Journal of Universal Computer Science, vol. 3, no. 8 (1997), 988-1006
submitted: 1/6/97, accepted: 10/8/97, appeared: 28/8/97 © Springer Pub. Co.

Profiling with the INFOrmer Text Filtering Agent

Humphrey Sorensen
(Department of Computer Science, University College Cork, Ireland
h.sorensen@cs.ucc.ie)

Adrian O' Riordan
(Department of Computer Science, University College Cork, Ireland
a.oriordan@cs.ucc.ie)

Colm O' Riordan
(Department of Computer Science, University College Cork, Ireland
c.oriordan@cs.ucc.ie)

Abstract: INFOrmer is an intelligent filtering system, currently being applied to the management of USENET News articles. An individual may have one or more profiles, each representing a long-term interest of that user. The user profile is then used to measure the relevance of incoming articles and filter out irrelevant documents. A user profile may be modified as a result of relevance feedback, so that it adjusts to users' changing interests. This paper discusses the architecture of INFOrmer and covers the profile/document representation and comparison techniques adopted within the system.

Key Words: Text filtering, semantic network, spreading activation, relevance feedback.
0ategory: H.3.3

1 Introduction

The increasing popularity of the Internet has led to an explosion in the amount of online data available to the user. One of the major sources of information is through the USENET news feed. The high traffic through many of these newsgroups (estimated at roughly 400 megabytes of textual data a day), coupled with the very coarse filtering provided by the newsgroup hierarchy, has resulted in a situation where users are flooded with information, much of which is irrelevant to him/her. This problem has led to the *productivity paradox* whereby the availability of more and more information has actually resulted in reducing the productivity of these users. The growing volume of available online information threatens to exceed the users' capability to sift through it. This increase has accentuated the need for automatic filters to separate the relevant from the irrelevant.

For such a system to be of practical benefit to the user, it must filter information in an intelligent manner on behalf of the user and must be capable of learning to provide more precise filtering (and to deal with users' changing information needs) through communication with the user. In fact, it is appropriate to view the filter as an intelligent agent operating on the user's behalf, as it possesses many of the properties associated with intelligent agents - being adaptive (the filtering agent adapts to suit user's changing information needs), autonomous (the agent filters without user intervention) and communicative (the agent communicates with the user through user feedback)[Beale 1994].

To protect the user from possible information overload, we have constructed an intelligent filtering agent, INFOrmer [O'Riordan and Sorensen 1995], to filter out information in an effective manner. To cater for a user's changing information need, and to improve precision with filtering, a learning mechanism is also incorporated.

2 Background

2.1 Information Filtering

Numerous approaches and implementations of text filtering and text retrieval systems exist. Earliest approaches utilised simple string searching - if an incoming article contained a user-specified string it was deemed relevant. This simple and easy-to-use approach has one major disadvantage, namely it "is based on the assumption that it is a simple matter for users to foresee the exact words and phrases that will be used in the documents they will find useful and only in those documents"[Blair and Maron 1985]. Most commercial systems are still based on pattern matching techniques, augmented by the use of Boolean expressions containing the "AND" and "OR" operators, and by proximity operators. The major drawbacks with this approach are that i) this type of input requires some skill on behalf of the user for any reasonably complex information need and ii) different vocabularies may result in low recall.

The vector space approach [Salton 1983] is based on the statistical occurrence of words in both the user query (profile) and the documents (incoming articles). The user profile consists of a collection of words, each with an associated weight, which occupies a slot in a vector. The incoming article is also viewed as a vector of weighted terms. The advantages of this approach are adaptability, robustness and minimal user intervention. The main disadvantages are the possibility of different terms in the article describing the same concept (synonymy) and the possibility of the same terms describing differing concepts based on differing context (polysemy), e.g., blind Venetian and Venetian blinds. Thesauri have been used to overcome the problem of polysemy by expanding the initial query or profile. This has proved beneficial but has the disadvantage that the additional context provided by associated terms in a profile is ignored.

Latent Semantic Indexing (LSI) [Dumais et al 1990] attempts to overcome the problems associated with word-based methods by organising textual information into a semantic structure more suitable to information filtering. The LSI approach attempts to filter/retrieve information at a semantic, rather than at a syntactic or lexical level by not basing the comparisons between documents on the terms in the document but on the domains within which these terms occur. For example, SFCs (Subject Field Codes) were used by Liddy, Paik & Yu to provide a feature set for incoming documents above that of word-level. Thus, if the word "people" occurred then the SFCs *sociology*, *political science* and *anthropology* are attached to the word. A weighted vector of the SFCs is then used to determine relevancy of documents [Liddy et al 1994]. This method avoids the problems present in filtering techniques based on natural language but suffers from other difficulties - such as problems in attaining fine-grained filtering without user-defined domains.

Probabilistic models for information filtering and retrieval have tended to stem from Robertson's "Probabilistic Ranking Principle" [Robertson 1977] which states that:

> 'For optimal performance, [a retrieval] system a should rank the documents according to their probability of being judged relevant or useful to the user's problem or information need.'

Connectionist Networks have been applied to the problems of information retrieval and information filtering by Mozer and Belew[Mozer 1984] [Belew 1989] [Belew 1986]. In Belew's AIR (adaptive information retrieval) system, nodes in the network representation of the document/article are set to a certain level if they occur in the user's profile. This weight ('activity') is then leaked out over the network. The learning mechanism in this system has been shown to be capable of learning some advanced semantic features, such as word stems, synonyms and simple phrases[Belew 1986].

2.2 Feedback Techniques

Relevance feedback has proven to be highly effective for improving information filtering and retrieval. Upon receiving filtered articles, the user may provide judgements for these articles. These relevance judgements may be employed to guide relevance feedback for the filtering system. Given the results from the filtering system, the user is asked to identify which documents are relevant and which are irrelevant. This information, along with the current user profile, P_k, is then used to form a new profile, P_{k+1} which is used as the user's profile in future filtering.

Relevance feedback techniques for two of the most popular filtering methods are:

- Vector-Space Model:
  The Rocchio feedback model [Rocchio 1971] is the most common method used. Rocchio showed that an more effective profile representation could be iteratively generated as follows:

$$\text{Rocchio:}\quad P_{k+1} = P_k + \beta\sum_{k=1}^{n_1}\frac{R_k}{n_1} - \gamma\sum_{k=1}^{n_2}\frac{S_k}{n_2}$$

  where P_{K+1} is the new profile, P_k is the old profile, R_k is a vector representation of a relevant article k, S_k is a vector representation for non–relevant article k, n_1 is the number of relevant documents and n_2 is the number of non–relevant documents. The values β and γ determine the relative contributions of positive and negative feedback, respectively.

Relevance feedback using this technique has been shown to result in a significant improvement in retrieval performance [Salton 1989].

- Probabilistic networks:
  A query can be modified by the addition of the first m terms taken from a list where all terms present in documents deemed relevant are ranked according to the formula[Robertson and Sparck-Jones 1976]:

$$w_i = log[\frac{r_i(N - R - n_i + r_i)}{(R - r_i)(n_i - r_i)}]$$

  where N is the number of documents retrieved, n_i is the number of documents with an occurrence of term i, R is the number of documents deemed relevant by the user, r_i is the number of relevant documents containing an occurrence of term i.

Feedback techniques for newer filtering models (connectionist) are discussed in [Biron and Kraft 1993].

3 System Architecture

3.1 Overview

The INFOrmer filtering agent described in this paper attempts to achieve effective filtering (high precision) by trying to overcome many of the problems associated with previous attempts - differing vocabularies (polysemy), comparisons based on terms and keywords only (and not a term's surrounding context). The system also provides a learning mechanism to incorporate user feedback.

The system, whose architecture is depicted in Figure 1, consists of four main components:

1. The preprocessing of the textual data (both user-provided and incoming articles).
2. Semantic network representation of user profile and articles.
3. A spreading activation method for profile/article comparison and ranking.
4. A user-feedback module which adapts machine learning techniques for profile adjustment in an attempt to attain higher precision filtering.

3.2 User Interaction

The user interacts with the INFOrmer filtering engine via a HTML interface (see Figures 2 and 3). A user-friendly interface is used to display filtered articles and to allow users to provide feedback judgements based on these articles (a cut-and-paste mechanism is provided for passage-level feedback and a toggle button is provided for full article feedback). The user interface also provides information about the system and guidelines to help the user interact with the system.

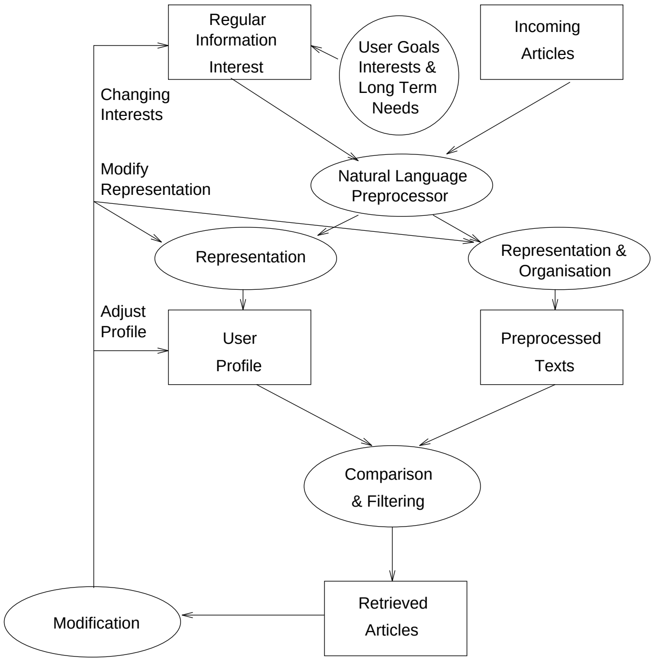

Figure 1: System Architecture

3.3 Preprocessor

The preprocessing phase of the system transforms the textual representation of the user's information need and the incoming articles into an intermediate form which is then used to construct the semantic graph representation. The components of the preprocessing phase are:

- **Stopword Removal:** Stopwords refer to words which occur with a very high frequency and are usually articles, prepositions and conjunctions. These words are said to have a *low resolving power*, i.e. they do not contribute to the overall semantic meaning of a document/article. Stopword removal is beneficial in that it increases accuracy of comparison and makes the system more computationally inexpensive.
- **Stemming Algorithm:** This is a computational procedure which reduces all words with the same root to a common form, usually by stripping each word of its inflectional and derivational suffixes (e.g., the words *computer*, *computerisation*, *computing* are all stemmed to a common form *comput*). The algorithm used in this system is based on Lovin's algorithm [Lovins 1968], a longest-match, context-sensitive algorithm, which uses a list of ordered endings, with context-sensitive rules associated with these endings. A second phase of the algorithm uses a set of respelling rules to convert stemmed words to the same root term (e.g., *absorption* and *absorbed* will be stemmed to *absorp* and *absorb* respectively; the respelling rules will cause the following transformation to occur: $absorp \rightarrow absorb$).

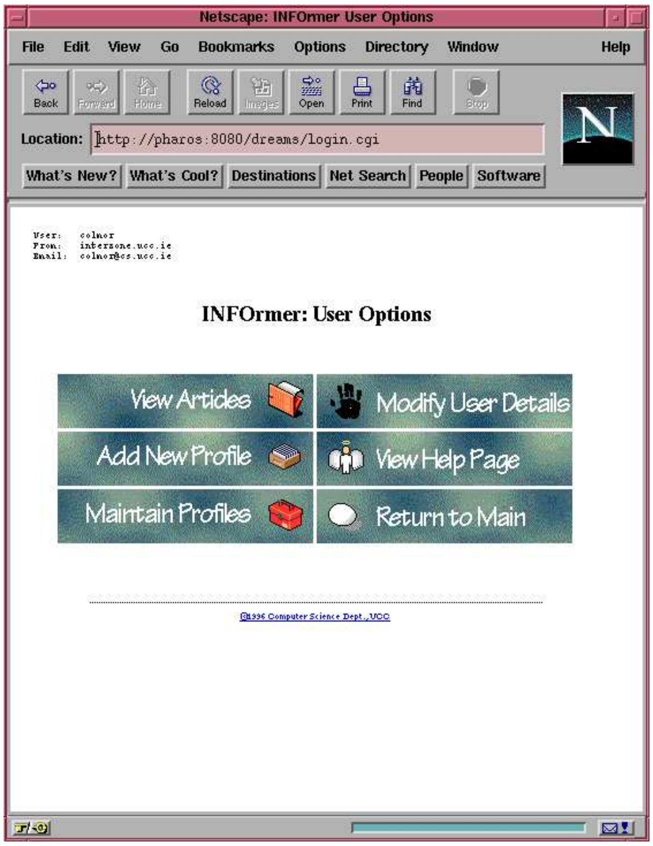

Figure 2: User Interface

– **Sentence Boundary Disambiguation:** This is a process of extracting sentence delimiters. This system utilises a series of rules to provide effective terminator detection. Exceptions like *Mr.*, *i.e.*, *Dr.* etc. are handled by hard-coded rules. Termination detection is important in aiding phrase recognition in the next phase.

The stemmed article/profile with stopwords removed, together with a record of sentence terminators, is passed to the second component of the system.

3.4 Semantic Network Representation

In the semantic network representation of the user profile and the incoming articles, each term is represented by a weighted node. Adjacent terms are linked by weighted edges. An initial semantic network is created from the output of the preprocessing phase, attention being given to sentence boundaries so as to prevent erroneous edges being added (incorrect 'phrases'). Slightly different representations are used for the user profile and the article:

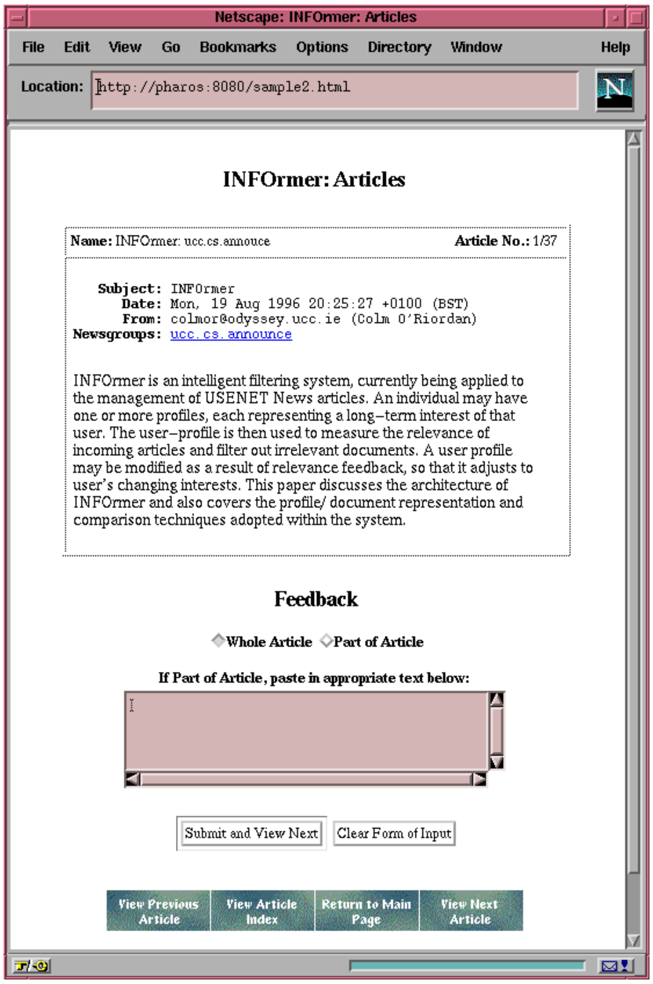

Figure 3: User Interface

– Profile representation: The initial representation is modified to merge together nodes representing the same term. In merging terms and phrases, attention is paid to the context in which they arise; phrases and terms from different contexts are not merged. This helps overcome the problem of polysemy and its negative effect on the precision of filtering algorithms. This merging of nodes causes increases in activity levels of the nodes and of the edge weights. Terms and phrases (linked nodes) with a high frequency of occurrence have their weights increased accordingly. This graph representation is useful in that not only are individual terms weighted, as in the vector space model etc., but importance is also attached to 'phrases', i.e., to groups of terms that occur together. This emphasis on phrases over isolated words is a relatively recent phenomenon in information retrieval and has been recognised as having benefit for improving system accuracy [Kelledy and Smeaton 1997]. While others have approached phrase recognition from the natural language processing (NLP) viewpoint [Strzalkowski and Carballo 1996], INFOrmer essentially takes a statistical approach.

For example, the following extracts from a user profile would be merged:

... the gravity of the situation ...due to the forces of gravity ... the force of gravity is ...

to give the graph segment depicted in Figure 4.

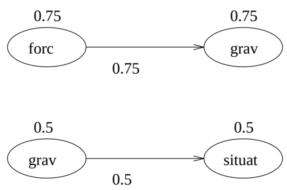

Figure 4: Graph Representation

As can be seen the phrase "forces of gravity" is given a higher weighting than the phrase "gravity of the situation" and also the two different occurrences of gravity are not merged because of different contexts.
The weighting scheme used in the graph is as follows: Each term and edge in the graph is given the initial value of 0.5. Upon merging two nodes n_i and n_j the weight of the new node $n_{(ij)}$ is defined by

$$n_{(ij)} = n_i + (1 - n_i)n_j$$

This ensures that the new value of the node is always increased and that the weight is always normalised between 0 and 1. The value of the node increases towards 1. A similar function is used for modifying the weights of edges when merged. For example, in the above profile, the values of the node *gravity* and *forc* and the edge between them are set to 0.5 (initial learning of terms and phrases). On encountering these phrases again, their values are all increased to 0.75. This represents reinforcement of these terms, i.e. the filtering agent learns that the terms *'gravity'* and *'force'* and the phrase *'force of gravity'* are more important than the phrase *'gravity of the situation'*.

– Article Representation: The representation of incoming articles differs from that of the profile in two respects: unweighted links are used, as the occurrence of a phrase in an incoming article is not a priori known to be significant; and its nodes do not have activation levels associated with them.

A more thorough example is provided: given the extracts of text depicted below, the corresponding networks created are depicted in Figures 5 and 6 respectively.

- Text used to create the profile graph:
  The main principles of using symbolic, fuzzy and neuro systems for problem solving .... Fuzzy and neural systems can improve ...

- Article Text:
  Neural networks, genetic algorithms and other subsymbolic approaches .... Genetic algorithms improve ....

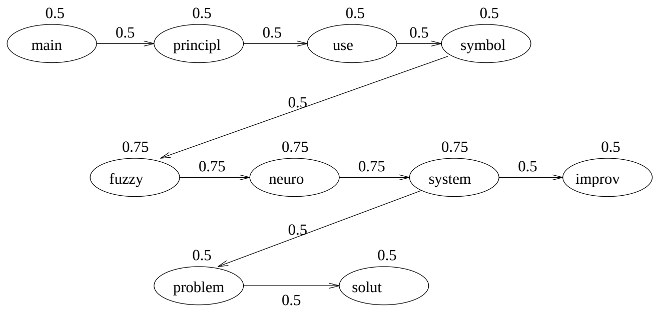

Figure 5: User Profile

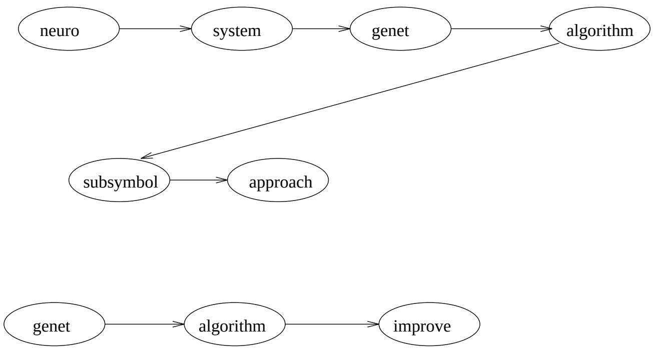

Figure 6: Incoming Article

3.5 Comparison Algorithm

In our graph representation both phrases and terms in the profile and articles are used as the feature set. Our comparison algorithm reflects this with priority given to similarity of groups of nodes (phrases) over that of individual nodes, i.e., the comparison of a user profile with an article involves localised matching of structural similarity between the profile graph and the article graph using the weights of edges and terms in the profile.

The comparison utilises a spreading activation algorithm to highlight portions of graphs (representing 'phrases') common to both profile and article. The algorithm consists of setting the activity level (weights) of nodes in the local article graph that are common to both article and profile graph. The activity is then 'leaked' out to its neighbouring nodes, depending on these neighbouring nodes' occurrences (and their position and weighting) in the profile graph. The leaking of the activity is controlled in such a manner that priority is given to occurrences of 'phrases' common to both.

Initially all nodes in the graph representation of the incoming article are set to zero. Let $\mathcal{P}(\mathcal{V}, \mathcal{E})$ be the profile graph and $\mathcal{A}(\mathcal{V}', \mathcal{E}')$ be the incoming article.

$$\forall n_i, n_i \in V':$$
$$(Wt_A(n_i) \leftarrow 0)$$

i.e., set the initial weight of all article nodes n_i to zero.

The first phase in the comparison algorithm involves setting the weight (activity level) of all article nodes also occuring in the user profile, to the weight of that same node in the user profile.

$$\forall n_i, ((n_i \in \mathcal{V}') \wedge (n_i \in \mathcal{V})):$$
$$(Wt_A(n_i)) \leftarrow (Wt_P(n_i))$$

The spreading activation mechanism involves spreading these weights out to the neighbouring nodes in the article graph. This causes nodes occuring in common phrases to have their weights increased, signifying higher relevance in the overall article, while terms not involved in phrases are adjusted to a different (and varying) extent.

$$\forall n_i, ((n_i \in \mathcal{V}') \wedge ((n_i, n_j) \in \mathcal{E}') \wedge ((n_i, n_j) \in \mathcal{E})):$$
$$Wt_A(n_j) \leftarrow (Wt_A(n_j)) + ((Wt_A(n_j)) \times (Wt_P(n_i, n_j)))$$

$$\forall n_i, ((n_i \in \mathcal{V}') \wedge ((n_i, n_j) \in \mathcal{E}') \wedge ((n_i, n_j) \notin \mathcal{E})):$$
$$(Wt_A(n_j)) \leftarrow \lambda \times (Wt_A(n_j))$$

This re-weighting mechanism causes an increase in weight of the nodes comprising phrases common to both profile and article. The nodes that do not constitute a section of a phrase remain with a positive weight but are not increased (as $\lambda \leq 1$). Other nodes remain at zero.

The net result of the spreading activation in the article graph is that terms that comprise phrases matching phrases in the profile graph will have their weights increased. The number of nodes and phrases above a threshold can be used as a relevance metric. The relevance of an incoming article depends on:

- the frequency of occurrence of certain phrases within the article.
- the relevant importance of those phrases (as depicted by their profile weights).
- to a lesser extent, the frequency of occurrence of certain terms and their occurrences in the profile graph.

The comparison algorithm causes a reweighting of nodes in the article graph as shown in Figure 7. The similarity measure between the graph representations of the profile and the article is calculated to be 22%. This is calculated from counting the number of nodes above a threshold θ (in this case $\theta = 0.4$).

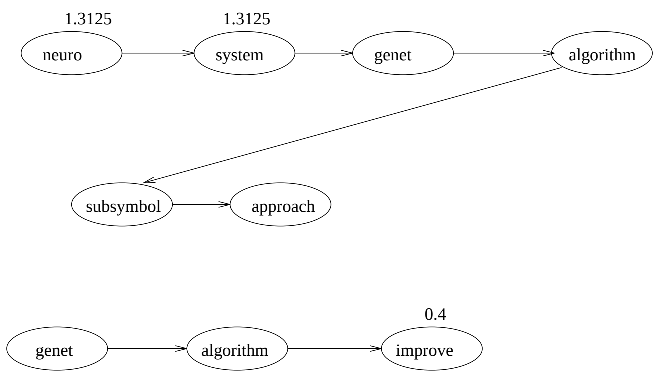

Figure 7: Weighted Article following Comparison

3.6 Relevance Feedback

To improve the precision of the filtering engine and to cater for a user's changing information needs, it is necessary to include a feedback utility. On viewing a filtered article, the user may indicate that this article is relevant to him/her. If an article is deemed relevant, then it is used to modify the profile. Modifying the profile consists of incorporating new phrases and new terms into it and reinforcing a phrase/term that already exists therein.

In the INFOrmer system, the user is offered two types of feedback:

- Full article feedback: The complete text of the current article is used to modify the current user profile.
- Passage level feedback: The user may select specific relevant sections of the text to be used for feedback (see Figure 3).

The feedback module consists of two rules - one for incorporating new terms and phrases (learning), and one for reinforcing already learned terms and phrases. All terms and phrases that occur in the article/passage offered as feedback with activity level greater than that of a specified threshold τ are used to modify the user profile. (The value of the threshold τ is different for the two types of feedback). These rules are more formally defined in section 3.6.1

Prior to applying these rules, the passage of text returned as feedback by the user is represented in a new semantic network. This network is created via the same mechanism as is used to create the original profile graph, i.e., edges and nodes are merged to reflect increased importance of certain phrases and terms.

3.6.1 Rules used for Relevance Feedback

The following rules are used to add new phrases and terms to the user profile. The first rule defines which phrases and terms are added to the user profile. It covers phrases and terms with a high relevancy within the articles graph, i.e. with weights above the threshold. This modification of the profile allows a better representation of the users' information need. The second rule allows the reinforcement of phrases and terms already present in the user profile. The weights of the nodes and edges comprising phrases are increased to reflect their increased relevancy, due to their occurrence in the feedback article.

The learning rate in both rules is governed by the variables δ and τ. τ, the threshold, is used to determine if a phrase/term is of sufficiently high importance in the feedback article to add to the profile. For passage-level feedback, τ has a lesser value. δ is the weight modification factor used when inserting values into the profile graph.

Given a graph $\mathcal{P}(\mathcal{V}, \mathcal{E})$, representing the user profile and a graph $\mathcal{F}(\mathcal{V}', \mathcal{E}')$ representing the feedback article, the following two rules are used to modify $\mathcal{P}$.

- Rule 1: New Phrase and term incorporation:
  - For all nodes in $\mathcal{V}'$ that do not exist in $\mathcal{V}$, add those terms to the graph:

$$\begin{aligned} &\forall n_i,\ ((n_i \in \mathcal{V}') \wedge (n_i \notin \mathcal{V}) \wedge (Wt_F(n_i)) > \tau):\\ &\qquad \mathcal{V} \leftarrow \mathcal{V} \cup \{n_i\}\\ &\qquad Wt_P(n_i) \leftarrow (\delta \times Wt_F(n_i)) \end{aligned}$$

  - For all edges in the feedback graph that are not in the profile graph, add those edges to the profile graph:

$$\begin{aligned} &\forall (n_i, n_j),\ (((n_i, n_j) \in \mathcal{E}') \wedge ((n_i, n_j) \notin \mathcal{E}) \ \wedge (Wt_F(n_i, n_j) > \tau)):\\ &\qquad \mathcal{E} \leftarrow \mathcal{E} \cup \{(n_i, n_j)\}\\ &\qquad Wt_P(n_i, n_j) \leftarrow (\delta \times\ (Wt_F(n_i, n_j))) \end{aligned}$$

- Rule 2: Phrase and term reinforcement:
  - For all edges common to both graphs modify the weight of that edge in the user's profile graph:

$$\begin{aligned}&\forall(n_i, n_j),\ (((n_i, n_j) \in \mathcal{V}) \wedge ((n_i, n_j) \in \mathcal{V}^{'})):\\ &\qquad Wt_P(n_i, n_j) \leftarrow Wt_P(n_i, n_j) + ((1 - Wt_P(n_i, n_j))\\ &\qquad\qquad \times \delta(Wt_F(n_i, n_j)))\end{aligned}$$

- For all nodes common to both graphs, modify the weight of that node in the user's profile graph:

$$\begin{aligned}&\forall n_i,\ ((n_i \in \mathcal{V}) \wedge (n_i \in \mathcal{V}')):\\ &\qquad (Wt_P(n_i)) \leftarrow (Wt_P(n_i))\ + ((1 - Wt_P(n_i)) \times (\delta(Wt_F(n_i))))\end{aligned}$$

3.6.2 Example of Relevance Feedback

Given the feedback graph for an article deemed relevant by the user (see Figure 8), modification of the user profile results in the graph depicted in Figure 9. (learning rate $\delta = 1$, i.e. very quick, and learning threshold $\tau = 0.7$).

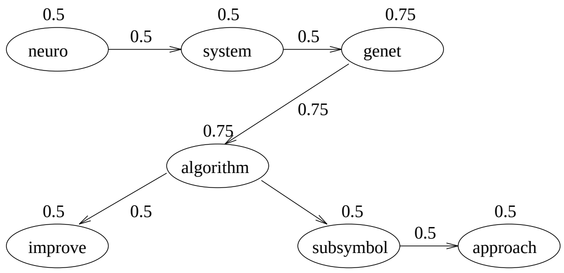

Figure 8: Feedback Article

The phrase "neuro systems " has been reinforced, and the phrase "genetic algorithms" has been learned by the profile. The user's information need should now be captured more fully by the inclusion of more relevant phrases and by appropriate reweighting of the more important (those deemed to have a higher resolving power) phrases/terms in the user's profile.

By comparing this new profile, to the original article graph, a similarity measure of 77% is returned (the new weighting of the nodes is depicted in Figure 10). This small example is used to illustrate the effect of feedback.

Experimentally, given 20 iterations of full-article feedback, the relevance feedback module has shown to improve the recall of the system by between 10-15%.

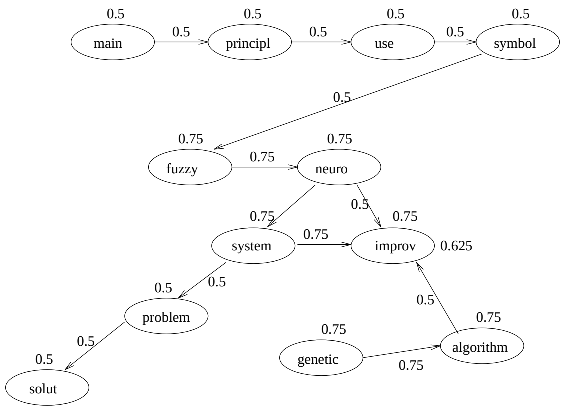

Figure 9: User Profile after feedback

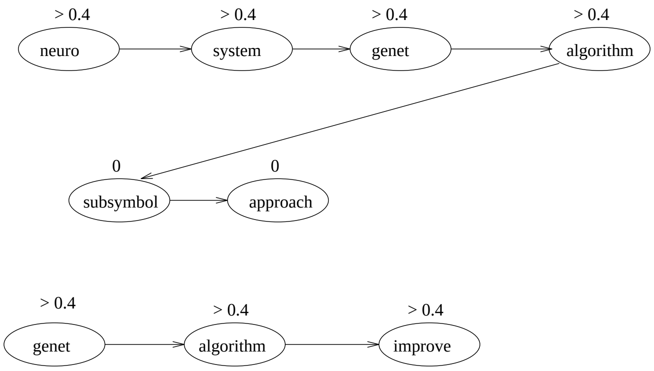

Figure 10: Weighted Article following Comparison (after Feedback)

4 Evaluation

To date the system has been tested on a small user base and has proved to be beneficial to users in helping protect against the problem of information overload, by filtering the USENET newsfeed and presenting the relevant articles to the user.

A more formal methodology for testing has also been employed, using the traditonal information retrieval measures of recall and precision [Salton 1989]. These measure the fraction of returned documents which were deemed relevant by the user and the fraction of documents deemed relevant by the user which were returned by the filter, respectively. It is the aim of retrieval (and filtering) systems to simultaneously maximise these measures, thereby maintaining a high accuracy. INFOrmer has been tested extensively against a large document set for which relevance estimates have been manually estimated (a subset of the TIPSTER data collection [Harman 1995]). It has attained an acceptably high level of precision and recall. Table 1 summarises the run statistics. Effectivness was found to be comparable to other systems tested in TREC-2 (see Table 2 and Figure 13).

Summary Statistics	
Run Data	Category B, automatic
Num of Queries	50
Total number of documents over all queries	
Retrieved	50000
Relevant	3913
Rel_ret	2357

Table 1: Summary Statistics

Recall Level Averages	
Recall	Precision
0.0	0.5917
0.1	0.4236
0.2	0.3773
0.3	0.3126
0.4	0.2534
0.5	0.1991
0.6	0.1564
0.7	0.0778
0.8	0.0575
0.9	0.0203
1.0	0.0021
Average precision over all relevant docs	
non-interpolated	0.2034

Table 2: Recall Level Averages

Category B ad-hoc (automatic)		
inform1	University College Cork	INFOrmer
dgros1	George Mason University	trigram weighting
UniNE1	Université de Neuchâtel	interdocument relations
UniNE2		
DCUNL1	Dublin City University	syntactic analysis
DCUNL2		
expst2	Mayo Clinic/Foundation	Expert Network/word matcher

Table 3: Key to understanding Figure 11

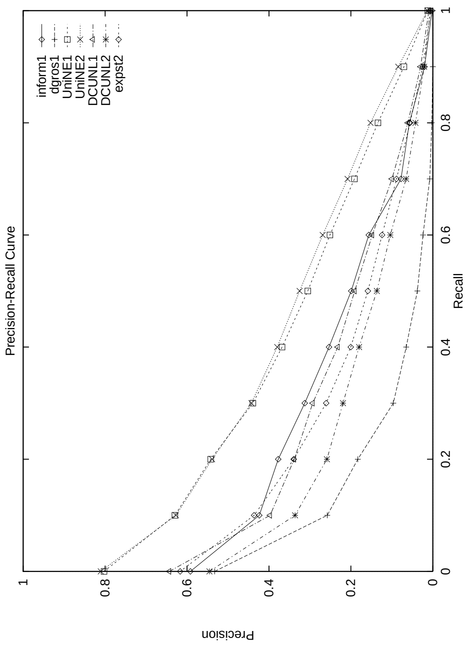

Figure 11: INFOrmer Compared to Other Systems

The level of precision and recall was shown to improve given relevance feedback from the user. Precision-recall graphs for sample runs are given in Figures 11 and 12. Figure 11 shows the decrease in precision as recall increases over a typical sample run (e.g. at recall of 0.7 the precision is 0.8). Figure 12 illustrates the improvement in precision after feedback. These results, in comparison to other filtering engines, are quite good (see Figure 13) and illustrate the advantages of a filtering algorithm using more than just keywords as a feature set for documents and profiles.

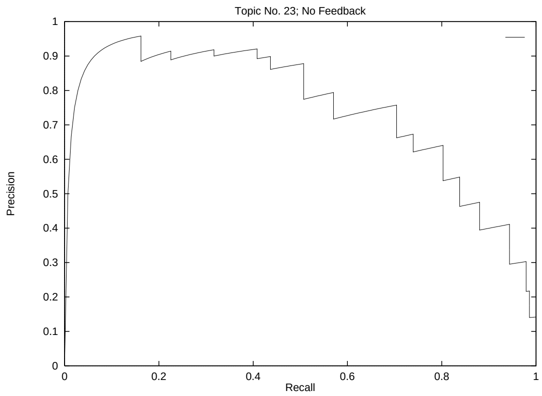

Figure 12: Precision-Recall Ggraph before Feedback

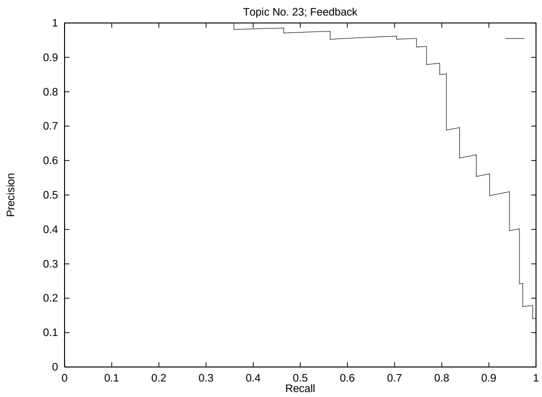

Figure 13: Precision-Recall Graph after Feedback

5 Current Extensions

The INFOrmer filtering engine is currently being extended in the following two directions:

Collaborative Filtering:
As described heretofore, the INFOrmer engine is used to achieve traditional content based filtering. This approach is used in conjunction with a multi-agent based system to allow collaborative (or social) filtering amongst agents filtering in the same domain. A contract net protocol is used to control co-operation between these filter agents. A detailed description of the system can be found in [O'Riordan and Sorensen 1997].

Mobile Filtering of Web-Based Information:
The INFOrmer engine described in this paper has also been applied to the filtering/retrieval of web-based information. The system uses pre-existing web-crawlers and indexes to attain a set of possibly relevant pages for a given information need. To allow a more effective fine-grained filtering, the INFOrmer engine is used to further filter the set of web-pages returned from querying the indexes. For efficiency, this system is being reegineered to allow a mobile approach to this filtering. This system, and future aims, are described in [O'Riordan and Hanafin 1997].

6 Conclusion

In this paper we have seen how effective filtering may be achieved by using a semantic network representation. The INFOrmer approach tries to overcome many of the problems associated with other filtering techniques - synonymy, polysemy, filtering based on terms only and filtering requiring a high level of expertise on the user's behalf. The system uses a semantic network for representation of both user's profile and the incoming articles. A spreading activation technique is used for article/profile comparison. A powerful and easy-to-use relevance feedback module is also included to increase the accuracy of the filtering.

References

[Beale 1994] R. Beale and A. Wood. Agent-based interaction. *People and Computers IX: Proceedings of HCI'94, Glasgow, UK*, pages 239–245, 1994.

[Belew 1989] R. Belew. Adaptive information retrieval: Using a connectionist representation to retrieve and learn about documents. In *Proceedings of the Twelfth Annual International ACMSIGIR Conference on Research and Development in Information Retrieval*, pages 11–20, 1989.

[Belew 1986] R. K. Belew. *Adaptive Information Retrieval: Machine Learning in Associative Networks.* Phd thesis, Univ. Michigan, CS Department, 1986.

[Biron and Kraft 1993] P. Biron and D. Kraft. New methods for relevance feedback: Improving information retreival performance. *Available at http://argo.gslis.ucla.edu/pbiron/pubs/newrf/newrf.html*, 1993.

[Blair and Maron 1985] D.C. Blair and M.E. Maron. An evaluation of retrieval effectiveness for a full document retrieval system. *Communications of the ACM*, 28(3), March 1985.

[Dumais et al 1990] S. Deerwester, S. Dumais, T. Landauer, G. Furnas, and R. Harshman. Indexing by latent semantic analysis. *Journal of the Society for Information Science*, 6(41):391–407, 1990.

[Harman 1995] D. Harman. Overview of the fourth text retrieval conference (trec-4). *Availble online : http://www-nlpir.nist.gov/TREC/trec4.papers/overview.ps*, 1995.

[Kelledy and Smeaton 1997] F. Kelledy and A.M Smeaton. Automatic phrase recognition and extraction from text. In *Proceedings of British Computer Science Information Retrieval Specialist Group, 19th Annual Colloquim on IR*, 1997.

[Liddy et al 1994] Liddy, W.Paik, and E. Yu. Text categorisation for multiple users based on semantic features from a machine-readable dictionary. *ACM Transactions on Information Systems*, 12(3):278–295, July 1994.

[Lovins 1968] J.B. Lovins. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 1:22–31, March 1968.

[Mozer 1984] M. Mozer. Inductive information retrieval using parallel distributed computation. Technical report, University of California, San Diego, 1984. Research Report ICS-8406.

[O'Riordan and Hanafin 1997] C. O'Riordan and M. Hanafin. A mobile agent approach to filtering web-based information. *Abakus, The Journal of the Computer Science Dept., University College Cork*, 1997.

[O'Riordan and Sorensen 1995] H. Sorensen A. O'Riordan. An intelligent agent for high-precision text filtering. *Fourth International Conference On Information and Knowledge Management*, pages 205–211, 1995.

[O'Riordan and Sorensen 1997] C. O'Riordan and H. Sorensen. A multi-agent based approach to collaborative filtering. *Abakus, The Journal of the Computer Science Dept., University College Cork*, 1997.

[Robertson 1977] Stephen F. Robertson. The probability ranking principle in IR. *Journal Of Documentation*, pages 294–304, 1977.

[Rocchio 1971] J.J Rocchio, Jr. Relevance feedback in information retreival. In G. Salton, editor, *The SMART REtreival System : Experiments in Auotmatic Document Processing*, chapter 14, pages 313–323. Prentice-Hall, 1971.

[Robertson and Sparck-Jones 1976] S. Robertson and K. Sparck-Jones. Relevance weighting of search terms. *Journal of American Society for Information Science*, 3(27):129–146, 1976.

[Salton 1989] G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retreival of Information by Computer.* Addison-Wesley, 1989.

[Salton 1983] G. Salton and M. McGill. *Introduction to Modern Information Retrieval.* McGraw Hill International, 1983.

[Strzalkowski and Carballo 1996] T. Strzalkowski J. Carballo. Natural language information retrieval. *TREC-4 Report*, 1996.