Autonomous Document Classification for Business

Chris Clack

Dept. Computer Science
University College London
Gower Street
London WC1E 6BT

C.Clack@cs.ucl.ac.uk

Jonny Farringdon

Dept. Computer Science
University College London
Gower Street
London WC1E 6BT

J.Farringdon@cs.ucl.ac.uk

Peter Lidwell

University College London
c/o Friends of the Earth
26-28 Underwood Street
London N1 7JQ

peterl@foe.co.uk

Tina Yu

Dept. Computer Science
University College London
Gower Street
London WC1E 6BT

T.Yu@cs.ucl.ac.uk

Abstract

With the continuing exponential growth of the Internet and the more recent growth of business Intranets, the commercial world is becoming increasingly aware of the problem of electronic information overload. This has encouraged interest in developing agents/softbots that can act as electronic personal assistants and can develop and adapt representations of users information needs, commonly known as profiles.

As the result of collaborative research with Friends of the Earth, a leading environmental campaigning organisation, we have developed a general purpose information classification agent architecture and are applying it to the problem of document classification and routing. Collaboration with Friends of the Earth allows us to test our ideas in a non-academic context involving high volumes of documents.

We use the technique of genetic programming (GP), (Koza & Rice 1992), to evolve classifying agents. This is a novel approach for document classification, where each agent evolves a parse-tree representation of a user's particular information need. The other unusual features of our research are the longevity of our agents and the fact that they undergo a continual training process; feedback from the user enables the agent to adapt to the user's long-term information requirements.

1. Introduction

Information overload is a common and pressing problem in industry, commerce and academia. A significant amount of effort has been expended to address this issue, with both academic and commercial development of agents (or "softbots") to provide information filtering.

Our work differs from others in four important aspects:

1. we use Genetic Programming to evolve parse trees as the key technology for our agents
2. we grow hybrid solutions that may emulate any combination of "standard" information retrieval techniques
3. we concentrate on long-lived agents with continuous training, using unobtrusive "natural feedback"
4. our work is aimed at routing information within organisations with large-scale information overload problems

In this paper we describe the development of a particular application, that of an Information Routing Agent which classifies and re-routes documents according to their textual content.

The paper is structured as follows: Section 2 provides the background to our research and places our requirements in context; Section 3 summarises related work; Section 4 explains our approach to the design of a document filtering application; Section 5 describes the internal working of our Information Routing Agent; Section 6 presents early results and discusses further work; and Section 7 concludes.

2. Background

Friends of the Earth (FOE) is a leading environmental campaigning organisation. FOE's UK activities are geographically distributed and it has formal links with FOE International offices across the world. The information requirements of the head office, the regional offices, and the international offices, are diverse and a large variety of information must be efficiently exchanged and shared between them. FOE responds to thousands of inquiries

each week from the public, press, teachers, academics, industry and government. Timeliness and accuracy of information is essential to FOE's strategic objectives.

Until recently FOE has used mainly paper-based information management systems. This is becoming unmanageable: research documentation alone currently totals about a million pages and is expected to double in the next three years. To manage this growth FOE plans to introduce a more efficient information management system involving innovative use of IT; the involvement of University College London (UCL) via a collaborative research project is an essential part of this plan.

2.1 The Working Environment

FOE's activities at head office are distributed between several groups each specialising in a different research area. Each group organises its own information and uses its own cataloguing and indexing procedures. Whilst this structure is essential to support the development and application of specialist knowledge and skills, there are occasions that require the sharing of information. The strong culture of mutual support at FOE means that the different groups are highly co-operative in sharing information, yet the process is very labour-intensive. Furthermore, the present system is highly reliant on individual knowledge of what information is available and where and how it is stored. If a person is ill or leaves, that knowledge is lost. The existence of information is not always clear and duplication (including the associated unnecessary costs) is a potential problem.

The initial perceived need is for information management systems that are not dependent on individual knowledge; however, the system must continue to support the fundamental need for each research group to determine the relevance or otherwise of any given document in a different way.

2.2 Classification and Routing

Our approach is to file inbound documents (including email) centrally, with a standard system for filing and cataloguing, but with the additional development of autonomous classification agents which will automatically re-route new documents to the appropriate research group. What is actually re-routed may be as simple as the location code for the document - our technique allows but does not require the physical routing of the document itself. In fact, retaining the documents in the central file store may be the best way to support sharing between the groups.

An interesting part of this system is the development of the autonomous agents. Our design has three aims:

1. to provide each research group with a percentage "confidence" value for each document, to indicate the degree of confidence with which the Information Routing System believes that this research group would be interested in that document. This is far more useful than a simple binary value.

2. to support the observation that each research group has a different perspective on the shared information which itself will shift with time. Thus, we require agents that are able to undergo continuous adaptation over long periods (perhaps the entire career of a researcher, or for as long as the research group is active).

3. to establish a general technique which can be used for a variety of documents, for example email, Web pages, Usenet etc.

Our Information Routing System consists of many Information Routing Agents per research group. All of these agents are identical in structure; they differ only in that they classify documents according to the different perspectives of the different research groups. Thus, for the remainder of this paper, we shall discuss the design and implementation of just one such agent. First, however, we provide a brief survey of related work.

3. Related Work

3.1 Adaptive Agents for Information Filtering

There are a number of existing software systems which use adaptive agent techniques for information filtering. The five systems briefly outlined below illustrate the wide range of different machine learning methods all of which generate or adapt a user profile (a representation of the users information need) with the changing requirements of the user.

1. NewT - a USENET news reader - uses a Genetic Algorithm to manage a population of profiles; each profile reflects the users interest in a different subject. NewT uses the vector space model to query incoming news articles. (Sheth 1994)

2. MAXIMS - an email assistant - uses machine based reasoning (MBR) to filter email. MAXIMS stores exemplars or instances (situation/action pairs) together with priority weightings. (Metral 1993)

3. NewsWeeder (MDL) - a USENET assistant - uses Minimum Description Length to classify documents. (Lang 1995)

4. Magi - a mail interface agent - uses Decision tree algorithms. (Payne 1994) concludes that there is a need for better feature extraction techniques and notes that

using all words of the document as features will cause the search space to be too large. Payne explores the idea of prototypical learning. (Payne 1994)

5. General Magic provide a commercial email assistant which sits on a mail server and applies filters to incoming documents, routing them to interested parties. (General Magic 1996)

3.2 Information Retrieval Techniques

Traditional information retrieval (IR) techniques (Sheth 1994, Belkin & Croft 1992) rely heavily upon their own a priori representation scheme for documents, as described below. These representations can be used to generate a query (a filter) to implement the document classification requirements of a user.

Standard representation and query profiles include:

1. Statistical - vector (tuple) weighted terms. (Vector Space model) (Salton & McGill 1983).

2. Latent Semantic Indexing (LSI) - vector but each vector represents a "concept". (Dumais, Furnas, Landaver & Harshman 1990)

3. Probabilistic - allows feature detection to be embedded within a probability framework. (Belkin & Croft 1992)

4. Natural language.

Boolean - can be extended to use contextual information such as words within titles, words within abstract, words within main body of text (assuming the document has these standard fields) (Marcus 1991). A similar technique using a thesaurus is used by (Gauch & Smith 1989).

All of these traditional methods impose an a priori determination of how the information need of the user is to be structured. The choice of method therefore determines how accurately the query represents the user's information need; the method which works well for one user's classification requirements may not work well for another user.

Our Information Routing System must deal with many different groups whose requirements will change with time; thus, we aim to develop a system which can determine the user's information need and use this to select the appropriate representation for the query.

We (superficially) abandon the standard techniques because their fixed structures do not reflect the rich variety of our users' possible requirements. Rather, a genetic programming (GP) approach is taken whereby the GP can evolve hybrid techniques appropriate for the context. Thus, the traditional techniques are generalised rather than truly abandoned.

4. A New Approach to Document Filtering Agent Design

Each agent in our Information Routing System is identical in design and adheres to the following model:

1. A document is presented to an agent and the agent generates a confidence score for that document.

2. According to the confidence score and a given threshold, the agent decides whether to route the document to the user.

3. The user receives notification of an interesting document, together with a confidence value.

4. After reading the document, the user provides feedback (explicitly or implicitly) to the agent.

5. The feedback is used to improve the performance of the agent or to modify the agent's classification behavior in the light of changing user requirements.

The agent's initial behavior is derived from a training set of example document classifications for a given user.

4.1 Genetic Programming

One of the primary ways in which our work differs from others is that we use the technique of Genetic Programming (GP) to derive the query representation used by each agent. That is, we use a Genetic Algorithm to evolve a parse tree - the parse tree is a program which, when executed by an interpreter, will take a document as input and produce a confidence value as its output.

The advantage of using GP is that, by making appropriate primitive functions available to the parse-tree evolver, the system may evolve a program which uses the best query representation method for the user's information need. Indeed, the parse tree might use more than one representation. For example, a vector based system has to generate a vector query, whereas a GP system could generate a parse tree equivalent to combinations of vector query, Boolean query, and any others that are supported by the tree primitives.

With genetic algorithms the evolution/learning is used to determine the best query with respect to a rigid "DNA" representation. By contrast, a GP system can evolve both a query and a unique query representation for each context.

The parse tree allows the maximum amount of flexibility enabling the agent to evolve a good (hybrid) representation. This is a generalisation of earlier fixed representation systems, and thus results in a substantially richer representation.

4.2 Feedback for a Long-Lived Agent

Feedback is necessary for adaptation, and for long lived agents is crucial. An adaptive agent uses feedback to change the filter profile over time, driven by the user's changing and developing interests. Thus, in addition to the initial period of training, long-lived agents use feedback for repeated top-up learning.[1]

Feedback for the agent is possible through both explicit and implicit user actions. For example, the user may press buttons for the purpose of giving feedback, or the system may monitor the user's routine actions and infer feedback (described later).

Long term feedback requires the occasional archival of whole populations of agents, even though from each generation only the best agent is used. If entire populations of agents were not archived (analogous to keeping the gene pool "in cold store") then subsequent top-up learning would have to start the evolution from scratch, a very costly process.

Thus, a necessary expense of longevity is the storage requirement of keeping entire populations of agents archived, ready for top-up learning. For example, if each agent initiated by a user requires a supporting population of several hundred for the evolution process, then several distinct agents would require tens of thousands of agents to be maintained; this is not unlikely for a large institution like FOE. By today's standards of disk and memory hungry applications this is quite a modest requirement, growing in a linear relationship with the number of active agents.

Additional training of an individual agent takes place when user feedback reaches a pre-determined threshold. Top-up training is then scheduled to occur when the user is absent (usually at night), thus masking entirely any associated processing costs.

4.3 Natural Feedback

The process of collecting feedback should not impinge on the user too much, since the agents are there to support, not hinder, the user. (Imagine an employee who required you to give graded feedback on every action they took!). By observing the user's routine actions it is possible to infer feedback (Farringdon 1996b), for example the agent can monitor the user's interaction with documents and based upon those actions make assumptions as to the quality of the service it is providing. A simple example is to compare the predicted classification with the (mail) folder into which the document was refiled. Other examples of natural feedback include monitoring how frequently a document is accessed and whether it is sent on to other recipients.

Of course, some documents will fall through the natural feedback net, where no inference may be made. For example, a user might immediately delete a document because a) it was junk, or b) privacy was very important.

Such natural feedback is unobtrusive and effort-free for the user, thus a tremendously valuable technique. Additionally some accommodation should be made for users to initiate explicit feedback when they deem it necessary.

5. Information Routing Agent

The overall design of the Information Routing System is given in Figure 1. Each agent is a program which interprets a genetically-evolved parse tree in order to give a confidence value to a document.

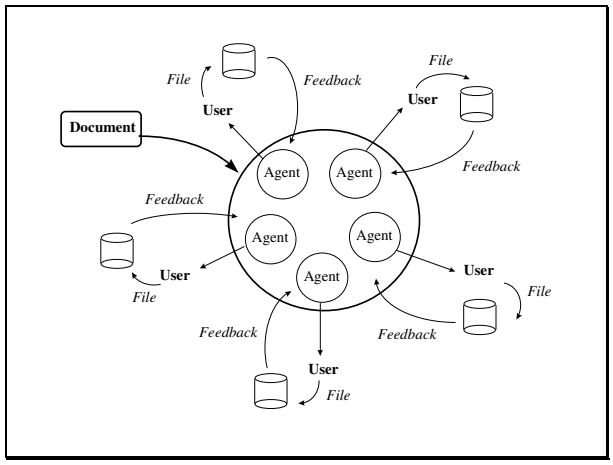

Figure 1. Design of the Routing Agent Environment

Documents may arrive at the Information Routing Agent from various sources. These can include inbound email, Usenet feeds, documents found by other searching agents, and so forth. In a large organisation like FOE individual research groups would also offer their internally generated documents or documents they had found during the course of their work to the routing agent for redistribution.

[1]Drawing an evolutionary analogy, Darwin is often misquoted with "survival of the fittest". Rather, he refers to survival of species most able to adapt.

The routing/filtering agents can monitor any central document repository. A special case of the routing agent is the case where the agent monitors the mail spool of one user; effectively the router becomes an email assistant. This mechanism is discussed in detail in (Clack et al 1996).

5.1 Parse Tree representation

In this system a parse tree is a program that can be evaluated through the use of an interpreter which matches nodes of the tree to aspects of a document. During evaluation against a document the tree ultimately reduces to a single numerical value - the classification or "confidence value". The method used here has operators at each branch of the tree, all of which produce a number. Leaves of the tree (terminals) are nodes containing word operators and arguments (words) or numerical-constant terminals. Thus any part of a tree can be interchanged with another part and the tree remains valid - perfect for flexible evolution and mutations. An example parse tree is shown in Figure 2.

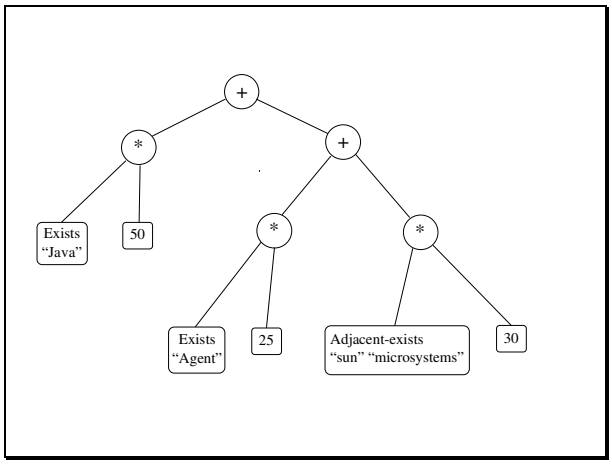

Figure 2. A Parse Tree Representation

One drawback to this method is that word operator nodes of the tree are strongly bound to their arguments, so that evolution operators can not easily (in this numeric based approach) manipulate the words in a tree independently of the word operator. For example, a leaf of the parse tree may evaluate to the frequency of "complaint". During evolution it is not a simple matter to exchange the word "complaint" for another word, rather the node frequency-of-"complaint" is exchanged in its entirety. This ridged approach is useful because an evolutionary exchange of a word operator may not be for another leaf - but for a complete sub-tree.

The parse tree operators all reduce to a single (real) number. Thus a tree of these operators reduces to one numerical value also. The available operators are listed below. It should be noted that while the following operators are made available to the system there is no requirement for any particular one to be used.

Feature Detectors: Word Operators

- frequency {word}.

Relative frequency (absolute-frequency / document-length) gives a comparable measure between documents.

- exists {word}(== frequency > 0).

Returns 0 for false and 1 for true. Testing for the existence of an individual word gives the same power as many other systems which use key-word-search and match for classification. Thus the evolved parse tree can be at least as powerful as a key word search.

- pair-distance {word , word}.

Mean distance between words in the document. The order that the words are used in the document should not matter, for example "search space" and "space to search" would both figure in the result of pair-distance(search, space).

- adjacent-exists {word , word}.

Returns 0 for false and 1 for true. Adjacent words exist in the document. Pairs of words are often their own units, for example {search space, world wide}.

- adjacent-frequency {word , word}.

Mean adjacent-words frequency. Pairs of adjacent words are important discriminators (more complex than single word discriminators).

Binding Groups of Features: Numeric Operators

- AND, OR, NOT.

Boolean operators. Interpret zero as false, non-zero as true. Return 0 for false and 1 for true.

- {+, -, *, /, =, <>, <, >=, Max, Min}.

Standard numerical and relational operators. The relational operators return 0 for false and 1 for true. The relational operators might be thought of as being used to compare the current document with learnt discriminators. The numerical operators bias the importance of branches of the tree and combinations thereof.

5.2. GP Evolution

In this section we briefly review the method by which we evolve the parse tree which the agent uses to effect the document classification.

We use standard GP evolution, starting with a random population of parse trees. All parse trees are run against all documents in the training set, with a fitness function selecting the best fraction of the population; the next generation is made up of these parse trees plus a certain number of parse trees generated using mutation and/or crossover. This standard method roughly approximates natural genetics except that in natural genetics mutation does not necessarily occur during breeding but may also occur during an organism's lifetime.

The fitness function takes the confidence value produced by a parse tree when given a certain document as input; the resulting confidence value is then compared with the expected value for that training document. The parse tree is then given a score for that document which is the square of the difference between the expected and actual confidence values. This process is repeated for all documents in the training set. The fitness score given to that parse tree is then the sum of all the scores obtained for all the documents in the training set. Thus, a parse tree giving perfect classification would have an overall score of zero. As a parse tree gets increasingly worse at classification (in some way) the overall score increases.

5.3 A Worked Example

This section illustrates how an information routing system would work in practice at Friends of the Earth.

Upon being installed, the first task undertaken by an agent is to generate an initial population to produce a stable first solution classifier (parse tree).

1. An existing store of pre-classified documents is required as a training set, commonly existing training sets include e-mail folders, web bookmark lists, and specific document directories. From these documents a set of key words is required to seed the initial populations. One method capable of generating such a set is to compare the relative frequencies of words (from the entire training set, or from individual documents) with their frequency in common usage (Quinlan 1992). Distinguished words will have maximum or significant deviation from common usage.

2. With a set of distinguished words prepared for seeding the first population, a significant period of learning is undertaken. This is exactly the same evolutionary learning process that takes place for top up learning (step 8) but is *ab abnito*. It is possible this process could take many days of background (low priority) processing. Impatient users could curtail this at any time, and allow top up learning to bring the agent up to satisfactory performance levels.

At this point the agent is ready for normal operation, the rest of this example considers the chain of events triggered by one particular document coming to the attention of the pool of similar agents situated at Friends of the Earth (FOE).

3. Tina e-mails a copy of this paper to FOE, where Peter has 3 agents on the lookout for articles pertaining to (i) autonomous agents, (ii) text filtering, and (iii) environmental issues.

4. The document is pre-processed, removing noise and reducing the search space. This is a typical first step in classification, and the precise actions of this agent are described in (Clack et al 1996).

5. Each agent takes the document as data, and using its parse tree returns a confidence score.

6. Peter is informed that his agents have an interesting candidate document, with the classifications {Text Filter: 90, Agents: 70, Environmental: 60}.

7. His curiosity aroused, Peter reads the paper and files it under "agents". The act of filing is sufficient natural feedback for the system to queue some top-up learning. Top-up learning is carried out to improve upon sub-optimal performance when enough new exemplars are available. If Peter had enough time to give explicit feedback, he could have hit a few buttons on an agents GUI to let it know that the environmental classification was way off the mark.

8. Hidden from any user, usually during the night, the agent can continue its evolutionary learning upon the parse trees, triggered by the feedback 7. using the new documents as additional training data.

6. Results and Further Work

A simple prototype has been tested on small test documents, taking about 6.5 minutes to learn to distinguish with 95% accuracy between test cases, and less than a second to provide a confidence score when applied to a larger file of about 6 kilobytes (all timings were carried out on a modest Sun Sparcstation IPC). These initial results are very encouraging and we are therefore pursuing the technique further; in particular, we need to pay attention to reducing the internal complexity of the algorithms (Farringdon 1996a, Clack & Yu 1996) in order to deal with

the extremely high numbers of large documents used every day at FOE. Our final aim is for a system that may take many days for the initial learning process but that will thereafter characterise any given document in a matter of seconds.

We recognise that there are some problems that are very hard to overcome. Documents can contain text which will confuse any automated classification system; for example, post-scripts added at the end of email messages are often irrelevant and can be totally misleading. An obvious area of further work is to apply heuristics to protect the agent from such misleading information.

Our prototype currently uses many agents for each research group. In future we might extend this to provide multiple agents per research group, thereby enabling sub-classification.

A specific area of interest for us is a quantitative comparison of this adaptive genetic programming method against more traditional methods such as vector space analysis.

7. Conclusion

We have produced a first prototype Information Routing Agent which learns to classify documents by giving them a percentage "confidence score" according to the presence or absence of words and other word relationships such as inter-word proximity. We have taken a novel approach to information retrieval, using Genetic Programming as a means to provide more flexible and powerful query representation.

The need for an adaptive text classification agent grows every day (in proportion to the size of the Internet and the increased use of on-line documents). Our early results show the viability of implementing such an agent using genetic programming methods. An agent capable of initial and continued long term top-up learning would be of great benefit to a considerable population of computer users, and is within the bounds of current machine learning technology.

Acknowledgments

This work has been partially funded by a Teaching Company Scheme grant from the Department of Trade and Industry (DTI) in conjunction with Friends of the Earth (FOE). We wish to thank both the DTI and FOE for their generous support.

References

Belkin, N., Croft, W. 1992. Information Filtering and Information Retrieval: Two Sides of the Same Coin? *Communications of the ACM*, Vol. 35 No. 12, 29-38.

Clack, C., Farringdon, J., Lidwell, P., Yu, T. 1996. An Adaptive Document Classification Agent, Research note, RN/96/45, Dept. of Computer Science, University College London.

Clack, C., Yu, T. 1996. Performance-Enhanced Genetic Programming, Research note, RN/96/116, Dept. Computer Science, University College London.

Deerwester, S.,Dumais, S.T., Furnas, G.W., Landaver, J.K., Harshman R. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, Vol. 41, No 6, 391-407.

Farringdon, J. 1996a. Genetic Programming Science, Internal note, IN/96/05, Dept. of Computer Science, University College London.

Farringdon, J. 1996b. Natural Feedback for Autonomous Agents, Internal note, IN/96/07, Dept. of Computer Science, University College London.

Gauch, S., Smith, J.B. 1989. An Expert System for Searching Full Text. *Information Processing and Management*, 25(3), 253-263.

General Magic, 1996. URL: http://www.genmagic.com/

Koza, J.R., Rice, J.P.. 1992. Genetic Programming - The Movie. MIT Press.

Lang, K. 1995. NewsWeeder, Learning to Filter Netnews. In *Proceedings of the 12th International Machine Learning Conference* (ML95), pp. 331-339, San Francisco, CA:Morgan Kaufman.

Marcus, R.S. 1991. Computer and Human Understanding in Intelligent Retrieval Assistance. *Proceedings of the 54th American Society for Information Science meeting*, Vol. 28, October, pp. 49-59.

Metral, M. 1993. Design of a Generic Learning Interface Agent. BSc Thesis, Department of Electrical Engineering & Computer Science, MIT.

Payne, T. 1994. Learning E-mail Filtering Rules with Magi, A Mail Agent Interface. MSc Thesis, Department of Computing Science, University of Aberdeen Scotland.

Quinlan, P.T. 1992. Oxford Psycholinguistic Database.

Salton, G., McGill, M. 1983. *Introduction to Modern Information Retrieval*. McGraw-Hill.

Sheth, B. 1994. A learning Approach to Personalised Information Filtering. Masters Thesis, Department of Electrical Engineering & Computer Science, MIT.