Personalization in A Semantic Taxonomy-Based Meta-Search Agent

Wooju Kima, Larry Kerschbergb, and Anthony Scimec

a Department of Industrial Engineering, Chonbuk National University
664-14 Deokjin, Chonju, Chonbuk, 560-756, Korea
Tel: +82-63-270-2332, Fax: +82-63-270-2333, E-mail: wjkim@chonbuk.ac.kr

b E-Center for E-Business, George Mason University
4400 University Drive, Fairfax, VA 22030, USA
Tel: +1-703-993-1640, Fax: +1-703-993-1638, E-mail: kersch@gmu.edu

c Department of Computer Science, SUNY-Brockport
E-mail: ascime@brockport.edu

Abstract

Providing most 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. Regarding 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 preferences based rating. To provide a learning mechanism, we represent the entire rating mechanism of semantic taxonomy-based meta-search agent approach as a feed-forward neural network model and adopt the generalized delta rule as our basic learning scheme by modifying it to conform to our framework. 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 Internet and WWW, the amount of information available on the Web grows daily. However, having too much information at one’s fingertips does not always mean good 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 uncustomizable ranking system, where users cannot “tell” the search engine what preferences to use for search criteria. 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 personalize user preferences. They cannot track large number of users. The personal agent approach can help to solve this problem.

To address the first three of these four problems, we propose a semantic taxonomy-based personalizable meta-search agent approach in our previous researches [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 famous 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 automatic personalization of both the user’s search intent as well as the user’s ranking preferences.

The typical search engine uses a term-frequency vector as part of the user profile, while our approach provides a richer preference representation scheme. Moreover, we feel that machine-learning techniques, specifically neural network learning algorithms such as generalized delta rule [4], can be adapted to learning the user preference and system parameters.

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 neural network-based learning method for the personalization under the pre-defined semantic taxonomy-based meta-search agent framework in Section 4. Section 5 addresses a brief introduction of WebSifter II system and the role of our personalization components in the entire system. 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 of 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 Siteseeer [17]. In addition, some hybrid approaches incorporate both approaches for example Fab [18], Lifestyle Finder [19], WebCobra [20].

The third category is the domain knowledge approach that uses user and organizational domain knowledge to improve the relevancy of search results. Yahoo! used 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

Many researches mentioned in the previous section incorporate 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) what representation of user profile they use for learning, 2) how they get feedback from the user, and 3) which algorithm they use to learn.

We focus only on the first aspect, the representation of 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 word frequency vector as their basis of user profile representation.

But as indicated in [28], learning 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 user profile. ifWeb [29] and SiteEF [30] use semantic networks and PSUN [28] uses a kind of associative network of words. Also, SmartPush [31] and OBIWAN [32] use a kind of ontology (hierarchical concept tree).

Even though this improved user profile representation, their user profile representation schemes are still based on word frequencies and their word ordering. Therefore, similar limitations mentioned in the end of the previous section are observed too as follows. First, there are so many important aspects in user profile for Web search, which cannot be easily represented by only using the word frequency concept. User’s preference to authority and popularity of page hit are the examples for this case. Second, even in the case of using domain knowledge such as ontology, it is fixed and not user-generated.

To maximize the personalization, the domain knowledge should be extracted from users and used to represent a part of user profile. In fact, our previous research 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 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 support readily query formulation and search.

Figure 1 - Four Phases of Decision Making Process

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 problems, 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 is 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. As we mentioned earlier, 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 - A Simple Example of Taxonomy Tree](image)

The taxonomy tree approach is already used in many search engines such as 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.

Figure 3 shows a realistic example of the businessman’s search intention using our WSTT scheme. Users can build their own hierarchical taxonomy tree, and assign importance levels to each term within the context of their antecedent terms. 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 the respective importance levels of 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 limitation, we introduce the notion of “word senses” from Wordnet [33] into our WSTT scheme to allow users to refine their search intention.
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 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 second concept from our example, according to the definitions from (1) and (2), pct(n) and nct(n) are as follows: \(pct(n) = \{\text{professorship, chair}\}\) and \(nct(n) = \{\text{seat, president, chairman, chairwoman, chairperson, electric chair, death chair, hot seat}\}\).

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.

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. Both Multi-Attribute Utility Technology (MAUT) [34] and Repertory Grid [35] 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 [36].
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 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.

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 did not address the detailed discussion about the rating mechanism and the reader can refer to [37] about this issue.

4. Learning for Personalization

So far, we have discussed how to represent a user search query and preferences as well as, how to rate the resulting page hits based on the search query and preferences. In order to learn a user’s real search intent and preferences, we define a user profile and feedback.

4.1 User Profile

In this semantic taxonomy-based meta-search agent approach, there are several search preferences that can be included into the user profile, in addition to the traditional word frequency vector. This is one of the major advantages of our learning scheme. We define user profile from the following information available from Section 3.

- Weights on the preference components, \(cw(\text{com})\): represent how important a user thinks each search preference component, \(\text{com}\) in his search.
- Weights on the nodes in the WSTT, \(w(n)\): represent how important a user thinks each concept node, \(n\) in his search.
- Weights on the search engines, \(s\text{w}(s)\): 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 semantic and categorical match component relevancy computation, \(\theta\) and \(\alpha\): \(\theta\) denotes a rate a user wants to consider the irrelevancy measure in his semantic rating and \(\alpha\) denotes how much a user prefers the co-occurrence level to the order consistency level in the categorical match rating.

Now, we can represent a user’s search preference profile by using the above five types of information and also, we can learn more accurate profiles by adjusting them by means of the following user feedback mechanism.

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 has to choose 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 formally as follows:

$$E_{pg} = \frac{1}{2} \left( rv^{U}(pg) - rv(pg) \right)^2$$  \hspace{1cm} (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, otherwise, 0. For example, if our relevancy expectation on a page $pg$ is 0.7 and user’s reply is 1, then $E_{pg}$ becomes $\frac{1}{2} \times 0.3^2 = 0.045$.

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

$$TE = \sum_{pg} E_{pg}$$  \hspace{1cm} (2)

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

4.3 Learning Mechanism

4.3.1 Feed-Forward Neural Network-Based Representation

To derive a learning mechanism for our user profile, we first represent our entire rating mechanism mentioned in Section 3.4 as a feed-forward neural network model as shown in Figure 6. In this figure, we assume the case where an example WSTT is the tree shown in Figure 7, where $t_i$ means a term assigned to the $i$-th node and $w^N_i$ is a normalized weight assigned to the $i$-th node, and this tree is already depicted as sub-networks outlined by the dotted box in Figure 6. Actually, there are three sub-networks, and this is because we use the same WSTT three times to compute semantic, categorical match, and search engine components.

Also, every node in Figure 6, except nodes labeled as ‘SRPV’, is represented as a simple summation node, which
is denoted by the $\Sigma$ symbol. Its functional form is defined as the following formula (3).

$$ o_{pg,j} = \sum_i w_{ji} o_{pg,i} $$ \hspace{1cm} (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 $j$.

**Figure 7 - An Example Structure of WSTT**

Actually the nodes labeled by ‘SRPV’, adopt a nonlinear exponent function as their activation functions to compute semantic relevance value for each tree path in Figure 7, and so, it is very different from (3). Every parameter used in our rating mechanism is also represented by the $\Sigma$ symbol. Its functional form is defined as the following formula (3).

$$ \delta_{pg,j} = \left\{ \begin{array}{ll} rv_i(pg) - rv(pg) & \text{if } j \text{ is an output node} \\ \sum_k \delta_{pg,k} w_{kj} & \text{otherwise} \end{array} \right. $$ \hspace{1cm} (4)

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 follow:

$$ w_{ji}^{updated} = w_{ji}^{old} + \eta \cdot o_{pg,j} \cdot \delta_{pg,j} $$ \hspace{1cm} (5)

where $\eta$ is a learning rate for the weight.

However, these two rules, (4) and (5) are not enough to address all our learning requirements. At first, since the above rules are only for the summation node, which can be represented by (3), we cannot apply them also to the cases of nodes denoted by ‘SRPV’, which adopt a non-linear exponent function as their activation functions. To update the parameter $\theta$ of this function according to a user’s feedback, we additionally define a delta for this ‘SRPV’ type node as follow:

$$ \delta_{pg,j}^\theta = \frac{1}{\eta} \left( -\frac{\sum_{k} rv_t(cM(tc,pg)) \cdot w_{kj}}{\text{ns}(tc,dp)} \right) \cdot \eta^{\gamma-1} $$ \hspace{1cm} (6)

where the node $j$ is a ‘SRPV’ type node.

We use (6) for updating the parameter $\theta$ in ‘SRPV’ function but since $o_{pg,j}$ becomes 1 in this $\theta$ case, its updating rule can be simplified as follows:

$$ \theta_j^{updated} = \theta_j^{old} + \eta \cdot \delta_{pg,j}^\theta $$ \hspace{1cm} (7)

where the $\theta_j$ is a corresponding $\theta$ in a ‘SRPV’ type node $j$.

Second problem in simply adopting generalized delta rule is that our rating mechanism requires the sum of the weights on the sibling nodes from a parent node should be 1 for normalization purposes. To satisfy this constraint, we devise a further re-normalization step after updating the weights according to (5) as follow:

$$ w_{ji}^{normalized} = \frac{w_{ji}^{updated}}{\sum_{k} w_{kj}^{updated}} $$ \hspace{1cm} (8)

where $children(j)$ is a set of child nodes of $j$.

One last problem to be considered is the fact that some parameters including $\theta$ in our rating mechanism appear multiple times in different sub-networks of Figure 6. For example, a weight $w_{i1}^t$ of a term $t_1$ in the example WSTT in Figure 7 appears three times in each dotted box. This implies three different adjustment directions for the same weight $w_{i1}^t$ may occur at the same learning cycle. To resolve this problem, we update such weights by using the average adjustment level of multiple suggested adjustments.

### 5. WebSifter II System Architecture

In this section we present the architecture of WebSifter II. Figure 8 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.

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.

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 [38] and EDR [39].

Our stemming agent is based on Porter’s algorithm [40]. 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.

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

The search preference elicitor, via a GUI, supports the process (cf. section 3.2) to capture 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.

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.

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.

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 search broker and page request broker.

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 user invokes learning or when user closes the system, the learning process starts and it updates various user preference parameters to reflect the user’s feedback information. 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 Example Learning

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 9 shows an illustrative screen where the user builds a WSTT using the WSTT elicitor. Figure 10 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 11 shows a sample screen for a user to specify his search preference using our search preference elicitor. Four tab windows in Figure 11 are for adjusting 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 11.

![Figure 11 - A Tab Window of Search Preference Elicitor](image)

Finally Figure 12 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 is also allowed to select “don’t know” if he doesn’t want to rate the page or feels he is not sure 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 learning 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.

6.2 Example Computation for Learning and Performance Evaluation

To show an example of the learning that occurs in real case, let’s use the cases appearing in Figure 12 and demonstrate how the learning occurs only for the weights on the first layer in Figure 6. Let’s assume a user rated the first ranked page, ‘www.officesuppliessuperstore.com’, as a relevant page.

![Figure 11 - A Tab Window of Search Preference Elicitor](image)

Since the suggested relevancy of that page by WebSifter, was 0.325, the delta value of the output node in Figure 6 can be computed as $1 - 0.325 = 0.675$ according to (4).

Then, since our five preference component relevancy values are 0.286, 1.0, 0.0, 0.034, and 0.0, respectively as shown in Figure 12, their weight adjustment levels become as follows, if we assume $\eta = 0.5$:

- $\Delta c_{\text{semantic}} = 0.5 \times 0.675 \times 0.286 = 0.097$
- $\Delta c_{\text{syntactic}} = 0.5 \times 0.675 \times 1.0 = 0.338$
- $\Delta c_{\text{match}} = 0.5 \times 0.675 \times 0.0 = 0.0$
- $\Delta c_{\text{search}} = 0.5 \times 0.675 \times 0.034 = 0.011$
- $\Delta c_{\text{popularity}} = 0.5 \times 0.675 \times 0.0 = 0.0$

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 since these new updated weights violate the constraint that requires their sum must be 1, so we additionally need to apply a renormalization process to these weights according to (8).

$$\sum \text{weights} = 1$$
After finishing this process, the weights for five components finally become 0.271, 0.397, 0.163, 0.129, and 0.040, respectively. As a result, the weight only for syntactic component increases but all other weights for the remaining components decrease in terms of their relative importance levels in the user preferences.

We just show partly what happens on the neural network model to learn a user profile using our learning method. Now, we are currently doing empirical experiments on our learning approach to evaluate how effectively the method works on our semantic taxonomy-based meta-search agent. Actually, the search performance of our meta-search approach has been validated empirically and also, we are performing empirical experiments to evaluate user search profile learning performance. So far, the experiments show promising results to our approach.

7. Conclusions

The semantic taxonomy-based meta-search agent approach [3] is proposed to achieve two important and complementary goals: 1) allowing users more expressive power in formulating their Web searches, and 2) improving the relevancy of search results based on the user’s real intent. It has been empirically proved that this approach achieves both goals through the real experiments.

However, one weakness to our approach is it does not support user profile learning for personalization even though it can represent user’s search intent and preference well. To overcome this shortness and to achieve a better personalization in search agent domain by taking advantage of the strong search intent and preference representation of the semantic taxonomy-based meta-search agent approach, we have proposed a neural network-based user profile learning mechanism for this search agent approach.

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 user’s profile of the search intent and preferences in a more sophisticated manner than previous approaches based on the vector space model.

Second, we present a neural-network-based user profile learning mechanism to learn user search intent and preferences in his Web search based on the proposed user profile representation scheme. To achieve this goal, we first represent the entire rating mechanism [37] as a feed-
forward neural network model and then, we devise a profile-learning method 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 are also doing some real world experiments of our system.

References


