PersonalSearcher: An Intelligent Agent for Searching Web Pages

Daniela Godoy[1] and Analía Amandi

ISISTAN Research Institute, Facultad de Ciencias Exactas
Universidad Nacional del Centro de la Prov. de Buenos Aires
Campus Universitario - Paraje Arroyo Seco – (7000) – Tandil, Bs. As., Argentina
[1]Also Facultad de Ingeniería, Universidad Nacional de La Pampa
Calle 9 esq. 110 – (6360) – General Pico, La Pampa, Argentina
{dgodoy,amandi}@exa.unicen.edu.ar

Abstract. The volume of information on the Internet is constantly growing. This fact causes that the search of interesting information becomes a time-consuming task. Generally, a user must revise a big number of uninteresting documents and consult several search engines before finding relevant information. A personalized agent, called PersonalSearcher, that assists the user in finding interesting documents in the World Wide Web is presented in this paper. This agent carries out a parallel search in the most popular Web search engines and filters their result, listing to the user a reduced number of documents with high probability of being relevant to him. This filtering is based on a user profile that the agent builds by observing the user behavior on the Web. The agent uses a textual case-based reasoning approach in order to detect specific subjects that the user is interested in and organizes them in a hierarchy that defines the user profile.

1 Introduction

The information available through Internet is constantly growing. This fact causes that the search of interesting information becomes a time-consuming task since this activity involves the analysis and separation of interesting pages from a great set of candidate pages.

Search engines are the most widely spread tools for searching web pages. Users provide a set of words to these engines and wait for a set of pages related to those words. This mechanism based on words that act as keywords in the searching process is easy to use. However, this simplicity for expressing search goals generally produces low levels of precision in the response from these engines.

In this context, users have to dedicate a considerable amount of both time and effort to browse a ranking list of documents. Generally, this list contains a lot of uninteresting documents and just a few really relevant ones.

For example, we can imagine a user looking for web pages about software agents. This user makes a query using the keyword *agents* chosen from many other words that refer to this subject (i.e. *softbots*). Traditional tools return to the user documents about *software agents, travel agents, insure agents*, etc. all at the same time. A

personalized system able to contextualize the user consult according to his preferences and subjects of interest could be preferred over traditional search engines. It could, for example, filter out documents about *travel agent* and *insure agents* for our user.

In this sense, personal agents have been developed to help the management of the increasing volume of information. They are intelligent assistants that make different tasks on behalf of the user to find, filter and access to a great amount of information from different sources, and finally present a reduced and potentially relevant part of this information to their users. These personalized agents use different learning mechanisms to capture users' interests and habits over time.

We present in this article an intelligent agent that learns about users' interests by observing users' behavior while they are carrying out regular activities on the Web. By a content-based analysis of the information extracted by observation, this agent is able to deduce the subjects that a user is interested in, and according to them filters the resulting list of web pages of a traditional search.

Our agent, named *PersonalSearcher*, builds user profiles using a technique for dynamic classification based on textual case-based reasoning. In this article, we present this agent and, particularly, our technique for dynamic classification that allows the agent to filter pages according to personal interests.

The article is organized as follows. Section 2 introduces the functionality of our PersonalSearcher agent. Section 3 treats the construction of a user profile. Section 4 shows some evaluations for PersonalSearcher. Section 5 compares our agent and technique with related works. Finally, conclusions are presented.

2 Agent Overview

Each agent, instance of PersonalSearcher, monitors Web activity of his associated user. This monitoring is made in order to collect documents, which are interesting to the user. For each document read by a user on the standard browser, the agent observes a set of given characteristics in order to determine its relevance degree for that user. These observed characteristics are basically the time consumed on the reading, its length, and so on.

The documents classified as interesting are analyzed to obtain other characteristics, which describe the subject treated on them. For achieving this goal, a textual case-based reasoning approach is used. In this approach, the main characteristics of textual documents are extracted to represent them as cases. Our case-based reasoner deals with these cases in order to learn interesting subjects for the user. At the same time, it organizes them building a subject hierarchy, which determines the user profile for such user.

Users interact with their *PersonalSearcher* expressing their information needs by keywords as usual. The agent in turn posts these keywords to the most popular search engines (Altavista, Infoseek, Excite, etc.), obtaining a set of documents covering a widely portion of the Web.

The relevance degree of each document in relation to the user profile is computed by the PersonalSearcher to determine the convenience of suggesting the document. Only

documents that surpass a given threshold of relevance as regards the user profile are sent back to the user as a result to his query. Figure 1 illustrates this process.

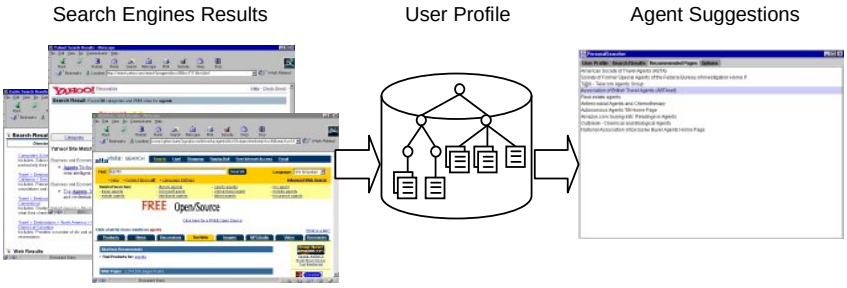


Fig. 1. *PersonalSearcher* task

Once the agent has presented a set of documents as a result for a given query, the user is again observed. This new observation can produce adaptations on the user profile in terms of the user's approval to the agent's suggestions. The complete functionality of our agent is illustrated in Figure 2.

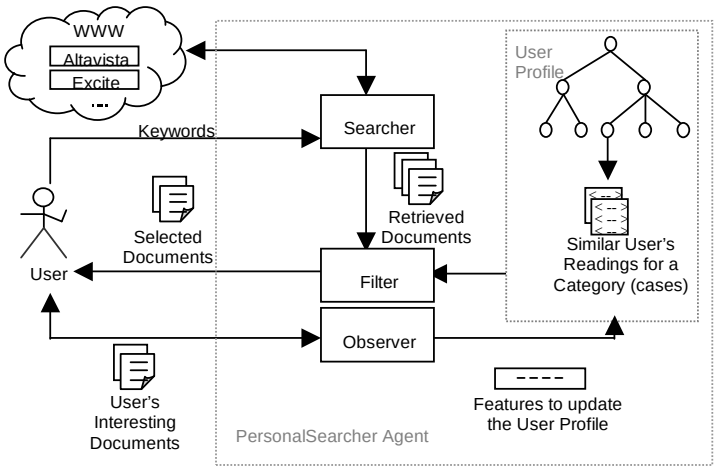


Fig. 2. PersonalSearcher Functionality

3 User Profiling

In order to build a user profile we define a technique based in Case-Based Reasoning (CBR). In CBR a new problem is solved by remembering a previous similar situation (case) and by reusing information and knowledge of that situation [4]. This technique has been used in several areas, including information retrieval and filtering [5,10]. In

the context of our agent, CBR is used to dynamically classify new documents according to their subject.

The assumption in the construction of PersonalSearcher is that the subjects that a user is interested in can be obtained by similarity and frequency analysis of his readings. In order to accomplish both kinds of analyses the readings of a particular user are represented as cases in the context of case-based reasoning. For our agent a case describes a particular experience of page reading on the Web. The agent collects these cases from the user observation and then these are recorded in a case base that defines the user profile.

From these experiences the agent can deduce very specific subjects in which the user is interested, for example into the *software agents* subject, the user could be interested only in *KQML* or *interface agents* documents. This level of specificity is accomplished by first classifying a document according to its generic subjects (*software agents* in the example) and then, analyzing its similarity with cases that represent particular experiences into this subject.

3.1 Building a Subject Hierarchy

A personalized hierarchy of increasing specificity is used to organize the case base according to the subjects which a user is interested in. This kind of organization is common in text collections and it has been successfully applied to Internet directories like Yahoo, Infoseek, etc. in an attempt to categorize the content of the Web.

The subject hierarchy could be seen like a tree. Each internal node on the tree holds features shared by their child nodes and the cases below it. Items without those features live in or below their sibling nodes. While leaf nodes hold cases themselves. This kind of organization of the case base is referred as a *shared featured network* [4]

For instance, a case representing a document about *computers* probably is characterized by the presence of words such as *computer*, *science*, *systems*, *software*, and so on, even when it talks about different sub-subjects like *agents*, *equipment*, *www*, etc. In turn, these sub-subjects will have another set of features in common with their own sub-subjects. For example, all the documents about *agents* will have the words *agent*, *environments*, etc. even when they are about *intelligent agents*, *mobile agents* or *multi-agent systems*. Figure 3 shows an example of user profile.

This hierarchy needs to be built automatically for PersonalSearcher starting from the scratch. To do this, as soon as new cases appear describing user interests, they are grouped by similarity in the case base. Each one of these groups represents a very specific subject of interest to this user. Then, a general inductive process automatically builds a classifier for this subject or category c_i by observing the characteristics of a set of cases that have been classified under c_i. Starting from these characteristics, the inductive process gleans the characteristics that a novel document should have in order to be classified under c_i.

A classifier for a category is composed of a function $CSV_i: D \rightarrow [0,1]$ that, given a document d_j, returns a *categorization status value* for it, i.e. a number between 0 and 1 that, roughly speaking, represents the evidence for the fact that d_j should be classified under c_i. A threshold τ_i such that $CSV_i(d_j) >= \tau_i$ is interpreted as a decision to

classify d_j under c_i, while $CSV_i(d_j)<\tau_i$ is interpreted as a decision of not classifying d_j under c_i [9].
Once a classifier is built representing a generic subject in the hierarchy, new cases belonging to this subject (those with $CSV>\tau_I$) are placed below it and new groups will be created. From these groups new classifiers will be obtained and added like child nodes of the first classifier defining a hierarchy of classifiers. Cases that do not belong to any subject (those with $CSV,<\tau_I$) in one level of the tree will be placed in this level inside the group of cases that correspond. For example, in Figure 3, a case that does not surpass the thresholds for *software agents* and *world wide web* classifiers is grouped into the group of cases identified with the code 45 (if it is similar to them) or a new group is created at this level.

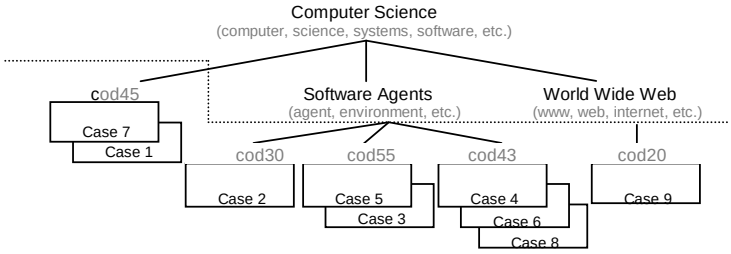


Fig. 3. An example of user profile

3.2 Induction of Classifiers

We use in this work lineal classifiers that represent a category or subject like a vector $c_i=<w_{1i},...,w_{ri}>$ where w_{ji} is the weight associated to the word j in the category i. The CSV function for this classifier is the cosine similarity measure.
Each of these classifiers is focus on a subset of relevant features extracted from the cases stored and grouped by similarity as we explain below. Previous works [3] have proved that a very small number of features are needed for accurate document classifications. For example, there is a small number of words, like *computer*, *science*, *dollar*, *market*, etc, whose presence or absence in a document clearly differentiates documents about *economy* from documents about *computers*. However, more specific words will be needed to distinguish among *computer* sub-subjects.
Each group of cases, representing an specific subject of interest to the user, have a subject definition that resume the features (words) presents on them. The agent creates this definition as soon as a new different case appears and it is updated with the subsequent similar cases. Each time that a new case is added, words on this case that already exist in the subject definition increase its weight in one, those that do not exist are added with weight equal to one and words in the subject definition that do not exist in the case are decreased in one.
The basic assumption is that rare terms (those that appear in few documents) are not relevant to build a category. However, terms that appear in the majority of the documents grouped on a category have a high probability to be important in the subject definition.

Only small set of words, those with weight higher than a given threshold in the subject definition, are selected to define the classifier for that subject. Each time that a definition must be updated for the addition of a new case this set of word is checked for changes. If it didn't suffer any change after a number of incorporations of cases the subject is considered stable and a classifier is obtain to it.

For example, we can suppose that our agent collects three pages about *software agent* read by the user. The set of words with an associated weight is presented in the three first tables of Figure 4. For simplicity, in this example we use the word frequency in each document as its weight, but our agent to obtain this weight uses other considerations. We will explain this later in this paper.

Starting from these pages, a subject definition, shown in the fourth table of the Figure 4, is obtained. On this table, in gray, the words set that finally will be chosen to define the classifier are shown.

Adaptive	5
Agent	4
Agents	3
Computer	3
Conventional	3
Develops	2
Differ	2
Group	2
Institute	2
Laboratory	2
Massachusetts	1
Proactive	1
Projects	1
Prototype	1
Multi-agent	1
Software	1
…	

Academia	16
Advances	14
Agent	7
Agents	6
Autonomous	5
Call	5
Computer	4
Conference	4
Demonstrations	4
Environments	4
Information	3
Program	2
Software	2
Student	2
Systems	2
Technical	1
…	

Address	12
Administration	16
Administrator	11
Agent	7
Agents	6
Agents-digest	6
AI	5
Alto	5
Asked	4
Available	3
Computer	3
Message	1
Messages	1
Multi-agent	1
Software	1
Subscribe	1
…	

→

Agents	3
Computer	3
Agent	3
Software	3
Systems	2
Multi-agent	1
Conference	0
Digest	1
Interface	1
Laboratory	-1
List	1
Proceedings	0
Program	1
Send	1
Subscribe	1
Autonomous	0
…	

≠

Travel	20
Agents	10
American	10
Society	10
Hotels	2
Guidebooks	2
Trademarks	2
Affiliated	1
Airlines	1
Parks	1
Associations	1
Campgrounds	1
Car	1
Chains	1
Dining	1
Firms	1
…	1

Fig. 4. An example of how a classifier is obtained starting from a set of documents

Note that, for the previous example, querying with the keyword *agents* and given a user profile containing reading about *intelligent agents* as this that we are showing, PersonalSearcher retrieves only documents about this subject. A document about another theme like *insure agents*, *travel agents*, etc. will be automatically discarded because it does not have the necessary features to belong to any of the categories the user is interested in. Then the agent can conclude that there is no previous reading on these last topics and these are considered uninteresting for the user. This is shown in the fifth table on Figure 4 where a representation of a document treating about *travel agents* is presented. This document hardly could be classified in the subject *software agent* because from the set of words in the classifier for this subject (*agents*, *computer*, *agent*, *software*, *systems*) just the word *agents* appears frequently in its content.

Returned documents about *intelligent agents* instead will be compared later with cases located below the node representing this subject. If the agent finds another similar case, it will be considered relevant because the user has read about this subject before. If the contrary happens, the case will be also discarded.

3.3 Document Representation as Cases

Cases represent specific knowledge that describes a situation [4]. The main parts of a case are the description of a problem that has been solved, the description of its solution itself and the feedback got from the user for that solution. These components permit to solve a new situation by retrieving relevant cases (the ones matching with the current problem) and adapting their solutions to the new situation.

Words in the content of a document permit to describe a particular situation in our subject classification problem. The solutions to this kind of textual cases are specific subject definitions. In this way when a new document appears with similar distribution of words into its content compared to another document already seen, the agent can deduce that both documents are about the same subject.

To reflect the importance of each word in the document a weight is associated to each of them in the case. Weights are the result of a function $weight(word_i,document_j)$ defined in terms of several characteristics of word *i* into the document *j*.

Our function *weight* considers the number of word occurrences, capitalization of words, word function in the sentence, location in the document, font size and style. Capital words are considered important to define a subject because they generally represent people, countries, cities, etc. Also, nouns are more relevant than other terms into the text like verbs, etc. Other important information is extracted from the HTML code, like word location in the page (title, heading, or normal text), font size and style (bold, underline, etc.). All these characteristics are used as additive factors in the function *weight* in order to get a weighted list of relevant words.

Previous to the document representation as cases, non-informative words such as prepositions, conjunctions, pronouns, very common verbs, etc. are removed using a standard stop-words list. Each word present in this list is not taken into account in the representation because they are topic independent words that appear with similar frequency in the majority of the documents.

Figure 5 shows two Web pages and their representation as cases. The page located at the right side in this Figure has a defined subject in its solution part, since it is a case in the current user profile. The left page needs to be compared with the other one to analyze whether the same subject could be applied to it.

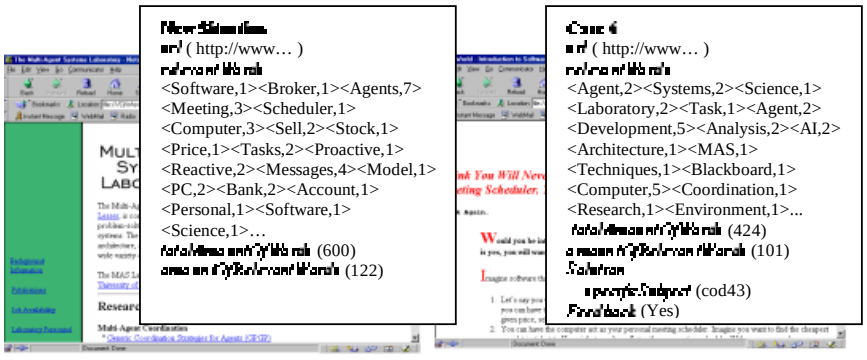


Fig. 5. Case representation for two Web pages

Our agent uses case-based reasoning for discovering the more similar previous cases and thus to suppose the topic treated by a particular page. To do this, the case-based reasoner retrieves from the base a set of cases to compare with the new one.

3.4 Case Similarity

Cases are compared through a number of dimensions that describe them. A function of similarity *Sim* is defined for each dimension on the cases, being the most important the similarity of respective relevant word lists. This similarity is calculated by the *cosine measure* also called *inner product with cosine normalization* [11]:

$$Sim(l_i, l_j) = \cos(\alpha) = \frac{\sum_{k=1}^{r} w_{ik} * w_{jk}}{\sqrt{\sum_{k=1}^{r} w_{ik}^2} * \sqrt{\sum_{k=1}^{r} w_{jk}^2}} \quad (1)$$

where l_i and l_j are the respective list of words, and w_{ik} y w_{jk} the weights of the word k in each list This similarity function measures the cosine of the angle α between the vectors representing both documents.
A numerical evaluation function, that combines the matching of each dimension with the value of importance given to that dimension, permits to obtain the global similarity of the entry case (C_E) and retrieved (C_R) respectively. The function used in our agent is the following:

$$S(C_E, C_R) = \sum w_r * Sim_r(f_r^E, f_r^R) \quad (2)$$

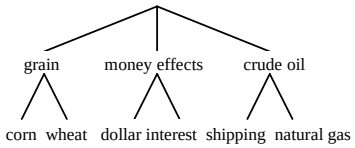
where w_i is the importance of the dimension i, Sim_i a similarity function for this dimension, and f_i^E, f_i^C are the values for the feature f_i on both cases.
If the similarity value obtained from S is higher than a given threshold, the cases are considered similar and the agent can conclude that both cases are about the same specific subject. For instance, in Figure 5, if the case representing the new situation and the case 6 are considered similar, the subject identified by the code 43 belonging to the case 6 is applied to the new situation and added to the case base as a new member of the group of cases treating the subject identified with this code.

4 System Evaluation

In order to test our technique for dynamic classification of topics we use the Reuters 21578[1] collection of newswire articles. Articles in this collection have been classified into 118 categories. We restrict ourselves to the categories in the hierarchy shown in Figure 6, also used in [3]. Those articles with no assigned category were removed from consideration and also those with more than two categories. Remaining articles were divided into training (70% of the total amount of articles in the category) and

[1] Reuters 21578 is available from David Lewis at http://www.research.att.com/~lewis

test (30%) set. The first set was used to create the hierarchy and the second one to test the classification on this hierarchy.
The standard precision and recall techniques from information retrieval were used to evaluate the performance on a single category. Precision measures the proportion of items placed in the category that really belong to this category, and recall measures the proportion of items in the category that are actually placed in the category. Table on Figure 6 summarizes average of precision and recall (also call breakeven point) for the selected categories.



Category	Training	Test	Breakeven Point
Grain	36	15	61%
Money effects	213	92	42%
Crude Oil	283	121	75%
Corn	65	28	45%
Wheat	114	49	69%
Dollar	92	40	52%
Interest	318	137	59%
Shipping	161	69	54%
Natural Gas	71	31	43%

Fig. 6. Results from the evaluation of PersonalSearcher

5 Related Works

A number of personal assistants that help users on query formulation or guide the navigation having some information about their personal interests have been built in the last years. Some recent developments include: Personal WebWatcher [7], Letizia [6], Syskill&Webert [8], FAQFinder [1], BROADWAY [2], PTV[10].
Most of these assistants use different models to represent documents coming from information retrieval area. For example, the vector space model is used in Letizia, Personal WebWatcher and Syskill&Webert. Our PersonalSearcher extended this model using case-based reasoning to the document representations. Cases contain a vector to represent the document content but contextual information is also added to enrich the representation.
Some assistants have applied Case-Based Reasoning to capture domain experiences during the search of relevant documents in different ways. For example, the FAQFinder goal is to answer natural language questions by retrieving these from frequently asked questions archives. This system compares question-answer pairs represented by cases with the query entered by the user. BROADWAY goes with a user during the navigation and advises him documents potentially relevant according to previous navigation experiences of a group of users represented by cases. PTV generated personalized TV guides using a combination of CBR and collaborative filtering approaches. TV programs are represented by cases and compared with the information in the user profile to be recommended by this system.
The main difference of our work with previous ones that apply CBR resides in the usage of this technique to identify subjects of interests from user readings,

dynamically classify incoming documents from a user query according to its subjects and finally establish its relevance with regard to the user profile.

6 Conclusions

PersonalSearcher, an agent able to observe user behavior relative to activities on the Web and build user profiles has been presented in this paper. This agent then filters the response of a given requirement to offer a reduced number of Web pages considered potentially relevant to the user based on his profile.
The main contribution of this work is the specification of a technique based on case-based reasoning to build a user profile.

Acknowledgements

This study was supported in part by the Cooperation Project between Argentina-Brazil under grant BR 17/00.

References

1. Burke, R., Hammond, K., Kozlovsky, J.: Knowledge-Based Information Retrieval for Semi-Structured Text. Working Notes from AAAI Fall Symp. AI Applications in Knowledge Navigation and Retrieval. AAAI Press, Menlo Park, California (1995).
2. Jaczynski, M., Trousse, B.: Broadway: A World Wide Browsing Advisor Reusing Past Navigations from a Group of Users. Proceedings of the Third UK Case-Based Reasoning Workshop, Manchester (1997).
3. Koller, D., Sahami, M.: Hierarchically classifying documents using very few words. Proceedings of the Fourteenth International Conference on Machine Learning (1997).
4. Kolodner, J.: Case-Based Reasoning. Morgan Kaufmann (1993).
5. Lenz, M., Hübner, A., Kunze, M.: Question Answering with Textual CBR. Proceedings of the International Conference on Flexible Query Answering Systems. Denmark (1998).
6. Lieberman, H.: Letizia: An agent that assists web browsing, Proceedings of the International Joint Conference on Artificial Intelligence, Montreal (1995).
7. Mladenic, D.: Machine learning used by Personal WebWatcher. Proceedings of ACAI-99 Workshop on Machine Learning and Intelligent Agents, Chania, Crete (1999).
8. Pazzani, M., Billsus, D.: Learning and Revising User Profiles: the Identification of Interesting Web Sites. Machine Learning (1997).
9. Sebastiani, F.: Machine Learning in Automated Text Categorization. Technical Report IEI-B4-31-1999, Istituto di Elaborazione dell'Informazione, Consiglio Nazionale delle Ricerche, Pisa, IT (1999).
10. Smyth, B., Cotter, P.: Surfing the Digital Wave: Generating Personalised Television Guides using Colaborative, Case-based Recommendation. In proceedings of the third International Conference on Case-based Reasoning. Munich, Germany (1999).
11. van Rijsbergen, C. F.: Information Retrieval, 2d ed. London: Butterworths (1979).