WebMate : A Personal Agent for Browsing and Searching*

Liren Chen and Katia Sycara
The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA. 15213
lchen@cs.cmu.edu, katia@cs.cmu.edu

September 30, 1997

Abstract

The World-Wide Web is developing very fast. Currently, finding useful information on the Web is a time consuming process. In this paper, we present WebMate, an agent that helps users to effectively browse and search the Web. WebMate extends the state of the art in Web-based information retrieval in many ways. First, it uses multiple TF-IDF vectors to keep track of user interests in different domains. These domains are automatically learned by WebMate. Second, WebMate uses the Trigger Pair Model to automatically extract keywords for refining document search. Third, during search, the user can provide multiple pages as similarity/relevance guidance for the search. The system extracts and combines relevant keywords from these relevant pages and uses them for keyword refinement. Using these techniques, WebMate provides effective browsing and searching help and also compiles and sends to users personal newspaper by automatically spiding news sources. We have experimentally evaluated the performance of the system.

Area: Software Agents

Keywords: Information Agents, Instructability, Knowledge acquisition and accumulation, long-term adaptation and learning, user modeling

*This research has been supported in part by ARPA contract F33615-93-1-1330, and by ONR Grant N00014-96-1222.

1 Introduction

The Web is full of information and resources. People have at least three ways to find information they need: (1) by browsing (following hyper-links that seem of interest to them), (2) by sending a query to a search engine, such as Altavista, (3) by following existing categories in search engines, such as Yahoo or Lycos. The problem is that people have to spend a lot of time and effort to navigate but may not find interesting personalized information. However, it is difficult to find the wanted information because a user can't accurately express what he wants and search engines don't adapt their search strategies according to different users. Moreover, the problem is exacerbated because the information sources have high "noise", i.e. most of the pages are irrelevant to a particular user's interests. Intelligent software agents are being developed to deal with these issues.

Intelligent agents are programs that act on behalf of their human users to perform laborious information-gathering tasks [1] and they are one of the "hot" topics in Information Systems R&D at the moment. The last ten years have seen a marked interest in agent-oriented technology, spanning applications as diverse as information retrieval, user interface design and network management.

In this paper, we present WebMate, a personal software agent that accompanies a user when he browses and searches and provides intelligent help [1].

For clarity of presentation, the WebMate capabilities will be presented in roughly two categories: (1) learning user interests incrementally and with continuous update and automatically providing documents (e.g. a personalized newspaper) that match the user interests, and (2) helping the user refine search so as to increase retrieval of relevant documents. In section 2, we describe the architecture of the system. The WebMate acts as a proxy and monitors a user's actions. In section 3, we describe the user profile representation and learning algorithm [3, 4]. In addition, we provide experimental results of compiling a personal newspaper. In section 4, we discuss how to use the Trigger Pairs Model to extract relevant words to use as keyword refinements to improve search. We also present utilizing relevance feedback [8] during search to dynamically enhance the search for relevant documents. Finally, related work and our future work are described.

2 WebMate architecture

WebMate is composed of a stand-alone proxy that can monitor a user's actions to provide information for learning and search refinement, and an applet controller that interacts with a user (See Figure 1).

[1]The WebMate system has been operating on Web and has been downloaded by more than 600 users since it was published in the middle of September 1997 (15 days ago). Its URL is http://www.cs.cmu.edu/~softagents/webmate.

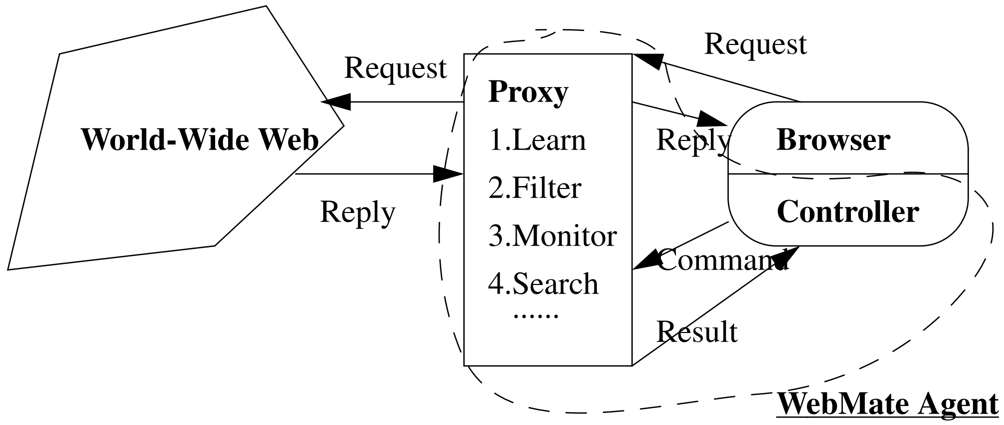

Figure 1: System Architecture

The stand-alone proxy is an HTTP proxy that sits between a user's web browser and the World-Wide Web. All HTTP transactions pass through WebMate which can monitor a user's browsing and searching activities and learn from them.

The applet controller is the interface between the user and the stand-alone proxy. Through it, the user can express his interests when he browses and provide relevance feedback when he searches. In addition, through the applet controller, the user receives intelligent help from WebMate.

3 Learning profile to compile personal newspaper

3.1 Profile Representation and Learning Algorithm

There are several machine learning approaches that can be used to learn a user profile, such as Bayesian classifier, Nearest Neighbor, PEBLS, Decision Trees, TF-IDF, Neural Nets [4, 5]. In order for a particular technique to be effective, it should match the characteristics of the task and the user.

The filtering task for our agent involves judging whether an article is relevant or irrelevant to the user based on the user profile, in an environment where the prior probability of encountering a relevant document is very low compared to the probability of encountering an irrelevant document. In such an environment, it would be very frustrating and time consuming for a user to interact with an agent that starts with no knowledge but must obtain a set of positive and negative examples from user feedback. When a user browses, he does not want to evaluate all web pages that might contain potentially interesting information. To reduce user evaluation burden, WebMate collects only examples that are interesting to the user (only positive training examples). This kind of interaction presents potential problems since the documents that a user might label as "I like It" might fall into many distinct domains (e.g fishing, computer science, soccer). Those subclasses correspond to the different interests a user has. There have been two methods to address the problem of multiple user interests. The first is to keep a single user profile where the keywords might come from different domains but are "averaged'. This method has the disadvantage that averaging the vectors from the different documents might decrease too much the weights of words that are important for only a few of the interest categories. The second method is to ask the user to explicitly provide labels for the sub-categories of interest. WebMate does not ask the user to label the category that the interesting document is in, but learns the categories automatically.

In contrast to other systems that learn a user profile and use it statically to determine relevant documents, WebMate learns the user profile incrementally and continuously. When a new positive example is known, the system updates the profile. In order to save on storage space, the system doesn't keep any of the previous positive example documents. It only keeps the profile learned from those positive examples. In this way, the system will adapt to the user's evolving and recent interests.

WebMate utilizes TF-IDF method [7] with multiple vectors representation. The basic idea of the algorithm is to represent each document as a vector in a vector space so that documents with similar content have similar vectors. Each dimension of the vector space represents a word and its weight. The values of the vector elements for a document are calculated as a combination of the statistics term frequency $TF(w,d)$ (the number of times word w occurs in document d) and document frequency $DF(w)$ (the number of documents the word w occurs in at least once). From the document frequency the inverse document frequency $IDF(w)$ can be calculated.

$$IDF(w) = \log \frac{|D|}{DF(w)}$$

$|D|$ is the total number of documents. The value $d^{(i)}$ of an element in the vector is then calculated as the product

$$d^{(i)} = TF(w_i, d) \times IDF(w_i)$$

We have developed an algorithm for *multi TF-IDF* vector learning. The algorithm follows.

We assume that a user has at most N domains of interest. [2] Assume the initial profile set is V, $|V| = 0$; the predefined number of TF-IDF vectors in the profile set is N, the preset number of elements of a vector is M. For each positive example (i.e. an HTML documents that the user has marked "I like It"), do:

[2]In the current implementation, N is heuristically set to 10

1. Preprocess: parse HTML page, deleting the *stop* words (or non-informative words) such as "a", "the", "is", "in", etc, stemming the plural noun to its single form and inflexed verb to its original form, extracting the words in *title*(<TITLE>), *head1*(<H1>), *head2*(<H2>), *head3*(<H3>) because they will be given more weights;

2. Extract the TF-IDF vector for this document, let it be V_i;

3. If $|V| < N$ ($|V|$ is the number of vectors in the profile set V), then $V \Leftarrow V \cup V_i$;

4. Otherwise, calculate the cosine similarity between every two TF-IDF vectors including the vectors in the profile set V and the new document vector V_i. Assume the profile set V is $\{V_1, V_2, \ldots, V_n\}(n = N)$.

$$\mathcal{S}im(V_j, V_k) = \frac{V_j \bullet V_k}{|V_j| \times |V_k|} \quad j, k \in \{1, 2, \ldots, n, i\}$$

5. Combine the two vectors V_l and V_m with the greatest similarity..

$$V_l = V_l + V_m \quad (l, m) = \arg\max_{(x,y)}(\mathcal{S}im(V_i, V_j)) \quad x, y \in \{1, 2, \ldots, n, i\}$$

6. Sort the weights in the new vector V_k in decreasing order and keep the highest M elements.

This algorithm is run whenever a user marks a document as "I like it". Thus, the user profile is incrementally, unobtrusively and continuously updated.

3.2 Compiling personal newspaper

We utilize the approach of learning user profile to compile a personal newspaper [9, 10, 11]. We do this in two ways.

One way is to automatically spide a list of URLs that the user wants monitored. An example of such a URL is one that consists of many news headlines like the home page of the NewsLinx Company[3]. WebMate (1) parses the html page , (2) extracts the links of each headline, (3) fetches those pages, (4) constructs the TF-IDF vector for each of those pages (using as additional heuristics that words in title, and headings are given additional weights), and (5) calculates the similarity with the current profile. If the similarity is greater than some threshold, it recommends the page to the user, and sorts all the recommended pages in decreasing order of similarity to form the personal newspaper. All operations are

[3] http://www.newslinx.com/

often performed in the middle of the night when the network traffic is low. In the morning, the user can read the recommended personal newspaper.

If the user does not provide any URLs that he would like to be the information sources, WebMate constructs a query[4] using the top several words in the current profile and sends it to popular search engines (e.g. Altavista, Yahoo). If the result is needed immediately, the results returned by the search engines are directly used as the recommended web pages. Otherwise, the system fetches the pages corresponding to each and every URL in the results. It then calculates the similarity of the profile and these web pages and recommends the pages whose similarity is greater than some threshold presenting the results in descending order of relevance.

3.3 Experiments

In our experiments, the system monitors about 14 news sites that contain articles about high technology including LAN time news[4], Media Central [5], PC magazine online [6], etc. We recorded the personal newspaper and evaluated whether a piece of news is interesting to us (Table 1). The first column is the date of the personal news, the second column is the percentage accuracy of how many pieces of news are interesting in the top 10 returned by WebMate, the third column is the percentage accuracy in the top 20. In order to evaluate the learning approach, the percentage accuracy in the whole recommended news (the number of interesting news articles divided by the total number of news articles in the newspaper) is given in the fourth column.

Date	Accuracy in top 10	Accuracy in top 20	Accuracy in whole
Sep.16	70%	60%	17/55=31%
Sep.17	40%	35%	11/42=26%
Sep.18	50%	35%	9/33=27%
Sep.19	60%	65%	18/76=24%
Sep.20	50%	40%	9/29=31%
Sep.22	40%	40%	12/49=25%
Sep.23	50%	50%	18/78=23%
Sep.24	60%	56%	10/18=56%
Average	52%	49%	30.4%

Table 1: Experiment Results

From Table 1, we see that the average accuracy (relevance rate) that the recommended news is relevant to our interests is between 50% and 60% in the top 10 news articles .

[4] http://www.lantimes.com/
[5] http://www.mediacentral.com/Magazines/MediaDaily/Archive
[6] http://www8.zdnet.com/pcmag/

Generally the system will spide more than 500 pieces of news for a day. In the whole recommended news, the average accuracy is about 30%. But if the news are randomly chosen from 500 pieces of news in which we assume there are 100 interesting news to us (this is based on our observation that for a typical news site such as LinkExchange, there are about 10 out of 50 pieces of news that are interesting to us in any given day), the default accuracy in the whole news is about 20%. So a 50% to 60% accuracy, achieved by WebMate, represents a two to three-fold accuracy increase.

There are several factors that lower the accuracy of the system. First, it is difficult to determine which links are the headlines of the news and which links are irrelevant stuff such as advertisements. We are currently working on heuristics to filter out advertisements. So, currently, all the links in the page are used to calculate the similarity, not just the links of the news headlines. Second, while calculating the TF-IDF vectors, the irrelevant stuff around the news affects the accuracy of the TF-IDF.

4 Search refinement by keywords expansion and relevance feedback

4.1 Trigger Pairs Model to extract relevant words

Single keywords are usually ambiguous, or too general. Moreover, they can occur in vast quantities of documents, thus making the search return hundreds of hits, most of which are irrelevant to the intended user query. Giving additional keywords can refine search providing considerable improvement in the retrieval results. Good refinement words must have meanings that help disambiguate or make more specific the original search word. For example, the word "stock" has more than 10 definition in the WordNet[7] including "the capital raised by a corporation through the issue of shares entitling holders to partial ownership", "gun-stock", "inventory", "stock certificate", etc. Providing the refinement words that correspond to each one of those meanings, would help a search engine, for example, to prune out documents where the word is used with any of its other meanings. There are three ways to expand the query: manual query expansion, semi-manual query expansion, and automatic query expansion [12]. No matter which method is used, the key point is to get the best refinement words. In manual query expansion, although the user knows the intended meaning of the keyword she is using, she may not be able to provide the best refinement words. "Best" here means refinement words that most frequently co-occur with the word in its intended meaning in large number of documents. In other words, one of the characteristics of good refinement words is that they be domain specific. In this section we present the method for automatically finding appropriate keywords to constrain and refine search for relevant documents.

[7] http://www.cogsci.princeton.edu/~wn/

We use the Trigger Pairs Model [13, 14]. If a word S is significantly correlated with another word T, then (S, T) is considered a "trigger pair", with S being the trigger and T the triggered word. When S occurs in the document, it triggers T, causing its probability estimate to change. That is, when we see the word S appearing at some point in a text, we expect the word T to appear somewhere after S with some confidence[8]. The mutual information (MI) that considers the words order is a measure of the correlation and used to extract trigger pairs from large corpus. The mutual information is given by the following formula:

$$\mathcal{MI}(s,t) = \mathcal{P}(s,t) \log \frac{\mathcal{P}(s,t)}{\mathcal{P}(s)\mathcal{P}(t)}$$

To evaluate the method, we used the Broadcast News Corpus of $140M$ words and set the maximum distance between S and T to 500. Some randomly selected trigger pairs which are sorted in decreasing order of the mutual information are shown.

product ← {maker, company, corporation, industry, incorporate, sale, computer, market, business, sell, machine, consumer, share, software, manufacture, electronic, base, million, manufacturer}

car ← {motor, auto, model, maker, vehicle, ford, buick, honda, inventory, assembly, chevrolet, sale, nissan, incentif, pontiac, plant, toyota, dealer, chrysler}

interest ← {rate, bank, loan, point, dollar, credit, bond, percent, investment, market, reserve, term, debt, investor, billion, exchange, higher, treasury, lower}

fare ← {airline, maxsaver, carrier, discount, air, coach, flight, traveler, travel, continental, unrestrict, ticket, texas, northwest, pettee, match}

music ← {musical, symphony, orchestra, composer, song, concert, tune, concerto, sound, musician, classical, album, violin, violinist, jazz, audience, conductor, play, audio, rock, cello, perform, dance}

pork ← {meat, hog, slaughter, livestock, mercantile, cattle}

plead ← {guilty, sentence, insider, indictment, indict, ivan, charge, attorney, fraud, boesky, lasker, criminal, pleas, investigation, plea, court, prosecutor, prison, felony, defendant, cooperate, palmieri}

We also extracted trigger pairs from the Wall Street Journal Corpus of 1M words. We found that the trigger pairs are domain specific. For example, the triggers to "Stock" in news and media domain (Broadcast News Corpus, 140M tokens) are {company, bond, buy, business, bank, dow, earning, composite, cent, analyst, big, chrysler, investor, cash,

[8] In the Trigger Pairs Model, (S,T) is different from (T,S), so the Trigger Pairs Model is different from the method of using co-occurrence of two words that is generally used in other keywords expansion experiments[12]

average, economy, close, capital, chip, ...}. However, in business and Economic (Wall Street Journal Corpus, 1M tokens) the triggers are {share, investor, index, exchange, price, dow, market, buy, point, jone, trade, trader, average, cent, industrial, gain, shareholder, company, board, ...}

4.2 Keywords Expansion Algorithm

The trigger pair method can provide several candidate refinement keywords. An additional question is, how many and which ones to use under any given circumstances. extract relevant words from large corpus. For a search with only one keyword, the top several triggers to the keyword are used to expand the search. But for a search with more than 2 keywords, the choice becomes more complicated. We use the following algorithm for keywords expansion based on the trigger pairs:

Let us assume that the keywords are $K_1, K_2, \ldots, K_m$, and the the expected number of refinement words is N. Initialize $n = m$, $\mathcal{S}$ is the empty set.

1. $\mathcal{S}_1 = \{s_{11}, s_{12}, \ldots, s_{1i}\} \rightarrow K_1$, $\mathcal{S}_1$ is the triggers set to K_1. $s_{11}, s_{12}, \ldots, s_{1i}$ are sorted in decreasing order of the mutual information.

   $\mathcal{S}_2 = \{s_{21} s_{22}, \ldots, s_{2j}\} \rightarrow K_2$, $\mathcal{S}_2$ is the triggers set to K_2

   ...

   $\mathcal{S}_m = \{s_{m1}, s_{m2}, \ldots, s_{mk}\} \rightarrow K_m$, $\mathcal{S}_m$ is the triggers set to K_m

2. $\mathcal{S} = \mathcal{S} \cup (\forall(\mathcal{S}_p, \mathcal{S}_q, \ldots, \mathcal{S}_r)(\mathcal{S}_p \cap \mathcal{S}_q \cap \ldots \cap \mathcal{S}_r))$, and $(\mathcal{S}_p, \mathcal{S}_q, \ldots, \mathcal{S}_r)$ is one of the combinations of n sets out of m. The words in the S are sorted in decreasing order of mutual information.

3. If $|S| \geq N$, let the top N words in the S be the refinement words and stop.

4. otherwise, let $n \Leftarrow n - 1$, goto 2.

This method can improve the recall rate of the search. For example, if a system uses TF-IDF to extract informative words to index documents, some K_i itself might be ignored because of its low weight. However, some words in $\mathcal{S}_i$ could be selected thus helping to recall documents where the ignored $K_(i)$ appears thus improving recall rate.

This method also provides disambiguation information for ambiguous query words. For example, $K_1 =$ *charge* and $\mathcal{S}_1 =$\{*federal, investigation, attorney, plead, indict, allege, fraud, guilty, indictment, jury, prosecutor, court, case, criminal, law, grand, commission, insider, conspiracy, ...*\}, $K_2 =$*fee* and $\mathcal{S}_2 =$ \{*pay, dollar, million, bank, service, tax, raise, federal, bill, require, percent, charge, paid, law, client, loan, money, legal, payment, ...*\}, then $\mathcal{K} = \{K_1, K_2\} =$ \{*Charge, Fee*\} and $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2 =$ \{*million, pay, dollar, tax, service, federal, client, law, loan, legal, payment, court, suit, file, cost, case, company, firm, ...*\}.

So triggers, such as million, pay, dollar, tax and service, help confine and disambiguate the meaning of the word “charge”.

4.3 Examples on keywords expansion

In this section, we present a typical example of how our refinement method indeed helps improve search results. Suppose the user is interested in documents where the word “stock” appears in its financial meaning. Inputting simply the keyword “stock” to Lycos and Altavista returns the following results.

From Lycos:

1) YOSEMITE STOCK PHOTOS, ROCK CLIMBING, Daniela Masetti PHOTOS

2) YOSEMITE STOCK PHOTOS, ROCK CLIMBING PHOTOS

3) YOSEMITE STOCK PHOTOS, FISHING PHOTO

*4) Stock information Java Applet

5) STOCK GRAPHICS & PHOTOS

*6) American Stock Transfer & Trust Home Page

*7) STOCK CHARTS

*8) GROWTH STOCK ADVISOR FULL DISCLAIMER

*9) Stock information Java Applet

10) Ocean Stock

Only 5 hits are relevant to the financial meaning of “stock” in the top 10.

From Altavista:

1. E. coli Genetic Stock Center

2. Michael Paras Photography: Photographs, Photography, stock photos,stock photo

*3. iGOLF Features - Stocks & Industry - Stock Report: Tuesday,September 5, 1995

4. Cedar Stock Resort Trinity Center Marina

*5. Stock 4 Art: HOME PAGE!

6. NET INFO - Luc Sala - Myster - stock footage

*7. The Official Vancouver Stock Exchange

*8. Stock Club

*9. NIAGARA MOHAWK DECLARES PREFERRED STOCK DIVIDEND

*10. The Italian Stock Exchange

There are 6 hits that are relevant to the financial meaning of the "stock" in the top 10.

At this time, it is difficult for a user to figure out what words should be used to expand or refine the current search. So the trigger pairs can be used to expand the current search. The triggers to "stock" are {share, investor, index, exchange, price, dow, market, buy, point, jone, trade, trader, average, cent, industrial, gain, shareholder, company, board, ...}. If we use the first word "share" in the ranked triggers list to expand the keyword "stock" and send {stock share} to the above two search engines, the following results get returned.

From Lycos:

*1) Share, Stock or CD Secured Loans

*2) Share / Stock Option Scheme Administration

*3) Allfinanz: Stock, Share Dealers

*4) One Share of Stock, Inc. - Ordering Info

*5) One Share of Stock - Product Line

*6) Akiko New Zealand: Stock And Share Market Links (12-Sep-1995)

*7) Akiko New Zealand: Stock And Share Market Links (12-Sep-1995)

*8) Money: $50 can buy share of stock in a company

*9) ONE SHARE OF STOCK - Order Form

*10) One Share of Stock, Inc. - Company Info

Those results are all relevent to the financial meaning of the word "stock".

From Altavista:

*1. South Africa: Stock market: Share price index (dissemination formats)

*2. Denmark: Stock market: Share price index (base page)

*3. ONE SHARE OF STOCK, INC.

*4. Chile: Stock market: Share price index (base page)

*5. Accounting financial software share stock market money portfolio bank mutual f

*6. Singapore: Stock market: Share price index (dissemination formats)

*7. Mexico: Stock market: Share price index (base page)

*8. Netherlands: Stock market: Share price index (base page)

*9. Ireland: Stock market: Share price index (dissemination formats)

*10. Japan: Stock market: Share price index (base page)

Those results are all relevent to the financial meaning of the word "stock".

We can see the results are better than before. We can also refine the search "stock share" if the results are not satisfactory. The intersection of the triggers sets of "stock" and "share" is {stake, outstanding, company, common, quarter, convertible, shareholder, cent, takeover, earning, exchange, incorporate, acquire, million, composite, dividend, percent, point}. Again we can use the words in this set to continue to expand the keywords "stock" and "share" by choosing one or more of them.

4.4 Relevance feedback

One of the most important ways in which current information retrieval technology supports refining searches is relevance feedback. Relevance feedback is a process where users identify relevant documents in an initial list of retrieved documents, and the system then creates a new query based on those sample relevant documents [14]. The idea is that since the newly formed query is based on documents that are similar to the desired relevant documents, the returned documents will indeed be similar. The central problems in relevance feedback are selecting "features" (words, phrases) from relevant documents and calculating weights for these features in the context of a new query [8].

In WebMate agent, the *context* of the search keywords in the "relevant" web pages is used to refine the search because we think that if a user tells the system some page is relevant to his searh, the context of the search keywords is more informative than the content of the page.

Given a relevant page, the system first looks for the keywords (assume K_i is one of the keywords) and context of the keywords (assume the context of the keyword K_i is $\ldots W_{-5}W_{-4}W_{-3}W_{-2}W_{-1}K_iW_1W_2W_3W_4W_5\ldots$). For each keyword $K_(i)$, the system then extracts the chunks of 5words $W_{-5}W_{-4}W_{-3}W_{-2}W_{-1}$ before K_i and the chunks of 5 words $W_1W_2W_3W_4W_5$ after K_i until all the keywords in the query are processed.

Then, a bag of chunks are collected and passed to the processes of deleting the stop words and calculating the frequency. After that, the top several frequent words are used to expand the current search keywords.

For example, the following text is part of the overview of our Intelligent Agents project at CMU[9]. Suppose a user gives this text as a relevance feedback to the search keywords "intelligent agent".

> Intelligent Software Agents
>
> The voluminous and readily available information on the Internet has given rise to exploration of Intelligent Agent technology for accessing, filtering, evaluating and integrating information.

[9] The URL of our project is: http//www.cs.cmu.edu/~softagents.

In contrast to most current research that has investigated single-agent approaches, we are developing a collection of multiple agents that team up on demand—depending on the user, task, and situation—to access, filter and integrate information in support of user tasks. We are investigating techniques for developing distributed adaptive collections of information agents that coordinate to retrieve, filter and fuse information relevant to the user, task and situation, as well as anticipate user's information needs.

Approach is based on:

adaptable user and task models

flexible organizational structuring

a reusable agent architecture

Underlying Technology

Our intra-agent architecture and inter-agent organization is based on the RETSINA multiagent reusable infrastructure that we are developing.

Using our method, the refinement words extracted from the text are {software, structure, reusable, architecture, technology, organizational, network, schedule, research, rise}. Most of the refinement words reflect well the characteristic of the project. But, if instead of using the context method, we considered the whole content of the page when calculating the frequency, then the expanding words would be {software, information, task, area, application, technology, user, current, develop, underlying}. Obviously, the context of the search keywords can reflect the relevance better than the whole content of the web page.

Subsequently, we used the top 5 words {software structure reusable architecture technology} to expand the search "intelligent agent". These are the results returned by Lycos. The content of links marked with "*" are similar to the content of the page given as the "relevant" feedback.

*1) The Agent Building Shell: Programming Cooperative Enterprise Agents

(http://www.ie.utoronto.ca/EIL/ABS-page/ABS-overvie)

*2) The Agent Building Shell: Programming Cooperative Enterprise Agents

(http://www.ie.utoronto.ca/EIL/ABS-page/ABS-overvie)

*3) An Architecture for Supporting Quasi-agent Entities in the WWW

(http://www.cs.umbc.edu/˜cikm/iia/submitted/viewing)

4) Knowledge Sharing Papers

(http://hpp.stanford.edu/knowledge-sharing/papers/R)

5) Knowledge Sharing Papers

(http://hpp.stanford.edu/knowledge-sharing/papers/i)

6) Knowledge Sharing Papers

(http://ksl.stanford.edu/knowledge-sharing/papers/i)

*7) The Agent Building Shell: Programming Cooperative

(http://www.ie.utoronto.ca/EIL/ABS-page/ABS-intro.h)

*8) Special Issue AI in Medicine Editorial Special Issue Artificial Intelligence in Medicine "Architectures for Intelligent Systems Based on Reusable Components"

(http://www.swi.psy.uva.nl/usr/Schreiber/papers/Mu)

*9) CS 791A – Agent Architectures for Information Gathering

(http://centaurus.cs.umass.edu/ig-seminar.html)

*10) Interaction Protocols for Software Agents on the World Wide Web

(http://rbse.jsc.nasa.gov/eichmann/www-s96/interact)

5 Related work

WebWatcher [10][16] is a tour guide for the web. It learns from experiences of *multiple users* to improve its advice-giving skills. Letizia [17] can recommend nearby pages by doing lookahead search. Syskill & Webert [4] is a software agent that learns to rate pages on the Web, deciding which pages might interest a user. Lira [3] works offline and returns a set of pages that match the user's interest. Daily Briefing [11] allows you to use Autonomy Intelligent Agents as Newshounds to sniff out stories and compile a personal daily newspaper with stories, features and articles selected from the Internet to match your requirements. WBI [12] is a personal web agent designed to personalize your web browsing. Metabot [13] is a Java-based, client-server application for searching the web by performing a simultaneous query on multiple web search services. CoolURL [14] is an exploratory technology that enables users to use agent technology to recommend cool URLs to a community of users. Beehive [18] is a distributed system for social sharing and filtering of information. Firefly [15] uses software agents that automate the process of retrieving data from the Web based on what they know about their owner's tastes and interests. Their core technology is the social filtering (or colaborative filtering). WiseWire [16] uses advanced neural net technology

[10] http://www.cs.cmu.edu/Groups/webwatcher/
[11] http://www.agentware.com/main/dailyme.html
[12] http://www.networking.ibm.com/iag/iaghome.html
[13] http://metabot.kinetoscope.com/docs/docs.html
[14] http://support.intel.com/oem-developer/internet/coolurl/COOL_FAQ.HTM
[15] http://www.firefly.com/
[16] http://www.wisewire.com/

and adaptive collaborative filtering to filter all types of digital content that is personally relevant to you.

6 Summary and Future Research

WebMate is a personal agent running on the end user machine. It accompanies users from page to page to provide assistance. It can learn the user profile and compile personal newspaper, help the user improve the search by keyword expansion and relevance feedback, and aid the user in other ways such as alias, reference, prefetch, and monitor bookmarks or web pages for changes.

Currently in WebMate, only words are used to represent a user's profile. We feel that new machine learning algothrims for classifying the new web pages are necessary to improve the accuracy of the recommendation. We are currently implementing phrases, bigram [13] of words and plan to explore the trigger pairs or relevant words to improve the learning. In addition, we are implementing heuristics to filter out advertisements and irrelevant content around web pages comtaining news.

References

[1] Katia Sycara, Anandeep Pannu, Mike Williamson, Dajun Zeng, Keih Decker. *1996, Distributed Intelligent Agents.* Published in IEEE Expert, Intelligent Systems & their applications, Dec, 1996

[2] Shaw Green, Leon Hurst, Brenda Nangle, Padraig Cunninggham, Fergal Somers, Richard Evans. *1997, Software Agents: A review.* http://www.cs.tcd.id/Brenda.Nangle/iag.html

[3] Marko Balabanovic, Yoav Shaham. *1995, Learning Information Retrieval Agents: Experiments with Automated Web Browsing.* Proceedings of the AAAI Spring Symposium Series on Information Gathering from Heterogeneous, Distributed Environments: 13-18.

[4] M. Pazzani, J. Muramatsu, D. Billsus. *1996, Syskill & webert: Identifying interesting web sites.* In AAAI conference, Portland, 1996

[5] Pannu, A. and Sycara, K. *1996, A Personal Text Filtering Agent.* Proceedings of the AAAI Stanford Spring Symposium on Machine Learning and Information Access, Stanford, CA, March 25-27, 1996.

[6] K.Lang. *1995, NewsWeeder: Learning to filter Netnews.* Proceedings of Machine Learning, Morgan Kaufman, San Francisco, 1995

[7] G.Salton and M.J.McGill. *1983, Introduction to Modern Information Retrieval.* McGraw-Hill, New York, 1983

[8] Gerard Salton, Chris Buckley. *1988, Improving Retrieval Performance by Relevance Feedback.* Cornell University, 88-898

[9] Marbo Balabonovic, Yoav Shoham. *1997, Combining Content-Based and Collaborative Recommendation.* Communications of the ACM, March, 1997

[10] Peter W. Foltz, Susan T. Dumais. *1992, Personalized Information Delivery: An Analysis of Information Filtering Methods.* Published in Comminucations of the ACM, 35(12), 51-60, 1992

[11] Paul Resnick. *1997, Filtering Information on the Internet.* http://www.sciam.com/0397issue/0397resnick.html

[12] Chengfeng Han, Hideo Fujii, W. Bruce Croft. *1994 Automatic Query Expansion for Japanese Text Retrieval.* UMass Technical Report

[13] Ronald Rosenfeld. *1994, Adaptive Statistical Language Modeling: A Maximum Entropy Approach.* Carnegie Mellon University, Ph.D. Thesis

[14] Susan Gauch, Robert P. Futrelle. *Experiments in Automatic Word Class and Word Sense Identification for Information Retrieval.* Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval

[15] Kathleen Webster, Kathryn Paul. *1996, Beyond Surfing: Tools and Techniques for Searching the Web.* Jan. 1996, Information Technology

[16] Thorsten Joachims, Dayne Freitag, Tom Mitchell. *1997, WebWatcher: A Tour Guide for the World Wide Web.* Proceedings of IJCAI97, August 1997.

[17] H.Lieberman. *1995, Letizia: An agent that assists web browsing.* In International Joint Conference of Artificial Intelligence, Montreal, Aug. 1995

[18] Bernardo A. Huberman, Michael Kaminsky. *1996, Beehive: A System for Cooperative Filtering and Sharing of Information*

[19] URL: *Collection of Bots in the Web*, http://www.botspot.com/