An Intelligent Agent for High–Precision Text Filtering

Adrian O'Riordan, Humphrey Sorensen
Department of Computer Science
University College Cork
Ireland
{adrian,sorensen}@odyssey.ucc.ie

Abstract

We present here an overview of a research project aimed at reducing information overload for individual computer users. High-precision information filtering software has been developed to disseminate on–line electronic information. While the robustness and scalability of statistical approaches to information retrieval were a major influence on our design, we looked to the AI literature to supply the necessary techniques for the creation of an adaptive system. The system, called INFOrmer, is based on an intelligent agent approach and embodies machine learning, adaptation and relevance feedback techniques in its construction. A weighted graph representation is used for documents, and graph manipulation algorithms are used in the processing.

1 Introduction

It is generally acknowledged that the volume of information which is accessible over various networks has exceeded the capability of users to sift through it in order to access that which is relevant to them. This problem has led to the *productivity paradox*, whereby making more and more information available to online users has actually resulted in reducing the productivity of these users.

We would claim that what is required is the provision of sophisticated *information retrieval* software (for accessing long–term online databases) and *information filtering* software (for routing more transiently occurring information on a network). It is to this latter process of filtering information to relevant users which INFOrmer addresses itself. The system currently filters an on–line News feed. In particular, an information filter was built which can be personalised by individual users and which models the user's interests so as to route through to him/her those articles which are deemed as relevant. The user may evaluate the significance of received information, thus providing *relevance feedback* which is used in fine–tuning the filter (or *user profile*) so as to improve its precision and to better model a user's changing interests. In this sense, the profile learns of a user's preferences through assimilation of an initial set of interesting documents and continues this learning process via relevance feedback throughout its lifetime.

2 Background

2.1 Information Filtering

Simple filtering systems have been based on manual keyword indexing or string matching techniques, generally augmented by the use of thesauri to cater for synonymy. More recent research efforts have evolved from perceived similarities between filtering and the more mature field of information retrieval [BC92]. Information retrieval (IR) has a long history, the manual indexing of books and documents in libraries being a well–known example [Ell90]. Today researchers in IR are well aware of the shortcomings of these approaches and either pay more attention to the ambiguity and vagaries that exist in natural language text, or they take greater advantage of the structure and position of words in the texts [LCB89].

Evolving primarily from the *word frequency model* [Luh58], techniques adopted in IR have included the *Boolean retrieval model* for article indexing and fuzzy logic extensions of same [SFW92]. A more recent research vehicle is the *vector space approach* and variations thereof [SM83]. Latent Semantic Indexing (LSI) [DDL$^+$88] represents a more sophisticated statistical framework for vector space systems. Methods based on Bayesian networks, such as the Inference Net system of Turtle and Croft [TC91], have given good performance results. Neural networks have also been applied to the

CIKM '95, Baltimore MD USA
© 1995 ACM 0-89791-812-6/95/11..$3.50

problem [Kwo89][Bel89]. All of these methods, and others, have been adopted for use in information filtering [Fol90][YGM94].

2.2 An Intelligent Agent Approach

In the recent past, the field of software engineering has witnessed the emergence of agent based computing. INFOrmer belongs within a specific category of these agent systems which has come to the fore very recently: ***intelligent agents*** [Mae94]. These are agents which embody techniques derived from the field of artificial intelligence (AI) such as machine learning, adaptation and user modelling. Intelligent agents have found use in such diverse areas as VLSI design, user interface design, mail routing and network management. The basic assumption is that a software agent acts on behalf of the user — embodying his/her beliefs, intentions and goals — behaving as an intermediary between the user and the system with which he/she is interacting. The agent adapts to a user's changing needs using the AI techniques listed above.

Intelligent agents have been advocated and developed for information locating, routing and filtering, particularly on the Internet [EW94]. Maes et al. have designed some agents which she specifically employs for News filtering [Mae94]. INFOrmer differs from this system and others in several significant ways. Other approaches have been largely concerned with simple keyword searching, e.g. making the assumption that a News article will contain a SUBJECT entry containing the important keywords to appear in the text. We would claim that this entry is generally either non-existent or is inadequately filled in. The user profile in INFOrmer is based on a more comprehensive and semantically rich analysis of relevant and incoming articles, and considers the context of terms occurring in the text rather than just their frequency of occurrence. Finally, adaptation in previous models has been somewhat simplistic due to the simple structure of the user profile — INFOrmer employs more sophisticated hybrid learning strategies.

3 System Architecture

We discuss here a prototype of INFOrmer, which was been constructed and tested on a sample user population. We are currently tuning and evaluating it in a formal setting and would ultimately see this prototype being developed into a more comprehensive and robust package.

Figure 1 depicts the overall high-level architecture of the system[1]. The initial user profile is constructed through his/her presenting a set of news articles, or any other text documents, which are considered relevant. Because the initial set of relevant articles and, more importantly, the incoming articles will comprise free-text documents, a ***natural language preprocessor*** is required for early morphological analysis. From the initially presented documents, a ***user profile*** is produced which acts as a representation of a user's interests and can serve as an index into the set of subsequently received articles. In reality, a number of user profiles may exist for each individual, corresponding to a set of separate interests that he/she might have. Each incoming document must also be analysed to produce a ***document representation***. Once this representation is complete, it can be compared with the user profile to determine the likely relevance of the article to the user. The results of this comparison are presented to the user via a ***user interface agent***, through which the user also returns relevance feedback which is processed as training data.

3.1 Natural Language Preprocessor

The primary use of this module lies in the analysis of incoming documents prior to the construction of a user profile or document representation. It essentially comprises a lexical analyser, a stopword removal algorithm for noise reduction, and a stemming algorithm.

The lexical analyser tokenises the input file, extracting words, dealing with punctuation and expanding acronyms. Next a sentence boundary disambiguation is performed on the articles so as to isolate individual sentences. Given the well established fact that the resolving power of significant words in an article follows a hyperbolic distribution [Luh58], stopword removal is applied to remove the high-frequency words, while the low-frequency words will be naturally filtered out by the approach taken. We use Lovins' stemming algorithm to strip inflectional and derivational word endings. Research in information retrieval has shown that the employment of a stemming algorithm increases recall [Pai94].

3.2 User Profile Representation

An associative network approach is applied for the representation of user profiles. An associative network is constructed containing as nodes the primary terms, or words, in which a user is interested and organising these terms into relevant phrases through a set of weighted links.

Associative networks differ from the semantic networks used widely in AI and cognitive science. Semantic networks have different generic link types such as synonymy, superclass-subclass, and also possibly disjunctive and conjunctive sets of links. Contrasting with

[1] While we would view our system in the context of it being an intelligent agent, it is in fact comprised of a society of sub-agents For example, we have separate agents acting as interfaces to the user and to a system-resident news server.

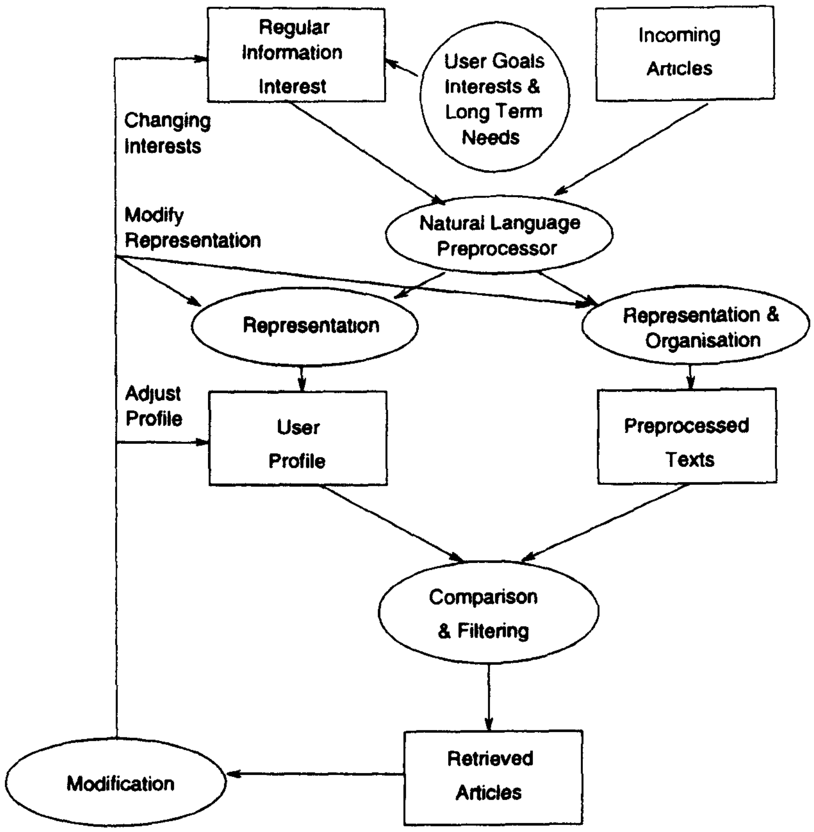

Figure 1: The Agent Architecture

this, associative networks (somewhat like artificial neural systems) have only a single link type, a weighted edge, the semantics being implicit in the structure of the network and the parameters associated with the processing.

The networks are constructed by first doing the preprocessing described above. Initially, each sentence is viewed as a chain of nodes linked by edges. Terms that occur in the same article more than once are merged into the same node if the words around them satisfy some measure of similarity. This similarity judgement is necessary because of the problem of ***homographs*** (words with the same spelling, but different meanings). A more extensive natural language processing capability could recognize phrases in a more robust fashion by recognising syntactic relationships, such as active and passive verb constructions, conjunctions, prepositional phrases, etc. We are investigating the use of a part–of–speech tagger in this regard. The rules used in the graph construction were arrived at empirically to ensure that a scalable scheme was chosen. The current set of rules, based on graph node neighbourhoods, has been used successfully to index both articles constituting a single sentence and articles with thousands of lines. The links have a certain fixed initial value (held by a system parameter). These values are later adjusted during the profile adaptation phase. In this way, the graph models the relationships between terms, both direct and indirect, and captures these in an appropriate context.

Part of a profile structure, constructed from the text in the example paragraph below, and other segments of text which are not shown, is given in Figure 2. The weights on the arcs vary because of reinforcement that some of the indexing phrases have received i.e. **system** and **problem** have reoccurred across the articles more often than the indexing phrase **neur** and **system**.

The main principles of using symbolic, fuzzy and neuro systems for problem solving will be discussed and compared. Then hybrid systems will be introduced. A hybrid environment will be used for demonstration and practical examples will begiven as illustrations. Different techniques for solving difficult problems in a hybrid environment will be demonstrated. Neural and fuzzy systems can compliment each other very well.

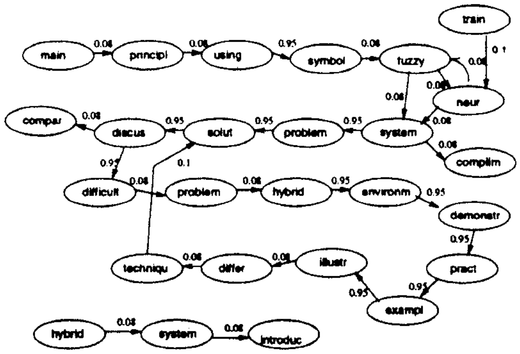

Figure 2: Example profile network

3.3 Document Representation

Again, an associative network is used here, with the words and phrases of each incoming article being organised into a graph–like structure. A document representation differs from a user profile in two respects: it uses unweighted links, since the occurrence of a phrase in an incoming article is not a priori known to be significant; and its nodes, representing terms, have activation levels associated with them which are initially set to zero but are adjusted during the comparison process.

3.4 Profile–Article Comparison

The comparison of a user profile with a document representation involves the localised matching of structural similarity between the profile network and incoming article networks, using profile weights to influence this comparison. Two basic properties of graphs can be singled out when graph comparison is the issue: *paths* and *neighbourhoods*[2]. Since we are predominantly concerned with identifying phrases in articles, associations within neighbourhoods are deemed the priority. This fact is reflected in the algorithm used. INFOrmer uses a general mechanism for measuring the similarity of two labelled directed graphs. The measure chosen is sensitive to structural information in addition to just the node contents.

Goldsmith and Davenport outline eight different algorithms for comparing labelled undirected graphs [GD91]. Here the assumption is made that the graphs are always composed of a common set of labelled nodes, but it is easy to relax this assumption as we do. Four of the algorithms are based on paths and four on neighbourhoods. For the algorithms based on paths, the key data is the distance between the two nodes v and v' in a graph $G = (V, E)$, which is defined as the minimum path length for all paths between v and v' provided a path exists, call it $\delta_G(v, v')$ say, for a graph $G = (V, E)$.

But we are more interested in the algorithms (C_1, C_2, C_3 and C_4) that are based on neighbourhoods. Note that Goldsmith and Davenport restrict themselves to neighbourhoods constrained by a path length of 1, i.e. let $\alpha_G(v, v') = 1$ if $\delta_G(v, v') = 1$, and 0 otherwise.

One of these algorithms gives the index of similarity for a common node in the two graphs as the cardinality of the intersection of the nodes' neighbourhoods divided by the cardinality of the union of the neighbourhoods. Let G_v denotes the set of nodes v' such that $\alpha_A(v, v') = 1$. Formally

$$C_1(A, B) = \frac{1}{n} \sum_{v \in V} \frac{|A_v \cap B_v|}{|A_v \cup B_v|}$$

where $A(V, E_1)$ and $B(V, E_2)$ are graphs with common node set V of cardinality n, v being a node in V. The measures C_2 and C_3 differ only in the normalization used. C_4 differs in that it is based on the number of edges that match, divided by the total possible number (which can be computed as $n^2 - n$).

$$C_4(A, B) = \frac{1}{n^2 - n} \sum_{v \neq v'} \alpha_A(v, v') \bigtriangleup \alpha_B(v, v')$$

$\bigtriangleup$ is defined as follows: $a \bigtriangleup b = 1$ if $a = b$, a/b if $a < b$ and b/a if $b < a$. This acts like a symmetric difference operator and implements a comparison of the nodes surrounding a particular node in two different graphs.

INFOrmer's similarity measure is based on C_4, which is extended to take profile edge weights into account. Let $A(V, E_1)$ be the profile. Now $\bigtriangleup$ has to be redefined. Given the sparseness of the data available in a text filtering environment, when $a = b = 0$ we want $a \bigtriangleup b = 0$, but the denominator $n^2 - n$ needs to be correspondingly decreased by 1 for each such occurrence. This focusses the measure on those relations that are actually present in the profile and incoming article data. Also if $a = b = 1$ then, to involve profile weights in the comparison, we have $a \bigtriangleup b = w(a)$ where $w(a)$ is the weight on the profile edge operated on by a. The definitions for a/b and b/a remain the same.

This comparison mechanism for graphs thus captures the fact that a phrase in a document that also occurs in the profile is an important signifier of the relevance of the articles to the profile. Thus we have a computationally tractable mechanism for recognising phrases of arbitrary length. Specifically, the relevance of a received article — as depicted by its similarity measure — depends on three factors:

- the frequency of occurrence of certain phrases within the article.
- the relevant importance of those phrases (as depicted by their profile weights).

[2] A path between two nodes, is a chain of edges connecting those two nodes. The neighbourhood of a node is defined as the set of nodes accessible from it, constrained by a specific path length.

- the relevance of these phrases based on their context within the article (due to the algorithm for constructing the graphs).

We believe that our concentration on this last issue is likely to give our system a significant advantage over the previous approaches to this problem.

We are also experimenting with a further extension of this measure, motivated by neural network approaches to information retrieval. Here, once the networks are in place, an appropriate control mechanism is required to supervise the processing — techniques such as having inhibitory connections and competitive activation have been used successfully by researchers. We use a scheme very like Mozer's [Moz84], where each unit computes the sum of its incoming activations and modulates it by its own current unit activity when it has a positive activation level. This was based on a model of word perception which McClelland and Rumelhart developed in parallel distributed processing [RM86]. The procedure is somewhat similar to spreading activation methods in semantic networks.

3.5 User Interface Agent

Those articles considered relevant to the user's needs are forwarded by the agent, while the others are screened out. Forwarded articles are also ranked according to estimated relevance. There has been considerable debate as to how such estimations should be made and presented. Note that the description of the comparison above has been simplified somewhat for the purposes of communicating the primary function of the comparison module clearly. In actuality there are three similarity measures. The scheme in INFOrmer involves the estimation of the percentage of an article considered to be 'very relevant', 'possibly relevant' and 'not relevant' to the user. Cut–off values are used for screening out articles, while these percentages are attached to returned articles. Thus the agent provides the user with a richer expression of its estimation of the incoming article's relevance, the judgment still being immediately understandable even by a novice user.

3.6 Relevance Feedback

Via the user interface, the user may provide relevance feedback on those articles routed to him/her. A tag may be attached to a received article specifying whether or not it is relevant. Based on this tag, the network weights are modified using a *vector space* (VS) relevance feedback model [SM83] so as to adapt the profile to better reflect the user's requirements.

Although it is stated above that the profile edges are weighted, these weights are calculated dynamically from weights associated with the terms in the profile. Adopting the terminology of the VS model, the profile is viewed as a vector of term weights, $\vec{P_j}$, the weight for each term being a *normalised* value calculated from a term's relative frequency in the profile and a dataset of articles typical of the newsgroup or document collection being examined. For an article D given as feedback, a term weight vector $\vec{D}$ is computed. $\vec{D}$ is used to shift the vector $\vec{P_j}$. Hopefully $\vec{P_j}$'s new position is more representative of the user's interests and needs. This is best viewed geometrically, with the vectors being points in the same multidimensional space.

$$\vec{P_{j+1}} = \begin{cases} \alpha\vec{P_j} + \beta\vec{D}, & \text{if D tagged as relevant;} \\ \alpha\vec{P_j} - \gamma\vec{D}, & \text{if D tagged as not relevant} \end{cases}$$

$\vec{P_{j+1}}$ is the new profile after this iteration of learning.

At present, the adaptation rate is parameterised by α, β and γ; so, a compromise is possible between oscillation and stagnation of user profiles. After each iteration of learning we need the new profile edge weights for use in the profile–article comparison algorithm. The translation of these term values to edge weights is done as follows:

$$w(k,l) = \frac{P_{j,k} + P_{j,l}}{2}$$

$P_{j,k}$ is the term value (or weight) for the k-th term in the profile $\vec{P_j}$; $P_{j,k}$ is the l-th. $w(k,l)$ is the new edge weight for an edge joining these two terms if an edge exists.

There is also a facility for incorporating new terms into a profile. This is crucial if the system is to be adaptive. For this we use a method similar to the techniques used in Belew's neural network information retrieval system AIR [Bel89].

4 Current System Status and Testing Methodology

Presently a working prototype of the system exists which is used to filter USENET Net News. The user interface, at present primarily text–based, needs to be improved though. We are currently building an X/Motif interface for the system, so that relevance estimations can be presented and relevance feedback given in a less obtrusive fashion.

Testing has already taken place which we believe has endorsed the approach we have taken. The same performance evaluation problems that have plagued text retrieval research also pose considerable hurdles to filtering research. Experience has uncovered the many difficulties involved in comparing information management systems, when so many factors are involved.

We have recently begun detailed comparisons between our system and other information filterers, regardless of their approach or architecture. Specifically, we wish to compare both the recall and precision of INFOrmer with those of other of other routing software. The effect of different profile adaptation rates on the performance of the system also needs to be examined. The fact that this is parameterised will make experimentation easier.

To enable meaningful comparisons to be carried out, we are one of the participating groups in this years TREC text retrieval project, which uses the TIPSTER document collection [Har94]. The TREC project is sponsored by the Software and Intelligent Systems Technology Office of the Defence Advanced Research Projects Agency (DARPA/SISTO) in an effort to advance the fields of information retrieval and data extraction from real–world document collections. Specifically, we will be using the routing environment which is concerned with retrieving information based on long–term information needs. This evaluation is now in progress and will continue over the next three months. Especially now, with the increasing application of AI techniques to information filtering, a rigor is needed in testing which we feel is absent in a lot of AI research. The testing methodologies of the TREC projects fill a gap in this respect by providing a uniform framework for experimentation.

5 Summary

In this paper, we have outlined the architecture of INFOrmer, a system used for filtering news articles based on a knowledge of a user's stated interests. We have depicted how the user's interests are represented, how incoming articles are represented, and how comparisons are made between these representations in order to evaluate the articles' relevance to the user. We have also indicated how the relevance feedback supplied by a user may alter the profile so as to model his/her changing interests. Finally, we have outlined how user testing has proved promising and that detailed performance testing is now underway both for evaluation and optimization purposes.

6 Acknowledgements

The first author was supported in part by Memorex Telex Ireland Limited.

References

[BC92] N.J. Belkin and W.B. Croft. Information filtering and information retrieval: Two sides of the same coin? *Communications of the ACM*, 35(12):29–39, December 1992.

[Bel89] R. Belew. Adaptive information retrieval : using a connectionist representation to retrieve and learn about documents. In *Annual Proc. of the ACM SIGIR*, pages 11–20, 1989.

[DDL$^+$88] S. Deerwester, S. Dumais, T. Landauer, G. Furnas, and L. Beck. Improving information retrieval using latent semantic indexing. In *Proc. of the Annual Meeting of the American Society for Information Science*, pages 36–40, 1988.

[Ell90] D. Ellis. *New Horizons in Information Retrieval.* Library Association, London, 1990.

[EW94] O. Etzioni and D. Weld. A softbot-based interface to the internet. Technical report, University of Washington, 1994.

[Fol90] P.W. Foltz. Using latent semantic indexing for information filtering. In *Proc. of the ACM Conference on Office Infrormation Systems*, pages 40–47, 1990.

[GD91] T.E. Goldsmith and D.M. Davenport. Assessing structural similarity of graphs. In R.W. Schvaneveldt, editor, *Pathfinder Associative Networks: Studies in Knowledge Organization.* Ablex Publishing, 1991.

[Har94] D. Harman. Overview of the third text retrieval conference (trec-3). In *Text Retrieval Conference (TREC-3).* National Institute of Standarda and Technology, 1994.

[Kwo89] K.L. Kwok. A neural network for probabilistic information retrieval. In *ACM SIGIR Forum*, volume 23, pages 21–30, 1989.

[LCB89] D.D. Lewis, W.B. Croft, and N. Bhandaru. Language-oriented information retrieval. *Interml. Journal of Intelligent systems*, 4:285–318, 1989.

[Luh58] H.P. Luhn. The automatic creation of literature abstracts. *IBM Jour. Res. Dev.*, 2(2), 1958.

[Mae94] P. Maes. Agents that reduce work and information overload. *Communications of the ACM*, 37(7):48–53, July 1994.

[Moz84] M.C. Mozer. Inductive information retrieval using parallel distributed computation. Technical report, University of California, San Diego, 1984. Research Report ICS-8406.

[Pai94] C.D. Paice. An evaluation method for stemming algorithms. In W.B. Croft and C.J. van Rijsbergen, editors, *Proc. of the 17th Int. ACM SIGIR Conf.*, 1994.

[RM86] D.E. Rumelhart and J.L. McClelland, editors. *Parallel Distributed Processing Vol. 1*, Cambridge, U.S.A., 1986. M.I.T. Press.

[SFW92] G. Salton, E. Fox, and H. Wu. Extended boolean information retrieval. *Communications of the ACM*, 26(12):1022–1036, March 1992.

[SM83] E.A. Salton and M.J. McGill. *Introduction to Modern Information Retrieval.* McGraw Hill International, 1983.

[TC91] H.R. Turtle and W.B. Croft. Evaluation of an inference network-based retrieval model. *ACM Trans. Inf. Sys.*, 3, 1991.

[YGM94] T.W. Yan and H. Garcia-Molina. Index structures for information retrieval. In *Proc. Internl. Conf. on Data Engineering*, pages 337–347, 1994.