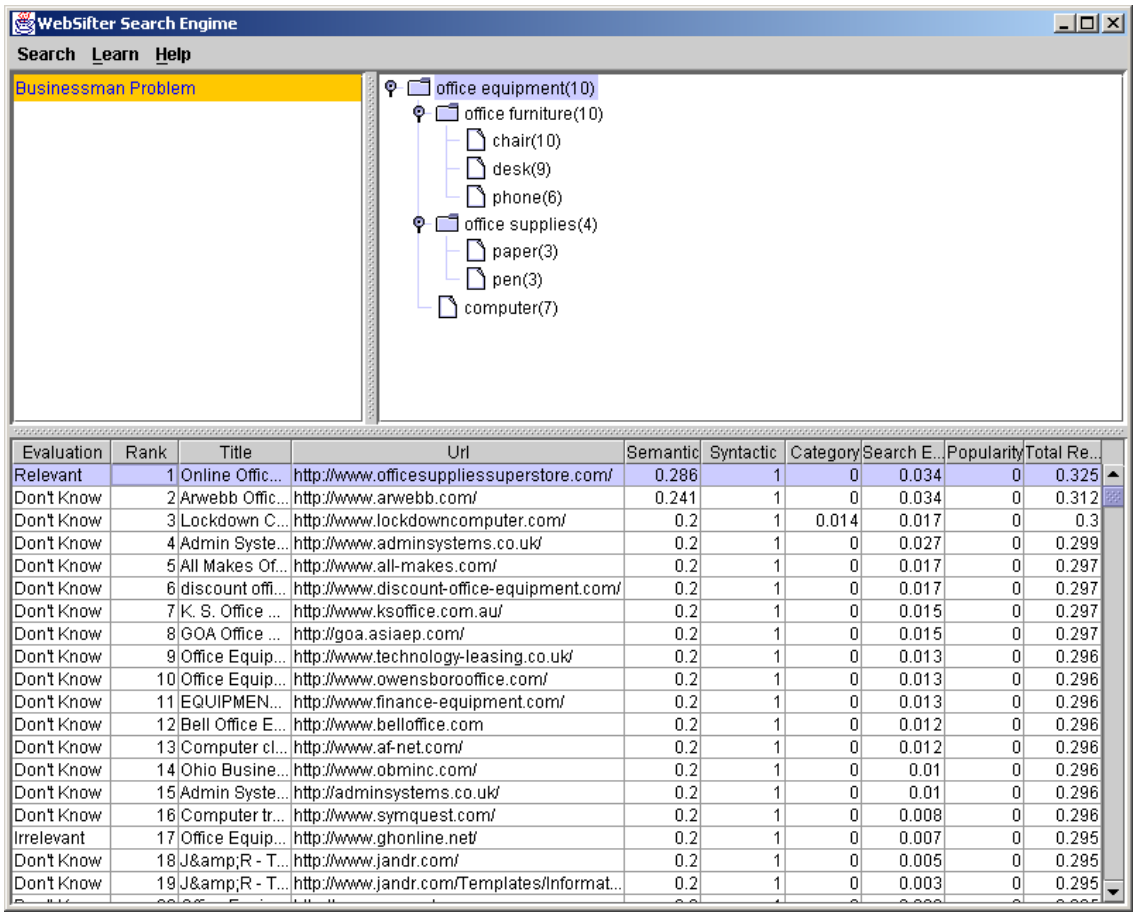


Figure 12 - A Tab Window of Search Preference Elicitor



Evaluation	Rank	Title	Url	Semantic	Syntactic	Category	Search E...	Popularity	Total Re...
Relevant	1	Online Offic...	http://www.officesuppliessuperstore.com/	0.286	1	0	0.034	0	0.325
Don't Know	2	Arwebb Offic...	http://www.arwebb.com/	0.241	1	0	0.034	0	0.312
Don't Know	3	Lockdown C...	http://www.lockdowncomputer.com/	0.2	1	0.014	0.017	0	0.3
Don't Know	4	Admin Syste...	http://www.adminsystems.co.uk/	0.2	1	0	0.027	0	0.299
Don't Know	5	All Makes Of...	http://www.all-makes.com/	0.2	1	0	0.017	0	0.297
Don't Know	6	discount offi...	http://www.discount-office-equipment.com/	0.2	1	0	0.017	0	0.297
Don't Know	7	K. S. Office ...	http://www.ksoffice.com.au/	0.2	1	0	0.015	0	0.297
Don't Know	8	GOA Office ...	http://goa.asiaep.com/	0.2	1	0	0.015	0	0.297
Don't Know	9	Office Equip...	http://www.technology-leasing.co.uk/	0.2	1	0	0.013	0	0.296
Don't Know	10	Office Equip...	http://www.owensborooffice.com/	0.2	1	0	0.013	0	0.296
Don't Know	11	EQUIPMEN...	http://www.finance-equipment.com/	0.2	1	0	0.013	0	0.296
Don't Know	12	Bell Office E...	http://www.belloffice.com	0.2	1	0	0.012	0	0.296
Don't Know	13	Computer cl...	http://www.af-net.com/	0.2	1	0	0.012	0	0.296
Don't Know	14	Ohio Busine...	http://www.obminc.com/	0.2	1	0	0.01	0	0.296
Don't Know	15	Admin Syste...	http://adminsystems.co.uk/	0.2	1	0	0.01	0	0.296
Don't Know	16	Computer tr...	http://www.symquest.com/	0.2	1	0	0.008	0	0.296
Irrelevant	17	Office Equip...	http://www.ghonline.net/	0.2	1	0	0.007	0	0.295
Don't Know	18	J&R - T...	http://www.jandr.com/	0.2	1	0	0.005	0	0.295
Don't Know	19	J&R - T...	http://www.jandr.com/Templates/Informat...	0.2	1	0	0.003	0	0.295

Figure 13 - An Illustrative Screen for the Result from Web Page Rater and Feedback Interface

Table 1 – Overall Performance Comparison

Search Engines	Average Performance
WebSifter	**63%**
Copernic	**40%**
Altavista	**43%**
Google	**40%**
Yahoo	**48%**
Excite	**43%**

Table 2 – Initial Retrieved Page Hit Results and User's Feedback

Rank	URL	Feedback	Relevancy
1	http://www.countryseat.com	Y	Y
2	http://www.infant-car-seat.com/	N	N
3	http://www.chairmaker.co.uk/	Y	Y
4	http://www.convertible-car-seat.com/	N	N
5	http://www.booster-car-seats.com/	N	N
6	http://www.booster-seats-online.com/	Don't Know	N
7	http://www.booster-car-seat.com/	Don't Know	N
8	http://www.podiatrychair.com/	Don't Know	N
9	http://www.carolinachair.com/	Don't Know	Y
10	http://www.chairdancing.com/	Don't Know	N
11	http://www.massage-chairs-online.com/	Don't Know	N
12	http://www.panasonic-massage-	Don't Know	N
13	http://www.fairfieldchair.com/	Don't Know	Y
14	http://www.gasserchair.com/	Don't Know	Y
15	http://www.chairtech.com/	Don't Know	Y
16	http://www.snugseat.com/	Don't Know	N
17	http://www.seat.com/	Don't Know	N
18	http://www.fifthchair.org/	Don't Know	N
19	http://www.painted-	Don't Know	N
20	http://www.jeanmonnetprogram.org/	Don't Know	N

Table 3 – Page Hit Results after Instant Learning and Its Relevancies

Rank	URL	Relevancy
1	http://www.fairfieldchair.com/	Y
2	http://www.chairmaker.co.uk/	Y
3	http://www.carolinachair.com/	Y
4	http://www.podiatrychair.com/	N
5	http://www.chairdancing.com/	N
6	http://www.gasserchair.com/	Y
7	http://www.chairtech.com/	Y
8	http://www.snugseat.com/	N
9	http://www.fifthchair.org/	N
10	http://www.ompchairs.com/	Y
11	http://www.cyberchair.com/	Y
12	http://www.massage-chairs-online.com/	N
13	http://www.panasonic-massage-	N
14	http://www.jeanmonnetprogram.org/	N
15	http://www.leap-chair.com/	Y
16	http://www.painted-	N
17	http://www.zackback.com/	Y
18	http://www.chair-ergonomics.com/	Y
19	http://www.countryseat.com	Y
20	http://www.infant-car-seat.com/	N

Table 4 – Page Hit Ratios from Five Learning Experiments

Search Term	Before Learning	After Learning	Performance Enhancement
chair	30%	55%	25%
paper	45%	60%	15%
pen	55%	85%	30%
rock	15%	75%	60%
phone	65%	80%	15%
Average	42%	71%	29%