User Modeling and User-Adapted Interaction **14:** 159–200, 2004.

Anatomy and Empirical Evaluation of an Adaptive Web-Based Information Filtering System

ALESSANDRO MICARELLI and FILIPPO SCIARRONE
Department of Computer Science and Automation, Artificial Intelligence Laboratory, University of Roma Tre, Via della Vasca Navale 79, 00146 Rome, Italy.
e-mail: {micarel, sciarro}@dia.uniroma3.it

(Received: 19 March 2002; accepted in revised form 18 July 2003)

Abstract. A case study in adaptive information filtering systems for the Web is presented. The described system comprises two main modules, named HUMOS and WIFS. HUMOS is a user modeling system based on stereotypes. It builds and maintains long term models of individual Internet users, representing their information needs. The user model is structured as a frame containing informative words, enhanced with semantic networks. The proposed machine learning approach for the user modeling process is based on the use of an artificial neural network for stereotype assignments. WIFS is a content-based information filtering module, capable of selecting html/text documents on computer science collected from the Web according to the interests of the user. It has been created for the very purpose of the structure of the user model utilized by HUMOS. Currently, this system acts as an adaptive interface to the Web search engine ALTA VISTA$^{\text{TM}}$. An empirical evaluation of the system has been made in experimental settings. The experiments focused on the evaluation, by means of a non-parametric statistics approach, of the added value in terms of system performance given by the user modeling component; it also focused on the evaluation of the usability and user acceptance of the system. The results of the experiments are satisfactory and support the choice of a user model-based approach to information filtering on the Web.

Key words. artificial neural networks, case-based reasoning, empirical methods, information filtering, user modeling

1. Introduction

It is often claimed that World Wide Web search engines are too sophisticated for the user's own good. They submerge her/him with an unmanageable number of documents ferreted out from sites all over the world–a phenomenon called 'information overload'. In reality, these search engines are not sophisticated enough from a human-computer interaction perspective. A truly refined engine would avoid retrieving the only marginally useful documents which abound in typical 'hit lists' and which get chosen simply because they happen to contain the key words appearing in the user's search request. A truly sophisticated engine would try to 'guess' exactly what kind of document the user desires, basing that guess not only on the key words provided by the user, but also on a profile of the user's background and interests and on evaluations of how the system satisfied or failed to satisfy the user's requests in the past. In other words, a truly sophisticated system would have an adaptive user modeling component. Moreover, it would retrieve only the specific kind of documents defined by the user modeling component, basing its selection of possible 'hits' on a relevance evaluation heuristic. The system might have, for example, an information filtering component which runs summary semantic analysis of all documents bearing promising titles. The combination of these two components would clearly offer the user a significantly higher probability of finding the best suited documents within those first presented on the screen; it should also permit the user to eliminate confidently any file from the bottom of the list, without wasting time inspecting it.

This paper presents a step forward in the realization of a system like the one just described. Technically, the system is a shell, coded in Java, which sits on top of ALTA VISTA[1] and transparently customizes searches and retrievals of computer science literature for users (who may not even be aware that they are using ALTA VISTA). The system comprises two self-contained modules–a user modeling component named HUMOS (Hybrid User Modeling System) and an information filtering component named WIFS (Web-oriented Information Filtering System)–plus an application specific module (the 'external retriever') which interfaces to ALTA VISTA.

The principal aim of this research project is therefore to lend support to a user-model based approach to information filtering on the Web, through the construction and empirical validation of a working system. This 'constructive' approach makes it possible to assess the added value that the user modeling component can offer. What is more, the approach presented here is characterized by the particular way in which (i) the user model is represented, (ii) the model itself is constructed and maintained, and (iii) the retrieved documents are ranked.

The user model managed by HUMOS represents the long-term information needs of the user. The model is structured as a frame containing informative words characterizing the domain. A distinctive trait of the model is its capacity to grow and modify itself dynamically from session to session, by adding slots and constructing semantic networks linking words which co-occur. It is our contention that such a trait makes the system most effective for filtering information 'in the wild', e.g. on the Web. The machine learning approach we propose for user modeling is based on stereotypes and uses an artificial neural network for the assignment of stereotypes to the user. Unlike other schemes based on a neural approach to user modeling (e.g., (Jennings and Higuchi, 1993; Chen and Norcio, 1997)), our method allows us to maintain a symbolic representation of the model; this facilitates handling co-occurring terms and non-monotonicity in the user's reasoning. The user model is dynamically updated by the system on the basis of relevance feedback that the user may provide with respect to the selected documents.

WIFS evaluates the documents initially retrieved by ALTA VISTA only on the basis of their contents, i.e., it is a *content-based* system (Oard, 1997; Hanani et al., 2001).[2]

[1] Trade Mark by Overture Services, Inc., www.altavista.com.

[2] Another important class of systems proposed in the literature is formed by *collaborative filtering* and *social based* systems (Goldberg et al., 1992; Hanani et al., 2001; Maes, 1994; Oard, 1997), which utilize other users' opinions and notes in order to obtain their assessment of a specific document.

The representation of the documents to be evaluated is based on the formalism chosen to represent the user model. Moreover, the present method for evaluating the relevance of documents is characterized by the fact that the implemented ranking function considers the user's two-fold information needs as represented in the user model and in the query formulated.

The system, as a whole, has been tested by means of a controlled experiment to evaluate the added value given by the user modeling component in terms of performance, according to well-known metrics proposed in the literature. As for the analysis of the experimental data, a non-parametric test for hypothesis testing has been chosen.[3] The reason of this choice is that we believe that non-parametric tests can be better applicable to computer science problems and systems (in particular, adaptive information filtering systems) where the 'human factors' play a crucial role and for which any kind of strong assumption concerning the distribution of the population might only be a matter of opinion. Our evaluation has shown that the system improves ALTA VISTA performance (according to the metrics used) up to 34%. An evaluation of the usability of the system and of user acceptance has been made through a questionnaire. The results of the experiments are satisfactory and support the choice of a user model-based approach to information filtering on the Web.

The next sections are organized as follows: Section 2 sets forth an overview of the entire system; Section 3 describes HUMOS, particularly with respect to the structure of stereotypes, the structure of the user model and the user modeling process; Section 4 describes WIFS and presents the filtering and feedback algorithms; Section 5 reports the empirical evaluation of the system; Section 6 contains comments and comparisons with respect to related work. Finally, our conclusions are set forth in Section 7.

2. General Architecture and Example Session

The general architecture of the system is shown in Figure 1. It is client-based, and comprises the following components:

- The *User Model*, which represents the 'information needs' of a particular user.
- The *User Modeling* component (HUMOS), which is capable of dynamically building the user model, as inferred by the system through the interaction.
- The *External Retriever*, which interfaces to ALTA VISTA.
- The *Information Filtering* component (WIFS), which selects the documents relevant to the user according to the content of the documents and of the user model.
- The *User Interface*, which manages the interaction.

The system is used to filter html/text documents on computer science collected from the Web, where a selection of the documents relevant to a particular user is performed on the basis of a model representing her interests. To perform its search the system

[3] We recall that non-parametric tests, unlike parametric ones, do not make restrictive assumptions (e.g., 'normality') about the population distributions (Devore, 1995).

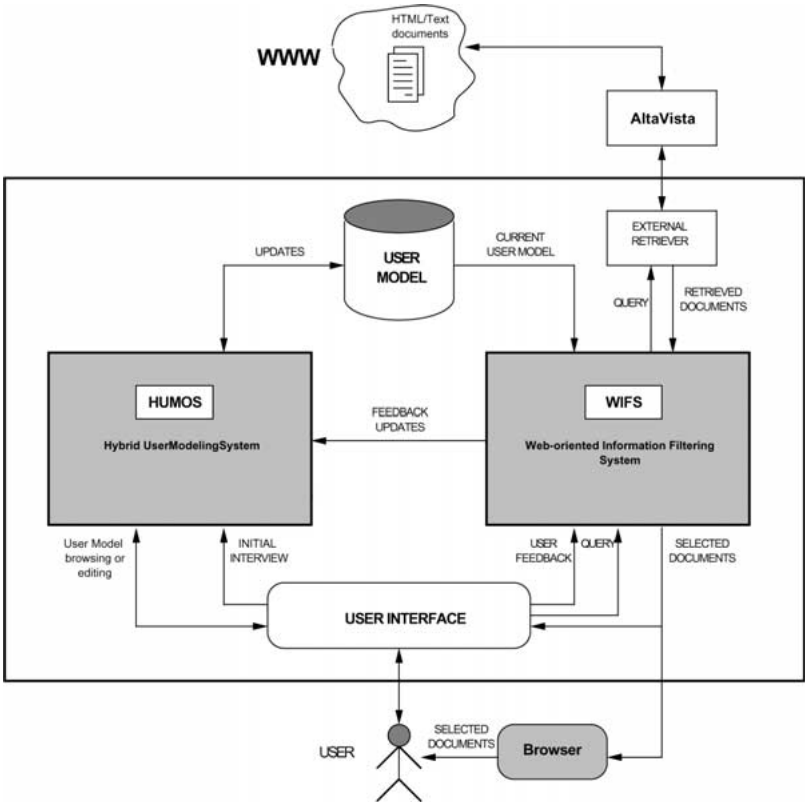


Figure 1. The architecture of the complete system.

exploits ALTA VISTA, used both in advanced and simple query modality to exploit the vast range of information gathered from the Web. The system is therefore used as an adaptive interface to search engines: it allows the user to insert a query and collect incoming documents to be filtered. This 'parasite' modality represents an alternative approach to other information filtering systems proposed in the literature (see Section 6), where documents are automatically collected from the Internet starting from a basic set of sites/documents, and 'surfing' to related links, in a way similar to Web crawlers (Chakrabarti et al., 1999). This choice was the result of a reasoning developed on the basis of the following ideas:

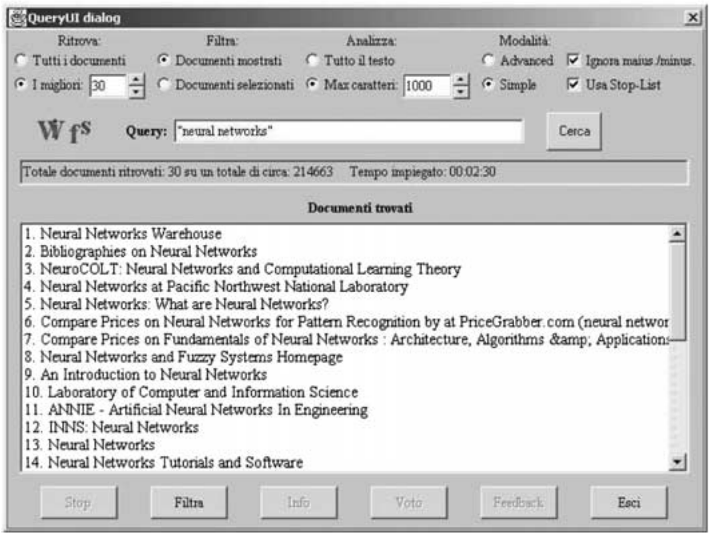
1. maximize the possibilities offered by the huge amount of information available on the Web (the documents indexed by search engines are hundreds of millions);
2. integrate standard technologies available on the Web (well known features, syntax, performances and limits of search engines);
3. increase the speed of user interaction. In fact, query-based searches are simpler and faster than auto-surfing modalities, especially if the user is not sure where to start her search from, and only knows the relevant items she is looking for.

For a better understanding of the tasks and interactions of the process, a simple example session with the system is illustrated here below.

During the initial phase of the process, the system obtains login information from the user (*Initializing* phase) and tries to retrieve the corresponding model from a library of user models (not represented in Figure 1) created by the system on the basis of previous interactions with former users. If the user is a *new user*, a preliminary interview ('Initial Interview' in Figure 1) is performed in order to obtain a first set of her 'information needs'. Different windows, which list a number of terms relative to computer science topics, are displayed on the user's screen. The user is then asked to specify, by clicking on combo boxes, her own 'interest' score for each topic using an integer relevance value ranging from -10 to $+10$, positive for 'interesting' topics, negative otherwise. The interview then continues and the user is asked to give her interests about more detailed topics.

In the next phase (*Querying* phase) the user is shown a window where she can set the searching and filtering modalities and input the query relative to computer science topics ('Query' in Figure 1). In accordance to ALTA VISTA syntax, the user can write both boolean queries (boolean AND/OR combinations of keywords) and structured queries (which allow the user to confine matches to certain attributes, such as the document type, title, host, etc.). In Figure 2 an example of a snapshot is shown: the user is looking for documents about 'neural network'. In this phase, she can set the following parameters in the upper section of the window:

- The maximum number of ALTA VISTA documents that may be retrieved ('Ritrova'). This cap substantially reduces the time necessary for the retrieval of a set of documents. In fact, the incoming documents are already ranked



Ritrova: ○ Tutti i documenti ● I migliori: 30

Filtra: ● Documenti mostrati ○ Documenti selezionati

Analizza: ○ Tutto il testo ● Max caratteri: 1000

Modalità: ○ Advanced ● Simple

☑ Ignora maius./minus.

☑ Usa Stop-List

Query: "neural networks"

Totale documenti ritrovati: 30 su un totale di circa: 214663 Tempo impiegato: 00:02:30

Documenti trovati

1. Neural Networks Warehouse
2. Bibliographies on Neural Networks
3. NeuroCOLT: Neural Networks and Computational Learning Theory
4. Neural Networks at Pacific Northwest National Laboratory
5. Neural Networks: What are Neural Networks?
6. Compare Prices on Neural Networks for Pattern Recognition by at PriceGrabber.com (neural networ
7. Compare Prices on Fundamentals of Neural Networks : Architecture, Algorithms &amp; Applications
8. Neural Networks and Fuzzy Systems Homepage
9. An Introduction to Neural Networks
10. Laboratory of Computer and Information Science
11. ANNIE - Artificial Neural Networks In Engineering
12. INNS: Neural Networks
13. Neural Networks
14. Neural Networks Tutorials and Software

Figure 2. The query and the list of retrieved documents.

by ALTA VISTA and, in most cases, only a minimal part of these are actually relevant to the user.
- The ALTA VISTA query modality ('Modalità') Simple or Advanced.

The search engine connection is activated by clicking on the 'Cerca' (Submit) button. The query input by the user is passed to ALTA VISTA, which returns the URLs of the retrieved documents. Figure 2 depicts the window after the search. The main text area contains the ranked list of titles of the documents retrieved by ALTA VISTA.

At this point, the next two phases (*Collecting* and *Filtering* phases) can be activated by clicking the 'Filtra' (Filtering) button. The following parameters can be set in the upper section of the window:

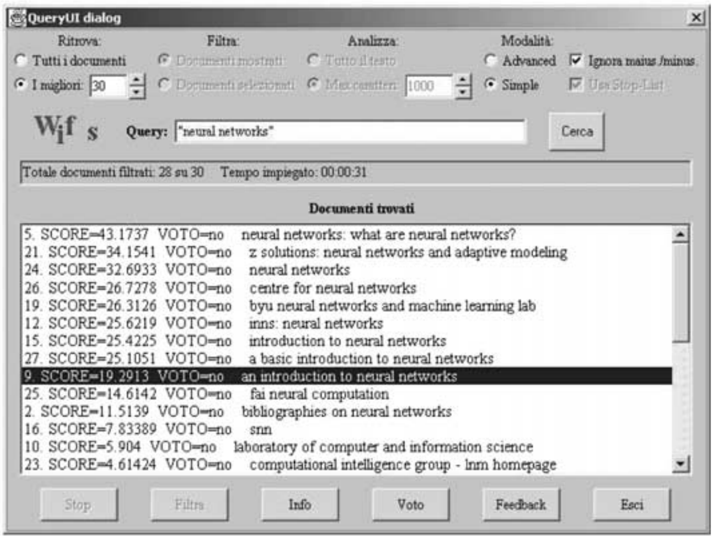
- the listed documents to be filtered: all the retrieved documents ('Documenti mostrati') or only the clicked documents ('Documenti selezionati');
- the number of characters of the texts to be analyzed ('Analizza');
- the 'ignore case' and the stop-list activation check-boxes (The stop-list is a list of words such as articles, conjunctions, etc. that can be skipped during the document analysis).

The *Collecting* phase is performed by the External Retriever which obtains from the hosts the documents to be analyzed. Note that the user can choose whether to filter (i.e., to analyze and select) all the listed documents or, alternatively, a subset selected by clicking the respective titles (in the example window, the user chose to filter all documents). Although WIFS sets up multi-threaded text-only connections to the Web sites (which are very fast since they skip the image files), it may be useful to limit the number of documents to be downloaded in order to reduce the waiting time. Another useful optimization is binding the maximum number of characters to be analyzed in each text as it prevents documents of several Mbytes from locking the process.

The *Filtering* phase automatically starts whenever a document is retrieved. The system activates a matching algorithm in order to assign a Score to each document, calculated in terms of the similarity between the document, the current user model and the query. All the selected documents are ranked by a descending score and the corresponding titles are listed on the screen, as shown in Figure 3. The number appearing on the extreme left of each line is the position as initially ordered by ALTA VISTA, whereas the number following 'Score=' represents the score assigned to the document by WIFS. The term following 'Voto=' states if the user has or has not expressed a relevance feedback on that document. The final outcome of this phase is thus a classification of documents on the basis of their potential relevance to the user.[4]

The user is allowed to view a document at any time, be it before, during or after the filtering process. In fact, a simple double-click on the title of a desired document will

[4] In actual fact, an information filtering system, proper, should only supply those documents that are relevant for the specific user. Naturally, WIFS can achieve this by presenting only those documents that pass a predetermined threshold on the basis of the evaluation made by the system during the filtering phase. However, currently WIFS returns the entire document list, appropriately ordered, in compliance with our users' specific request.



QueryUI dialog

Ritrova: ○ Tutti i documenti ● I migliori: 30

Filtra: ● Documenti mostrati ○ Documenti selezionati

Analizza: ○ Tutto il testo ● Max caratteri: 1000

Modalità: ○ Advanced ● Simple ☑ Ignora maius./minus. ☑ Use Stop-List

Query: "neural networks"

Totale documenti filtrati: 28 su 30 Tempo impiegato: 00:00:31

Documenti trovati

```
5. SCORE=43.1737  VOTO=no   neural networks: what are neural networks?
21. SCORE=34.1541  VOTO=no   z solutions: neural networks and adaptive modeling
24. SCORE=32.6933  VOTO=no   neural networks
26. SCORE=26.7278  VOTO=no   centre for neural networks
19. SCORE=26.3126  VOTO=no   byu neural networks and machine learning lab
12. SCORE=25.6219  VOTO=no   inns: neural networks
15. SCORE=25.4225  VOTO=no   introduction to neural networks
27. SCORE=25.1051  VOTO=no   a basic introduction to neural networks
9. SCORE=19.2913  VOTO=no   an introduction to neural networks
25. SCORE=14.6142  VOTO=no   fai neural computation
2. SCORE=11.5139  VOTO=no   bibliographies on neural networks
16. SCORE=7.83389  VOTO=no   snn
10. SCORE=5.904  VOTO=no   laboratory of computer and information science
23. SCORE=4.61424  VOTO=no   computational intelligence group - lnm homepage
```

Figure 3. The list of selected documents.

prompt the browser to proceed to the corresponding URL. Significantly, many of the documents on top of the rank-ordered list[5] reported in Figure 3 were ranked at the bottom of the list by ALTA VISTA.

The user may also give an evaluation of the viewed documents (*Feedback* phase). When the user clicks on the 'Feedback' button shown in Figure 3, the system asks for a relevance value (*relevancefeedback*), expressing the user's evaluation for the selected document. The user feedback ('User Feedback' in Figure 1) is used by the system to update the user model with new information concerning the user's interests.

At any time during the interaction the user is allowed to browse or edit her user model. We shall address this feature in the next section, which specifically describes the user modeling component.

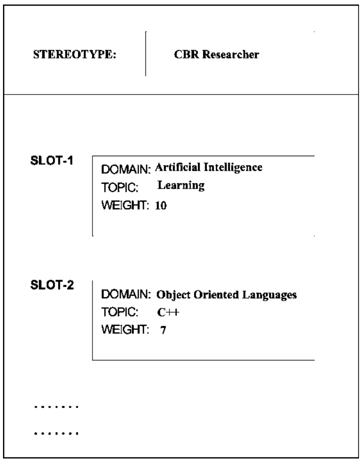
3. The User Modeling Component

The next Subsections (3.1–3.4) describe the knowledge bases and the features that characterize HUMOS. Specifically, these involve: the knowledge base of stereotypes, the structure of the user model, the system for the maintenance of consistency of user models and the complete user modeling process.

3.1 THE KNOWLEDGE BASE OF STEREOTYPES

Similarly to other systems presented in the literature (see for example (Finin, 1989; Brajnik and Tasso, 1994; Kobsa and Pohl, 1995)), HUMOS can be classified as a

[5] Ordered according to the model of the user who input the query.



STEREOTYPE: CBR Researcher

SLOT-1
DOMAIN: Artificial Intelligence
TOPIC: Learning
WEIGHT: 10

SLOT-2
DOMAIN: Object Oriented Languages
TOPIC: C++
WEIGHT: 7

....
....

Figure 4. An excerpt of the ‘CBR Researcher’ stereotype.

stereotype-based user modeling system (Rich, 1989). Stereotyping is a way of default reasoning about the user: by classifying the user we exploit an extensive amount of default information we possess on that particular class of users. This information may be revised later on by the system when it obtains more accurate knowledge about the user’s interests. In our system, a stereotype describes the ‘information needs’ of a prototypical user belonging to the class represented by that stereotype, who is interested in useful html/text documents from the Web. The specific field of interest considered by our system is Computer Science. The knowledge base of stereotypes was built by using the *behavioral approach* (Shapira et al., 1997), through consultation of human experts in the domain. A stereotype is represented in the form of a *frame*, whose *slots* describe the various interests (or non-interests) of the prototypical user. Each slot comprises three *facets*: *domain*, *topic* and *weight*. The values of the *domain* and *topic* facets are ‘terms’ (simple or compound) relevant to the domain of interest, whereas the value of the *weight* slot indicates the degree of interest for the topic (or better, for both $\langle domain, topic \rangle$), ranging from -10 to $+10$, positive for interesting topics, negative for non interesting ones. Figure 4 shows an example excerpt of the ‘CBR Researcher’ stereotype. In this example, Slot-1 relates to the `Artificial Intelligence` domain and to the Learning topic. The value of the weight facet is $+10$, a value that expresses the highest possible interest score for the respective subject-matter. Instead, Slot-2 concerns the Object Oriented Languages domain and the C++ topic. The value of the weight facet is $+7$ (a value that indicates a decidedly high interest rate for the respective item since, for example, various CBR shells are implemented using C++).

The *complementary tool method* (Shapira et al., 1997) was used for incorporating stereotypes in user modeling. The knowledge base of stereotypes is organized as ‘flat memory’, i.e., there is no explicit hierarchy among stereotypes. This organization promotes the use of our proposed approach to user modeling, as shown in the Section

3.4. A stereotype that fits the current user description is called *active stereotype*[6]. When a stereotype becomes active, it is used to build or update the user model according to the default values present in its slots.[7] The methods for the assignment of active stereotypes are also described in Section 3.4.

3.2 THE USER MODEL

A user model may be viewed as a folder where the user preferences are stored. It is represented as a frame, and is divided into two parts: the *header*, representing the user's personal data and the list of active stereotypes, and the *body*, representing the user's 'information needs'. The *body* consists of a collection of slots with the following facets:

- *Domain*: a subject of interest of the user.
- *Topic*: for each domain the user's interests may be represented in the form of a 'term'.
- *Weight*: each slot is embedded in the model with a weight expressing how much that slot has to be trusted as interesting for the user. Weights are real numbers and can be positive (an interesting topic for the user) or negative (not interesting); the score range is $[-10, +10]$.
- *Semantic Links*: terms co-occurring in a document with a topic of a slot are linked to such a slot. This allows for the creation and maintenance of semantic networks (this data structure will be described later on).
- *Justification Links*: a justification link is merely a pointer from a slot to the component on which its presence in the model directly depends. Currently we are using four possible values: initial interview, one or more active stereotypes, relevance feedback, direct editing. The Justification Links are used to maintain the consistency of the user model (see Section 3.3).

Figure 5 shows an excerpt of an example user model, where only one active stereotype ('CBR Researcher') is present. We shall now examine the contents of the slots represented in the figure. Slot-1 relates to the Artificial Intelligence domain and the Learning topic. The weight facet is +9 (a value that expresses the user's considerable interest for the respective topic). There is one Justification Link, shown as Interview. This means that the facet values of Slot-1 have been deduced through the initial interview with the user. Slot-2 concerns the Object Oriented Languages domain, and the C++ topic. The value of the weight facet is +7. There is one Justification Link, constituted by the CBR Researcher active stereotype. This means that the facet values of Slot-2 have been obtained by default from the active stereotype. Finally, Slot-s relates to the Internet domain, and the Web topic. The weight facet is +7. A Justification Link is present, indicated as Feedback. This means that the Slot-s facet

[6] In our system, it may happen that two or more stereotypes are activated simultaneously.

[7] In the event of more than one active stereotype, the user model inherits from them the slots with the highest weight facet value.

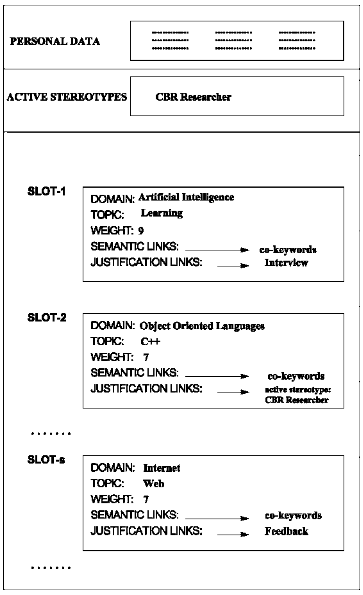


Figure 5. An except of an example user model.

values have been obtained by the system on the basis of a relevance feedback provided by the user in respect to a selected document.[8]

Semantic Links, summarily indicated as co-keywords, are present in the slots. The system enables the association of a slot's $\langle$*domain*, *topic*$\rangle$ pair to a set of terms (appropriately named co-keywords) that concern terms co-occurring in a document with the one specified by such a slot: this mechanism is represented as a semantic network. Figure 6 illustrates an example of such a network in relation to Slot-s of Figure 5. As clearly shown, the network's core node concerns the topic represented by the slot (the *planet*), whereas the *satellite* nodes represent the co-keywords that are linked to the planet by means of weighted arcs, where the weight indicates the rate of affinity between the respective satellite and the planet. The affinity rate is a real number in the range [0, +10]. Basically, this structure represents a sort of a localized semantic context where the topic specified in the slot may be found. It is built dynamically by the system and, as further detailed in Section 4.4, the system may add or eliminate co-keywords and change their respective affinity rates, in accordance with the user's behavior.

The reader may have noticed that the weight of the model's Slot-1 is different from the one of the active stereotype, despite the fact that the latter owns the slot relating to the $\langle$ `Artificial Intelligence, Learning`$\rangle$ pair. What happened in this case is that, during the model update process, the system gave a higher priority to the information (+9 weight) originating from the initial interview than the one

[8] The specifics of the feedback updates are explained in greater detail in Section 4.4.

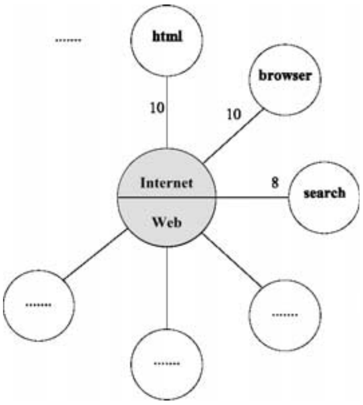
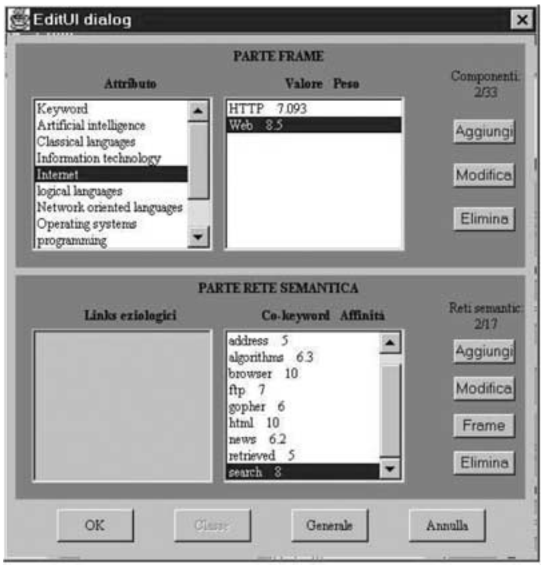


Figure 6. An example of a semantic network.

in the active stereotype (+10). These problems, relating to the priority of the various information sources and to the maintenance of data consistency in the user model, will be soon addressed in Section 3.3.

The contents of a user model are not static, but actually evolve dynamically during the interaction with the user (see Section 4.4). The user is given the possibility to browse or edit her model (for an example of user models partially generated by automatic methods and partially by direct user involvement see (Waern, 2004)). Accordingly, Figure 7 illustrates the interface which the user has at her disposal for the



PARTE FRAME

Attributo

Keyword
Artificial intelligence
Classical languages
Information technology
Internet
logical languages
Network oriented languages
Operating systems
programming

Valore	Peso
HTTP	7.093
Web	8.5

Componenti: 2/33

PARTE RETE SEMANTICA

Links eziologici

Co-keyword	Affinità
address	5
algorithms	6.3
browser	10
ftp	7
gopher	6
html	10
news	6.2
retrieved	5
search	8

Reti semantic: 2/17

Figure 7. The interface for browsing or editing a user model.

purposes of such operations, where the model is made actually visible, as suggested in (Kay, 1995). This figure depicts a browsing of the co-keywords of the above-mentioned ‘Web’ topic. By means of this interface the user may also add, modify or remove the various slots of the model, the weights, the co-keywords and the affinity rates.

3.3. CONSISTENCY MAINTENANCE OF THE USER MODEL

The user model may be viewed as a knowledge base where the user information needs are represented in an explicit way. During the operation of the system, this knowledge base may undergo various changes, which may occur due to the following reasons:

- Feedback updates: the system updates the user model on the basis of the relevance feedback given by the user after the analysis of a particular document.
- Direct editing: the user may update directly the model by adding, modifying or removing slots.

Any modification of the user model may however result in contradictions occurring with the facts already present in the knowledge base. Consequently it is necessary to maintain the consistency of the model using *belief revision* techniques. We have managed the belief revision problem by means of a simple Truth Maintenance System (TMS).

A TMS is a module capable of keeping track of dependencies among beliefs and retracting them efficiently when necessary (Doyle, 1979; McAllester, 1980). Several types of TMSs have been proposed in the literature. In HUMOS we implemented a Justification-based Truth Maintenance System (JTMS), i.e., a kind of TMS where the *justification* is the only logical constraint allowed. The JTMS is the simplest kind of TMS; its implementation is efficient from a computational point of view and its mechanisms are in any event sufficient for the purposes of HUMOS.[9]

We have already examined justification examples in Section 3.2. Slot-2 of Figure is justified by the fact that ‘CBR Researcher’ is the active stereotype:

$$\textit{CBR-Researcher} \overset{\textit{default}}{\Longrightarrow} \langle \textit{Object Oriented Languages}, C{+}{+}, +7\rangle$$

Let’s now assume that the user gives relevance feedback on a document supplied by the system, thus causing a feedback update of the model. If, due to this update, the ‘CBR Researcher’ active stereotype is removed (and replaced with another active stereotype), the JTMS will eliminate all those assertions that depend on the ‘CBR Researcher’ stereotype from the user model. In other words, the slot ⟨Object Oriented *Languages*, C++, +7⟩ will be removed. This slot will generally adopt the value that could issue from the new active stereotype, or from the feedback update, following the method described here below.

[9] The interested reader may consult (Forbus and De Kleer, 1993) for the technical details relating to the implementation of the different types of TMS proposed in the literature.

Where a slot may be updated in more than one way, the system will give a different priority to the different mechanisms having an ability to perform updates. The priority hierarchy used is the following (in descending order of priority):

1. Direct Editing of the model.
2. Feedback Update.
3. Initial Interview.
4. Default values from active stereotypes.
5. Renting.

Where a hierarchy level is equal (e.g., two direct editing or two feedback updates), the 'most recent' update is selected. Therefore, the user may modify, through a direct editing operation, one or more slots that have already been directly edited. Similarly, a feedback update may revise a slot justified by a previous feedback update. *Renting* is a modality for the updating of a slot's weight (see Section 4.4) whereby each topic *t* must necessarily pay a 'toll' to stay in the model, when *t* does not appear in the document evaluated by the user.

3.4. THE USER MODELING PROCESS

The machine learning approach used for the user modeling process will now be presented. It takes inspiration from Case-Based Reasoning (CBR), as applied for example in Help Desk Systems (Aamodt and Plaza, 1994), which consists in the use of *retrieve* and *adapt* as a core problem solving model. In our case, the retrieve and adapt phases are:

1. Stereotype(s) Assignment (*retrieve* phase).
2. User Model Refinement (*adapt* phase).

Figure 8 shows these phases, which are described in the next sub-sections.

3.4.1. *Stereotype(s) Assignment*

The current input (the *new problem* in CBR terms) is the 'user description', formed by a pattern of weights (each of which corresponds to a ⟨*domain*, *topic*⟩ pair). It

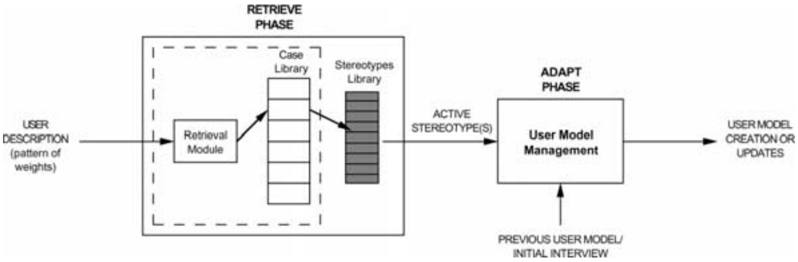


Figure 8. Stereotype assignment and user model refinement.

may be obtained by means of an initial interview or by updating an existing user model (consequently, it could result in a partial or inconsistent description). The case library contains the old cases provided by domain experts, which have the ⟨*old problem description*, *old solution*⟩ structure. The *old problem description* is a description of the user that must be modeled. The *old solution* should be (in principle) the complete user model of the user in question. However, since it is not feasible to determine a specific user model for every possible old case (a task that is too exacting for the domain expert), the solution part of our old cases is constituted in such a way as to indicate (i.e., to point at) one or more stereotypes, as shown in Figure 8. When a new user description is input, the old case that more closely matches the current one, according to a specific metric, is retrieved from the case library and the corresponding stereotype is assigned to the user. Basically, what is performed is a classification of the user, who is assigned a stereotype (or stereotypes) that most resembles the preliminary description of the user. On the basis of the above-mentioned description, the stereotype assignment technique used by the system qualifies as a *full resemblance method* (see (Shapira et al., 1997)).

This approach involves a substantial problem, i.e., the use of a valid metric for the retrieval of old cases. We solved the problem by means of a function-replacing hybrid (Goonatilake and Khebbal, 1995), where an artificial neural network implements (i.e., is functionally equivalent to) the components within the dash-lined box in

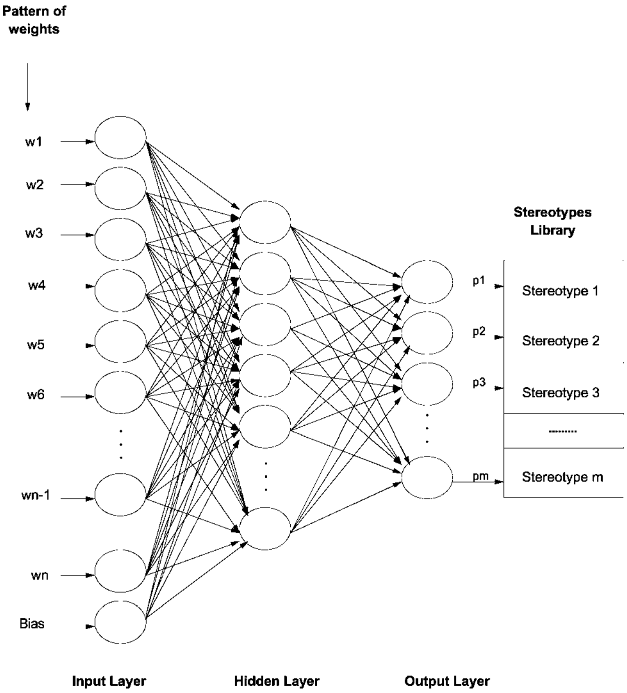


Figure 9. The artificial neural network used for stereotype assignment.

Figure 8. The old cases in the case library have been used as training records to train the network. The knowledge of the case library is therefore represented ‘monolithically’ in the network weights. As a result, the metric of the indexing module in Figure 8 has been replaced by the ‘generalization’ capability of the neural network (Haykin, 1994).

Thanks to this choice, the system can avail itself of the typical advantages offered by the use of artificial neural networks for pattern recognition, in particular to their fault tolerance attributes (Haykin, 1994). Since this kind of classification problem is generally not linearly separable, a Multi-Layer-Perceptron (Rumelhart and McClelland, 1986) with three distinct layers was used. Figure 9 shows the topology of the neural network we used. The first layer, the input layer, comprises the neurons relative to the weights w_i of the slots (99 in our case) which may be present in the various stereotypes (and therefore in the user model after the initial interview). We stress the fact that the same slot can appear in more than one stereotype, even if, in general, with different weights. The output layer comprises as many neurons as the number of stereotypes. The output values p_i are computed by the network according to a given input; this corresponds to the computation of a rank-ordered list of stereotypes contained in the library. The output values above a certain threshold identify the active stereotypes. The reader is referred to (Micarelli et al., 1998) for more details concerning the determination of the hidden layer, the training phase and the testing phase of the neural network.

3.4.2. *User Model Refinement*

After the stereotype (or stereotypes) is assigned to the user during the retrieval phase, the adaptation phase begins. The aim of this phase is to ‘refine’ the (where available) preceding version of the user model or the result of the initial interview (in the event of a non-registered user) by using the active stereotype and the user description given in input. Figure 10 shows this phase (and the entire user modeling process, taking into

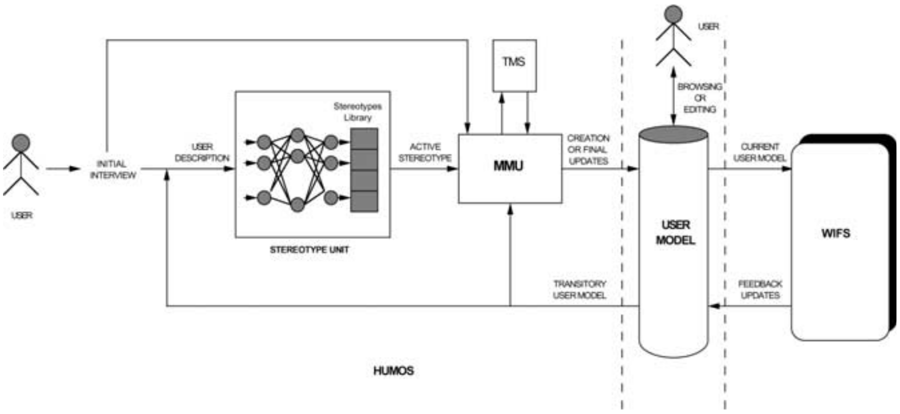


Figure 10. Functional diagram of the user modelling process.

account the project choices indicated in the preceding subsection). This figure is a refinement and completion of Figure 8. In particular, it emphasizes the two functionalities: (i) the Stereotype Unit, formed by the artificial neural network and the Stereotype Library, which implements the retrieval phase; (ii) the MMU (Model Management Unit) module, to which the TMS is linked, which handles the entire user model creation or update, thus essentially implementing the *adapt* phase.

For a better grasp of the process, let us suppose that a user accesses the system for the first time. As we well know, a preliminary interview is performed. The information obtained through the interview is input to the Stereotype Unit that identifies the active stereotype(s).[10] It is input to the MMU module that in turn, using the interview results, creates a first version of the user model by means of a stereotype overriding technique, meaning that it is formed by the active stereotype where the *weight* facets of the slots relating to the $\langle domain, topic \rangle$ pair, dealt with during the initial interview, acquire the values provided by the user during the interview. In this phase the 'Transitory User Model' is obviously empty. The user model thus created ('Current User Model') is input to WIFS so as to allow the adaptive filtering of the documents retrieved from the Web. As explained earlier, the user may, if she so wishes, give a rating in terms of usefulness (relevance feedback) of the selected documents. This rating is computed by WIFS, which in turn provides the 'Feedback Updates' that entail a modification of the user model. However, this updated model (Transitory User Model) is not yet final, the reason being that the feedback updates could have caused inconsistencies. Therefore it must be input to the Stereotype Unit and the MMU, as shown in Figure 10. The Stereotype Unit selects the active stereotype corresponding to the Transitory User Model (possibly confirming the previous active stereotype). The MMU takes into account the Transitory User Model and the selected active stereotype and it uses the TMS to perform the necessary updates ('Final Updates'), while respecting the priority order described in Section 3.3. In particular, if the new active stereotype is different from the previous one, it removes the slots justified by the old stereotype from the model and replaces them with those of the new active stereotype. Therefore, such updated model is final and newly ready for use. The same process is performed if the model updates are the result of direct editing by the user. Indeed, Figure 10 emphasizes the possibility for the user to browse or directly edit the model. However, it must be specified that these functionalities, as well as the model updates made on the basis of Feedback Updates, are performed by means of a call to HUMOS (as clearly illustrated in Figure 1), although this is not detailed in Figure 10 for sake of design simplicity.

4. The Information Filtering Component

This section focusses on WIFS (Web-oriented Information Filtering System).

[10] Hereinafter, for sake of simplicity, we shall assume that there is only one active stereotype.

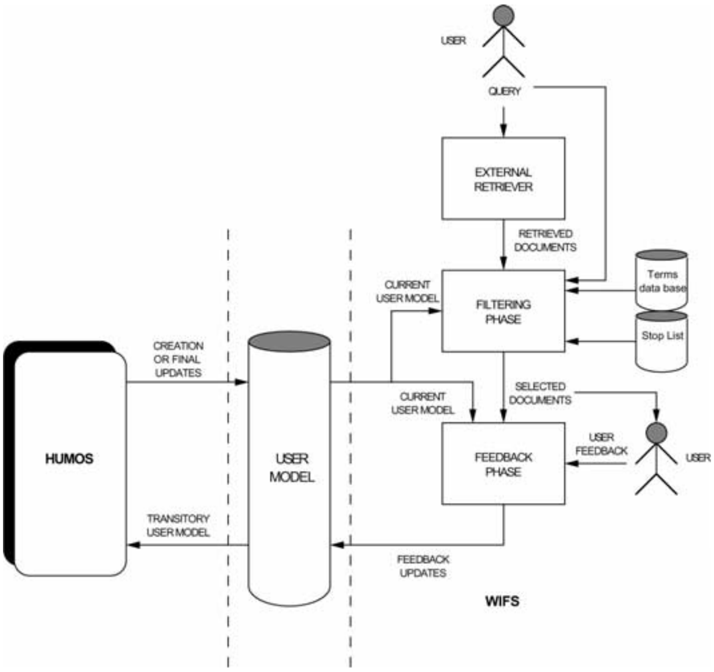


Figure 11. Functional diagram of the information filtering process.

Figure 11 describes (in the form of a functional diagram) the steps characterizing the information filtering process, informally presented in Section 2 in the example session. The next subsections provide a description of the main data structures and the filtering and feedback algorithms used by the system. In particular, Section 4.1 briefly describes the cleaning process of the document. Section 4.2 concerns the representation of the document, with reference to the various actors. Section 4.3 describes the filtering algorithm MAF (Matching Algorithm for Filtering), which processes the document and assigns it a final score. Section 4.4 details the algorithm SAF (Semantic net/DB-based Algorithm for Feedback) which produces as output the 'feedback updates' of the user model on the basis of the relevance feedback expressed by the user on the document. SAF and MAF take advantage of the user model structure provided by HUMOS, with particular reference to the co-occurrence relationships.

4.1. THE DOCUMENT CLEANING PROCESS

The retrieved document initially undergoes preprocessing by the system, relative to standard text operations (Baeza-Yates and Ribeiro-Neto, 1999), to reduce the complexity of the document representation. These operations involve essentially the deletion of the stop-list terms and the elimination of html tags, accents, punctuation marks, letter cases, etc.. We specifically chose not to use stemming for the remaining words (Harman and Candela, 1990; Frakes, 1992). This choice was not solely dictated by reasons of document processing efficiency, but also to improve the reading process which occurs when the user browses the user model content.

4.2. DOCUMENT REPRESENTATION AND EVALUATION

Information retrieval literature abounds with methods for representing documents and ranking algorithms (Baeza-Yates and Ribeiro-Neto, 1999). Many are based on the *Vector Space Model* (Salton and Lesk, 1968; Salton, 1971) which represents both the verbal contents of documents and the verbal contents of queries as vectors of fixed length, made up of weighted items (the words in the document or in the query). However this procedure does not appear suitable for information 'in the wild', that is, for the unpredictable verbal contents of documents found on the Web. It would be difficult to determine *a priori* the verbal items to be weighted in a vector when the semantic domain cannot be fixed in advance. Our working hypothesis is that, in such a case, a dynamic model is needed – one that can develop and modify itself from session to session.

In the system proposed here, both the representation of the documents and the evaluation of their contents are based on the method used for modeling the user. Given the list $\langle t_1, t_2, \ldots, t_n \rangle$ of the terms in the document *Doc* after the cleaning process, the system builds two vectors, $\overrightarrow{Occ}$ and $\overrightarrow{Rel}$. The element Occ$_i$ is the occurrence of a term t_i in the document, while the element Rel_i is the relevance of a term t_i with respect to the information needs of the user. The occurrence of a term is weighed in a different way depending on whether the term belongs to the Title or to the Body of the document. The relevance is computed taking into account the following data:

- The user model. The long term interests that characterize the specific user, represented by the set of slots $\langle domain, topic, weight \rangle$, together with the co-keywords linked to them, contribute to the final score by means of weights.
- The Query. The user formulates it by specifying terms of interest belonging to the domain of computer science. Essentially it allows the user to consider a subset of documents of the domain of interest.
- The Terms Data Base (TDB). It is a database of $\langle domain, topic \rangle$ pairs known to the system, which may appear in the slots of the user model and in those of the stereotypes. The TDB was built concurrently to the stereotype knowledge base through consultations with domain experts.

In the following section, we present in detail the ranking algorithm used for document representation and evaluation.

4.3. THE FILTERING ALGORITHM

The MAF algorithm computes a score for each retrieved document on the basis of the content of the user model, of the query, of the TDB and, obviously, of the document itself. First, this algorithm calculates the occurrence and the relevance of each term t in the document. Subsequently, it computes the global score adding up the contribution of every term. The constants that appear in the formulae used by MAF, shown in the next sections, have been tuned by means of informal trial-and-error experimental tests.

4.3.1. *Calculation of the Terms Occurrence*

The value of the occurrence of a term t in a document is obtained as follows:

$$OCC(t) = k_1 * freq_{body}(t) + k_2 * freq_{title}(t)$$

where $\mathrm{freq}_{\mathrm{body}}(t)$ is the frequency with which the term t appears in the body, while $\mathrm{freq}_{\mathrm{title}}(t)$ is the frequency with which the term t appears in the title of the document. The values for the constants are $k_1 = 1$ and $k_2 = 3$.

4.3.2. *Calculation of the Terms Relevance*

To calculate the relevance $Rel(t)$ of a term t in a document, the MAF algorithm considers the terms that are also present in the user model and, among these, it confers a particular emphasis on those present in the query. Another important contribution, furthermore, is given by the semantic networks which allow MAF to privilege situations wherein various terms co-occur in the document. The algorithm acts sequentially in four steps as illustrated in Table I that shows the formulae used to compute the relevance of a term t. The symbol $\in$ ($\notin$) means that, for the step in question, the algorithm evaluates whether the term t belongs (or not) to the corresponding component (UM, TDB, Query, Doc). Where this evaluation is not performed, the symbol '-'is displayed.

- *Step* 1. The algorithm verifies whether the term t belongs simultaneously to the user model (as a topic of a slot) and to the document. For every term t that satisfies this condition, the relevance $Rel_{new}(t)$ is calculated by the formula given in Table I. The value determined for the constant is $k_1 = 2$.
- *Step* 2. If the term t of the document belongs to the query and to the model (as a topic of a slot), then its new relevance as calculated in *Step 1* is significantly strengthened. In fact, it will be further multiplied by the weight of its proper slot w_{slot}.

Table I. MAF – Relevance calculation for each term of a document

Step	UM	TDB	Query	Doc	Update
1	$\in$	–	–	$\in$	$\mathrm{Rel}_{\mathrm{new}}(t) = \mathrm{Rel}_{\mathrm{old}}(t) + k_1 * \sum_j w_j \forall w_j : t \in \mathrm{slot}_j$
2	$\in$	–	$\in$	$\in$	$\mathrm{Rel}_{\mathrm{new}}(t) = \mathrm{Rel}_{\mathrm{old}}(t) * w_{\mathrm{slot}}$
3	$\notin$	$\in$	$\in$	$\in$	$\mathrm{Rel}_{\mathrm{new}}(t) = \mathrm{Rel}_{\mathrm{old}}(t) + k_2$
4	$\in$	–	–	$\in$	$\mathrm{Rel}_{\mathrm{new}}(t) = \mathrm{Rel}_{\mathrm{old}}(t) + w_j * \sum_i A_i \forall \mathrm{co} - \mathrm{keyword}_i$ of the $\mathrm{slot}_{\mathrm{j}}$: $co - keyword_i \in \mathrm{doc}; \forall \mathrm{slot}_j : \mathrm{topic}_j \in \mathrm{doc}$

- *Step* 3. The relevance of the term t of the document, which belongs to the TDB and to the query, but that is not present in the user model, is updated by adding a constant k_2 to it. The TDB contains terms of the area of interest of the user that could give a contribution even though they do not belong to the user model. Obviously such a contribution is of minor significance and therefore we decided on the sum. The value of the constant is fixed at $k_2 = 2$.
- *Step* 4. Finally, if the term t is a topic of a slot, the contribution of the semantic network associated with such a topic is also considered. The relevance of the term is thus updated with the contribution of the affinities A_i of its co-keywords that also belong to the document. This enables the recovery of the contribution furnished by the terms unknown to the system but nevertheless linked to the user model slots.

4.3.3. *Final Score*

Once the relevance $Rel(t)$ and the occurrence $Occ(t)$ of each term are calculated, a final score is assigned to the document Doc. This score is calculated as follows:

$$Score(Doc) = \overrightarrow{Occ} \cdot \overrightarrow{Rel} = \sum_{\forall t \in Doc} Occ(t) * Rel(t)$$

4.4. FEEDBACK ALGORITHM

The SAF algorithm updates the user model on the basis of the relevance feedback given by the user on a document. The renting mechanism, already mentioned in Section 3.3, is included in such operations. Such a mechanism decreases the weight of a slot when its topic does not belong to the document in question and decreases the affinity of a co-keyword when this too does not belong to the document. In this way it eliminates terms incidentally inserted into the model, and lessens the weight of terms rarely recurrent in the documents of interest for the user (Baclace, 1992). In the following we shall illustrate the steps of the SAF algorithm in detail. Even in this case the constants present in the formulae used by the algorithm have been tuned by means of trial-and-error experimental tests.

4.4.1. *Slots Update*

Table II sets out the criteria used by the algorithm to update the weights of already existing slots. The variable *feedback* is the value of the relevance feedback given by the user to the document. For each term t that is a topic of a slot in the user model the following steps are performed:

- *Step* 1. If the term t belongs both to the user model and to the document, the new weight of the slot w_{new} is proportional to the weighed difference between the user feedback and the absolute value of the old weight w_{old} of the slot. This process

Table II. Slots update

Step	UM	Query	Doc	Update
1	$\in$	–	$\in$	$w_{\text{new}} = w_{\text{old}} + \frac{k_1 * (\text{feedback} - \lvert w_{\text{old}} \rvert) * \text{Occ}_{\text{doc}}}{\text{Occ}_{\text{doc}} + 1}$
2	$\in$	$\in$	–	$w_{\text{new}} = w_{\text{old}} + k_2 * (\text{feedback} - \lvert w_{\text{old}} \rvert)$
3	$\in$	$\notin$	$\notin$	$w_{\text{new}} = w_{\text{old}} - k_3 * \lvert \text{feedback} \rvert$

shows that as soon as the number of occurrences increases, the w_{new} value stabilizes proportionally to the difference. The determined value of the constant is $k_1 = 1.3$. The difference between feedback and absolute value of the slot signifies, for that slot, the difference between the relevance of the term for the user and for the model.

- *Step* 2. If t belongs to both the user model and the query, the new weight of the slot, w_{new}, is proportional to the weighed difference between the user feedback and the absolute value of the old weight, w_{old}, of the slot. So if the term t belongs also to the document, the algorithm strengthens further its weight w_{old} after the step 1. The determined value of the constant is $k_2 = 1$.
- *Step* 3. If t belongs only to the user model, it is weakened by the *renting* mechanism, i.e., by reducing its weight proportionally to the user feedback. If the user gives a very positive feedback and t is not present in that document, it looses considerable importance in the model. The value of the constant is $k_3 = 0.11$.

All the slots of the user model are thus updated. It should be pointed out that, once each slot is updated, the system checks the new value: if it does not fall within an expected range, the slot itself is deleted from the user model together with all its co-keywords.

4.4.2. *Co-keywords Update*

Table III illustrates the updating rules for the co-keywords affinity A. The symbols $\in$ and $\notin$ refer to the topic of the slot to which the co-keyword is linked.

The algorithm processes the model as follows:

- *Step* 1. This step concerns the affinity update of those co-keywords whose slots have their topic in the document. If the co-keyword also belongs to the document, its occurrence *Occ* is computed using the same rules for the computation of

Table III. Co-keywords update

Step	UM	Query	Doc	Update
1	$\in$	–	$\in$	$A_{\text{new}} = A_{\text{old}} + \frac{k * \text{feedback}}{\lvert \text{Occ}_{\text{doc}} - \text{Occ}_{\text{ref}} \rvert + 1} \forall_{\text{co-keyword}} \in \text{doc}$ $A_{\text{new}} = A_{\text{old}} - k * \lvert \text{feedback} \rvert \forall_{\text{co-keyword}} \notin \text{doc}$
2	$\in$	$\in$	$\in$	$A_{\text{new}} = A_{\text{old}} + k * \lvert \text{feedback} \rvert \forall_{\text{co-keyword}} \in \text{doc}$ $A_{\text{new}} = A_{\text{old}} - k * \lvert \text{feedback} \rvert \forall_{\text{co-keyword}} \notin \text{doc}$

the occurrence of the terms. The affinity value is modified by a quantity that is proportional to the user feedback and inversely proportional to the difference between the calculated occurrences and a given threshold Occ_{ref}. The value used for the constant is $k = 1$. The quantity Occ_{ref} is the number of occurrences of the respective topic in the document. If the co-keyword appears in the document with a frequency close to the one of its slot, it can be assumed that the two terms are strongly correlated. In this case, a maximum increase of the affinity must be given. If the co-keyword does not belong to the document, then A_{new} is calculated by subtracting an amount proportional to the feedback expressed by the user from the old value. In this way the renting mechanism weakens the link between co-keyword and slot given that they do not appear in the same document. If the new value does not remain within a pre-established range, the co-keyword will be deleted from the model. The determined value of the constant is $k = 0.15$.

- *Step* 2. If the topic of a slot *t* belongs to the query as well as the document, then the new affinity of its co-keywords is either increased or decreased, depending on whether or not the co-keyword belongs to the document. The constant is $k = 0.6$.

4.4.3. *Insertion of New Slots*

In the following, we describe the rules governing the insertion of a new slot in the user model.

The steps for the calculation of new weights are the following:

- *Step* 1. Each term *t* of the document that is also a topic in the TDB is inserted as a new slot. The associated weight w_{new} is proportional to the user feedback and weighed according to its occurrence in the document itself. The value of the constant used is $k_1 = 1$. As the number of occurrences increases, the weight gets closer to the one given by the user.
- *Step* 2. Each term *t* of the query that is also a topic in the TDB is inserted as a new slot. The new weight w_{new} is computed proportionally to the user feedback. The constant is $k_2 = 0.7$.
- *Step* 3. If a term *t* of the document is present in the query but not in the model nor in the TDB (hence it is unknown to the system), a new slot is added to the model. The fields of that slot are set as follows: *domain* = *'filler'*; *topic* = *t*; *weight* =

Table IV. Calculation of the weights of new slots

Step	UM	TDB	Query	Doc	New entry weight
1	$\notin$	$\in$	–	$\in$	$w_{\text{new}} = \dfrac{k_1 * \text{feedback} * \text{Occ}_{\text{doc}}}{\text{Occ}_{\text{doc}} + 1}$
2	$\notin$	$\in$	$\in$	$\in$	$w_{\text{new}} = k_2 * \text{feedback}$
3	$\notin$	$\notin$	$\in$	$\in$	$w_{\text{new}} = k_3 * \text{feedback}$

Table V. Calculation of affinities of new co-keywords

Step	TDB	Query	Doc	Affinity
1	$\notin$	$\notin$	$\in$	$A_{\text{new}} = \dfrac{k_1 * \lvert\text{feedback}\rvert}{\lvert\text{Occ}_{\text{doc}} - \text{Occ}_{\text{ck}}\rvert + 1}$

the computed weight w_{new}. The constant is $k_3 = 0.43$. In all of the above cases, if the new weight does not exceed a given threshold it will not enter the user model.

4.4.4. *Insertion of New Co-Keywords*

In this subsection we illustrate the insertion mechanism for new co-keywords in the user model.

As can be seen from Table V, the algorithm acts in one step:

- *Step* 1. All the terms that belong to the document (but not to the query), and are unknown to the system, are inserted as co-keywords in all the slots whose topics are in the document. The formula in Table V takes into account a constant k_1 that is the constant of proportionality to the user feedback (in our case the tuning gave $k_1 = 1$). The variable Occ_{doc} is the occurrence of the topic of the slot in the document, while the variable Occ_{ck} is the occurrence of the co-keyword in the document.

The analysis given in this Section 4.4, shows that the user model can evolve dynamically both in respect to its co-keywords and its slots.

Empirical Evaluation

Research into user modeling and user-adapted interaction is essentially of an empirical experimental nature (Chin, 2001). For this reason, such research requires the use of precise statistical analysis techniques, new methodological approaches and the development of increasingly refined experimental designs[11]. We performed an evaluation of our system through real-time access to the Web. This evaluation was based on the following two points:

1. Hypothesis Testing for the evaluation of the added value to the system (in terms of performance measured according to metrics proposed in the literature) offered by the user modeling component.
2. Study of the usability of the system, performed through the analysis of the responses to a questionnaire submitted to users and of the data collected from system log files.

These points are illustrated in the following subsections.

[11] To help project planners and developers exchange views on methods and evaluation of adaptive systems, a database is currently available at http://art7.ph-freiburg.de/easy-d, *EASy-D* (Weibelzahl and Weber, 2001), containing structural data on a set of adaptive systems. In (Jameson, 1999) Jameson gives a classification of the systems present online at *EASy-D*.

5.1. ADDED VALUE OF THE USER MODEL

The experiment was organized into the following phases, which are typical of quantitative experimental research methods: formulation of the research question, choice of the statistical model, formulation of the statistical question, formation of the user sample, experimentation and collection of statistical data, application of the statistical method, statistical conclusion, research conclusion. Here below is a detailed description of each of these phases.

5.1.1 *Research Question*

It is important when evaluating adaptive systems to assess whether the system works better (and to what extent) with the user modeling component as opposed to a system deprived of this component (Chin, 2001).

This is our exact research question:

'Is there an improvement in the performance of WIFS operating with the user modeling module HUMOS, compared to its operation without this adaptive component; and if so to what extent?'.

As concerns the criteria used for performance evaluation, we considered the sorting of retrieved documents. We chose the metric defined in (Kemeney and Snell, 1962) and reported in (Yao, 1995) in the *Perfect Ranking* hypothesis. The same evaluations were made subsequently using the above-mentioned metric defined in the *Acceptable Ranking* hypothesis (Yao, 1995). We also computed and graphically displayed the *11 pt. average Precision–Recall* curve (Baeza-Yates and Ribeiro-Neto, 1999). Consequently, given a set D of documents provided by ALTAVISTA, in response to a query q, our observation focused on the correlation among the orders of the set D proposed, respectively, by ALTAVISTA, by WIFS and by the user who input the query. Given a set of documents D, let $\Gamma(D)$ be the set of all orders $\succ$ defined on D. The distance function we use is a real-valued function defined as β: $\Gamma(D) \times \Gamma(D) \rightarrow R$ (Yao, 1995). In the case of *Perfect Ranking*, which computes the distance between the order proposed by the system and that proposed by the user, the best situation sought is the one where: $\succ_S \equiv \succ_U$ and $\beta(\succ_S, \succ_U) = 0$; in other words, the order proposed by the system ($\succ_S$) and that proposed by the user ($\succ_U$) are perfectly equal. In this case, the distance is calculated by taking into account all the possible orders of D. Instead, the *Acceptable Ranking* is based on the consideration that very often a retrieval system is only required to place the more relevant documents in a higher position than the not-relevant ones (Wong et al., 1988; Wong and Yao, 1990). In this case, the distance does not take into account the individual positions of the documents in the list but instead considers the equivalence categories in which the list itself is sorted. Accordingly, the first method obviously gives more restrictive results than the second method in terms of distance calculation. In fact, the latter gives $\beta(\succ_S, \succ_U) = 0$ for all the distances between the acceptable orders and not only for the ones having the exact same documents listed in the same place. In both cases, the distance performance measure β is normalized to the maximum possible distance, obtaining the variable *ndpm* (normalized distance-based performance measure) (Yao, 1995). The phases described below refer to the *Perfect Ranking* case.

5.1.2. *The Choice of the Statistical Model*

We chose hypothesis testing as the statistical method for our experiment. We believe that, for the type of experiment under review, no strong assumption can be made as, for example, normality, about the nature and the shape of the probability distribution of the random variables involved in the experiment. Essentially, we find ourselves in a situation where we have a minimal knowledge regarding the probability distribution of the random variable to be studied. We therefore decided to use non-parametric statistics techniques and methods and we chose a method for the hypothesis testing that is not based on the assumption of normality. The consequence is that the statistical test for our experiment must be chosen according to the following criteria:

- It needs to be non-parametric as not to force strong assumptions (e.g., normality) about the distribution of the involved populations.
- It must be able to perform the Hypothesis Testing on coupled measures (we must compare the performance of WIFS vs. ALTA VISTA).
- It must be applicable to small samples, such as groups of scores associated to the single orders in the current investigation.

We believe that the *Wilcoxon-Signed-Rank test* (Wilcoxon, 1947) is the most appropriate test for such requirements.

5.1.3. *The Statistical Questions*

Given the following random variables:

- $X =$ distance between the order proposed by WIFS and the one proposed by the user on a set of documents D, produced in response to a query q.
- $Y =$ distance between the order proposed by ALTA VISTA and the one proposed by the user on a set of documents D, produced in response to a query q.

we verified whether the statistical distributions that involve the variables X and Y, to which the populations yielding the samples belong, were different. On the basis of the above considerations regarding the choice of the statistical model, we formulated the following *Statistical Questions*:

- *Null Hypothesis* H_0: The differences observed in the measures of the statistical parameters for the two samples x_i and y_i are due to chance and the two variables X and Y belong to the same statistical population: $\Phi_X \equiv \Phi_Y$, being Φ_X and Φ_Y the two statistical functions describing the population distributions to which the samples x_i and y_i respectively belong (same functional form). In particular:

$$\mu_X = \mu_Y \tag{1}$$

- *Alternative Hypothesis* H_1: The population yielding the sample x_i, is different from the population yielding the sample y_i: $\Phi_X \neq \Phi_Y$ and also:

$$\mu_Y = \mu_X + \Delta \tag{2}$$

where $\Delta > 0$. As for the significance level α, we chose $\alpha = 0.01$ (which means a 1% probability of rejecting the null hypothesis in case it turns out to be true) in order to have a very strong corroboration of our experiment's results.[12]

5.1.4. *The Statistical Sample*

The sample of users who experimented the system was selected from the population of computer science and electronic engineering students of our university. A total of 24 users were selected using sample statistics procedures.

5.1.5. *Experimentation and Data Collection*

The experiment was prepared in such a way as to avoid, insofar as possible, the introduction of disturbance variables (Chin, 2001) by adopting some precautionary measures as, for example, giving each user the right amount of time for the experiment, setting up a comfortable working environment, assigning casual time slots to users and optimizing the network to avoid long delays on the Web.

Each user performed 15 working sessions. In each of these sessions, the user formulated a query that resulted in the retrieval of 30 documents and subsequently ordered them.[13] Therefore, a total of 360 queries were effected and 10,800 documents analyzed. The users were asked, for each session, to provide the relevance feedback to a single document among those retrieved in answer to the query. The users also completed a questionnaire during and after the use of the system, which we analyzed as described in Section 5.2.

Let D_i be the set of 30 documents retrieved by ALTA VISTA following a query q_i input by the user. For each query, the user ordered the set of retrieved documents D_i according to their relevance. Therefore, for each query, we have three orders of the set D_i: the one proposed by the user, the one proposed by ALTA VISTA and the one proposed by WIFS. From the two variables X and Y, a new statistical variable Z is built, defined as follows:

$$Z \equiv Y - X$$

[12] Many statisticians consider these α values as arbitrary. They therefore believe that researchers should merely synthesize data, reporting the type of test used and the value of the obtained p-value, instead of making comparisons with the above-mentioned threshold values. In our case, in addition to the test, we report also the values obtained for the p-values that are in any event compared with the selected α threshold.

[13] It must be specified that for each query the users furnished the order of the documents retrieved by ALTAVISTA before the processing performed by WIFS, in other words, before knowing the order of the documents proposed by our system. In this way we ensured the independence of the two statistical variables X and Y.

Table VI. Wilcoxon test results (Perfect ranking)

Statistics	Value
T^+	51125
μ_{T^+}	42127.5
σ_{T^+}	2149
p-value	0.00005

5.1.6. *Application of the Statistical Method*

In the Wilcoxon test, the statistical variable T^+ is defined as the reference variable of the test (Wilcoxon, 1947). For a sample with our cardinality, the estimator T^+ is normally distributed, thus enabling the use of the *Large-Sample Approximation* (Siegel and Castellan, 1988). With such an approximation, valid for samples in the amount of $n > 15$, the mean and the variance of the T^+ distribution are easily determined through the standardization of the variable T^+ and the subsequent reading of the p-value in the Gauss distribution table (Siegel and Castellan, 1988). Table VI shows the results of the test.

At this point we can calculate numerically the value of Δ of Equation 2, using an inferential estimation method of the confidence intervals of the mean. The starting point of our evaluation is the one synthesized in Table VII which summarizes the statistical parameters that are characteristic of the sample. As shown in Table VII, the values of the mean and of the median almost coincide, showing symmetry in the distributions. The results of the calculation are the following:

$$\mu_X \sim 0.40$$
$$\mu_Y \sim 0.57$$

from which we obtain:

$$\mu_Y \sim \mu_X + 0.17$$

The computed value for Δ is therefore $\Delta = 0.17$.

5.1.7. *Statistical Conclusions*

The statistical results obtained by means of the Wilcoxon test are summarized in Table VI, where we have p-value $\ll \alpha$ for our tested hypothesis. Consequently, we must reject the null hypothesis H_0 and accept the H_1 hypothesis. This means that the samples

Table VII. Statistical parameters of the samples

Variable	Mean μ	Median θ	Standard Deviation s
X	0.40	0.39	0.06
Y	0.57	0.55	0.09

of the order distances between the user and the system WIFS, x_i, and between the user and ALTA VISTA, y_i, originate from two different distributions Φ_X and Φ_Y. Moreover, we have $\mu_y > \mu_x$. This means that the two modalities are significantly different. The estimated value of Δ corresponds to an added value of about 30%.

Figure 12 shows the trend of the average distance computed on all users for the 15 sessions in the *Perfect Ranking* hypothesis. The following points may be emphasized from the graph:

- The curve of the variable X, 'User-WIFS' distance, with the exception of its initial phase, always stays beneath the curve Y, 'User-ALTA VISTA' distance.
- The trend of the curve Y, relative to ALTA VISTA, assumes an essentially constant value. The curve X, represented by the inferior linear interpolation, instead shows a downward trend.

Figure 13 illustrates the results for *Acceptable Ranking* that we obtained by means of a computation procedure similar to the previous one (which we shall not detail here for sake of brevity).

All the qualitative considerations on the curve relating to Perfect Ranking are valid. Obviously, a decrease of both curves was expected due to the difference in the computation performed on the variable *ndpm*. The statistical procedure is the same and again includes the Wilcoxon test to coupled measures. The statistical results are highlighted in Table VIII.

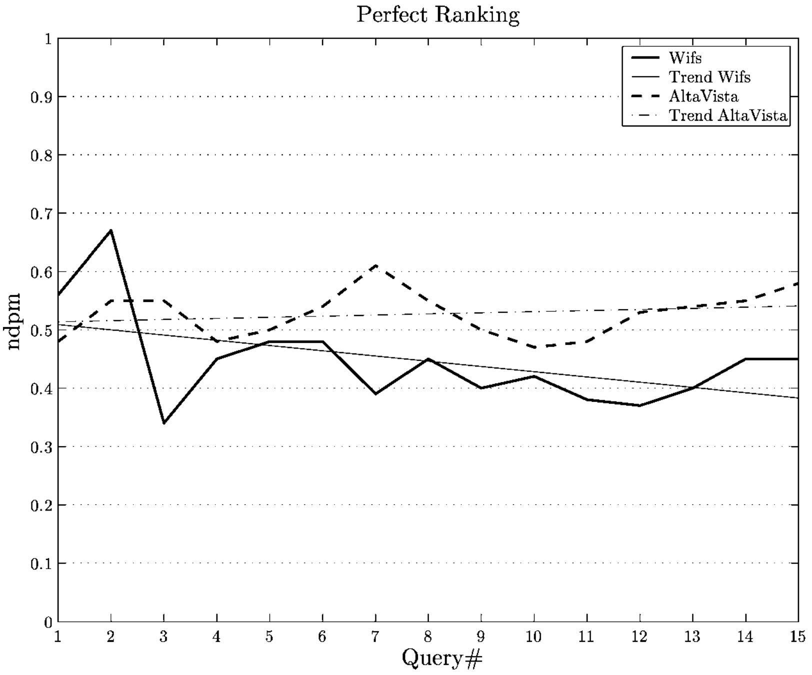


Figure 12. Ranking mean distance (Perfect Ranking).

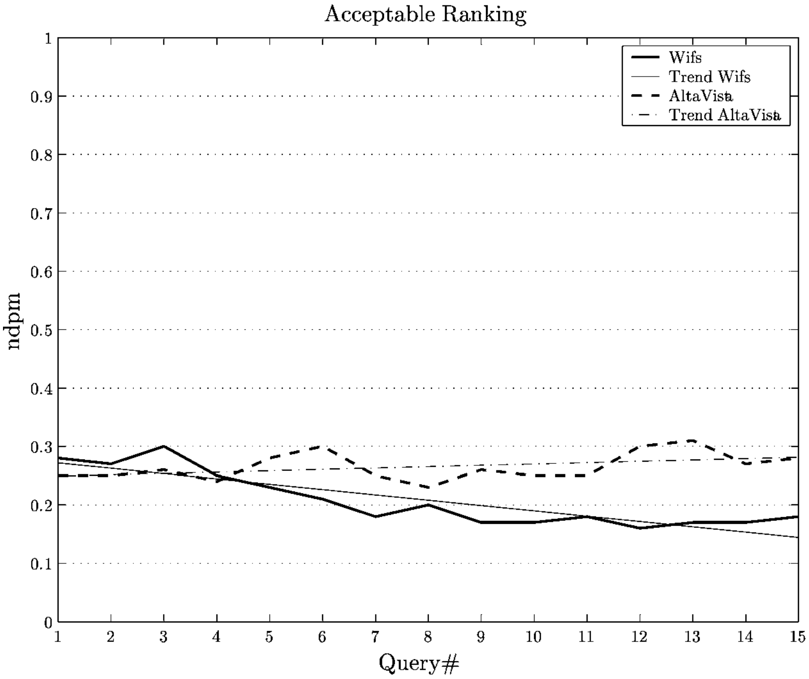


Figure 13. Ranking mean distance (Acceptable Ranking).

The null hypothesis H_0 must be rejected and the alternative hypothesis H_1 accepted and thus Equation 2 holds. Using the same procedure we estimated the added value Δ with the usual confidence interval technique. After similar calculations to the estimations for the perfect ranking case, we obtained:

$$\mu_Y = \mu_X + 0.21$$

This means that, according to this measure, the added value Δ is about 34%.

The results obtained in the analysis of two classic variables of the Information Retrieval field: *Precision* (*P* hereinafter) and *Recall* (*R* hereinafter) (Baeza-Yates and Ribeiro-Neto, 1999) offer a further confirmation of the results given by the Wilcoxon test.[14] In fact, if as demonstrated, WIFS produces a better document order than ALTA VISTA, this should also be reflected in the two quantities *P* and *R*. To this end, we used the *11 pt. average P–R* curve method (Baeza-Yates and Ribeiro-Neto, 1999). In practice, for each group of documents retrieved by ALTA VISTA in response to a query, the user's score creates a division of the group into two sets: relevant and not relevant documents. Subsequently, the orders are analyzed and the two systems are compared in the curve *R* vs. *P*. The interpolated average curve *P–R* is the one illustrated in Figure 14.

[14] The Recall and Precision variables are defined as follows: Recall is the fraction of the relevant documents which has been retrieved and Precision is the fraction of the retrieved documents which is relevant (Baeza-Yates and Ribeiro-Neto, 1999).

Table VIII. Wilcoxon test results (Acceptable ranking)

Statistics	Value
T^{+}	62211
$\mu_{T^{+}}$	52234.6
$\sigma_{T^{+}}$	2000
p-value	0.00003

As thus demonstrated, the ability to order the retrieved documents according to the relevance given by the user's scores is best in WIFS: in respect to both the high values and low values of recall, the precision is always higher. In particular, this means that WIFS considers a higher number of relevant documents in the first positions for the purpose of its orders.

5.1.8. *Research Conclusions*

The statistical conclusions indicate that the average performances of the two operational modalities are different. To proceed with the research conclusions we must understand why these differences occur. However, the setup of the experiment (system with user model and system without user model) provides the evidence for the conclusion that the difference in performance is due to the presence of the user modeling module. The module produces an increase in system performance, as

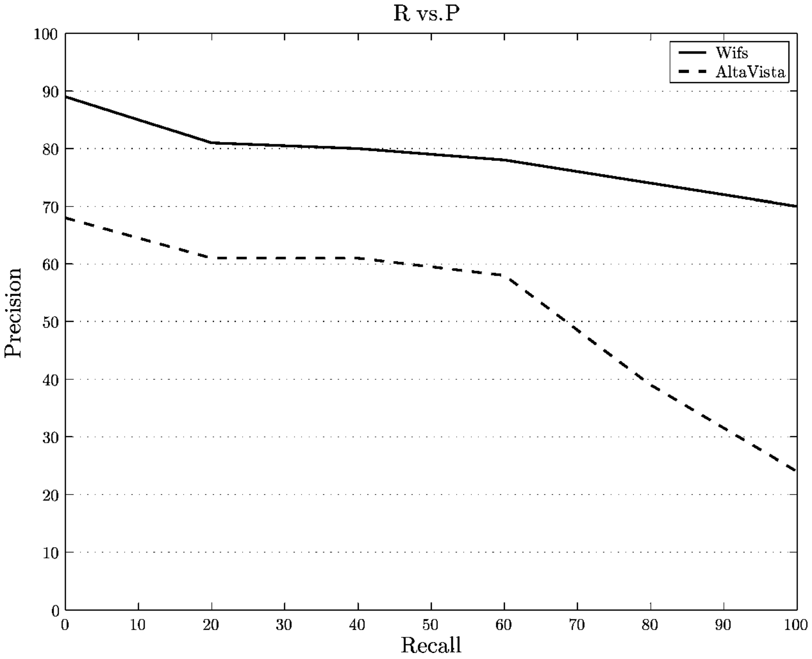


Figure 14. 11 pt. Average precision–recall curve.

assessed in the previous section. This is the answer to our initial research question. As concerns the trend of performance in time, reported in Figures 12 and 13, we can make the following observations. At the beginning of the session there is no appreciable help given by the user modeling system. This could be due to the users' very limited use of the initial interview. In the course of the sessions, however, there is a steady improvement of the system performance clearly due to the relevance feedback (one per session), which increasingly refines the model.

Having formulated the above-mentioned conclusions about causality, we must now consider the problem of generalizing results. Strictly speaking, the results of an experiment apply only to the specific circumstances in which the study is developed. Therefore, in our case, the results may only be applied to the category of users involved. Nevertheless, we believe that this category, essentially involving students of technical-scientific matters, is significant.

5.2. USER ACCEPTANCE AND USABILITY OF THE SYSTEM

This section illustrates a first descriptive measurement of the level of user acceptance and usability[15] of the WIFS system. The guidelines we used in formulating this short study are those indicated by (Shneiderman, 1998) as 'The Eight Golden Rules', relating to interactive systems projects.

Numerous metrics and evaluation techniques exist as criteria to test the usability of a system. Among these possible evaluation techniques (*Videoing, Think Aloud, Questionnaire*, etc.),[16] we chose the questionnaire method, because it offers both quality and quantity indications, it has a low implementation cost and is appropriate for our sample cardinality. As concerns the choice of metrics, an interesting set of selection criteria is the one illustrated in (Dix et al., 1993; Shneiderman, 1998). Our questionnaire included metrics taken from QUIS (Questionnaire for User Interaction Satisfaction) (Harper and Norman, 1998), designed to assess user's subjective satisfaction with specific aspects of the human-computer interface. The QUIS team successfully addressed these problems, creating a metric that is highly reliable across many types of interfaces to measure user satisfaction for a system in a valid, standard and reliable way (see also http://www.lap.umd.edu/QUIS/index.html).

5.2.1. *The Questionnaire*

The questionnaire was proposed taking into account the above considerations.

It is subdivided into parts that we deemed, among those set forth in QUIS, the most applicable to our study, i.e.: System Experience, Overall User Reactions, Screen and Usability. Each of the specific factors of interface and optional sections provides a main question followed by correlated sub-component questions. Furthermore, each

[15] Following the meaning expressed in http://www.dcs.napier.ac.uk/marble/Usability.

[16] See, for example, http://www.dcs.napier.ac.uk/marble/Usability/Table.html.

Table IX. Part 1: How long have you worked on this system?

Item	% of users
less than 1 h	0
1 h to less than 1 day	0
1 day to less than 1 week	5
1 week to less than 1 month	5
1 month to less than 6 months	90
6 months to less than 1 yr	0
1 yr to less than 2 yr	0
2 yr to less than 3 yr	0
3 yr or more	0

item was selected using a scale from 1 to 9, with positive adjectives on the right and negative ones on the left. Finally, the user was given the possibility of selecting a *Not Applicable* option for each item.

- *System Experience*: The first section contains generic questions on the use of the system, as for example, 'How long have you worked on this system?' The distribution of the data relating to the answers are illustrated in Table IX and in Table X.
- *Overall User Reactions*: The second section contains questions on 'Overall User Reactions'. The aim of this section is to have certain general indications on the user-system relationship. These data are reported in Table XI.
- *Screen*: The third section contains the part 'screen', i.e., the typical questions on user interface. Table XII summarizes the results.

Table X. Part 1: On average, how much time do you spend per week on this system?

Item	% of users
less than 1 h	4
one to less than 4 h	8
4 to less than 10 h	84
over than 10 h	4

Table XI. Part 2: Overall user reactions

Item	Mean	Variance
Terrible–wonderful	7.5	0.1
Frustrating–satisfying	6.2	0.3
Dull–stimulating	3.6	0.3
Difficult–easy	6.0	0.3
Inadequate–adequate power	7.0	0.3
Rigid–flexible	5.0	0.5

Table XII. Part 3: Screen

Item	Mean	Variance
Hard to read-easy to read	6.7	0.25
Highlighting on the screen	6.0	0.03
Screen layouts were helpful	5.8	0.3
Sequence of screen	6.5	0.01

- *Usability*: The forth section proposes questions both on the general performance of the system and on the feedback modalities available to the user. The experimental data are those summarized in Table XIII. On average the times were calculated by 'session', understood as the time required to formulate the query and the retrieval of the documents from the Web (without considering the time spent by the user in browsing for selected documents).

5.2.2. *Results Analysis*

The questionnaires indicate that the users' assessments were generally positive. In particular, the system is given as reliable and sound. Table XI emphasizes some significant data on the user–system relationship: the system is shown as adequate for processing power, efficient in terms of speed of response and relatively simple to use, while, instead, requiring a better interface for flexibility. It shows weakness both in terms of on-line help and of its capability to be stimulating for the user. As for the screen, the characters selected for the interface are positively accepted by the users and likewise, so are the screen sequences and the layouts. Useful indications are given regarding the method proposed by the system to express relevance feedback. On average, the users did not take advantage of all the scale $[-10, +10]$ to express a rating on the document: in most cases, they preferred to give a negative score for not relevant documents, a 0 score for neutral documents and a very positive score for relevant documents. This suggests that improved results would be obtained by providing only three levels (relevant, not relevant, neutral) for the feedback formulation.

Table XIII. Part 4: Usability

Item	Value
Time to complete a session per user	$\sim$ 3 min
Frequency of help and documentation used per session and per user	6.7
Frequency of repetitions of failed commands per session and per user	1.2
Number of available commands not invoked per session and per user	5.3
Number of times the user loses control of the system per session and per user	0.4
Number of times the user expresses satisfaction per session	6.9
Percent of use of the whole feedback range $[-10, +10]$ per user	6.1

6. Related Work

Systems proposed in the literature for adaptive searches on the Web are numerous (Hanani et al., 2001; Mladenic, 1999; Oard, 1997). A first noteworthy one is Web Watcher (Armstrong et al., 1995; Joachims et al., 1997), a tour guide software agent, developed to assist user navigation on the Web. WebWatcher follows the users between the html pages, offering suggestions on hyperlinks and learning from experience. Furthermore, the user can communicate with the system and provide feedback. The system therefore filters the information gathered on the Web, in particular the current html page, the interests of the user and the links of the page. The higher scoring links of the current page are the ones proposed by the user. There is a learning model based on user feedback at the end of the tour. Thus, the system possesses a dynamic knowledge model that contains paths, hyperlinks, keywords, selected during the tour. WebWatcher uses a vector space model for both documents and the user model. Unlike WIFS, that operates throughout the Web sites indexed by ALTA VISTA, WebWatcher operates within a clearly delimited domain, constituted by the CMU School of Computer Science Web pages. This permits it a sort of preprocessing of the collection of documents.

Syskill and Webert (Pazzani and Billsus, 1997) is an intelligent agent designed to learn user profiles and to determine interesting Web sites according to the user's interests. The system builds user profiles by collecting user evaluations of pages on the Web. An information-based approach is used to determine which words to use as features. The system learns a naive Bayesian classifier to determine the level of interest of pages. The authors have also developed an interesting experiment that shows the advantages of using and consulting a thesaurus for the selection of features.

Both of the previous systems devise their user model without considering the eventual co-occurrence of terms. This aspect, however, is indeed dealt with in ifWeb (Asnicar et al., 1997), a prototype of a user model-based intelligent agent capable of supporting the user in the navigation on the World Wide Web. Such a system, like WIFS, avails itself of the co-occurrences of terms, and represents the user model in the form of a sole weighted semantic network whose nodes correspond to terms found in documents. The weights of the arcs of the network represent the frequency of co-occurrences of the connected terms in previously analyzed documents. ifWeb does not avail itself of stereotypes for user modeling nor addresses the non-monotonic aspects of the user reasoning. An important difference between WIFS and ifWeb is that, for WIFS, the mode of Web access is of a 'parasite' type, while ifWeb performs a surfing navigation on sites related to the specific document pointed out by the user.

Other systems build their semantic networks utilizing the WORDNET (Fellbaum, 1998) semantic thesaurus. One of those is INFOS (Mock and Vemuri, 1997). The system applies keyword hill climbing methods, collaborative filtering, knowledge-based conceptual representation via WORDNET and partial parsing via index patterns. INFOS uses a quite simple user modeling method, based on user feedback. The experiments of the system yielded interesting comparisons between the document representation models. The authors report that the use of WORDNET supports higher recall through conceptual understanding of the text, but the precision is lower. Another of these systems is SiteIF (Magnini and Strapparava, 2004) that makes use of MULTIWORDNET) (Artale et al., 1997), a multilingual lexical database where English and Italian senses are aligned. SiteIF builds a user model in the form of a semantic network whose nodes represent senses (not just words) of the documents. The system takes advantage of the word senses to retrieve new documents with high semantic relevance for the user.

Fab (Balabanović, and Shoham, 1997; Balabanović, 1998), is an interesting system that searches documents in the Net through a central server. It uses the vector space model for the representation of documents. It does not deal with consistency maintenance of the model or with co-occurrence of terms. However, it avails itself of a *collaborative* filtering method that operates in synergy with a *content-based* one. As a result, whenever the collaborative approach is applicable, it offers the advantages of both methods.

Another system which, like WIFS, filters computer and information science documents, is SIFTER (Mostafa et al., 1997). This system, that does not act on the Web, uses the vector space model for representing documents. The algorithm used to construct the user model is based on a 'reinforcement learning' approach proposed in the literature (Narendra and Thathachar, 1989). SIFTER does not address the non-monotonic aspects in the user reasoning nor the term co-occurrences. The system was evaluated using the normalized recall and the normalized precision. It has been tested by six users on a static collection of 6000 records. Users were requested to run the system for 40 sessions. For each session, the system presented users with 20 documents. Simulation experiments have also been performed. The authors report that the performance of the system is satisfactory when the users are reasonably familiar with the domain of information.

As far as the use of stereotypes is concerned, an interesting proposal is advanced in (Shapira et al., 1999), which describes a prototype of a filtering and sorting system for the analysis of e-mails belonging to a group of computer scientists. The stereotypes are formed by way of clustering techniques applied to the group of users being interviewed. From these useful information may be inferred about the user for document filtering purposes. Such a clustering mechanism, along with the type of representation chosen, is suited to the considered domain relative to e-mails. We contend, however, that such an approach is not easily adaptable to systems which operate throughout the Web.

A distinctive characteristic of WIFS, which is not present in any of the other described systems, is the direct use of the query by the user. This additional element influences both the human-machine interaction and the matching and feedback mechanisms. Indeed, while systems such as Fab foresee a subsystem interfacing to external search engines that is transparent to the user and plays a minor role compared to other parts of the system, in WIFS the user query insertion represents an essential part of the ranking algorithm that influences its most important functionalities.

In the field of adaptive IR interactive systems, we find the Lighthouse system (Leuski and Allan, 2004). Lighthouse is a system that integrates three new models of user relevance with the aim of helping users to quickly locate the relevant documents among those retrieved by a Web-based search engine. The crucial point of the work performed by Leuski and Allan is the demonstration of the added value given by the interactive environment of the proposed IR system through a modeling relevance that is based, in all three of the proposed models, on the relevant/non-relevant value input by the user. The user interests are exclusively based on the retrieved documents and are represented by the status of the document set that changes over time following user feedback. Unlike WIFS, the Lighthouse system does not construct an initial user model based on interviews and therefore on stereotypes: the knowledge base is obtained from the dynamic document set updated through feedback. From a perspective of domain-to-domain portability, the flexibility of this system is an advantage. Instead, WIFS requires the construction of a more refined knowledge base of the domain and the user, based on stereotypes and initial interviews. In our view, however, this element makes it more efficient on specific domains on a regular working basis. Moreover, the representation of the user interests in WIFS is based on a more well-constructed structure of dynamic data, which also takes into account the terms unknown to the system that co-occur. Another adaptive IR system is InfoWeb (Gentili et al., 2003). It is designed to retrieve documents of interest to users from a digital library on *Cultural Heritage* domain, available on the Net. This system uses a typical bottom-up approach in order to construct the user model: first of all the user provides a relevance value through a specific user interface environment on *n* documents that represent the centroids of *n* clusters, which follow a subdivision of the library as determined by the domain expert. Even in this case, the user model is represented through a semantic network that evolves in time, whereas the relevance of a document is obtained by the system through a generalization of the classic vector space model. Thus, unlike WIFS, this system constructs the initial model using the most representative documents of the digital library and not through an interview on the user's explicit interests. Moreover, the calculation algorithms for the relevance of documents in the ranked lists produced by a specific query are based, as mentioned above, on the generalization of the vector space model, which takes into account the user model and the query. InfoWeb was designed for digital libraries, i.e., for environments where the information sources do not vary significantly over time and that are restricted to specific domains, as reflected in the clustering of documents. On the contrary, WIFS was designed to filter documents on the Web and, as such, originating from dynamic information sources. This dynamic factor would make it impossible to apply a cluster approach, based on stable documents, in such an environment.

As concerns the use of artificial neural networks for adaptive information seeking applications, the work described in (Jennings and Higuchi, 1993) is certainly worthy of note. It sets forth a proposed user modeling method for personal news services based on neural networks, which can be performed on an incremental basis. The system analyzes the content of articles kept in an information store and determines the ranking of the articles in accordance to the user model neural network. This model consists of a series of nodes, each of which corresponds to a term that can be found in the documents under review, representing the user's information needs. Each node is assigned a certain energy depending on the number of articles, previously accepted by the user during a specific session, which contain the term associated to the node. The nodes are linked to each other by arcs weighted in accordance to how often a certain pair of nodes (i.e., terms) appears in the articles read by the user. The ranking of new articles taken into consideration is established by comparing the terms of each document with the network nodes. An activation mechanism is initiated, at the end of which the energy of the active nodes is added up and the resulting sum constitutes the rating assigned to the document. Chen and Norcio (Chen and Norcio, 1997) present a user modeling architecture based on a combination of artificial neural networks that we believe of interest here, even if this particular approach is not used for information filtering applications. The reported architecture is used to represent and infer users' task-related characteristics. The networks take up the function of associative memories capable of associating a user's domain knowledge pattern with a completely hypothetical pattern characterizing the user. Both of the previous systems use a 'pure' neural approach. One advantage of this approach is the inherent fault tolerance capability of the networks. The reverse side of the coin is that a symbolic representation of the user model must be renounced.

7. Summary and Future Work

This paper presented an information filtering system capable of selecting html/text documents on computer science available on the Web by making use of the document's content and the long-term interests of the user represented in the user model. The system operates as an adaptive interface to a search engine (ALTAVISTA). This work addressed two main issues: (i) user modeling, i.e., how to represent, construct and maintain models appropriate for the needs of Internet users and (ii) document representation and ranking, i.e., how to represent online documents and compute their relevance with respect to user needs.

As concerns the user modeling problem, we chose to represent the user model as a frame. The slots of the frame represent the domain terms associated with a relevance value specified for the particular user. A distinctive characteristic of the proposed representation, which we believe to be particularly suited to operate on the Web, is its ability to develop and evolve itself. In fact, the model is enhanced by a dynamic semantic-net construction, relevant to co-occurrences. It can also vary dynamically in respect to the slots. As for the user modeling process, we proposed a stereotype-based approach, characterized by a machine learning technique which takes inspiration from a CBR framework that uses *retrieve* and *adapt* as a core problem solving model. The proposed user modeling method, based on neural networks for the *retrieve* phase, is different from other studies that employ a 'pure' neural approach (see, for example, (Jennings and Higuchi, 1993) and (Chen and Norcio, 1997)). Undoubtedly, a 'pure' approach such as this presents advantages linked primarily to the inherent fault tolerance capability of the neural networks. However, we chose to maintain a symbolic representation of the user model, using a neural network essentially as a means of scoring alternative 'symbolic views' of the user. We believe that, while remaining in the field of user modeling for information filtering, our approach makes it easier to represent the 'context' of a term. It allows for a user model that is transparent, i.e., visible to the user and available for direct manipulation functions (Shneiderman, 1998), a property that our users seem to appreciate. Finally, the non-monotonicity in user reasoning is more manageable.

With respect to problems connected to document representation and relevance evaluation, the approach we adopted is unlike other methods given in the literature, and is essentially based on the representation modality chosen for the user model, to which documents are matched.

The system assigns a relevance value to each document retrieved on the basis of the terms it contains (by duly assessing their occurrences), to the terms (with the respective weights) present in the user model and to the terms present in the query. Therefore, the query plays an important role for the purposes of the document evaluation. The system also allows the user the option of expressing a vote (relevance feedback) on the documents selected by the system. A feedback update of the model is performed following each vote.

The system was assessed, using a controlled experiment, for the evaluation of the added value offered by the user modeling component of the system, compared to the version of the system deprived of such an adaptive component (basically, WIFS vs. ALTAVISTA). The data collected from the experiment were subject to a non-parametric test for hypothesis testing (Wilcoxon–Signed–Rank test). The reasoning behind this particular choice of test was that, for this type of system, we maintain that only a minimal knowledge of the distribution function of the chance variable can be hypothesized and therefore no kind of assumption regarding the form (e.g., normal distribution) of the function can be made. In both cases the results of the experiment were statistically significant ($p \ll \alpha$). Furthermore, the added value provided by the system was assessed in about 30–34% for the metrics used. We deem this result satisfactory, especially considering the project choice of not using stemming for document representation process, a choice that we initially feared could have excessively penalized the retrieval performance of the system.

A second evaluation concerned the usability of the system. Clearly, the use of the system requires extra work from the user compared to sessions involving only the search engine. However, the analysis of the questionnaires revealed a general acceptance by users. The only point that caused some uncertainty was the request for a relevance feedback on an integer scale judged to be too extensive. A scale based on three values is definitely preferable (relevant, not relevant, neutral) or even restricted to two values (relevant, not relevant). Instead, the setting of the various parameters (number of documents to analyze, number of characters to download per document, activation of stop-list) did not raise any problem, since the users performed it only once, at the beginning of the first working session. Lastly, our experimentation brought out the need to integrate the system with a browser window, if for no other reason to facilitate a rapid comparison of documents retrieved.

As future work, we plan to effect an experiment of a *sensitivity analysis* (Cohen, 1995), obtained by degrading the data structures or parameters of the system and examining the produced effects. For example, we aim to investigate the effects provoked by a progressive reduction of the scale available to the user for relevance feedback purposes. We also plan to effect a comparison between the performance of the current system and a version of the system without the TMS, or of the initial interview results or the stereotype knowledge base. This should enable us to assess the usefulness and the added value provided by the various individual components, while taking into consideration the fact that the system's user model can, in any event, be created and updated solely by means of the feedback algorithm (in addition to direct editing). Another experiment which we intend to conduct pertains to the evaluation of the system to react and adapt rapidly to changes in the interests of users. We already performed a pilot experiment, involving four users, the results of which are worthy of attention. This evaluation must in any event be verified through more extensive experiments. We also envisage a re-implementation of the system in a *n*-tier architecture, the use of a thesaurus and more extensive experimentation, particularly with lay users.

Acknowledgements

The authors acknowledge the editors and the anonymous reviewers for their valuable comments. We also wish to thank Leonardo Ambrosini, Luciana Bordoni and Vincenzo Cirillo who have contributed to various stages of the realization of this work.

References

Aamodt, A. and Plaza, E.: 1994, Case-based easoning: Foundational issues, methodological variations, and system approaches. *AI Communications* **7**(1), 39–59.

Armstrong, R., Freitag, D., Joachims, T. and Mitchell, T.: 1995, WebWatcher: A learning apprentice for the world wide web, *AAAI Spring Symposium on Information Gathering from Hetereogeneus,Distributed Environments*, March 1995.

Artale, A., Magnini, B. and Strapparava, C.: 1997, WordNet for Italian and its use for Lexical Discrimination. In: M. Lenzerini (ed.): *AI*IA-97: Advances in Artificial Intelligence*, Lecture Notes in Artificial Intelligence, Vol. 1321. Springer Verlag, pp. 346–356.

Asnicar, F., Di Fant, M. and Tasso, C.: 1997, User model-based information filtering. In: M. Lenzerini (ed.): *AI*IA 97: Advances in Artificial Intelligence*, Lecture Notes in Artificial Intelligence, Vol. 1321. Springer Verlag, pp. 242–253.

Baclace, P. E.: 1992, Competitive agents for information filtering. *Communications of the ACM* **35**(12), 50.

Baeza-Yates, R. and Ribeiro-Neto, B.: 1999, *Modern Information Retrieval.* Addison-Wesley.

Balabanović, M.: 1998, Exploring versus exploiting when learning user models for text recommendation. *User Modeling and User-Adapted Interaction* **8**(1–2), 71–102.

Balabanović, M. and Shoham, Y.: 1997, Fab: content-based, collaborative recommendation. *Communications of the ACM* **40**(3), 66–72.

Brajnik, G. and Tasso, C.: 1994, A shell for developing non-monotonic user modeling systems. *International Journal of Human–Computer Studies* **40**, 31–62.

Chakrabarti, S., Van den Berg, M. and Dom, B.: 1999, Focused crawling: a new approach to topic-specific Web resource discovery. *Computer Networks* **31**, 1623–1640.

Chen, Q. and Norcio, A. F.: 1997, Modeling a user's domain knowledge with neural networks. *International Journal of Human–Computer Interaction* **9**(1), 25–40.

Chin, D. N.: 2001, Empirical evaluation of user models and user-adapted systems. *User Modeling and User-Adapted Interaction* **11**, 181–194.

Cohen, P. R.: 1995, *Empirical Methods for Artificial Intelligence*. MIT Press, Cambridge, Massachusetts.

Devore, J. L.: 1995, *Probability and Statistics for Engineering and the Sciences*. Brooks/Cole Publishing Company, Monterey, California, fourth edition.

Dix, A., Finlay, J., Aboud, G. and Beale, R.: 1993, *Human–Computer Interaction*. Prentice-Hall.

Doyle, J.: 1979, A truth maintenance system. *Artificial Intelligence* **12**, 231–272.

Fellbaum, C.: 1998, *WordNet. An Electronic Lexical Database*. The MIT Press.

Finin, T. W.: 1989, GUMS–A General User Modeling Shell. In: A. Kobsa and W. Wahlster (eds.): *User Models in Dialog Systems*, Springer, Berlin, Heidelberg, pp. 411–430.

Forbus, K. D. and De Kleer, J.: 1993, *Building Problem Solvers*. The MIT Press.

Frakes, W.: 1992, Stemming Algorithms. In: W. Frakes and R. Baeza-Yates (eds.): *Information Retrieval: Data Structures & Algorithms*, NJ, Prentice Hall, Englewood Cliffs, pp. 131–160.

Gentili, G., Micarelli, A. and Sciarrone, F.: 2003, InfoWeb: An adaptive information filtering system for the cultural heritage domain. *Applied Artificial Intelligence* **17**(8–9), 715–744.

Goldberg, D., Nichols, D., Oki, B. M. and Terry, D.: 1992, Using collaborative filtering to weave the information tapestry. *Communications of the ACM* **35**(12), 61–70.

Goonatilake, S. and Khebbal, S. (eds.): 1995, *Intelligent Hybrid Systems*. John Wiley & Sons.

Hanani, U., Shapira, B. and Shoval, P.: 2001, Information filtering: Overview of issues, research and systems. *User Modeling and User-Adapted Interaction* **11**, 203–259.

Harman, K. D. and Candela, G.: 1990, Retrieving records from a gigabyte of text on a minicomputer using statistical ranking. *Journal of the American Society for Information Science* **41**(8), 581–589.

Harper, B. D. and Norman, K. L.: 1998, Improving user satisfaction: The questionnaire for user interaction satisfaction version 5.5. *Proceedings of Mid Atlantic Human Factors Conference*, Virginia Beach, February 1998, pp. 224–228.

Haykin, S.: 1994, *Neural Networks–A Comprehensive Foundation*. Prentice Hall International.

Jameson, A.:1999, User-adaptive systems: An integrative overview. Tutorial presented at: *Seventh International Conference on User Modeling UM-99*, Banff, Canada, June 20.

Jennings, A. and Higuchi, H.: 1993, A user model neural network for a personal news service. *User Modeling and User-Adapted Interaction* **3**(1), 1–25.

Joachims, T., Freitag, D. and Mitchell, T.: 1997, WebWatcher: A Tour Guide for the World Wide Web. *Proceedings of the International Joint Conference on Artificial Intelligence IJCAI-97*, pp. 770–775, August 1997.

Kay, J.: 1995, The UM toolkit for cooperative user modeling. *User Modeling and User-Adapted Interaction* **4**(3), 149–196.

Kemeney, J. G. and Snell, J. L., 1962, *Mathematical Models in the Social Sciences*. New York, Blaisdell.

Kobsa, A., and Pohl, W.: 1995, The user modeling shell system BGP-MS. *User Modeling and User-Adapted Interaction* **4**(2), 59–106.

Leuski, A. and Allan, J.: 2004, Interactive information retrieval using clustering and spatial proximity. *User Modeling and User-Adapted Interaction* **14**(2–3), 259–288.

Maes, P.: 1994, Agents that reduce work and information overload. *Communications of the ACM* **37**(7), 30–40.

Magnini, B. and Strapparava, C.: 2004, User modelling for news web sites with word sense based techniques. *User Modeling and User-Adapted Interaction* **14**(2–3), 239–257.

McAllester, D. A.: 1980, *An Outlook on Truth Maintenance. AI Memo 551*, MIT AI Laboratory, Cambridge, Massachusetts, 924–929.

Micarelli, A., Sciarrone, F., Ambrosini, L. and Cirillo, V.:1998, A Case-Based Approach to User Modeling. In: B. Smyth and P. Cunningham (eds.): *Advances in Case-Based Reasoning*, Lecture Notes in Artificial Intelligence, Vol. 1488, Springer-Verlag, Berlin, pp. 310–321.

Mladenic, J.: 1999, Text-learning and related intelligent agents: A survey. *IEEE Intelligent Systems* **14**(4), 44–54.

Mock, K. J. and Vemuri, V. R: 1997, Information filtering via hill climbing, wordnet and index patterns. *Information Processing and Management* **33**(5), 633–644.

Mostafa, J., Mukhopadhyay, S., Lam, W. and Palakal, M.: 1997, A multi-level approach to intelligent information filtering: Model, system and evaluation. *ACM Transactions on Information Systems* **15**(4), 368–399.

Narendra, K. S. and Thathachar, M. A. L.: 1989, *Learning Automata – An Introduction*. NJ:Prentice-Hall, Englewood Cliffs.

Oard, D. W.: 1997, The state of the art in text filtering. *User Modeling and User-Adapted Interaction* **7**(3), 141–178.

Pazzani, M. and Billsus, D.: 1997, Learning and revising user profiles: The identification of interesting web sites. *Machine Learning* **27**, 313–331.

Rich, E.: 1989, Stereotypes and User Modeling. In: A. Kobsa and W. Wahlster (eds.): *User Models in Dialog Systems*, Springer-Verlag, pp. 35–51.

Rumelhart, D. E. and McClelland, J. L. (eds.): 1986, *Parallel Distributed Processing*. MIT Press, Cambridge, Massachusetts.

Salton, G.: 1971, *The SMART Retrieval System – Experiments in Automatic Document Processing*. NJ: Prentice Hall Inc., Englewood Cliffs.

Salton, G. and Lesk, L.: 1968, Computer evaluation of indexing and text processing. *Journal of the ACM* **15**(1), 8–36.

Shapira, B., Shoval, P. and Hanani, U.: 1997, Stereotypes in information filtering systems. *Information Processing & Management* **33**(3), 273–287.

Shapira, B., Shoval, P. and Hanani, U.: 1999, Experimentation with an information filtering system that combines cognitive and Sociological filtering integrated with user stereotypes. *Decision Support Systems* **27**, 5–24.

Shneiderman, B.: 1998, *Designing the User Interface – Strategies for Effective Human-Computer Interaction*. Addison-Wesley, Third Edition.

Siegel, S. and Castellan, N. J.:1988, *Nonparametric Statistics for the Behavioral Sciences*. New York: McGraw-Hill Inc..

Waern, A.: 2004, User Involvement in Automatic Filtering–an experimental study. *User Modeling and User-Adapted Interaction* **14**(2–3), 201–237.

Weibelzahl, S. and Weber, G.: 2001, A Database of empirical evaluations of adaptive systems. In: R. Klingerberg, S. Ruping, A. Fick, N. Henze, C. Herzog, R. Molitor and O. Schroeder (eds.): *Proceedings of Workshop Lernen-Lehren-Wissen LLWA 01*, Research Report in Computer Science nr. 763, University of Dortmund, pp. 302–306.

Wilcoxon, F.: 1947, Probability tables for individual comparisons by ranking methods. *Biometrics* **3**, 119–122.

Wong, S. K., Yao, Y. Y. and Bollmann, P.: 1988, Linear Structure in Information Retrieval. *Proceedings of The 18th International ACM Conference on Research and Development in Information Retrieval SIGIR-88*, pp. 219–232.

Wong, S. K. and Yao, Y. Y.: 1990, Query formulation in linear retrieval models. *Journal of the American Society for Information Science* **41**(5), 334–341.

Yao, Y.Y.: 1995, Measuring retrieval effectiveness based on user preference of documents. *Journal of the American Society for Information Science* **46**(2), 133–145.

Author's vitae

Alessandro Micarelli is an Associate Professor of Computer Science at the University of *Roma Tre*, where he heads the Artificial Intelligence Laboratory at the Department of Computer Science and Automation. His research interests are adaptive Web-based systems, Information Filtering, Adaptive Hypermedia, Artificial Intelligence in Education.

Filippo Sciarrone is a Ph.D. student. He received his degree in mathematics and a specialisation in Computer Science. Since 1994, he has been collaborating with the Department of Computer Science and Automation of the University of *Roma Tre*. His research interests are User Modeling, Information Filtering and Machine Learning. He is currently Software Division Manager at S.T.E. S.p.A.