A Learning Personalised Information Filter

Adrian O'Riordan

A Learning Personalised Information Filter

Adrian O'Riordan
Department of Computer Science
University College Cork
adrian@odyssey.ucc.ie

Abstract

We present here an overview of a research project which is ongoing at University College Cork. Information filtering software is being constructed which interfaces to the USENET Net News utility and which screens out irrelevant articles prior to their being presented to a user. The system 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 articles, and graph manipulation algorithms are used in the processing.

1 Introduction

Together with the much-vaunted benefits of the 'information superhighway' has come one major drawback: *information overload*. 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. Taking the Internet as an example, the increase in node numbers by approximately 9% per month suggests that the problem is rapidly escalating. 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 that we address ourselves. In particular, we aim to build an information filter 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.

As a testbed for this filtering application, we chose the USENET Net News service, which is the primary information server to Internet users. While the organisation of this service into newsgroups represents a first level of filtering, an empirical study has shown that reading just a few newsgroups requires perusing megabytes of information each month [FS91]. Clearly, then, this represents a case where accurate filtering of information might repay the user in terms of reduced browsing time.

2 Information Filtering

While the concept of information filtering is not new, the widespread access to computer networks (particularly the Internet) has heightened interest in practical filtering software. 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. [BC93]. Information retrieval (IR) has a long history, the manual indexing of books and documents in libraries being a well-known example. As computers increased in power, researchers began looking at automatic techniques for indexing and retrieving information. Today researchers in IR are well aware of the inadequacies of these approaches and powerful techniques involving *statistical analysis* and *artificial intelligence* have been developed. These 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.

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]. Queries consist of expressions involving the logical operators **and**, **or**, and **not**, with the possibility of terms being weighted. A more recent research vehicle has employed the *vector space approach* and variations thereof. The Smart system is an example of a text processing and retrieval system based on this approach [SM83]. Latent Semantic Indexing (LSI) [Fol90] 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. All of these methods, and others, have been adopted for use in information filtering. Other information filtering systems worthy of mention are Tapestry [GNOT92], a system which supports collaboration between a community of users, and the email-based News filter currently accessible at Stanford University.

3 Intelligent Agent Approach

In the recent past, the field of software engineering has witnessed the emergence of agent based computing. This may be viewed as an extension to conventional structured software design and its more modern counterpart, object-oriented software engineering, and is based on the premise that complex systems can be best viewed as a society of autonomous software agents which communicate and cooperate in order to fulfil their desired task. These agents have been shown to be particularly pertinent within distributed environments.

We are concerned with a specific category of these agent systems which has come to the fore very recently: *intelligent agents* [MCF+94]. 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 - acting 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 [Mae94].

Intelligent agents have been advocated and developed for information locating, routing and filtering, particularly on the Internet [EW94, SM93]. Maes has designed some agents which she specifically employs for News filtering [12]. We would view our system in such a light: a user agent represents a user's interests, having induced those interests from an initial set of relevant articles; this agent then filters incoming News articles, passing them on to the user or filtering them out, depending on their perceived relevance; as a result of relevance feedback (concerning received articles) on the part of the user, the agent behaviour is adapted so as to improve filtering precision and to better reflect a user's changing interests. We would see our system differing from those of Maes and others in several significant ways. Other approaches have been largely concerned with simple keyword searching, on the basis that a News article will contain a SUBJECT entry citing the important keywords to appear in the text. We would claim that this entry is generally either non-existent or is inadequately filled in. Our user profile is to be 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. The comparison techniques (between the profile and an incoming article) will be considerably more complex than that used by Maes et al. Finally, adaptation in previous models has been somewhat simplistic due to the simple structure of the user profile - in our system, we shall employ more sophisticated hybrid learning strategies.

4 System Architecture

Development of the system has been ongoing for more than a year. An initial prototype is approaching completion. After testing and evaluation, we would see

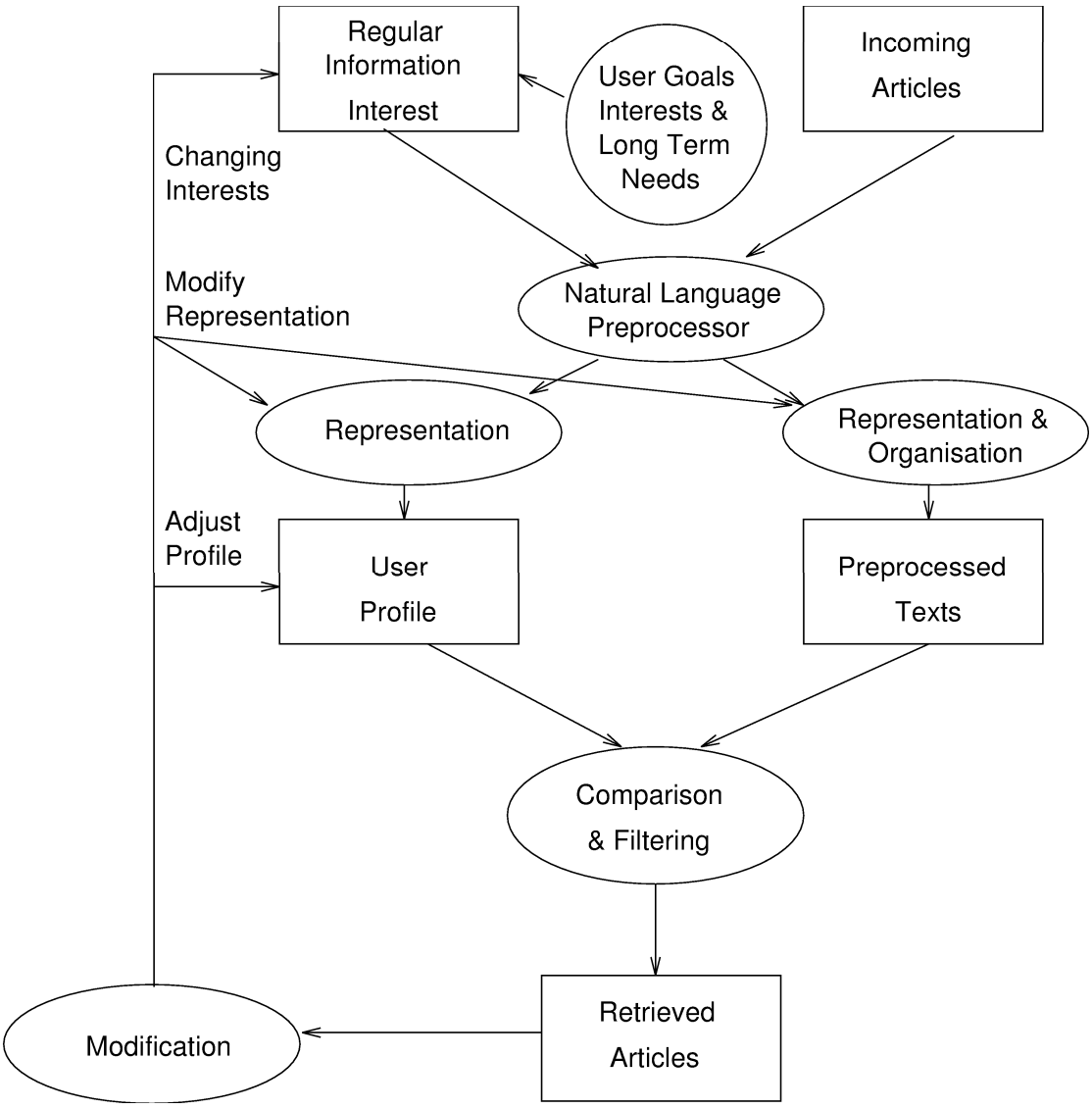

Figure 1: The Agent Architecture

this prototype being developed into a more comprehensive and robust package. Nevertheless, we do not envisage the finished product deviating much from the prototype architecture described below.

Fig. 1 depicts the overall high-level architecture of the system[1]. Because the initial set of relevant articles and, more importantly, the incoming articles will comprise free-text documents, we require a *natural language preprocessor* for early morphological analysis. From the initially presented documents, we produce a *user profile* which acts as a representation of a user's interests and can serve as an index into the set of subsequently received documents[2]. These incoming documents must

[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 the system resident news server.

[2]In reality, a number of user profiles may exist for each individual, corresponding to a set of

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. This *comparison* involves techniques derived from memory-based reasoning [SW86, Kol84]. The results of this comparison are presented to the user via a *user interface agent*, through which the user also returns relevance feedback as to the accuracy of the system. This relevance feedback then results in modifying the user profile.

4.1 Natural Language Processor

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 stemming algorithm and a stopword removal algorithm for noise reduction.

The lexical analyser tokenises the input file, extracting words and dealing with punctuation. The stemming algorithm strips inflectional and derivational word endings. Research in information retrieval has shown that the employment of a stemming algorithm increases recall [Pai94]. We use Lovins' algorithm [Lov68]. Also, it is a well established fact that the resolving power of significant words in an article follows a hyperbolic function [Luh58]. If the distinct words are ranked by frequency, words with very low or very high frequency are poor indicators of the subject matter of an article. Stopword removal is aimed at the removal of high-frequency words, while the low-frequency words will be naturally filtered out.

4.2 User Profile Representation

We have adopted a connectionist approach to the representation of user profiles. A connectionist 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.

Connectionist 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, association and also possibly disjunctive and conjunctive sets of links. Contrasting with this, connectionist networks (best exemplified by artificial neural systems) have only a single link type, a weighted edge. The semantics in this case are implicit in the structure of the network and the parameters associated with the processing.

In our system, the networks are constructed by first doing a sentence boundary disambiguation on the articles so as to isolate individual sentences. Next, 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

separate interests that he/she might have.

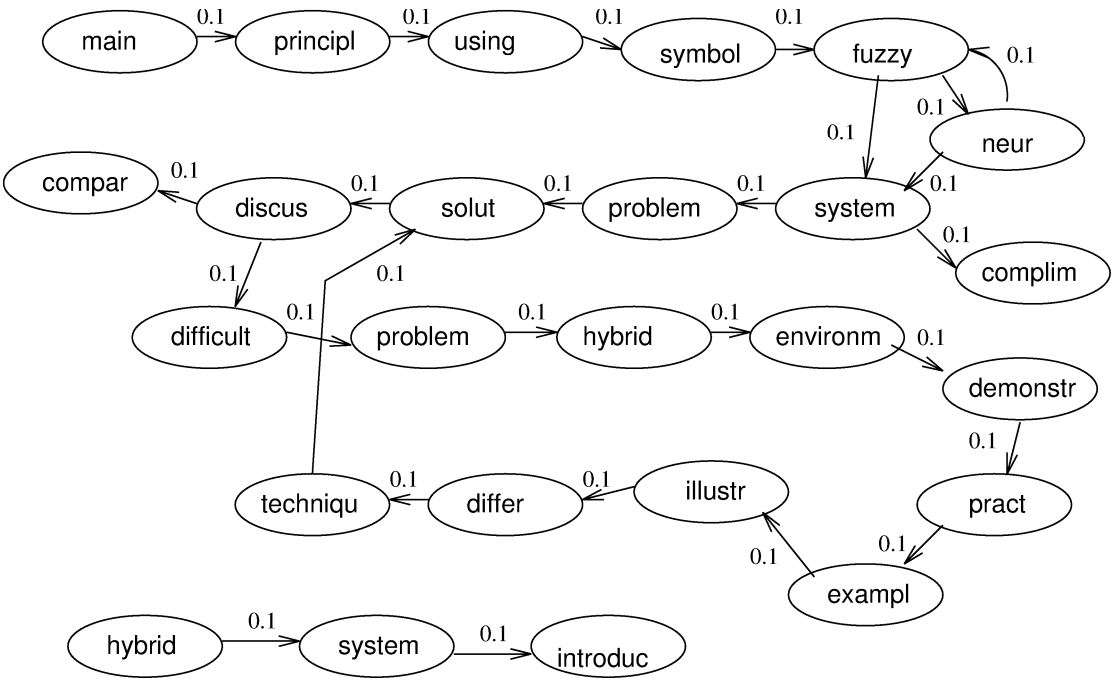

Figure 2: A network representation for free-text

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 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, our graph models the relationships between terms, both direct and indirect, and captures these in an appropriate context.

4.2.1 The Process in Motion

A short example will explain the representation more clearly. A paragraph of text, taken from an article on the **comp.ai.neural-nets** newsgroup is taken as the input to the graph constructor for our purposes here. Actually the full-text would be taken, but the representation generated would be too large for illustration of anything much larger than a paragraph. 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 text for the example paragraph is given below:

```
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 demon-
```

```
stration and practical examples will be given 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.
```

This 62–word section of text is converted into the 22–node graph shown in Fig. 2. Empirical trials show that the number of nodes generated for an article representation is typically 30–40% of the original number of words. This is a figure similar to the number of distinct words in an article and is not unreasonable on computer memory requirements in comparison to other indexing schemes. Initially all edge values are set to 0.1, as can be seen in Fig. 2.

4.3 Document Representation

Again, the connectionist approach is adopted 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.

4.4 Comparison

The comparison of a user profile with a document representation then involves localised matching of structural similarity between the networks using profile weights to influence the activities at incoming article graph nodes. We are essentially employing a *spreading activation model*, based on labelled graph comparison mechanisms.

Two basic properties of graphs can be singled out when graph comparison is the issue: paths and neighbourhoods[3]. 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.

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. We use a scheme very like Mozer's [Moz84]; here, 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 Rumelhardt developed in parallel distributed processing [RM86]. Other variations on spreading activation can be found in [DRL90].

The activities at nodes thus represent the fact that a phrase in a document exists in the profile or its contextual words do. Specifically, the relevance of a

[3]The neighbourhood of a node is defined as the set of nodes accessible from it, constrained by a specific path length.

received article — as depicted by its graph activation levels — 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)
- the relevance of these phrases based on their context within the article

We believe that our concentration on this last issue is likely to give our system a significant advantage over the previous agent-based approaches cited above.

4.5 User Interface

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. We have chosen a mechanism whereby we estimate 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.

4.6 Relevance Feedback

Again employing 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 the *vector space* relevance feedback model [SM83] so as to adapt the profile to better reflect the user's requirements.

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 being examined. An article, D, given as feedback is also viewed as such a term weight vector $\vec{D}$. $\vec{D}$ 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.

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

At present, the adaptation rate is parameterised by α,β and γ; so, a compromise is possible between oscillation and stagnation of user profiles.

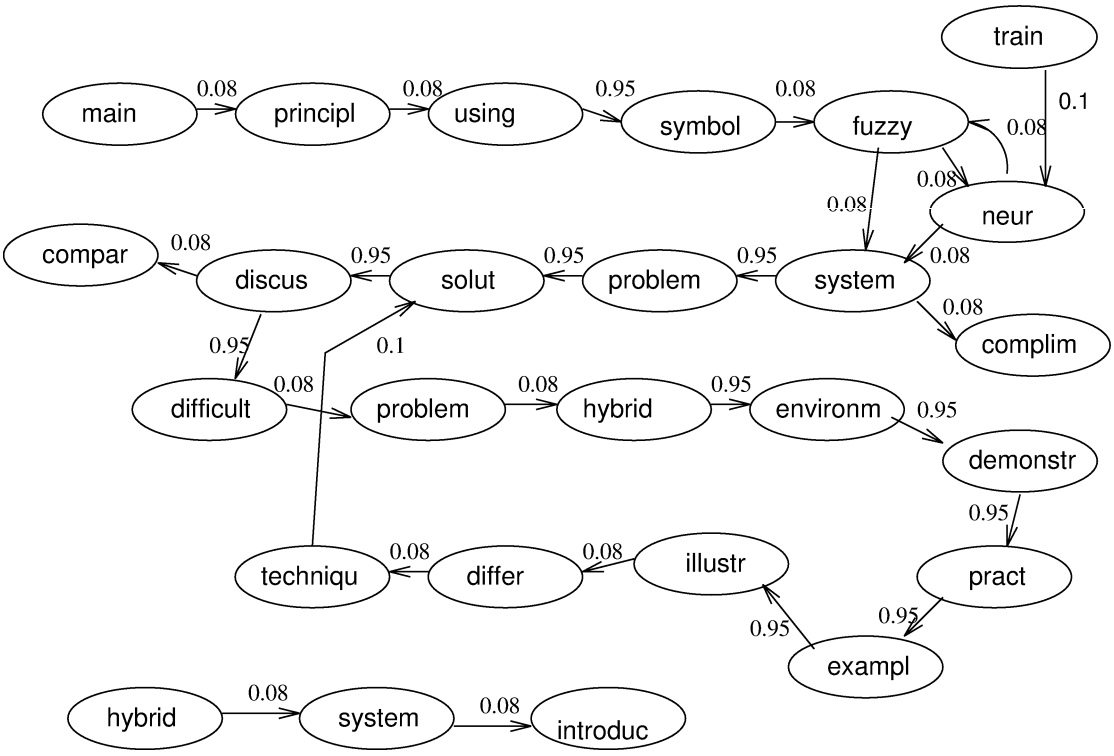

Figure 3: Profile network after one iteration of learning

4.6.1 The Short Example continued...

We now let the paragraph of text given earlier be a user profile. We take another paragraph from another article in the same newsgroup, **comp.ai.neural-nets**, and offer this as relevance feedback. The text of this is given here:

```
I have been training neural nets (using BP) to predict typing errors.
My approach thus far has been simple:  use all the data available (split
into equal training, validation and testing sets), throwing it all at
the network and seeing how well it can cope.  I have also considered
using different training techniques like genetic algorithms or possibly
a hybrid system.
```

Fig. 3 shows the profile graph with the edge weights adjusted. Note also that a node, with the label "train", has been automatically added to the profile. An appropriate place was found using the method outlined previously. This addition was caused by the repeated occurrence of the word `training` in the passage given as feedback. Also note how some edge weights are updated from 0.08 to 0.95: this resulted from the computation of $\alpha\vec{P}_j + \beta\vec{D}$, and the translation of these term values to edge weights.

$$\mathcal{E}_{k,l} = \frac{P_{j,k} + P_{j,l}}{2}$$

$P_{j,k}$ is the term value (or weight) for the k-th term in the profile $\vec{P_j}$; $P_{j,k}$ is the l-th. $\mathcal{E}_{k,l}$ is the new edge weight for an edge joining these two terms if an edge exists. Negative feedback works similarly, with edge weight decreases. A mechanism for the removal of nodes with very low edge weights was programmed, so old interests and information 'mistakingly' put in the profile can be pruned. Some reorganisation of the network is necessary here in the implementation.

5 Current and Projected System Status

At present, a working prototype of the system exists. The natural language processing component, network constructor and comparator, relevance estimator and relevance feedback module all function satisfactorily, together with the interface agent to the Net News service. Testing has already taken place which we believe has endorsed the approach we have taken.

We are preparing to carry out detailed comparisons between our system and other information filters, regardless of their approach or architecture. We also need to examine the effect of different profile adaptation rates on the performance of the filter. The fact that this is parameterised will make experimentation easier. To enable meaningful comparisons to be carried out, we intend to use the TIPSTER document collection [CC93], for which known statistics exist. The TIPSTER 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 scheduled to take place over the next three months. As a result, we may need to fine-tune certain parts of the current prototype.

The user interface, at present primarily text-based, also needs to be improved. We intend to place an X-windows interface on the system, so that relevance estimations can be presented and relevance feedback given in a less obtrusive fashion. In doing this, we shall heed the advice of [KSC94], whose architectural principles for bottom-up agent design has already been taken on board.

Finally, we intend to investigate the issue of *social learning*, and how it might benefit a system such as this. This feature obtains when learning is not limited to isolated user profiles, but can take into account similarities between profiles of different users (e.g. for users organised into work-groups), traffic analysis, etc. Some initial investigation into this learning strategy has taken place within the Tapestry project and by Maes, who investigated the role of serendipity in such systems. More specifically we are investigating the k-means clustering algorithm to group user profiles according to the topics represented. Users would then have the option of incorporating terms from profiles in the same cluster into their own to increase recall. Users need not be aware of the profiles of other users with whom they are grouped. While we recognise that arguments relating to security and privacy might arise here, we believe that the topic is worthy of research.

6 Acknowledgements

I would like to thank Humphrey Sorensen for many helpful suggestions on earlier drafts.

References

[BC93] 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 1993.

[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.

[CC93] J.P. Callan and W.B. Croft. An evaluation of query processing strategies using the tipster collection. In *Proc. of the 16th Int. ACM SIGIR Conf.*, June 1993.

[DRL90] T.E. Doszkocs, J. Reggia, and X. Lin. Connectionist models of information retrieval. In M.E. Williams, editor, *Annual Review of Information Science and Technology*, volume 25, 1990.

[EW94] O. Etzioni and D. Weld. A softbot-based interface to the internet. *Communications of the ACM*, 37(7), July 1994.

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

[FS91] G. Fischer and C. Stevens. Information access in complex, poorly structured information spaces. In *Human Factors in Computing Systems, CHI*, 1991.

[GNOT92] D. Goldberg, D. Nicols, B.M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12), December 1992.

[Kol84] J. Kolodner. *Retrieval and Organisation Strategies in Conceptual Memory.* Lawrence Erlbaum Associates, 1984.

[KSC94] H.A. Kautz, B. Selman, and M. Coen. Bottom-up design of software agents. *Communications of the ACM*, 37(7):143–147, July 1994.

[Lov68] J.B. Lovins. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 1:22–31, March 1968.

[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.

[MCF+94] T. Mitchell, R. Caruana, D. Freitag, J. McDermott, and D. Zabowski. Experience with a learning personal assistant. *Communications of the ACM*, 37(7):80–91, 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.

[SM93] B. Sheth and P. Maes. Evolving agents for personalised information filtering. In *Proc. of the 9th IEEE Conference on Artificial Intelligence for Applications*, 1993.

[SW86] C. Stanfill and D. Waltz. Towards memory-based reasoning. *Communications of the ACM*, 29(12), December 1986.

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