Exploiting E-mail Structure to Improve Summarization

Derek Lam
IBM Research
1 Rogers Street
Cambridge, MA 02142
dsl@alum.mit.edu

Steven L. Rohall
IBM Research
1 Rogers Street
Cambridge, MA 02142
steven_rohall@us.ibm.com

Chris Schmandt
MIT Media Lab
20 Ames Street
Cambridge, MA 02139
geek@media.mit.edu

Mia K. Stern
IBM Research
1 Rogers Street
Cambridge, MA 02142
mia_stern@us.ibm.com

ABSTRACT

This paper presents the design and implementation of a system to summarize e-mail messages. The system exploits two aspects of e-mail, *thread reply chains* and *commonly-found features*, to generate summaries. The system uses existing software designed to summarize single text documents. Such software typically performs best on well-authored, formal documents. E-mail messages, however, are typically neither well-authored, nor formal. As a result, existing summarization software gives a poor summary of e-mail messages. To remedy this poor performance, our system pre-processes e-mail messages using heuristics to remove e-mail signatures, header fields, and quoted text from parent messages. We also present a heuristics-based approach to identifying and reporting names, dates, and companies found in e-mail messages. Lastly, we discuss conclusions from a pilot user study of the summarization system, and conclude with areas for further investigation.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Abstracting methods*

Keywords

E-mail, e-mail thread, feature extraction, knowledge management, named entity extraction, text summarization

1. INTRODUCTION

Automatic summarization of e-mail messages is motivated by information overload. Whittaker and Sidner [23] and Ducheneaut and Bellotti [5] both describe how e-mail inboxes have become overloaded as personal information management devices. According to an Institute for the Future study [13], 97% of workers report using e-mail every day, or several times each week. The average user gets 24 messages a day, and "high-volume" users can easily get several hundred messages [11]. Users also feel pressured to reply quickly to e-mail messages, reporting that 27% of messages received "require" immediate attention [19]. The goal of this system is to improve users' interaction with their e-mail, by helping them prioritize new unread messages better and recall old read messages with better precision. To this end, our system exploits e-mail threads and commonly-found features to generate a more usable summary.

E-mail threads provide valuable context for summarizing e-mail messages, and allow summarization systems to exploit the structure of e-mail not found in other documents. E-mail threads are groups of replies that, directly or indirectly, are responses to an initial e-mail message. Threads are useful because they have the potential to organize groups of messages around a single topic. Ideally, e-mail threads can reduce the perceived volume of mail in users' inboxes, enhance awareness of others' contributions on a topic, and minimize lost messages by clustering related e-mail.

E-mail messages, especially in the enterprise, tend to center around people and events. *Commonly-found features* provide clues to the user about the subject matter of e-mail messages. Since much of corporate e-mail centers around collaboration, the system reports names of people and companies, and dates mentioned in e-mail messages. Reporting commonly-found features is intended to be a first-order approximation of the more general goal of reporting important aspects mentioned in e-mail messages.

Unlike archival documents, e-mail messages are often short, informal, and not well-authored. We have found that, knowing we are dealing with e-mail messages, manipulating the input can result in a more usable summary from tools which were not originally designed for this task. When a new message belonging to a thread is received, the system produces a summary of the new message in the context of its enclosing thread.

2. RELATED WORK

There have been no known explorations into the problem of summarizing e-mail by exploiting thread structure. Salton [18] did seminal research into the problem of text summarization in general. While others such as Farrell *et al.* [7] have researched discussion group thread summarization, e-mail threads differ from discussion groups in interesting ways. First, discussion databases archive all of the content of discussion groups, so discussion group threads are typically complete, whereas e-mail threads may be missing deleted messages. Second, discussion groups do not have to ad-

dress the thread computation problem, because they have a true parent-child hierarchy. Others in IBM Research are addressing the thread computation problem using heuristics for parsing both message headers and content. Lastly, the discussion group summarization algorithm relies on access to the engine underlying the summarization software, whereas our research treats the summarization software as a black box.

E-mail threads are similar to discourse, and Grosz *et al.* [10] have explored discourse theory and its representation in computer models. Rino and Scott [16] present an artificial-intelligence-based approach to a discourse model for preserving gist in the face of summarization. Ducheneaut [6] offers sample characterizations of e-mail messages, both by purpose and by structure. Meta-information about content in a similar manner might offer additional hints for summarization in future work.

Goldstein *et al.* [9], Stein *et al.* [20], and Radev [14] discuss multiple document summarization, where the system addresses summarizing related sets of documents, for example a set of news articles about an unfolding event. Multi-document summarization differs in intent from an e-mail summarization system that exploits threads. For example, in Stein *et al.*'s study [20], their summarization system is developed from the fundamental assumption that the original documents are text-only, news documents that are well-formed. In Radev's study [14] of finding new information in threaded news, the summarization system assumes that each document is labeled with the main location where the news story occurs. Lastly, messages in an e-mail thread are distinct replies to one another, thus each message contributes a unique viewpoint to the subject matter of the thread. In other words, the subject matter of a document set is fixed, whereas the subject matter of a thread may vary.

Boguraev *et al.* [1, 2, 3] explore the engine, TEXTRACT, behind the single-document summarization software used to implement our system. Ravin and Wacholder [15] and Wacholder *et al.* [22] discuss the name-recognition technology behind a feature extraction module named Nominator. We use a commercial implementation of both of these modules in our system, and so we found it useful to have background information on how they work.

3. IMPLEMENTATION

The prototype implementation of our system uses Lotus Notes and Domino from IBM, along with IBM Intelligent Miner for Text, a commercial product based upon TEXTRACT, as a back-end for processing e-mail messages. This algorithm, however, is not specific to either Domino or Intelligent Miner for Text, and could be implemented using any number of e-mail systems and summarization software. In particular, we have tested three summarization software programs: Intelligent Miner for Text, from IBM, Extractor, from the National Research Council of Canada, and SpeedRead, from Mirador Systems. The summarization results returned by each of these summarization packages have been very similar, and so we describe the implementation of the system keeping the choice of summarization software as a black box.

3.1 Summarization Algorithm

This algorithm makes use of knowledge specific to the e-mail domain to pre-process an e-mail message so that summarization software can generate a more useful summary from the message. The algorithm uses heuristics to remove extraneous headers, quoted text, forward information, and e-mail signatures to leave more useful text to be summarized. The resulting text represents our system's best guess at the relevant, new content of the message. Furthermore, if an enclosing e-mail thread exists, this algorithm processes the e-mail message's ancestors in the same way, to provide additional context for summarizing the e-mail message. The summarization software is then used to summarize each processed body.

A sample thread of messages is shown in Figures 1 through 5. Figure 1 shows the initial message in the thread, and Figure 2 is a response to Figure 1. Figures 3 and 4 are responses to Figure 2, and Figure 5 is a response to Figure 4. A system-generated summary of Figure 5 is shown in Figure 6. In contrast, a naive summarization of Figure 5, generated by summarizing the message's headers and body, yields an empty summary, which is not a useful result. The summarization software we use returns an empty result whenever the original document is too short to be summarized. Summaries generated by our system can involve multiple senders, even when only a single message is selected for input. The system exploits the threaded nature of e-mail to deduce enough context in the summary to remind the user of events that occurred before the input message.

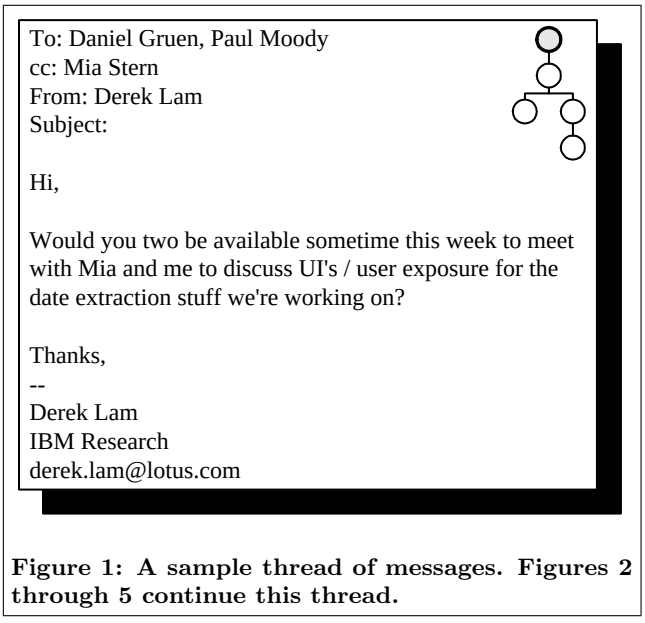

Figure 1: A sample thread of messages. Figures 2 through 5 continue this thread.

3.2 Thread Reply Structure Details

The system exploits an e-mail message's location in its enclosing thread to generate summaries. The system chooses to process only those messages that are ancestors of the original message, to shorten the lengths of the generated summaries. Farrell *et al.* [7] note that discussion group documents in a thread, similar to e-mail messages, are too short and numerous simply to offer document summarization of each document.

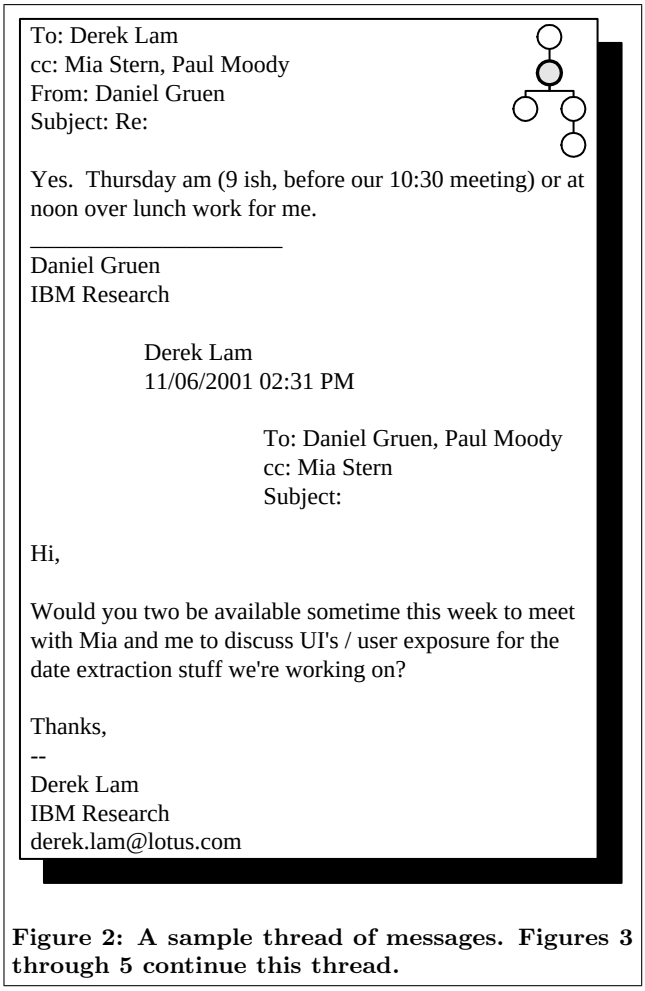

Figure 2: A sample thread of messages. Figures 3 through 5 continue this thread.

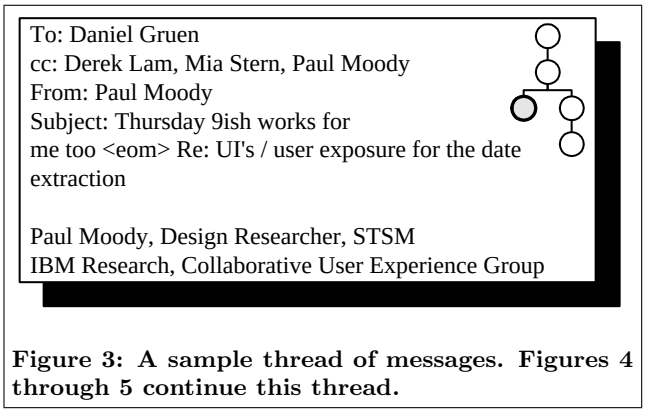

Figure 3: A sample thread of messages. Figures 4 through 5 continue this thread.

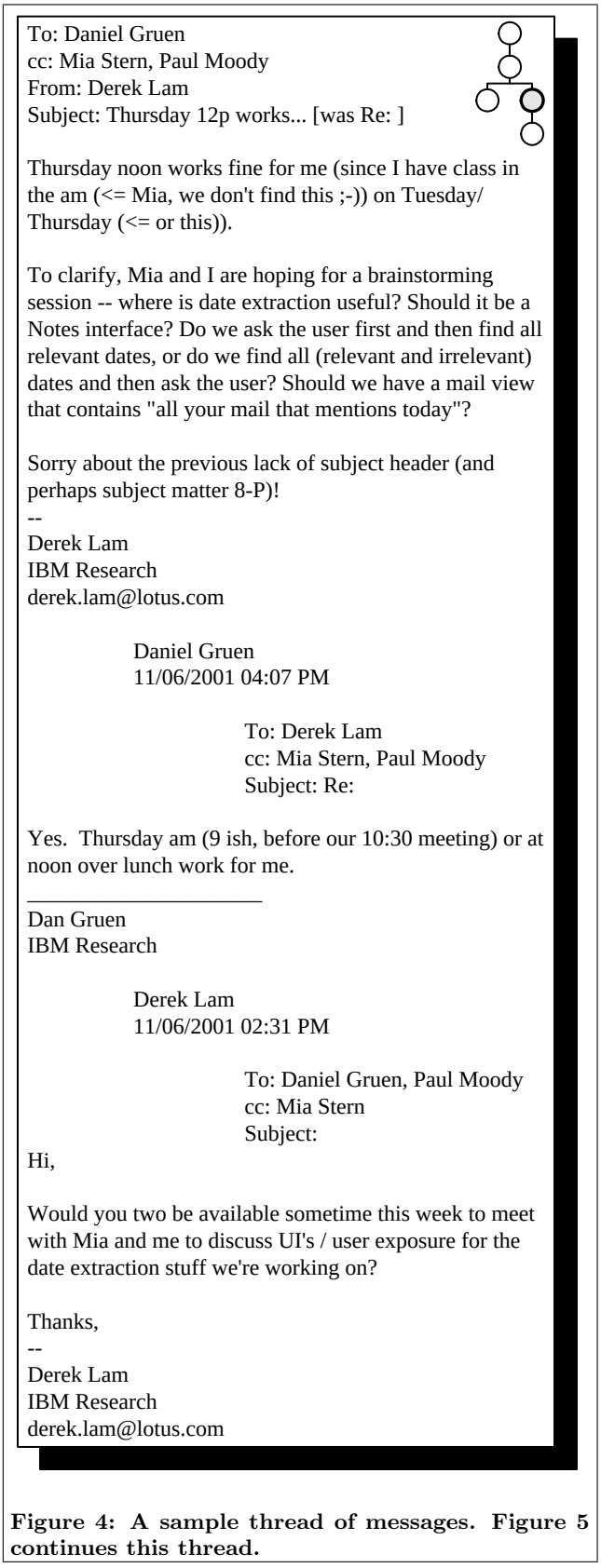

Figure 4: A sample thread of messages. Figure 5 continues this thread.

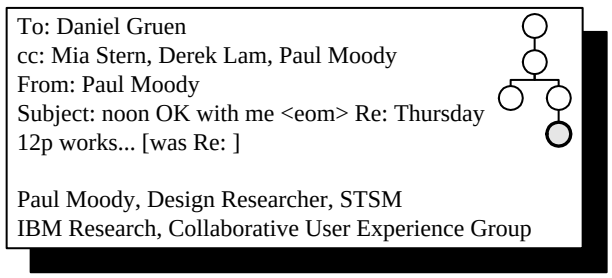

Figure 5: A sample thread of messages. Figure 6 contains a summary of this message.

The decision to summarize every ancestor from the message to the root yields a more useful summary that provides the user with more context about the thread. By the nature of e-mail threads, messages in threads are most often replies only to their ancestors in the thread. We choose not to summarize the entire thread because e-mail messages in a thread are often too short and numerous to result in a short summary. Our system's approach does not summarize the message's children for the same reason, although the children might potentially provide useful information for a summary. Choosing to summarize the message and only its ancestors was a design decision resulting in summaries of average length whose content was still usable. One issue with our current algorithm is that the length of the summary increases, as the message's position in its enclosing thread deepens.

Derek Lam wrote:
```
Hi,
Would you two be available sometime this week
to meet with Mia and me to discuss UI's / user
exposure for the date extraction stuff we're
working on?
Thanks,
```

Daniel Gruen wrote:
```
Re:
Yes.  Thursday am (9 ish, before our 10:30
meeting) or at noon over lunch work for me.
Derek Lam
11/06/2001 02:31 PM
```

Derek Lam wrote:
```
Do we ask the user first and then find all
relevant dates, or do we find all (relevant and
irrelevant) dates and then ask the user?
```

Paul Moody wrote:
```
noon OK with me <eom> Re:  Thursday 12p works...
[was Re:  ]
```

Figure 6: A summary of an e-mail message, generated by the system. The full thread of messages is shown in Figures 1 through 5.

3.3 Feature Extraction Details

E-mail messages, especially in the enterprise, tend to center around people and events. Since most summarizers do not have the functionality to understand e-mail messages, the system instead makes a first-order approximation and extracts names, dates, and company names mentioned in e-mail messages.

3.3.1 Name and Company Extraction

Our system relies on existing feature extraction software to find names of people and companies mentioned in certain messages. If any names are found, they are shown at the bottom of the summary. (The message in Figure 4, summarized in Figure 6, contained no features to report.) Potentially, the names and companies mentioned at the bottom of a summary might offer clues to the user that the original message is worth reading.

Feature extraction can be improved by *training* the feature extraction software to recognize names it might otherwise have skipped. This system's approach recognizes that named entities for training feature extraction software can be found in seemingly unrelated repositories. For example, users can pre-train the feature extraction software by aggregating contact data from their organizer information, including e-mail inboxes, electronic address books, and instant messaging buddy lists. After extracting names from users' electronic repositories, these contact data are synthesized into a training document, to train the software to recognize acquaintances listed in the user's contact lists.

3.3.2 Date Extraction

In some instances, commercially-available feature extraction software does not contain the functionality needed to identify complex dates in documents. IBM Intelligent Miner for Text does indeed recognize dates, but in a simplistic manner. On the other hand, our system applies regular expressions to identify more complex dates, and then uses the identified dates in novel ways.

Nardi *et al.* [12] have previously investigated the problem of identifying dates in documents, and Ge *et al.* [8] have previously investigated the problem of identifying dates in meeting e-mail messages. However, we believe the manner in which the identified dates are used, to aid in summarization, to be novel.

Our use of regular expressions in the implementation is perhaps too simple, because regular expressions simply match substrings in documents. Ideally, the system would use surrounding context to deduce that some false matches are in reality not dates. For example, regular expressions identify the string `3-2001` as the month of March in the year 2001. However, surrounding context might dictate that the string is not a date, for example if it is contained in "`Call me at 253-2001`." Also, regular expressions do not combine related dates or times if they are not directly adjacent in the e-mail message.

Currently, the system uses "false positive" regular expressions to handle the above contextual issue. We write regular expressions to match cases which might be tagged incorrectly as dates, such as phone numbers or newspaper URL's, where articles are often filed into a directory structure by date. Our system compares potential date matches against the false positive regular expression. If there is a match, the system rejects the string.

Each date and time, once it has been identified with regular expressions, is parsed semantically to determine its meaning. For example, if an email message received on December 5, 2001 contains the phrase "`next Monday at 2`," the date extraction system will process this phrase as December 10, 2001 2:00PM. Heuristics are used to make this semantic analysis, as well as to fill in under-specified information, such as the missing AM/PM in the previous example. Some other systems only assume the current year when parsing dates. Thus, if an e-mail message received in December 2001 reads "`Let's get together in January`," some software interprets "`January`" as January 2001. Our system instead interprets the above date as January 2002, since the e-mail was received in December 2001.

We anticipate many future uses for dates extracted from e-mail, such as being able to search one's inbox for e-mail mentioning a certain date, regardless of its format. For example, a user could search for `12/5/01` and find messages containing `12-05`, `December 5, 2001`, `Dec. 5, '01`, or even `tomorrow`, if that message was sent on December 4, 2001. Another application might be to add messages mentioning dates automatically, as appointments, to calendar software, with a system-generated e-mail summary as the subject of the appointment. The subject of an appointment is often different from the *Subject:* line of the message motivating the appointment.

4. USER STUDY

Boguraev and Neff [4] and Stein *et al.* [20] both note that evaluating summarization results is non-trivial, because there is no such thing as a "canonical" best summary. This section presents the results of a user study, in which we sat down with four users of our system and documented their experiences using the system. We suggest a partial taxonomy of e-mail adminstration tasks, for which summarization systems might be particularly well-suited. Lastly, we describe some interesting conclusions for summarization research in general.

We performed a pilot installation of the tool in four sample users' mailboxes. An ideal implementation of this user test would have been to generate summaries of new mail as it is received. Unfortunately, such an implementation was not technically feasible. Instead, we distributed an *agent*, an add-on to Lotus Notes, which provides summaries when the agent is invoked from the Notes menu. To generate a summary, the user clicks the message to be summarized, and then chooses a "Summarize..." menu item.

Participants performed test runs of the system on approximately ten of their own messages each, before we asked open-ended questions about the system. Most participants were given the chance to use the system for a few days before their interviews. We felt it was important for users to use their own mail when interacting with the system, as opposed to evaluating an inbox pre-seeded with sample e-mail messages and summaries. Since the nature of e-mail is extremely personal, users tend not to appreciate the value of a system unless they can use it on their own e-mail content. This choice also ensured that the results would not be dependent on how well sample messages approximated the contents of users' inboxes.

Typical interview lengths ranged from half an hour to an hour. During the interview process, all four participants ran the system on a random sampling of their own messages, mostly to provide examples for discussion during some of the questions.

4.1 Overall Impressions

All four participants felt that the system would help prioritize which e-mail messages to read, but they did not feel that the system would obviate the need to read certain classes of e-mail messages. [P1] remarked that the summarization system would change his behavior by better helping him decide whether to allow new mail to interrupt his workflow. That is, the decision for which the system helps is, "Do I deal with this message now, or not?" All participants felt that the system would save them time, especially if it were better integrated into their e-mail experience.

4.2 Tasks

We asked the participants about various hypothetical task scenarios, and they commented based on their experience using the system. Here we describe the *cleanup*, *calendaring*, and *triage* tasks. In the cleanup task, participants discussed using summaries to deal with mail that they had already read. In the calendaring task, participants commented on using summaries as subjects for automatically-generated appointments. In the triage task, participants discussed dealing with an overwhelming volume of unread mail, and also dealing with new mail belonging to a familiar thread.

4.2.1 Cleanup

Most participants found summaries useful for cleaning up and organizing old e-mail (3 subjects). The task is to decide whether to keep or delete previously-read messages. [P2] commented that "[summaries] could [help], if I'm doing a cleanup and I don't feel like reading all my messages, [the summary] tells me I'm not interested, and I can throw that [message] away." "If I'm [skimming] through my messages to figure out what to get rid of, I don't want to read the whole message; I think reading the summary would be sufficient." The disagreeing participant intentionally saves all of his e-mail, and noted as a result that the system would not make him any more or less likely to delete e-mail.

4.2.2 Calendar

Some participants would use summaries as system-generated calendar appointments (2 subjects). Some subjects would not find the system useful, because of the length of the resulting summaries (2 subjects). [P3] remarked that "[the summaries] are too long to be included as [short] items, or hover-over pop-up windows." All participants remarked that whether they would use summaries as appointment subjects would depend on the quality of the message's *Subject:* line. This observation suggests that the priority for appointment descriptions is that they be short, almost to the point of terseness, so that multiple appointments can be shown on a particular day or hour. The summaries, as they are currently shown, tend not to be as short as a human-generated summary which might paraphrase the content of an e-mail message.

4.2.3 Triage

We asked participants for their opinions of the system in two hypothetical situations involving mail triage. The triage task describes actions on new, unread messages. Example actions might include prioritizing, reading, responding to, or deleting or filing new messages. In the first hypothetical situation, the participant had returned from vacation to find a large volume of messages waiting in his inbox. The second hypothetical situation was set during a typical day, when the participant received a new message in a thread with which he was already familiar. Participants had different reactions to these two situations.

In the first situation where the task was to triage a large volume of messages, some participants thought that summaries would help (2 subjects). [P3] noted that, "if the summaries were shorter, and there were a good [user interface] to get to them, ... I think [summaries] would help to choose which messages were important ones to deal with." Participants who did not think summaries would help decide which messages to read had mixed reasons. [P1] commented that summaries would not help with messages that he would read regardless of the summary quality (for example, if the message were from his manager). He added that summaries might be helpful for newsletters, and that summaries might be better if the document summarization software recognized a "summary" at the top of a newsletter message. [P4] remarked that "in [the triage] situation, the most useful [feature] would actually be even a level before [summaries]. I've got so much mail that these kind of summaries, or this amount of text, wouldn't be enough. That would be one part of the solution."

In the second situation where participants are already active participants in a thread, then most participants did not feel that the system's results would help them decide whether to read new messages in that thread. [P3] explained that "the context provided in the summaries is unnecessary information. What you'd really like in some sense is to generate a summary, and then pull out context that you know you've already shown to the user before." [P1] mentioned that summaries would help if a message belonged to a "broad" thread, with many participants. In that case, summaries would help decide whether to read messages from infrequent contributors to the thread.

4.3 Summarization Software Quality

All participants noted that summary quality tends to be hit-or-miss. [P2] said that one summary of a newsletter "is actually an excellent summary. Instead of having to read the whole [message], this tells me, 'I'm *so* not interested in this!'" However, on other messages, summaries were not as informative. As a result of the above issues, a user interface which allows users to skim the summaries quickly becomes even more important.

Another aspect of interest is that all participants thought summaries were judged best on either very short or very long e-mail messages. Messages "in the middle" of the length spectrum tended to have the largest occurrence of mixed results. Typically on short messages, our chosen summarization software reported blank results, so the system uses the processed body as a summary whenever the software returns no results. The summarization software tends to perform better on longer messages, because longer documents tend to approximate the documents for which the summarization software was tuned.

Most subjects noted that their interaction with the system might improve if the document summarization software could describe why it chose certain sentences to include in a summary. Unfortunately, since most of the document summarization software with which we have experimented is statistically-based, the possibility that it could be augmented to explain its choice of sentences seems unlikely.

Lastly, all participants noted that the summaries would be more useful if they could somehow be "action-oriented." That is, if there are items in the e-mail message requiring the recipient to take some action, it would be useful for those items to appear in the summary. This feature request is interesting, because it highlights one area where using domain-independent summarization software may not have been the best choice for this domain-specific task.

4.4 Thread Reply Structure

All participants found the additional background context represented by the message's ancestors to be useful. [P1] noted that the context helps when dangling anaphora in one message's summary refer to content described earlier in the background thread context. For example, one summary contained a sentence starting with "These pictures," and a background message's summary happened to elaborate on exactly which pictures were being referenced in the original message. [P4] commented that "there are times in which [the background context] would be useful, particularly if it were displayed in a somewhat different way. It seems [in this UI] like the background context actually takes precedence."

While acknowledging the utility of including background context, all participants also noted that summaries tended to run long, sometimes prohibitively so. In some cases, particularly where the original message was short, reading the summaries took longer than reading the message itself. [P4] noted that "in fact, this [summary] is in some ways *more* confusing, and probably about as wordy, as actually just looking at the messages with [preview panes]... The entire messages here are very, very brief."

4.5 Feature Extraction

Unfortunately, no participants found feature extraction as useful as originally hoped. Common complaints were, "I don't use it because there's too much noise." Some participants noted that, if feature extraction software worked perfectly, then they would use the reported entities. According to [P4], the commonly-found features reinforce what he already knows. [P4] elaborates that "the 'names' list is a large, large list. Without the noise, the commonly-found features might be a decent index. This [feature extraction] is not as useful, at this stage. If [the features found] were cleaner, in some cases it could be useful in certain types of messages, where one of my questions would be, 'Who's mentioned in this? What companies? What names?' There's a lot of stuff I've got to read through that's *not* names."

There appear to be two issues with the use of feature extrac-

tion to approximate choosing action-oriented items in e-mail messages. The first issue is that the feature extraction itself is "noisy." That is, the feature extraction software appears to extract substrings that are neither names, dates, nor companies. This behavior is most likely due to the fact that the feature extraction is using heuristics based on capitalization in documents to deduce that a substring might be a name, and we anticipate that this noise will occur less as the feature extraction software improves. The second issue is that the approximation of action-oriented items by commonly-found features might not be a valid approximation.

4.6 User Interface

One way in which this system could be better integrated into an e-mail client might be to allow users to rest the mouse over a message listing, and have that message's summary appear in a tooltip window. This suggestion acknowledges the hit-or-miss nature of the current e-mail summaries, empowering users to decide quickly whether the summaries are good or bad, and act on the decision. Good summaries, then, could potentially save users some time. Tooltips ensure that the user has not wasted a large amount of time waiting for a bad summary to appear.

One participant also remarked that summaries might be useful when an e-mail client uses a particular "thumbnail" visualization of a thread, and a user is trying to find and navigate to a particular message in the thread based on information in its tooltip window. [P4] said for this task that "one of the things I'm finding is that the subject line alone is often meaningless." Some users are very conscientious about changing the subject line of a reply to reflect the message's current subject matter. Others, however, tend not to look at the *Subject:* line, especially for replies which already have *Subject:* lines set.

4.7 Final Impressions

Many of the concerns of quality and length could have been obviated by an improved user interface which separated background context for users, and allowed users to find the position of summary sentences in the original document. Our work did not focus on a user interface for visualizing threads or e-mail summaries. Others, such as Rohall *et al.* [17] and Venolia *et al.* [21] are working on such user interfaces. However, this realization points to an interesting conclusion, which suggests that summarization researchers might be served better by focusing on improving the quality of a user interface, even more so than improving the quality of a summary. Both Boguraev *et al.* [2] and Goldstein *et al.* [9] hint at this conclusion in their discussion of user interfaces, but they do not make the conclusion explicit.

Another future user study might focus on the difference between generating summaries using document summarization software, and simply reporting the first few lines of an e-mail message. Boguraev and Neff [3] agree that simple approaches, such as taking the first sentence from each segment, can have a remarkable impact on the quality of the resulting summary.

5. FUTURE WORK

5.1 Summary Length

All participants in the pilot user study mentioned the prohibitive length of some summaries. A system that made more effort to curb summary length would address the *compression factor* issue described by Goldstein *et al.* [9]. Aside from trying semantic analysis of some background context found in summaries, one idea is to summarize only certain ancestors of the original message, instead of summarizing all the ancestors. The problem of deciding which ancestors to summarize, and the number of ancestors to summarize, merits further research. The root message should definitely be summarized, and the input message should definitely be summarized, but deciding which other ancestors to summarize is an interesting problem.

5.2 User Interface

The user interface to the summary results was not the focus of this research. However, the user study pointed out the importance of a user interface when displaying summaries. Participants unanimously requested an improved user interface which separated background context for users, and allowed users to find the position of summary sentences in the original messages.

5.3 User Testing

More user studies might perform a useful in-depth analysis of this work. In particular, comparing this system to a system that simply extracted the first few lines of an e-mail message would yield interesting comparisons. We anticipate that, for users who put the important content of an e-mail message in the first few lines of the message, this summarization method would be sufficient. However, for longer messages, we suspect that the summarization software might perform better at extracting the key ideas.

A test of this system's performance using different summarization software would also prove interesting. Although an informal test yielded similar performance from all software packages, a more formal test might find that one software package performs better than others at extracting action-oriented items in e-mail messages.

6. CONCLUSION

We have presented a system that leverages structure inherent in e-mail messages to provide a better summary than simply running the unprocessed message through the summarization software. We find feature extraction and message pre-processing to be the best way of generating useful summaries thus far. Our system uses existing single-document summarization software to generate a summary of the discourse activity in an e-mail message and thread dynamically. The summary is augmented by also reporting any names, dates, and companies present in the message.

Our summarization system is a hybrid between single- and multi-document summarization systems. As such, it addresses some concerns put forth by proponents of multi-document summarization systems, such as *anti-redundancy methods*, the *co-reference* issue, and the *temporal dimension* [9]. The system removes as much redundant information as possible from reply e-mail messages, so that the only material summarized is what was written by the sender of the e-mail message.

The research performed for this paper has exposed two interesting conclusions. The first conclusion is practical for defining a set of tasks for which summarization might prove useful. Example tasks include prioritization and cleanup, and potentially calendaring or triage. The application of summarization to this set of tasks would be vastly improved by a better user interface. In fact, our second conclusion is that summarization researchers might be served better by focusing on improving the quality of a user interface, even more so than improving the quality of a summary.

7. ACKNOWLEDGMENTS

Thanks to Daniel Gruen and Paul Moody for providing useful dialogue and ideas during the development and implementation of this system. Thanks to Michael Muller for help designing the pilot user study of this system, and for discussing further user studies.

8. REFERENCES

[1] B. Boguraev, R. Bellamy, and C. Kennedy. Dynamic presentations of phrasally-based document abstractions. In *Hawaii International Conference on System Sciences (HICSS-32): Understanding Digital Documents*, Maui, Hawaii, January 1999.

[2] B. Boguraev, C. Kennedy, R. Bellamy, S. Brawer, Y. Y. Wong, and J. Swartz. Dynamic presentation of document content for rapid on-line skimming. In *Proceedings of AAAI Symposium on Intelligent Text Summarization*, pages 118–128, Stanford, CA, 1998.

[3] B. K. Boguraev and M. S. Neff. Discourse segmentation in aid of document summarization. In *Proceedings of Hawaii International Conference on System Sciences (HICSS-33)*, Maui, Hawaii, January 2000. IEEE.

[4] B. K. Boguraev and M. S. Neff. Lexical cohesion, discourse segmentation and document summarization. In *RIAO-2000*, Paris, April 2000.

[5] N. Ducheneaut and V. Bellotti. E-mail as habitat: An exploration of embedded personal information management. *ACM Interactions*, 8(1):30–38, September-October 2001.

[6] N. B. Ducheneaut. The social impacts of electronic mail in organizations: a case study of electronic power games using communication genres. *Information, Communication and Society (iCS)*, 5(1), 2002.

[7] R. Farrell, P. G. Fairweather, and K. Snyder. Summarization of discusion groups. In *Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM 2001)*, pages 532–534, Atlanta, GA, USA, 2001.

[8] N. Ge, X. Luo, S. Roukos, N. Kambhatla, and J. Chai. Extracting information about meetings from email messages. `niyuge@us.ibm.com`.

[9] J. Goldstein, V. Mittal, J. Carbonell, and J. Callan. Creating and evaluating multi-document sentence extract summaries. In *Proceedings of the Ninth International Conference on Information and Knowledge Management (CIKM 2000)*, pages 165–172, McLean, Virginia, United States, 2000.

[10] B. J. Grosz, M. E. Pollack, and C. L. Sidner. *Foundations of Cognitive Science*, chapter 11: Discourse. MIT Press, Cambridge, MA, 1989.

[11] M. Levitt. Email usage forecast and analysis, 2000-2005. IDC Report 23011, IDC, September 2000.

[12] B. A. Nardi, J. R. Miller, and D. J. Wright. Collaborative, programmable, intelligent agents. *Communications of the ACM*, 41(3):96–104, March 1998.

[13] "Pitney Bowes study reveals increased use of electronic communications tools among North American and European workers". Pitney Bowes press release, August 7, 2000.

[14] D. R. Radev. Topic shift detection - finding new information in threaded news. Technical Report TR CUCS-026-99, Columbia University, 1999.

[15] Y. Ravin and N. Wacholder. Extracting names from natural-language text. IBM Research Report 20338, IBM Research Division, 1997.

[16] L. H. M. Rino and D. Scott. A discourse model for gist preservation. In *Brazilian Symposium on Artificial Intelligence*, pages 131–140, 1996.

[17] S. L. Rohall, D. Gruen, P. Moody, and S. Kellerman. Email visualizations to aid communications. In *Late-Breaking Hot Topics Proceedings of the IEEE Symposium on Information Visualization*, pages 12–15, San Diego, CA, October 2001.

[18] G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*, chapter 12.3: Automatic Abstracting Systems, pages 439–448. Addison-Wesley Publishing Company, Reading, Massachusetts, 1989.

[19] "The Spam Within: Gartner says one-third of business email is 'occupational spam'". Gartner, Inc. press release, April 19, 2001.

[20] G. C. Stein, A. Bagga, and G. B. Wise. Multi-document summarization: Methodologies and evaluations. In *Proceedings of the 7th Conference on Automatic Natural Language Processing (TALN '00)*, pages 337–346, October 2000.

[21] G. D. Venolia, L. Dabbish, J. Cadiz, and A. Gupta. Supporting email workflow. Technical Report MSR-TR-2001-88, Microsoft Research, Collaboration & Multimedia Group, One Microsoft Way, Redmond, WA 98052 USA, December 2001.

[22] N. Wacholder, Y. Ravin, and M. Choi. Disambiguation of proper names in text. In *Proceedings of the 5th Applied Natural Language Processing Conference*, pages 202–208, Washington, D.C., March 1997.

[23] S. Whittaker and C. Sidner. Email overload: exploring personal information management of email. In *Conference proceedings on Human factors in computing systems*, pages 276–283, New York, NY, USA, 1996. ACM Press.