AlterEgo E-Mail Filtering Agent - Using CBR as a Service

Michael A. Redmond and Beth Adelson
Computer Science, and Psychology
Rutgers University
Camden, NJ 08102
(609) 225-6122 and (609) 225-6485
redmond@crab.rutgers.edu and adelson@crab.rutgers.edu

Abstract

With the growth and commercialization of the Internet, many busy people find themselves receiving more e-mail than ever before. Thus, e-mail filtering has become an important problem. Many mailers provide filtering based on the mail header: the sender, the domain of the sender, and the subject. In the first phase of the AlterEgo project, we are developing an e-mail filtering agent that uses the content of the *whole* message, plus a model of the world and a model of the user, including their important goals. Initially, Case-Based Reasoning seemed like a promising *master* approach; messages that are similar to messages that were deemed important in the past should be important now. However, since the user's world may often change significantly and sometimes suddenly, the relevant similarity to a past message may be at a very abstract level. This will result in an expensive search for similarity. Hence, in our initial prototype, we have developed an approach that is driven by model-based reasoning, and is aided by case-based reasoning.

Introduction

Life in the modern world, the *information age*, has among its problems, the explosion of inputs – information overload. With the growth and commercialization of the Internet, many busy people find themselves receiving more e-mail than ever before. This is one aspect of a more general information overload problem faced by many people. The AlterEgo project was initiated, partially in response to this problem, at the Rutgers Wireless Information Network LAB (WINLAB). The project's long-range goal is to develop a personal assistant, managing a user's communications over multiple modes, such as e-mail, paging, voice-mail, etc. Capabilities envisioned include paging a user when they get important but unexpected e-mail or voice mail, delivering important e-mail messages via wireless communication to the currently appropriate device.

E-mail filtering was chosen as one of the important first steps on the project. This has become an important problem; many mailers provide simple filtering capabilities. This filtering is based on the mail header: the sender, the domain of the sender, and the subject (Maes 1994). Alternatively, some research efforts focus on knowledge-free inductive classification of the texts (Lewis 1995; Lewis et. al. 1996; de Kroon, Mitchell, and Kerckhoffs 1996). In the first phase of AlterEgo, we are developing an e-mail filtering agent that uses the content of the whole message, *plus* a model of the world and a model of the user, including the user's important goals. The focus has been on time critical situations, such as when a user only has five minutes to read e-mail. In these situations, it is particularly crucial to recognize *important* mail, as opposed to providing a loose filter intended mainly to avoid junk.

Initially, Case-Based Reasoning seemed like a promising approach; messages that are similar to messages that were deemed important in the past should be important now. However, since the user's world may often change significantly, and sometimes suddenly, the relevant similarity to a past message may be at a very abstract level. For instance, e-mail from the boss is frequently significant because he or she is the boss. If the user gets a new boss, the similarity needs to be based on the boss relationship, rather than the surface feature of who the sender is. Similarly, the user may change projects and have a different set of goals that they are now pursuing. Thus, a CBR-master approach would require abstract comparisons to each previous case that is to be evaluated for retrieval. This will result in an expensive search for similarity.

Hence, in our initial prototype agent, we have developed an approach that is driven by model-based reasoning, and aided by case-based reasoning. The approach involves first consulting the user and world models to determine if a message fits into one of a set of predetermined categories of important messages Then, AlterEgo assesses the importance of the message within the category. Three criteria are used to estimate the importance of the message: the benefit of a response, the impact of a delay, and the likelihood of being able to deal with the message. The determination of the impact of delay uses case-based reasoning – *repair cases* are used to predict the cost of overcoming an obstacle to a goal. Thus, this work represents a task-driven application of a previously identified CBR integration strategy - where model-driven reasoning is the *master*, and CBR the *slave*.

```
Date: Mon, 21 Jul 1997 11:54:31 -0400 (EDT)
From: "Dr Joe Smith" <drbossman@crab.rutgers.edu>
To: profdude@crab.rutgers.edu
Subject: research on machine learning

Len,
I'll be including a section on machine learning
in the brochure that includes Darren's research
on neural networks. Could you please give me a
description of your own research in this area.
A half-page will do.  Since the
audience will be primarily students and
parents, please include the motivation behind
the research (i.e., practical applications).

Please send the description to me by e-mail
asap, preferably sometime today.

Thanks,
Joe
```

Figure 1: Example Message Requiring Change in Plans.

Example

Figure 1 shows an example message [1]. In this message, the user's boss asks for something that will take more than a few minutes, and he wants it as soon as possible. This requires a change in the user's plans, since this task was unforeseen. AlterEgo will go through two stages of processing this message, classifying it into a category, then determining the actual importance within the category.

Message Classification

This is an important message to the user. It fits into the category that we have identified as messages related to "FurtherSupportersGoals". Correctly classifying the message requires several things. First, the sender needs to be recognized as the user's boss. In AlterEgo, this recognition is accomplished by looking up the sender in the user model. There, significant people to the user are listed along with the relationship. When the sender is found to be the user's boss, there certain categories of important messages or "emergencies" that this message may concern.

Second, the message has to be identified as relevant to an important goal. In this instance, it is related to the boss's goal of attracting students. With the message found to be a request relating to one of the the boss's goals, this message fits into the category of "FurtherSupportersGoals"; next the importance and urgency of the message must be determined.

[1] Any published messages are made up or edited for privacy reasons

Rating Importance

For this example message, the goal of attracting more students is an important one to the boss, the action of supplying text for a brochure plays a small part in that goal, and there are several loose connections between the boss's goal and the user's goals (such as obtain tenure, have good students to teach, etc). These factors combine in evaluating the benefit of immediate response to the message. Each of these factors are examined using knowledge of goals that is included in the user and world models.

Further, in this message, the deadline is implicitly defined through "asap". The existence of such a deadline increases the impact of a delay in responding to the message. Considering further the impact of a delay, in this instance not providing a description wouldn't stop the brochure, or stop the attraction of students, nor would it stop the user from having good students to teach or stop him/her from getting tenure; it is merely an obstacle to those goals. The impact is the cost of recovering from or getting around that obstacle. Here, CBR comes in handy as a way to estimate that cost without resorting to complicated reasoning from first principles. If a similar *repair case* can be found, that can be used to estimate the cost of a delay. For instance, there may have been a previous case in which a brochure was to be created, and the user was late; the cost included having to FedEx material to the printer, and an additional relationship-building cost, regaining the boss' trust. This previous repair case can be used to help estimate the current cost.

There may be more to determining the importance with which the message should be handled. For instance, suppose that the user receives the message in Figure 1 when he is 5 minutes away from boarding a plane to head to a conference. He may not have the time to respond with text for the brochure. Then, should this message be ignored as unimportant? Currently, we think not. Perhaps the user may be able to call and negotiate deadlines, or get a friend and colleague to act as an agent and put something together based on something the user has previously written. In other situations, the event already scheduled might be able to be skipped or delayed in order for the user to respond to the boss's request. Thus, a penalty results only if the user is unable to even read the message.

The Model

The agent model is organized around the different categories of important messages. These categories were identified via brainstorming sessions at a number of meetings regarding the entire AlterEgo project. The participants were researchers, and potential supporters from industry. The focus was on identifying types of e-mail that these potential users would like to see if they had only a short window of opportunity to read e-mail. (Adelson and Redmond 1998). Ten categories were identified; these are listed in Table 1. We antici-

Table 1: Current Categories of Important Messages Under 5-Minute E-Mail Window.

1. Don't Get Scooped By a Rival
2. Detecting an about to Expire High Value Opportunity
3. Keeping a Project Alive/Moving
4. Further Supporter's Goals to Further User's Goals
5. Bounced E-Mail
6. Family Emergencies
7. Romantic Opportunities
8. Financial Opportunities
9. Mail About Destinations User is Heading To
10. Mail About Critical Appointments I am Trying to Make

Table 2: General Pseudo-Code for Sub-Agents for Message Categories.

- Classify Message via ...
  - Pull out Sender/Receivers as necessary for Sub-Agent's category
  - From user model, determine if senders/receivers play a role in the message class
  - From the user model, identify goals of the people above that are potentially involved in determining if the message goes in the class
  - For Each Known Goal of Relevant People
    - Parse Message for Indication that it Concerns Goal
- If the message concerns identified goal(s)
  - Calculate Benefit of Immediate Response
  - Calculate Impact of Delay
  - Calculate Likelihood of Success
- Combine to Form Total Importance Score for Message

pate that other categories will be identified. However, we believe we have gotten more than the tip of the iceberg. Structured interviews with different classes of potential users are planned in order to get a more systematic and complete classification of important messages.

Each category of important message essentially has a sub-agent, whose responsibility is to look out for messages that fit in its category, and to determine the scoring for those messages that fit into its category. Together, the set of sub-agents combine to serve the purpose of the e-mail filtering agent. This approach has the benefit of being very modular; categories are handled independently of each other, so extension to more categories is easy, as is parallelization. It also means that the categories may not be mutually exclusive; more than one sub-agent could identify a message as a member of its category, thereby increasing its importance. This makes for a more important message, in general.

Table 3: User Model Structure.

- DAILY TASK: description
- TIME OF YEAR: Season (fiscal, meteorological, whatever matters)
- SCHEDULE: list of Date, Time, Title, Length, Description
- PROJECTS: (name, potential rewards (funding/talks/pubs), current goals, priority)
- GENERAL GOALS: (goal name, importance)
- CONTACTS: (name, e-mail, default-importance, phone, organization, **user's current goals with respect to them**, their projects and goals)
- FRIENDS/FAMILY (name, e-mail, default-importance, phone, **user's current goals with respect to them**, their projects and goals)
- CO-WORKERS:
  - STUDENTS:
  - SUBORDINATES: (name, e-mail, default-importance, phone, project, role, **user's current goals with respect to them**, their projects and goals, mgmnt style
  - PEERS: (name, e-mail, default-importance, phone, project, my-proj-team/no, friend/foe/competitor, heat-of-rivalry, **user's goals with respect to them**, their projects and goals **(with indicator if goal is subject of rivalry)**)
  - SUPERORDINATES: (name, e-mail, default-importance, phone, position, authority-relationship, mentor/receptive/suspicious/hostile, **user's goals with respect to them**, their projects and goals
- MACHINES OF INTEREST: (machine, default-importance)
- TOPICS OF INTEREST: topic names

Determining Category of Message

Walking through the pseudo code in Table 2, the classification of a message is a result of the next four steps.

Pull Out Senders and Receivers Different categories have different people that are relevant. For instance, in "FurtherSupportersGoals", only the sender is relevant. For "DontGetScoopedByRival", the sender and the other recipients are both relevant. The sub-agent pulls out of a given message-header the sender and/or receivers as appropriate. This first step is very syntactic.

Determine Senders/Receivers Role The second step is to look up the sender and/or recipients in the user model, to see if they could play a role in the sub-agent's category. The user model has contents as shown in Table 3. As can be seen, people significant to the user are listed along with their relationship. The user model has been knowledge-engineered by the developers for the first prototype; in the future the user will specify parts of the user model, while other parts will be inferred. For the example above, the sub-agent for "FurtherSupportersGoals looks up the sender and determines that he is the user's boss. This allows the sub-agent to continue. Other sub-agents need to perform other look-ups. For instance, the "DontGetScooped" sub-agent needs to look up other recipients of the message to see if they are rivals. Note, that this means that the user model must include the sender if a sender-dependent category (such as "FurtherSupportersGoals") is to be recognized; however, other categories do not depend on the sender being known (e.g. RomanticOpportunities, DestinationsUserIsHeaderTo).

Identify Category-Relevant Goals The third step is to look up relevant goals. As can be seen in Table 3, included in the user model are the user's goals, including goals that the user has with respect to other people and goals that are subject to rivalry with other people. The goals to be accessed depend on the sub-agent. For instance, for "DontGetScooped" the goals of interest are those that are subject to rivalry between the user and one of the other recipients of the message. Included with the goal are features that allow the sub-agent to determine whether a message refers to the goal.

Determine Message Relevance to Goals If the senders or recipients' goals are relevant to the given category, the next step is to parse the message for indicators that the message concerns those goals. Currently this is done based on matching of hand-coded keywords identified as indicative of the goal. In the long-range, we are interested in tying in either machine learning of keywords from example messages (perhaps using techniques such as in RIPPER (Cohen 1996)), or real parsing using techniques such as DMAP (Riesbeck and Martin 1986). At this point, if the message appears to be of relevance to the goals of the senders and recipients listed in the message category, then the message is classified as being in the category.

Rating Importance

Now, the sub-agent must determine the actual importance within the category. Three issues are involved in determining importance/urgency – the benefit of responding, the impact of delaying and the likelihood of success in dealing with the message. If there is little accomplished by responding, then the message must not be too important. If response can be delayed without must cost or impact, then the message must not be too urgent. And if there is little hope of the user being able to respond to the message given their schedule, then even if the message would normally be important, it is not important to deliver it *now*.

Benefit of Immediate Response As with classifying the message into a category, this varies somewhat among the different categories. Factors used in calculating the benefit of immediate response can include three things: the importance of the goal the message is about (to the sender and user, and in some categories, to other people as well), and if requests are involved, [2] how much the requested action furthers the relevant goals, and in some categories the degree of inter-relatedness of the relevant goals.

For instance, under the FurtherSupportersGoals category, the benefit of responding involves the importance to the sender as well as to the user. In fact, the above three factors are interpreted by this sub-agent as: the importance of the goal to the sender (the boss), how much the requested action would help the sender's (boss's) goal, and how much the sender's (boss's) goal further's one of the user's goals.

The importance of specific goals to the user must be specified by the user; the user also estimates the importance of various goals to people that play important roles in their world.

Impact of Delay Calculating the impact of delay is where our current use of CBR comes into play. First, the sub-agent must extract a deadline from the message or from the user model [3]. The user's schedule is also of importance, since a very tight schedule during the upcoming period of time suggests that a slip could lead to significant lateness. The potential delay has to be classified as either disabling or an obstacle to achieving each involved goal. If a delay disables (or stops) the achievement of the goal then the impact of delay equals the importance of the goal. However, if the delay is merely an obstacle to the goal, then the impact is the cost of recovering from or getting around that obstacle. Doing such a calculation could involve complicated reasoning from first principles about human affairs. Instead, we have chosen to embed CBR here – to fulfill the well-known role of saving effort through cacheing previous experience. If a similar *repair case* can be found, that can be used to estimate the cost of a delay.

The case retrieval is based on the goals and obstacles involved, as well as contexts involved. The context includes things such as where the user was, what communications media they had available, and their schedule. Schedule similarity is based on number and priorities of events on the schedules at the time. In the future we would like to add an analysis dealing with the tightness of the schedule in a more detailed way. A previous repair case can be adapted and used to help estimate the current cost.

Likelihood of Success Finally, the likelihood of Success must be calculated. As discussed in the Example section, this is still very much an open issue. It is not clear whether the request actually has to be fulfillable by the user by the deadline in order for the response to be successful. People are very resourceful and reactive and may be able to do *something* appropriate in response even if they cannot fulfill the request. Thus,

[2]Requests are to be detected using simple specialized extraction, for example, searching for "could you", "would you", "I need", etc.

[3]Deadlines for known projects are stored in the user model. When deadlines are specified in a message, we plan on using a specialized deadline extraction method. This is an open issue

currently, we only make likelihood of success low if the user wouldn't even have time to *read* the message given their schedule and the deadline.

CBR

In the current prototype, CBR is used to estimate the impact of delay in dealing with a message. A specific kind of case, repair-cases, are used in this process. Repair cases contain information about a previous problem or failure and how it was overcome. It can be used to estimate how much it would take to get around an obstacle presented by a message. Table 4 shows an example repair case. It contains the context at the time of the failure, and the actions taken to recover, including some measure of their cost. The major aspects of successfully using such repair-cases are successful retrieval, and adaptation. There are currently open issues in each of these areas; these a discussed below.

Retrieving a similar repair-case involves calculating the similarity between the current situation and the situations in which the cases in memory occurred. This is an important issue for any case-based reasoner. For AlterEgo's use of CBR, the similarity judgement includes the necessary condition that the same goal was blocked by the same obstacle, then a partial match is done using other features of the contexts: locations, equipment available, other resources available, time, day, and date, and upcoming schedule. The latter is a form of similarity judgement that has not been addressed in CBR research. Currently, the schedule similarity is only measured by how many events are on the schedule, clearly something that can be improved upon. Improvements to be made include considering how tightly the events are scheduled, and how important the commitments are. These are relevant factors in judging whether the situation a user was in before was similar to the situation in which they currently find themself. It would also be desirable to enable judging similarity based on what the scheduled events actually are/were. While the user's schedule is intended to be kept as part of a real calendar program, and thus should be kept current and have some form of title or description for commitments, reasoning about what a meeting is all about in order to judge similarity seems problematic. It is possible that examining the known roles of known participants in a meeting (e.g. meetings with the boss, with a complete project team, etc) might provide a starting point for a solution.

Estimating the cost of getting around an obstacle to a goal via CBR, involves first retrieving a repair-case from a similar situation. However, it also should include adapting the retrieved case to make up for the differences between the current situation or context and the context at the time of the previous case (Kass 1989; Hinrichs 1991; Alterman 1990; Converse, Hammond, and Marks 1989). Generally, case-adaptation involves reasoning using detailed background domain knowledge to determine what contextual differences are important, and how the case can be modified to deal with those important differences. In AlterEgo, adaptation will need to be based heavily on knowledge of human relationships, and of communication resources. Inclusion of such sophisticated adaptation would help AlterEgo more accurately assess the impact of delay, and thus it would be beneficial for AlterEgo to be able to make the best decisions about messages. Thus, inclusion of adaptation strategies is a central and very important part of future work.

Table 4: Example Repair Case.

- CONTEXT:
  - DATE/TIME: 7/31/1994 17:00:36
  - SCHEDULE:
    - 1.
      - DATE/TIME: 7/31/1994 19:00:00
      - EVENT: pack for trip
      - DESCRIPTION: for trip to Seattle for conference
    - 2.
      - DATE/TIME: 8/01/1994 10:00:00
      - EVENT: USAir Flight 734 to Seattle
      - DESCRIPTION: 6 hour plane trip - 3 hour time change
  - DEVICES AVAIL:
    - Office Workstation
    - Office Phone
  - LOCATION: Office
  - MONEY: 120.00
- GOAL: get home
- OBSTACLE: car towed
- REPAIR STEPS:
  - Find Out How to Get Car; Cost = 15
  - Take Bus; Cost = 45
  - Pay; Cost = 30
  - Get Back; Cost = 30
  - Get Replacement Money; Cost = 15
  - Dial in to Finish What Was Working On; Cost = 30

Future Work

We are preparing a plan for interviewing real users to determine their classification of important messages, and how they recognize messages as being important. This work will help to inform the research, as well as to help develop the human-computer interaction for a realistic prototype system. At the same time, the current prototype agent will undergo initial testing with semi-real messages. After initial testing, deeper reasoning (such as discussed in the CBR section above) will be added, and further sub-agents will also be developed. We expect we may identify further uses of CBR as a service to some of these other agents. Other work includes grounding the system with a parser or a machine learning approach so that not as much of a real system would have to be knowledge engineered. This effort is part of a larger project involving multiple researchers from different departments; in the long-run this work is to be integrated with their efforts. For instance, one investigator is developing a method of determining whether (and how) to send a message, given its importance and the status, and cost of various available communication devices. The importance judgements developed here could be passed on to that component (which is currently using Bayesian Networks) for a send/no send decision.

Acknowledgements

This research has been supported in part by Rutgers University, including a SROA grant to Beth Adelson. Stephan Samuel has done much of the low level programming (pre-processing messages, reading goal files, etc). The whole AlterEgo group, including Haym Hirsh, Narayan Mandayam, David Goodman, and Dick Frenkiel, has provided insight and feedback on the work.

References

Adelson, B., and Redmond, M. A. 1998. What would it take to have a personal assistant who fit in your world?. In *Proceedings of the 1998 AAAI Spring Symposium on Intelligent Environments*. AAAI Press.

Bruninghaus, S., and Ashley, K. D. 1997. Using machine learning for assigning indices to textual cases. In *Case-Based Reasoning Research and Development, Proceedings of the Second International Conference on Case-Based Reasoning*, 303–314. Providence, RI: Springer-Verlag.

Cohen, W. W. 1996. Learning rules that classify e-mail. In *Proceedings of the 1996 AAAI Spring Symposium on Machine Learning in Information Access*. AAAI Press.

Daniels, J. J., and Rissland, E. L. 1997. What you saw is what you want: using cases to seed information retrieval. In *Case-Based Reasoning Research and Development, Proceedings of the Second International Conference on Case-Based Reasoning*, 325–336. Providence, RI: Springer-Verlag.

de Kroon, H. C. M.; Mitchell, T.; and Kerckhoffs, E. J. H. 1996. Improving learning accuracy in information filtering. In *Proceedings of the International Workshop on Machine Learning - Workshop on Machine Learning Meets HCI.*

Kivinen, J., and Warmuth, M. K. 1994. Exponentiated gradient versus gradient descent for linear predictors. Technical Report UCSC-CRL-94-16, Basking Center for Computer Engineering and Information Sciences; University of California, Santa Cruz, CA.

Kolodner., J. 1993. Case-based reasoning. Los Altos, CA: Morgan Kaufmann.

Lewis, D. D.; Schapire, R. E.; Callan, J. P.; and Papka, R. 1996. Training algorithms for linear text classifiers. In *Proceedings of SIGIR '96.*

Lewis, D. D. 1995. Evaluating and optimizing autonomous text classification systems. In *Proceedings of SIGIR '95*, 246–254.

Maes., P. 1994. Agents that reduce work and information overload. *Communications of the ACM* 37(7):31–40.

Marx, M., and Schmandt, C. 1996. Clues: Dynamic personalized message filtering. In *Proceedings of the Conference on Computer Supported Cooperative Work (CSCW '96)*, 113–122.

Redmond, M. 1990. Distributed cases for case-based reasoning; facilitating use of multiple cases. In *Proceedings of the National Conference on Artificial Intelligence (AAAI-90).* Boston, MA: Morgan Kaufmann.

Redmond, M. 1992. *Learning by Observing and Explaining Expert Problem Solving.* Ph.D. Dissertation, Georgia Institute of Technology, Atlanta, GA.

Riesbeck, C., and Martin, C. 1986. Direct memory access parsing. In Kolodner, J., and Riesbeck, C., eds., *Experience, Memory, and Reasoning.* Hillsdale, NJ: Lawrence Erlbaum Associates. 209–226.

Rosch, E., and Mervis, C. B. 1975. Family resemblances: Studies in the internal structure of categories. *Cognitive Psychology* 7:573–605.

Rosch, E.; Mervis, C. B.; Gray, W. D.; Johnson, D. M.; and Boyes-Braem, P. 1976. Basic objects in natural categories. *Cognitive Psychology* 8:382–439.

Appendix

1. **Integration name/category**: AlterEgo
2. **Performance Task**: E-Mail Filtering
3. **Integration Objective**: Efficiency in amount of developer knowledge-engineering required, also expect some savings in cpu time.
4. **Reasoning Components**: Model-based reasoning for detecting message category, and for part of determining message importance. CBR for determining part of message importance (impact of delay)
5. **Control Architecture**: CBR as slave
6. **CBR Cycle Step(s) Supported:**
7. **Representations**: schemas for user models, world model. goals organized in directed acyclic graph. "Repair cases".
8. **Additional Components**: mailer.
9. **Integration Status**: Proposed, partially implemented.
10. **Priority future work**: Evaluation of effectiveness. Interviews for cognitive foundation. Deeper reasoning, particularly for similarity measurement and adaptation. Future integration with rule induction and/or Bayesian Nets? CBR as master, with model-based reasoning used for knowledge-based matching for retrieval?